QODIQA-STD-1.0 · VERSION 1.0

The QODIQA *Specification*

Deterministic runtime consent enforcement for artificial intelligence systems: the complete corpus of 32 documents as one versioned edition.

I ORIENTATION AND FOUNDATIONS	II NORMATIVE TECHNICAL SPECIFICATIONS	III OPERATIONAL DOCUMENTS	IV GOVERNANCE AND LEGAL INSTRUMENTS	V CONTEXTUAL, ANALYTICAL AND PROCESS SUPPORT
01 IDX	04 FRAMEWORK	12 IMPL	21 GOV	26 EB
02 WP	05 ARCH	13 CL	22 RISK	27 EIP
03 CORE	06 SEC	14 RIS	23 CERT	28 RAM
	07 CTSS	15 FEM	24 PSLI	29 AUDIT
	08 CVS	16 FCM	25 PSL	30 ARP
	09 SEC	17 SBTM		31 UCD
	10 TM	18 INTEROP		32 CCL
	11 GEI	19 TND		
		20 WE		

NORMATIVE 28 · INFORMATIVE 2 · NOT STATED 2

32 DOCUMENTS

5 PARTS

1219 PAGES

EDITION 1.0 · 2026-10-29 · STATUS: PUBLIC RELEASE · LICENCE: CC BY 4.0
WWW.QODIQA.COM · CONTACT@QODIQA.COM

QODIQA-STD-1.0 | V1.0 | 2026-10-29 | PUBLIC RELEASE

DOCUMENT CONTROL

QODIQA-STD-1.0

TITLE	The QODIQA Specification (QODIQA-STD-1.0)
VERSION	1.0
EDITION	First compiled edition: the 32 documents of the public corpus in one PDF, in dependency order. Presentation only; the text of every document is reproduced verbatim.
DATE	2026-10-29
STATUS	Edition 1.0, compiled from the public corpus. Each document keeps the Type and Status declared in its own header; a marker on every document opener and running header states NORMATIVE, INFORMATIVE or NOT STATED, derived only from those header values (rule below).
SOURCE	`https://github.com/qodiqa/qodiqa` at commit `d3b5066a29b1a32f20d30301311dcbd4e30d2ed9`. SHA-256 of every source file: Appendix C.
LICENCE	Creative Commons Attribution 4.0 International (CC BY 4.0), as stated in the LICENSE file of the source repository. The corpus also contains the document QODIQA-PSLI-2026-001 (Public Specification License), reproduced in Part IV exactly as published; its opening notice states that CC BY 4.0 supersedes its terms (see the issues register).
HOW TO CITE	QODIQA. *QODIQA-STD-1.0: The QODIQA Specification*, Version 1.0, 2026-10-29. https://github.com/qodiqa/qodiqa. Licensed under CC BY 4.0. To cite one document, give its Document Identifier (for example QODIQA-CORE-2026-001) and section.
CONTACT	www.qodiqa.com · contact@qodiqa.com
REPRODUCIBILITY	The build is deterministic: the PDF carries fixed metadata dates and no timestamps; its SHA-256 is published beside it in `QODIQA-STD-1.0.pdf.sha256`.

Change history

Corpus entry reproduced verbatim from QODIQA-CCL-2026-001 (Corpus Change Log and Version History), Section 4, followed by the entry for this compiled edition.

ENTRY TYPE	INITIAL CORPUS PUBLICATION: GENESIS EVENT
ENTRY REFERENCE	QODIQA-CCL-ENTRY-001

PUBLICATION DATE	2026
CORPUS VERSION	1.0: all 28 corpus documents
DOCUMENTS AFFECTED	All 28 corpus documents: QODIQA-WP, QODIQA-CS, QODIQA-EF, QODIQA-CF, QODIQA-IP, QODIQA-RA, QODIQA-SC, QODIQA-CT, QODIQA-RM, QODIQA-GC, QODIQA-EI, QODIQA-TM, QODIQA-ID, QODIQA-TD, QODIQA-PL, QODIQA-PS, QODIQA-RR, QODIQA-UC, QODIQA-NC, QODIQA-CV, QODIQA-AE, QODIQA-SB, QODIQA-FH, QODIQA-AT, QODIQA-ID-B, QODIQA-TD-B, QODIQA-PS-B, QODIQA-RR-B.
CHANGE TYPE	Genesis Publication. This is not an amendment to any prior version. No prior version of any affected document exists in the governance record.
RATIONALE	Initial formal and synchronized release of the complete QODIQA specification corpus as a publicly available, internally consistent, normatively operative standard for deterministic runtime consent enforcement in artificial intelligence systems.
PRIOR VERSION	None. This entry records the genesis state of the corpus. No prior formal publication exists for any corpus document.
GOVERNANCE AUTHORITY	QODIQA Governance Charter (QODIQA-GC v1.0)
CLASSIFICATION	Public Release
ENTRY STATUS	CANONICAL AND IMMUTABLE. This entry may not be amended, annotated, superseded, or deleted by any future governance action. It constitutes the permanent opening record of the QODIQA Corpus Change Log.
QODIQA-STD-1.0 1.0	2026-10-29: first compiled edition. Compiled from the 32 corpus files at the commit above. No change to the text of any document.

Normative and informative markers

NORMATIVE

The document's own Status or Type header (or its status block) contains the word "Normative".

INFORMATIVE

The document's own Status or Type header (or its status block) contains the word "Informative".

NOT STATED

Neither word appears in the document's header; the Status it declares is shown verbatim on its opener.

Rule order: hero Status, status-block Document Status, hero Type, status-block Document Type; the first one containing "Informative" or "Normative" decides. The marker never overrides what a document says about itself in its text.

CONFORMANCE LANGUAGE

Requirement keywords

The requirement keywords used throughout this edition are defined by the corpus itself. The statements below are reproduced verbatim from the documents named, in their own words and numbering; nothing is added or harmonised. Where they differ, the difference is recorded in the issues register, not resolved here.

QODIQA-CORE-2026-001: Core Standard

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC 2119] [RFC 8174]. Non-implementation of any MUST-level requirement constitutes non-Conformance with this standard across all profiles. Deviation from any SHOULD-level requirement MUST be explicitly documented with written justification and recorded in the Conformance Statement (Section 13).

QODIQA-FRAMEWORK-2026-003: 68-Point Enforcement Framework

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals.

QODIQA-SEC-2026-001: Security and Cryptographic Profile

(statement not found)

QODIQA-TND-2026-001: Terminology and Definitions, Section 1 (Normative Language Conventions)

SECTION 1

Normative Language Conventions

The following terms, when appearing in any QODIQA normative document in uppercase form, SHALL be interpreted strictly as defined below. These conventions follow the model established in RFC 2119 but are restated here with binding force specific to the QODIQA framework. The absence of a normative keyword in a statement does not imply permissiveness; it implies that the statement is informative and carries no binding requirement.

TABLE 1: NORMATIVE KEYWORD DEFINITIONS

KEYWORD	BINDING STATUS	INTERPRETATION
SHALL	**Mandatory**	The requirement is unconditionally binding. Non-conformance constitutes a violation.
MUST	**Mandatory**	Equivalent to SHALL. Both forms carry identical binding force and are interchangeable within this corpus.
SHALL NOT	**Mandatory Prohibition**	The action or condition is unconditionally prohibited. Any occurrence constitutes a violation.
MUST NOT	**Mandatory Prohibition**	Equivalent to SHALL NOT. Identical binding force.
PROHIBITED	**Absolute Prohibition**	Stronger than SHALL NOT. Used where a violation would constitute a fundamental non-conformance that invalidates certification status or audit integrity. Context-independence is assumed.
MAY	*Permissive*	The action or condition is permitted but not required. Implementers are free to include or omit the described behavior without affecting conformance status.
SHOULD	*Recommended*	The action or condition is recommended where technically feasible. Deviation from a SHOULD-qualified requirement does not constitute non-conformance but SHALL be documented in conformance disclosures.

> Informative content within normative documents is explicitly marked with the prefix **"Informative Note:"** and does not constitute a requirement. Absence of this prefix implies normative status for all definitional and requirement statements.

1.1 Semantic Drift Prevention

Normative keywords SHALL NOT be used in lowercase form within normative documents when the intent is to invoke binding requirements. Use of lowercase "shall," "must," or "must not" in normative sections is treated as informal language and carries no binding force. Document authors are responsible for consistent application of uppercase normative keywords.

CONTENTS

Table of contents

· Document control 2

· Conformance language 4

· Table of contents 7

I Orientation and foundations 17

QODIQA-IDX-2026-001 · INFORMATIVE 18
Master Index: Reader's Guide
Abstract 18
Preamble: How to Use this Document 18
Section 1: Purpose of this Document 19
Section 2: Corpus Overview 20
Section 3: Complete Document Index 21
Section 4: Reading Paths by Audience 25
Section 5: Cross-Reference Map 27
Section 6: Consolidated Glossary 29
Section 7: Corpus Amendment History 33
Document Status: Document Status 33

QODIQA-WP-2026-001 · NORMATIVE 37
QODIQA · Consent as Infrastructure for Artificial Intelligence: Technical Whitepaper
Abstract 37
Section 1: Executive Summary 38
Section 2: Introduction 40
Section 3: The Consent Gap in Artificial Intelligence 41
3.1 Decision-Making Without Explicit Mandate 45
3.2 Implicit Consent and Its Failure Modes 46
3.3 The Difference Between Authorization and Consent 47
Section 4: Requirements for Enforceable AI Consent 49
4.1 Explicitness 49
4.2 Contextual Binding 49
4.3 Time-Bounded Validity 50
4.4 Deterministic Evaluation 50
4.5 Separation of Responsibility and Enforcement 50
4.6 Fail-Closed Semantics 50
4.7 Auditability and Verifiability 51
Section 5: The QODIQA Framework 51
5.1 Architectural Positioning 52
5.2 Input Contract 52
5.3 Consent Resolution 53
5.4 Policy Evaluation 53
5.5 Decision Outcomes 54
5.6 Failure Semantics 54
5.7 Scope and Boundaries 54
Section 6: QODIQA System Architecture 55
6.1 Architectural Overview 55
6.2 Intent Declaration: Closed by Design 56
6.3 Consent Resolution as Explicit State 57
6.4 Policy Evaluation as a Pure Function 58
6.5 Frozen Evaluation Context 59
6.6 Decision Artifacts and Identity 59
6.7 Failure Semantics 60
6.8 Why This Architecture Is Novel 61
Section 7: Determinism, Auditability, and Proof 62
7.1 Determinism as a First-Class Property 62
7.2 Decisions as Stable Artifacts 63
7.3 Replay as a Correctness Invariant 63
7.4 Auditability Without Interpretation 64
7.5 Proof Over Explanation 64
7.6 Intentional Limits 64
Section 8: Integration, Adoption, and Operational Reality 65
8.2 Latency and Performance Constraints 66
8.3 Failure Modes and Fail-Closed Behavior 66
8.4 Adoption Across Teams and Organizations 66
8.5 Resistance to Bypass 67
8.6 Operational Transparency 67
Section 9: Limitations, Trade-offs, and Non-Goals 68
Section 10: Conclusion and Outlook 70
Document Status: Document Status and Corpus Alignment 72

QODIQA-CORE-2026-001 · NORMATIVE 75
Core Standard for Deterministic Runtime Consent Enforcement
Abstract 75
Section 1: Executive Summary 76
Section 2: Normative Scope and Conformance 79
Section 3: Threat and Risk Model 82
Section 4: Core Architecture 85
Section 5: The Twelve Core Control Points (QCC-01 through QCC-12) 87
Section 6: Formal Determinism and Replay Invariants 99
Section 7: Implementation Model 101

Section 8: Interoperability and Ecosystem Compatibility 105
Section 9: Adoption Model 107
Section 10: Regulatory Alignment 108
Section 11: Economic Model Overview 109
Section 12: Governance and Versioning 110
Section 13: Conformance Statement Template 111
Institutional Closing Statement: Institutional Closing Statement 115
Document Status: Document Status and Classification 117

II Normative technical specifications 120

QODIQA-FRAMEWORK-2026-003 · NORMATIVE 121
68-Point Enforcement Framework for Runtime Consent Governance
Abstract 121
Section 1: Introduction 122
Section 2: Terminology and Definitions 122
Section 3: Architectural Overview 124
Section 4: Governance Domains (D01–D10) 125
Section 5: Lifecycle Model 134
Section 6: Cryptographic Layer and Protocols 135
Section 7: Protocol Flow (Six-Gate Sequence) 136
Section 8: Threat Model 137
Section 9: Conformance and Profiles 140
Section 10: Maturity Model 141
Sections 11–12: Interoperability and Attestation 142
Section 13: Security Considerations 143
Section 14: Privacy Considerations 145
Section 15: Normative References 146
Appendices A–E: Normative Annexes 148
Document Status: Document Status and Corpus Alignment 151

QODIQA-ARCH-2026-001 · NORMATIVE 154
Reference Architecture for Deterministic Runtime Consent Enforcement
Abstract 154
Section 1: Purpose and Scope 155
Section 2: Architectural Principles 157
Section 3: Canonical Component Model 160
Section 4: Logical Data Flow Model 167
Section 5: Trust Boundary Definition 171
Section 6: Deployment Topology Variants 174
Section 7: Deterministic Decision Boundary Model 180
Section 8: Scaling and Performance Model 183
Section 9: Failure and Degradation Model 186
Section 10: Interoperability and Extensibility Model 188
Section 11: Reference Invariants Summary 191
Section 12: Non-Coverage Disclosure 193
Section 13: Institutional Closing Statement 196
Document Status: Document Status and Classification 199

QODIQA-SEC-2026-001 · NORMATIVE 202
Security and Cryptographic Profile for Runtime Consent Enforcement
Abstract 202
Section 1: Purpose and Security Scope 203
Section 2: Threat Model Definition 206
Section 2.1: Security Invariants 207
Section 2.2: Attack Surface Mapping 209
Section 2.3: Failure Mode Enumeration and Deterministic Response 210
Section 3: Cryptographic Algorithm Requirements 213
Section 4: Consent Token Structure Security 219
Section 5: Trust Anchor and Key Management Model 225
Section 5.5: High-Assurance Key Management Model 228
Section 6: Revocation Registry Security 231
Section 7: Artifact Hashing and Integrity Model 232
Section 8: Audit Ledger Integrity Requirements 236
Section 9: Time and Clock Security 238
Section 10: Network and Transport Security Requirements 239
Section 10.3: Performance and Latency Constraints 242
Section 11: Side-Channel and Non-Deterministic Risk Mitigation 243
Section 12: Secure Deployment Profiles 245
Section 13: Compromise and Incident Response Model 246
Section 14: Non-Coverage Disclosure 248
Closing Statement: Institutional Closing Statement 250
Annex X: Normative Reference: Cryptographic Trust Model 252
Annex A: Non-Normative: Informative Cryptographic Reference Mapping 254
Document Status: Document Status and Classification 260

QODIQA-CTSS-2026-001 · NORMATIVE 263
Conformance Test Suite Specification
Abstract 263
Section 1: Purpose and Scope 264
Section 2: Conformance Testing Principles 267
Section 3: Test Environment Requirements 268
Section 4: Control Point Test Matrix 270
Section 5: Determinism Verification Protocol 288
Section 6: Fail-Closed Enforcement Tests 290
Section 7: Revocation Propagation Race Condition Tests 291
Section 8: Artifact Integrity Validation 292
Section 9: Audit Ledger Integrity Tests 293
Section 10: Negative and Adversarial Testing 294
Section 11: Performance Boundary Tests and Deterministic Benchmark Methodology 296
Section 12: Cross-Architecture and Runtime Determinism Validation 299
Section 13: Formal Gateway State Machine Model 302
Section 14: Control-to-Test Coverage Mapping Matrix 307

Section 15: Canonical Serialization Requirements 310
Section 16: Evidence Packaging Requirements 312
Section 17: Conformance Result Classification 313
Section 18: Institutional Closing Statement 315
Document Status: Document Status and Classification 318

QODIQA-CVS-2026-001 · NORMATIVE 321
Conformance Verification Specification
Section 0: Abstract 321
Section 1: Scope 322
Section 2: Normative Foundations 323
Section 3: Definition of Conformance 324
Section 4: Verification Model 325
4.1 Deterministic Verification Principle 325
4.2 Verification Layers 326
4.3 Verification Boundaries 327
Section 5: Conformance Verification Matrix 328
Section 6: Deterministic Verification Conditions 330
Section 7: Evidence Requirements 333
7.1 Enforcement Decision Artifacts 333
7.2 Audit Log Records 334
7.3 Decision Traces 334
7.4 Cryptographic Validation Records 334
7.5 Evidence Integrity Requirements 334
Section 8: Verification Failure Conditions 335
Section 9: Verification Observability 337
Section 10: Verification Execution Model 338
10.1 Pre-Execution Verification 338
10.2 Runtime Verification 339
10.3 Post-Execution Verification 339
Section 11: Certification Integration 340
11.1 Verification as Prerequisite to Certification 340
11.2 Mapping to Certification Levels 340
11.3 Mandatory Verification Requirements for Certification 340
11.4 Verification Failure Impact on Certification 341
Section 12: Prohibited Verification Behaviors 341
Section 13: Conformance Assertion 343
Section 14: Residual Verification Limitations 344
14.1 What Cannot Be Verified 344
14.2 Required System Behavior Under Residual Limitations 345
Document Status: Document Status and Corpus Alignment 347

QODIQA-SEC-TM-2026-001 · NORMATIVE 350
Threat Model and Abuse Case Specification
Abstract 350
Section 0: Introduction 351
Section 1: Threat Modeling Methodology 352
1.1 STRIDE Integration 352
1.2 Trust Boundary Analysis 353
1.3 Misuse Case Modeling and Adversarial Objective Decomposition 353
Section 2: System Boundary Definition 353
2.1 Components Inside the Enforcement Boundary 354
2.2 Components Outside the Enforcement Boundary 355
Diagram: Enforcement Boundary Overview 356
Section 3: Modeling Assumptions 357
Section 4: Severity Classification 359
Section 5: Threat Actor Taxonomy 360
5.1 External Attacker 361
5.2 Malicious Integrator 361
5.3 Model Operator Attempting Bypass 361
5.4 Compromised Infrastructure Provider 362
5.5 Governance Abuse Actor 362
Section 6: Attack Surface Enumeration 362
6.1 Inference Request Path 363
6.2 Consent Verification Interface 363
6.3 Cryptographic Key Infrastructure 363
6.4 Policy Evaluation System 364
6.5 Audit Logging Infrastructure 364
6.6 Governance and Certification Processes 364
Section 7: Adversarial Objective Model 365
Section 8: Runtime Bypass Scenarios 367
8.1 Direct Inference Endpoint Bypass 367
8.2 Forged Authorization Token Injection 368
8.3 Enforcement Gateway Removal 369
8.4 Replay Attack on Consent Artifacts 370
Section 9: Cryptographic Compromise Analysis 371
Section 10: Governance and Certification Abuse 373
Section 11: Integration and Dependency Failure 374
Section 12: Invariant Violation Analysis 376
Mitigation Mapping 377
Section 13: Residual Risk Summary 378
Section 14: Threat Coverage Matrix 380
Section 15: Formal Security Posture 382
Document Status: Document Status and Corpus Alignment 385

QODIQA-TM-2026-001 · NORMATIVE 387
Extended Adversarial Threat Model
Abstract 387
Section 1: Scope and Threat Modeling Boundaries 388
Section 2: System Context and Trust Boundaries 389
Section 3: Adversary Classification Framework 392
3.1 Classification Axes 392
3.2 Adversary Tiers 393
Section 4: Attack Surface Enumeration 394
4.1 Input Interfaces 394
4.2 Policy and Consent Store 394
4.3 Enforcement Engine 395
4.4 Audit and Logging Subsystem 395
Section 5: Threat Taxonomy 395
Section 6: Adversarial Objectives and Capabilities 397
Section 7: Inference-Time Threat Vectors 399

7.1 Request Manipulation 399
7.2 Context Flooding 399
7.3 Timing Attacks 400
Section 8: Consent Bypass Strategies 400
Section 9: Model Manipulation and Prompt Injection Threats 402
Section 10: Data Exfiltration Vectors 404
Section 11: Side-Channel and Indirect Leakage Risks 405
Section 12: Enforcement Failure Modes 406
Section 13: Multi-System and Supply Chain Threats 408
Section 14: Insider and Privileged Actor Threats 410
Section 15: Adversarial Automation and Scaling Risks 411
Section 16: Detection and Observability Constraints 413
Section 17: Residual Risk and Non-Eliminable Threats 414
Section 18: Assumptions and Limitations 416
Section 19: Appendix: Threat Mapping Tables 417
Document Status: Document Status and Corpus Position 420

QODIQA-GEI-2026-001 · NORMATIVE 422
Global Enforcement Invariants
Abstract 422
Section 1: Purpose and Scope 422
Section 2: Relationship to QODIQA Corpus 423
Section 3: Invariant Model Overview 424
Section 4: Global Enforcement Invariants 424
Section 5: Invariant Interdependency 436
Section 6: Non-Compliance Conditions (Global) 437
Section 7: Cross-Document Traceability 437
Section 8: Enforcement Implications 439
Section 9: Conclusion 440
Document Status: Document Status and Corpus Alignment 441

III **Operational documents** 444

QODIQA-IMPL-2026-001 · NORMATIVE 445
Implementation Playbook for Deterministic Runtime Consent Enforcement
Scope and Purpose 445
Section 1: Executive Technical Overview 446
Section 2: System Architecture Blueprint 449
Section 3: Deployment Topologies 458
Section 4: API Integration Model 464
Section 5: Enforcement Lifecycle: Operational Detail 469
Section 6: Performance and Scalability Model 471
Section 7: Security Hardening Guidelines 476
Section 8: Determinism Assurance Procedures 481
Section 9: Observability and Monitoring 486
Section 10: Conformance Readiness Checklist 492
Section 11: Implementation Maturity Levels 498
Section 12: Institutional Closing Statement 501
Document Status: Document Status and Classification 503

QODIQA-CL-2026-001 · NORMATIVE 506
Implementation Conformance Checklist
Purpose & Scope 506
Section 1: How to Use This Checklist 507
Section 2: Profile Overview 508
Section 3: CORE-MIN Conformance Checklist 508
Section 4: CORE-VER Conformance Checklist 512
Section 5: CORE-ENT Conformance Checklist 516
Section 6: Scoring and Conformance Determination 519
Document Status: Document Status and Corpus Alignment 522

QODIQA-RIS-2026-001 · NORMATIVE 524
Reference Implementation Specification (Minimal Deterministic Enforcement Stack)
Abstract 524
Section 1: Executive Overview 525
Section 2: System Scope and Constraints 526
2.1 Included 526
2.2 Excluded 527
Section 3: Architectural Overview 528
3.1 System Topology 528
3.2 Trust Boundaries 530
3.3 Enforcement Boundary 530
Section 4: Core Components 531
4.1 Enforcement Gateway 531
4.2 Policy Engine (Deterministic) 532
4.3 Consent Registry (Mock / Minimal) 532
4.4 Request Validator 533
4.5 Decision Engine (Allow / Deny Only) 533
4.6 Audit Log (Append-Only) 534
4.7 Artifact Generator 535
Section 5: Deterministic Enforcement Flow 535
Section 6: Fail-Closed Guarantees 538
Section 7: Audit and Evidence Model 540
Section 8: Minimal Data Model 542
8.1 Consent Object 542
8.2 Request Context 542
8.3 Policy Structure 543
8.4 Decision Record 544
Section 9: Deployment Model 544
Section 10: Non-Goals 545
Section 11: Conformance Alignment 546
Section 12: Validation Scenarios 547
Section 13: Conclusion 550

Document Status: Document Status and Corpus Alignment 552

QODIQA-FEM-2026-001 · NORMATIVE 555
QODIQA · Formal Enforcement Model · Deterministic Execution Semantics
Abstract 555
Section 1: Purpose and Scope 555
Section 2: Relationship to QODIQA Corpus 556
Section 3: Deterministic Execution Model 557
Section 4: Canonical State Machine 558
4.1 Defined States 558
4.2 Allowed Transitions 559
4.3 Prohibited Transitions 561
Section 5: Deterministic Decision Function 562
5.1 Formal Definition 562
5.2 Input Constraints 563
5.3 Output Constraints 563
Section 6: Execution Flow Constraints 564
Section 7: Formal Enforcement Invariants 566
7.1 Consent Invariants 566
7.2 Audit Invariants 567
7.3 Sequencing Invariants 568
7.4 Decision Invariants 569
Section 8: Invalid Execution States 570
Section 9: Non-Compliance Conditions 571
Section 10: Cross-Document Traceability 573
Section 11: Enforcement Implications 574
Section 12: Conclusion 575
Document Status: Document Status and Identity 576

QODIQA-FCM-2026-001 · NORMATIVE 578
Non-Compliance Conditions and Failure Modes
Abstract 578
Section 1: Scope 579
Section 2: Normative Foundations 580
Section 3: Definition of Non-Compliance 581
3.1 Hard Non-Compliance 581
3.2 Soft Non-Compliance 582
3.3 Conditional Non-Compliance 582
Section 3A: Failure State Transition Model 583
Section 4: Failure Mode Taxonomy 585
4.1 Enforcement Failure 585
4.2 Decision Integrity Failure 587
4.3 Data Integrity Failure 588
4.4 Control Plane Failure 590
4.5 Audit Failure 591
4.6 Cryptographic Failure 593
4.7 Boundary Failure 595
Section 5: Deterministic Failure Conditions 597
Section 6: Fail-Closed Requirements 598
Section 7: Non-Compliance Severity Levels 600
7.1 Level 1 - Critical Non-Compliance 600
7.2 Level 2 - Major Non-Compliance 601
7.3 Level 3 - Minor Non-Compliance 602
Section 8: Certification Impact 602
Section 9: Detection and Verifiability Constraints 604
Section 10: Prohibited System Behaviors 605
Section 11: Recovery Constraints 606
Section 12: Residual Failure Surfaces 607
Section 13: Conformance Requirements 608
Document Status: Document Status and Corpus Alignment 610

QODIQA-SBTM-2026-001 · NORMATIVE 613
System Boundary and Trust Model Specification
Abstract 613
Section 1: Executive Definition 614
Section 2: System Boundary Definition 615
2.1 Formal Boundary Definition 615
2.2 Inclusion Criteria 616
2.3 Exclusion Criteria 616
2.4 Boundary Immutability Constraint 617
Section 3: Canonical Trust Zones 618
Section 618
Section 619
Section 619
Section 619
Section 620
Section 620
Section 621
Section 621
Section 622
Architecture Reference: System Boundary and Trust Architecture 623
Section 4: Trust Model Definition 624
4.1 Trust Minimization 624
4.2 Explicit Trust Contracts 625
4.3 Revocation Requirement 625
4.4 Verifiability Requirement 626
4.5 Non-Transitive Trust Rule 627
Section 5: Enforcement Boundary 627
Section 6: Cross-Boundary Interaction Model 631
6.1 Data Flows 631
6.2 Control Flows 632
6.3 Trust Downgrade Rules 632
6.4 Validation Requirements 633
Section 7: Untrusted Component Containment 634
Section 8: Trust Violation Scenarios 637
Section 9: Trust Reduction Strategy 639
Section 10: System Invariants 641
Section 11: Non-Compliance Conditions 643
Section 12: Conformance Requirements 644
Section 13: Final Deterministic Statement 646

Document Status: Document Status and Corpus Alignment 648

QODIQA-INTEROP-CONS-2026-001 · NORMATIVE 651
Interoperability and Deployment Constraints · Version 1.0
Abstract 651
Section 0: Purpose and Scope Boundary 652
Section 1: Definitions and System Boundary Vocabulary 654
Section 2: Layered Integration Model 655
Section 3: Authorized Integration Surfaces 662
Section 4: Non-Override Guarantees 664
Section 5: Multi-Model and Multi-Vendor Deployment Constraints 666
Section 6: Agent Framework Constraints 669
Section 7: Data Flow Constraints 671
Section 8: Cryptographic and Identity Constraints 674
Section 9: Failure Mode Handling and Recovery Constraints 676
Section 10: Anti-Bypass and Adversarial Integration Controls 677
Section 11: Certification Implications 683
Section 12: Non-Coverage and Architectural Limitations 685
Declaration: Architectural Declaration 686
Document Status: Document Status and Classification 688

QODIQA-TND-2026-001 · NORMATIVE 691
Terminology and Normative Definitions
Scope 691
Section 0: Scope of Terminology Control 692
Section 1: Normative Language Conventions 5
Taxonomy: Taxonomy of Terms 695
Section 2: Core Conceptual Definitions: Foundational Layer 697
Section 3: Architectural Component Definitions 707
Section 4: Security and Integrity Terms 711
Section 5: Regulatory and Governance Terms 716
Section 6: Prohibited Interpretations 720
Section 7: Semantic Integrity Requirements 722
Section 8: Change Control of Terminology 724
Conclusion: Semantic Authority Statement 726
Document Status: Document Status and Classification 727

QODIQA-WE-2026-001 · NORMATIVE 730
QODIQA · Worked Example: End-to-End Enforcement Scenario
Purpose & Scope 730
Section 1: Scenario Setup 731
Stage 1: Intent Declaration 734
Stage 2: Consent Resolution 736
Stage 3: Policy Evaluation 738
Stage 4: Decision Artifact Production 741
Stage 5: Execution Gate 743
Stage 6: Mid-Session Revocation 745
Section 8: Audit Replay 748
Section 9: What QODIQA Did Not Do 749
Document Status: Document Status and Corpus Alignment 751

IV **Governance and legal instruments** 753

QODIQA-GOV-CH-2026-001 · NORMATIVE 754
Governance Charter for the QODIQA Standard Corpus
Preamble 754
Article 01: Institutional Purpose 755
Article 02: Governance Principles 757
Article 03: Standard Corpus Definition 759
Article 04: Versioning Model 761
Article 05: Amendment Process 763
Article 05b: Emergency Amendment Protocol 767
Article 06: Governance Structure 770
Article 06b: Standards Steward Institutionalization 773
Article 07: Change Impact Analysis Requirements 775
Article 08: Certification Oversight 776
Article 09: Transparency and Publication Requirements 778
Article 10: Safeguards Against Capture 780
Article 11: Governance Evolution and Scaling Mechanism 782
Article 12: Dissolution and Transfer Provisions 784
Article 13: Institutional Closing Declaration 786
Document Status: Document Status and Classification 789
Institutional: Scope 792
Institutional: Terminology 792
Institutional: Conformance Language 793
Institutional: Standard Corpus 794
Institutional: Institutional Disclaimer 796
Institutional: Recommended Citation 796
Institutional: Revision History 797

QODIQA-RISK-2026-001 · NORMATIVE 798
QODIQA - Residual Risk and Assumption Disclosure Annex
Abstract 798
Section 00: Executive Disclosure Statement 799
0.1Affirmative Scope 799
0.2Negative Scope - Formal Exclusions 800
Section 01: Foundational Assumptions Register 801
1.1Assumption Register 801
Section 02: Deterministic Scope Boundary 805
2.1What Deterministic Runtime Consent Enforcement Means 806
2.2What Deterministic Runtime Consent Enforcement Does Not Mean 806
2.3Formal Enforcement Perimeter Definition 807
Section 03: Residual Technical Risk Surfaces 810
3.1Prompt Injection Within Boundary (RT-04) 811
3.2Infrastructure Compromise (RT-08) 811
3.3Model Capability Overhang (RT-10) 811
3.4Risk Interdependency and Compounded Failure 812
Section 04: Regulatory Interpretation Risk 814
Section 05: Organizational Risk Dependencies 816
Section 06: Non-Coverage Declaration 817
Section 07: Cryptographic Assumption Disclosure 819

Section 08: Operational Misuse Risk 821
Section 09: Abuse and Adversarial Exploitation Scenarios 822
Section 10: Residual Societal Risk 825
Section 11: No Guarantee Clause 827
Section 12: Transparency Commitment Statement 828
Annex: Closing Statement 830
Document Status: Document Status and Classification 832

QODIQA-CERT-2026-001 · NORMATIVE 835
Certification Framework for Deterministic Runtime Consent Enforcement
Scope and Purpose 835
Section 1: Executive Certification Overview 836
Section 2: Conformance Definition Model 838
Section 3: Certification Levels 840
Section 4: Certification Evidence Requirements 844
Section 5: Conformance Test Methodology 847
Section 6: Determinism Verification Protocol 854
Section 7: Audit and Ledger Integrity Validation 858
Section 8: Revocation Propagation Certification 861
Section 9: Certification Authority Model 865
Section 10: Renewal and Continuous Compliance 867
Section 11: Public Registry Model 871
Section 12: Institutional Closing Statement 874
Section 13: Certification Lifecycle 875
Section 14: Certification Assessment Procedure 879
Document Status: Document Status and Classification 889

QODIQA-PSLI-2026-001 · NORMATIVE 892
Public Specification License
Section 892
Preamble 892
Section 1: Definitions 893
Section 2: Grant of Public Access 896
Section 3: Permitted Uses 896
3.1 Commercial Implementation 897
Section 4: Restrictions 898
4.1 Anti-Confusion Clause 899
Section 5: Derivative Specifications 899
5.1 Designation Restriction 899
5.2 Deviation Disclosure Obligation 899
5.3 Compatibility Claims 900
5.4 Prohibition on Silent Semantic Modification 900
Section 6: Conformance Claims 900
6.1 Self-Attestation Disclosure Requirement 901
6.2 False Claims 901
Section 7: Certification Boundary Protection 902
7.1 Protected Instruments 902
7.2 Certification Engagement 903
Section 8: Trademark and Naming Policy 903
8.1 Permitted Descriptive Reference 904
8.2 Prohibited Naming Patterns 905
Section 9: Intellectual Property and Patent Boundary 905
9.1 No Implied Patent License 905
9.2 Non-Infringement Disclaimer 906
9.3 Contributor Neutrality 906
Section 10: No Warranty 907
Section 11: Limitation of Liability 908
Section 12: Governance Authority 909
Section 13: Amendment Process 911
Section 14: Severability 912
Section 15: Governing Law and Jurisdiction 912
Section 16: Termination 914
Section 17: Enforcement and Remedies 915
Section 18: Institutional Continuity Clause 918
Document Status: Document Status and Classification 920

QODIQA-PSL-2026-001 · NORMATIVE 923
Positioning and Scope Limitation Statement
Preamble 923
Document Status: Intended Audience and Review Bodies 924
Section 00: Executive Positioning Summary 925
Section 01: Core Functional Positioning 927
Section 02: Explicit Non-Claims 929
Section 03: Boundary of Technical Scope 931
Section 04: Regulatory Positioning Boundary 933
Section 05: Security Scope Limitation 935
Section 06: Organizational Responsibility Retention Clause 937
Section 07: Misuse and Abuse Limitation 938
Section 08: Architectural Non-Expansion Clause 940
Section 09: Interaction with Existing Standards 941
Section 10: Certification Boundary Clarification 942
Section 11: Residual Risk Acknowledgment 944
Section 12: Formal Scope Freeze Declaration 946
Document Status: Document Status and Classification 949

V **Contextual, analytical and process support** 952

QODIQA-EB-2026-001 · NORMATIVE 953
QODIQA. Executive Brief for Deterministic Runtime Consent Enforcement. v1.0
Abstract 953
Section 0: Executive Overview 954
Section 1: Problem Definition 955
Section 2: Failure of Current Consent Models 956
Section 3: Deterministic Runtime Consent Enforcement 957
Section 4: System-Level Implications 959
Section 5: Risk Containment Capabilities 960
Section 6: Regulatory Alignment Positioning 962
Section 7: Deployment Model 964

Section 8: Strategic Impact 965
Section 9: Limitations and Non-Coverage 966
Section 10: Conclusion 967
Document Status: Document Status and Corpus Alignment 969

QODIQA-EIP-2026-001 · INFORMATIVE 971
Economic Impact Analysis of Deterministic Runtime Consent Enforcement
Abstract 971
Methodological Note: Methodological Positioning and Limitations 972
Governance Context: Relationship to Existing Governance Frameworks 974
Section 976
Section 1: Executive Economic Overview 978
Section 2: Risk Surface Categories 980
Section 3: Baseline Economic Exposure Model 982
Section 4: Economic Impact of Deterministic Runtime Enforcement 985
Section 5: Audit and Compliance Cost Model 989
Section 6: Incident Response Containment Modeling 991
Section 7: Multi-Agent System Risk Amplification Economics 993
Section 8: Implementation Cost Envelope 995
Section 9: Scenario Analysis 997
Section 10: Residual Risk and Economic Non-Coverage 1000
Section 11: Long-Term Infrastructure Stabilization Model 1002
Economic Interpretation: Economic Interpretation 1004
Section 12: Institutional Closing Statement 1006
Document Status: Document Status and Classification 1009
Institutional: Scope 1013
Institutional: Terminology 1014
Institutional: Conformance Language 1015
Institutional: Non-Coverage 1016
Institutional: QODIQA Standard Corpus 1017
Institutional: Institutional Disclaimer 1020
Institutional: Recommended Citation 1021

QODIQA-RAM-2026-001 · NORMATIVE 1022
Regulatory Alignment Matrix for AI Governance Frameworks · Version 1.0
Abstract 1022
Section 1: Purpose and Scope 1023
Section 2: Mapping Methodology 1026
Section 2.X: QODIQA Control Index and Traceability Registry 1029
Section 3: GDPR Technical Alignment 1034
Section 4: EU AI Act Technical Alignment 1039
Section 5: NIST AI Risk Management Framework Alignment 1043
Section 6: ISO/IEC 27001 Control Family Alignment 1046
Section 7: ISO/IEC 42001 AI Management System Alignment 1048
Section 8: OECD AI Principles Structural Support 1050
Section 9: Residual Risk and Non-Coverage Disclosure 1052
Section 10: Cross-Framework Comparative Matrix 1054
Section 10.X: Regulatory Ambiguity Surface Reduction Analysis 1058
Section 11: Institutional Closing Statement 1060
Document Status: Document Status and Classification 1062

QODIQA-AUDIT-2026-001 · NORMATIVE 1065
Audit and Evidence Generation Model
Abstract 1065
Section 1: System Role of Audit and Evidence 1066
1.1 Architectural Position 1066
1.2 Relationship with the Enforcement Gateway 1067
1.3 Trust Boundary Implications 1068
1.4 System Boundary and Trust Isolation 1069
Section 2: Deterministic Audit Principles 1070
2.1 Immutability 1070
2.2 Verifiability 1071
2.3 Non-Repudiation 1071
2.4 Completeness 1071
2.5 Time Consistency 1072
Invariants: Deterministic Audit Invariants 1072
Section 3: Evidence Generation Model 1074
3.1 Definition of Evidence 1074
3.2 Evidence Types 1075
Section 4: Audit Event Taxonomy 1076
4.1 Event Class: Inference Request Received 1076
4.2 Event Class: Policy Evaluation 1077
4.3 Event Class: Consent Lookup 1078
4.4 Event Class: Enforcement Decision 1079
4.5 Event Class: Block / Allow Outcome 1080
Section 5: Evidence Structure Specification 1081
5.1 Canonical Schema 1081
5.2 Field-Level Definitions 1082
5.3 Deterministic Serialization 1083
Section 6: Cryptographic Integrity Model 1083
6.1 Hash Chaining 1084
6.2 Tamper-Evidence Guarantees 1084
6.3 Signing Requirements 1085
6.4 Optional External Anchoring 1085
Section 7: Append-Only Audit Log Architecture 1086
7.1 Strict Append-Only Behavior 1086
7.2 Ordering Guarantees 1087
7.3 Replay Consistency 1087
Section 8: Evidence Retrieval and Verification 1088
8.1 Evidence Access Model 1088
8.2 Verification Flows 1089
8.3 Third-Party Validation Capability 1089
Section 9: Failure Modes and Audit Gaps 1090
Section 10: Privacy and Data Minimization 1092
Section 11: Integration with the Certification Framework 1093
Section 12: Implementation Constraints 1095
Section 13: Residual Risk and Non-Coverage 1097

Document Status: Document Status and Corpus Alignment 1099

QODIQA-ARP-2026-001 · NORMATIVE 1102
Audit Readiness and Evidence Pack
Abstract 1102
Section 1: Purpose and Scope 1103
Section 2: Audit Readiness Framework 1104
2.1 Readiness Dimensions 1104
2.2 Evidence Categories 1105
2.3 Readiness Maturity Levels 1105
Section 3: Evidence Architecture 1106
3.1 Decision Artifact Evidence 1107
3.2 Consent State Evidence 1108
3.3 Policy Version Evidence 1109
3.4 Replay Verification Evidence 1110
Section 4: Regulatory Compliance Mapping 1110
4.1 General Data Protection Regulation (GDPR) Mapping 1111
4.2 EU AI Act Mapping 1111
4.3 Sectoral Obligations 1112
Section 5: Audit Trail Requirements 1113
5.1 Structural Requirements 1113
5.2 Retention Policy 1114
5.3 Integrity Verification 1114
Section 6: Verification Procedures 1115
6.1 Pre-Audit Checklist 1115
6.2 Decision Replay Protocol 1116
6.3 Policy Immutability Check 1117
6.4 Consent Coverage Analysis 1118
Section 7: Evidence Pack Structure 1118
7.1 Required Documents 1119
7.2 Artifact Inventory 1119
7.3 Submission Format 1120
Section 8: Operational Readiness 1120
8.1 Continuous Readiness Practices 1121
8.2 Incident Evidence Handling 1121
8.3 Denial Record Preservation 1122
Section 9: Limitations and Scope 1122
Section 10: Conclusion 1124
Document Status: Document Status and Corpus Alignment 1125

QODIQA-UCD-2026-001 · NOT STATED 1128
Use Case Dossiers for Runtime Consent Enforcement Deployments
Scope and Purpose 1128
Dossier I of IV: QODIQA Use Case Dossier - Healthcare and Clinical AI Systems 1129
D1 - Section 1: Sector Context Overview 1129
D1 - Section 2: Current Runtime Consent Failure Surface 1130
D1 - Section 3: Execution Path Without Deterministic Enforcement 1131
D1 - Section 4: Execution Path With QODIQA Runtime Enforcement 1131
D1 - Section 5: Risk Surface Reduction Analysis 1133
D1 - Section 6: Evidence and Artifact Model 1134
D1 - Section 7: Operational Constraints 1135
D1 - Section 7a: Enforcement Bypass and Adversarial Risk Considerations 1137
D1 - Section 8: Residual and Out-of-Scope Risks 1139
D1 - Section 9: Institutional Closing - Healthcare Dossier 1139
Appendix A1 - Dossier I: Enforcement Deployment Maturity Levels (Illustrative) 1140
Dossier II of IV: QODIQA Use Case Dossier - Financial Services and Algorithmic Decisioning 1141
D2 - Section 1: Sector Context Overview 1142
D2 - Section 2: Current Runtime Consent Failure Surface 1143
D2 - Section 3: Execution Path Without Deterministic Enforcement 1144
D2 - Section 4: Execution Path With QODIQA Runtime Enforcement 1145
D2 - Section 5: Risk Surface Reduction Analysis 1146
D2 - Section 6: Evidence and Artifact Model 1147
D2 - Section 7: Operational Constraints 1148
D2 - Section 7a: Enforcement Bypass and Adversarial Risk Considerations 1149
D2 - Section 8: Residual and Out-of-Scope Risks 1151
D2 - Section 9: Institutional Closing - Financial Services Dossier 1151
Appendix A2 - Dossier II: Enforcement Deployment Maturity Levels (Illustrative) 1152
Dossier III of IV: QODIQA Use Case Dossier - Media and Content Generation Platforms 1153
D3 - Section 1: Sector Context Overview 1154
D3 - Section 2: Current Runtime Consent Failure Surface 1155
D3 - Section 3: Execution Path Without Deterministic Enforcement 1156
D3 - Section 4: Execution Path With QODIQA Runtime Enforcement 1156
D3 - Section 5: Risk Surface Reduction Analysis 1158
D3 - Section 6: Evidence and Artifact Model 1159
D3 - Section 7: Operational Constraints 1160
D3 - Section 7a: Enforcement Bypass and Adversarial Risk Considerations 1162
D3 - Section 8: Residual and Out-of-Scope Risks 1164
D3 - Section 9: Institutional Closing - Media and Content Generation Dossier 1164
Appendix A3 - Dossier III: Enforcement Deployment Maturity Levels (Illustrative) 1165
Dossier IV of IV: QODIQA Use Case Dossier - Enterprise AI Assistants and Knowledge Systems 1166
D4 - Section 1: Sector Context Overview 1167
D4 - Section 2: Current Runtime Consent Failure Surface 1168

D4 - Section 3: Execution Path Without Deterministic Enforcement 1169
D4 - Section 4: Execution Path With QODIQA Runtime Enforcement 1169
D4 - Section 5: Risk Surface Reduction Analysis 1171
D4 - Section 6: Evidence and Artifact Model 1172
D4 - Section 7: Operational Constraints 1173
D4 - Section 7a: Enforcement Bypass and Adversarial Risk Considerations 1174
D4 - Section 8: Residual and Out-of-Scope Risks 1176
D4 - Section 9: Institutional Closing - Enterprise AI Assistants Dossier 1177
Appendix A4 - Dossier IV: Enforcement Deployment Maturity Levels (Illustrative) 1178
Document Status: Document Status and Classification 1180

QODIQA-CCL-2026-001 · NOT STATED 1183
Corpus Change Log and Version History
Abstract 1183
Section 1: Initial Corpus Release 1184
Section 2: Corpus Synchronization State 1187
Section 3: Version Baseline Lock 1189
Section 4: Change Log Entry 1190
Section 5: Governance Implication 1192
Section 6: Prohibited States 1193
Section 7: Forward Compatibility 1194
Governance Statement: Governance Statement 1195
Document Status: Document Status and Corpus Alignment 1197

A **Glossary** 1201

B **Index of requirement identifiers** 1209

C **Hash manifest** 1217

PART I

I

Orientation and foundations

3 documents

01 Master Index and Reader's Guide 18

02 Technical Whitepaper 37

03 Core Standard 75

CHAPTER 01 PART I · ORIENTATION AND FOUNDATIONS

Master Index Reader's Guide

Navigation reference, corpus hierarchy, reading paths by audience, and consolidated glossary for the standard corpus.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-IDX-2026-001	1.0	2026	Navigation Instrument	Informative	INFORMATIVE

ABSTRACT

The QODIQA standard corpus comprises twenty-nine documents spanning five normative tiers. Each document addresses a defined aspect of deterministic runtime consent enforcement for artificial intelligence systems. The corpus is internally coherent and cross-referenced throughout, but it does not contain a navigational instrument that orients a new reader, defines the reading order appropriate to a given role, or provides a consolidated reference for terms used across documents.

This document, the Master Index and Reader's Guide, fills that structural gap. It provides a complete inventory of all corpus documents with their normative status and tier classification, four role-specific reading paths, a cross-reference map of document dependencies, and a consolidated glossary of all defined terms drawn from their authoritative definitions across the corpus.

PREAMBLE

How to Use this Document

This document is a navigational instrument, not a normative specification. It does not introduce new requirements, does not modify any existing normative statement in the corpus, and does not supersede any document it references. Its sole function is to make the corpus accessible.

> If you are reading the QODIQA corpus for the first time, begin with Section 4 (Reading Paths). Identify the path that corresponds to your role and follow it. Return to Section 3 (Complete Document Index) when you need to locate a specific document, and to Section 6 (Consolidated Glossary) when you encounter an undefined term.

Readers who are already familiar with the corpus and are seeking a specific document or term may proceed directly to Section 3 or Section 6 respectively.

SECTION 1

Purpose of this Document

A technical corpus of twenty-nine documents presents a navigational challenge that is distinct from the challenge of understanding any single document within it. Individual documents are self-contained. The corpus as a whole is not. Dependencies between documents are declared through cross-references but are not visualized. The normative hierarchy is implied by document type but is not summarized in any single location. Terms are defined where they first appear but are not consolidated. Reading paths appropriate to specific professional roles are not described.

These omissions do not impair the correctness of the corpus, but they impose a non-trivial orientation cost on any new reader. An implementation engineer who begins with the wrong document will acquire correct but poorly-sequenced

knowledge. A regulator who encounters an undefined term must search across twenty-nine documents to locate its authoritative definition. An executive seeking a strategic overview has no indicated starting point.

> **SCOPE OF THIS DOCUMENT**
>
> This document provides: (i) a complete inventory of all corpus documents with normative tier and status classification; (ii) a visual hierarchy of document dependencies; (iii) four role-differentiated reading paths; (iv) a cross-reference map identifying which documents must be read before which; and (v) a consolidated glossary of all defined terms drawn from their authoritative source documents. It does not introduce any normative requirement and does not modify any existing document in the corpus.

This document is issued as a companion instrument to the corpus and should be made available wherever the corpus is distributed. It is classified as informative and carries document identifier QODIQA-IDX-2026-001.

SECTION 2

Corpus Overview

2.1 Corpus Statistics

The QODIQA standard corpus at Version 1.0 (2026) consists of twenty-nine documents. These documents are distributed across five normative tiers, from foundational conceptual documents through governance and legal instruments. The corpus is versioned as a unit under the QODIQA Governance Charter; individual documents carry their own version numbers but are released and amended as part of the corpus as a whole.

TABLE 1. CORPUS COMPOSITION BY TIER

TIER	CLASSIFICATION	DOCUMENT COUNT	STATUS
Tier I	Foundational	2	NORMATIVE
Tier II	Normative Technical	8	NORMATIVE

TIER	CLASSIFICATION	DOCUMENT COUNT	STATUS
Tier III	Operational	7	NORMATIVE
Tier IV	Governance and Legal	5	NORMATIVE
Tier V	Informative	7	INFORMATIVE

2.2 Normative Hierarchy

The corpus establishes a strict normative hierarchy. In the event of apparent conflict between documents, the following precedence order applies: Tier I documents take precedence over all others; within Tier II, the Core Standard (QODIQA-CORE-2026-001) takes precedence over all other Tier II documents; Tier IV governance documents take precedence over Tier III operational documents on matters of governance and process. Informative documents (Tier V) carry no normative force and cannot override any normative statement.

CONFLICT RESOLUTION PRECEDENCE: NORMATIVE ORDER

1. QODIQA Technical Whitepaper (QODIQA-WP-2026-001): conceptual foundation
2. QODIQA Core Standard (QODIQA-CORE-2026-001): primary normative instrument
3. 68-Point Enforcement Framework (QODIQA-68P-2026-001): detailed control specification
4. Tier II technical specifications: architecture, security, conformance
5. Tier III operational documents: implementation, deployment, failure handling
6. Tier IV governance instruments: charter, license, change log
7. Tier V informative documents: no normative force

SECTION 3

Complete Document Index

The following index lists all twenty-nine documents in the QODIQA corpus, organized by tier. Each entry includes the document identifier, full title, normative status, and a brief functional description. Documents are listed in the recommended reading order within each tier.

3.1 Tier I: Foundational Documents

Tier I documents establish the conceptual and definitional foundation of the QODIQA framework. All other corpus documents presuppose familiarity with these two documents. They should be read first by any reader approaching the corpus for the first time.

TIER I	Foundational: read first	
QODIQA-WP-2026-001	Consent as Infrastructure for Artificial Intelligence Technical Whitepaper	NORMATIVE
QODIQA-CORE-2026-001	Core Standard for Deterministic Runtime Consent Enforcement	NORMATIVE

3.2 Tier II: Normative Technical Specifications

Tier II documents specify the technical architecture, control framework, security profile, conformance requirements, and threat model of the QODIQA enforcement system. These documents expand the normative requirements established in the Core Standard into implementable technical detail. Readers undertaking implementation should read all Tier II documents after completing Tier I.

TIER II	Normative Technical Specifications	
QODIQA-68P-2026-001	68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement	NORMATIVE
QODIQA-ARCH-2026-001	Reference Architecture for Deterministic Runtime Consent Enforcement	NORMATIVE
QODIQA-SEC-2026-001	Security and Cryptographic Profile for Runtime Consent Enforcement	NORMATIVE
QODIQA-CTS-2026-001	Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement	NORMATIVE
QODIQA-CVS-2026-001	Conformance Verification Specification	NORMATIVE
QODIQA-TM-2026-001	Threat Model and Abuse Case Specification	NORMATIVE
QODIQA-ETM-2026-001	Extended Adversarial Threat Model	NORMATIVE
QODIQA-GEI-2026-001	Global Enforcement Invariants for Deterministic Runtime Consent Enforcement	NORMATIVE

3.3 Tier III: Operational Documents

Tier III documents provide the operational guidance required to implement, deploy, and maintain a QODIQA-conformant system. They address implementation patterns, failure handling, reference stack specification, system boundaries, deployment constraints, and non-compliance conditions. These documents are intended for engineering, operations, and assurance teams undertaking active implementation.

TIER III	Operational: implementation and deployment guidance	
QODIQA-IMPL-2026-001	Implementation Playbook for Deterministic Runtime Consent Enforcement	NORMATIVE
QODIQA-RIS-2026-001	Reference Implementation Specification - Minimal Deterministic Enforcement Stack	NORMATIVE
QODIQA-FHS-2026-001	Failure Handling and Recovery Specification	NORMATIVE
QODIQA-NCC-2026-001	Non-Compliance Conditions and Failure Modes	NORMATIVE
QODIQA-SBT-2026-001	System Boundary and Trust Model Specification	NORMATIVE
QODIQA-IDC-2026-001	Interoperability and Deployment Constraints	NORMATIVE
QODIQA-TERM-2026-001	Terminology and Normative Definitions	NORMATIVE

3.4 Tier IV: Governance and Legal Instruments

Tier IV documents establish the institutional governance of the corpus, define certification boundaries, specify the legal terms under which the corpus is distributed, and provide the residual risk disclosure required of all conformant implementations. These documents are primarily addressed to legal, compliance, and institutional governance functions, but all implementors are required to acknowledge the Residual Risk Annex as a condition of claiming conformance.

TIER IV	Governance and Legal: institutional instruments	
QODIQA-GC-2026-001	Governance Charter for the QODIQA Standard Corpus	NORMATIVE
QODIQA-CF-2026-001	Certification Framework for Deterministic Runtime Consent Enforcement	NORMATIVE
QODIQA-RISK-2026-001	Residual Risk and Assumption Disclosure Annex	NORMATIVE
QODIQA-PSLI-2026-001	Public Specification License	NORMATIVE
QODIQA-POS-2026-001	Positioning and Scope Limitation Statement	NORMATIVE

3.5 Tier V: Informative Documents

Tier V documents provide contextual, analytical, and process support for the corpus. They do not introduce normative requirements and carry no binding force. They are intended to support evaluation, adoption, and stakeholder communication. Readers may engage with these documents selectively based on their role and purpose.

TIER V	Informative: no normative force	
QODIQA-EB-2026-001	Executive Brief for Deterministic Runtime Consent Enforcement	INFORMATIVE
QODIQA-EIA-2026-001	Economic Impact Analysis of Deterministic Runtime Consent Enforcement	INFORMATIVE
QODIQA-RAM-2026-001	Regulatory Alignment Matrix for Deterministic Runtime Consent Enforcement	INFORMATIVE
QODIQA-AEM-2026-001	Audit and Evidence Generation Model	INFORMATIVE
QODIQA-ARP-2026-001	Audit Readiness and Evidence Pack	INFORMATIVE
QODIQA-UCD-2026-001	Use Case Dossiers for Runtime Consent Enforcement Deployments	INFORMATIVE
QODIQA-CCL-2026-001	Corpus Change Log and Version History	INFORMATIVE

SECTION 4

Reading Paths by Audience

The following four reading paths are designed for distinct professional roles. Each path identifies the minimum set of documents required to achieve functional understanding for that role, presented in the recommended reading order. Documents are listed by identifier; full titles are provided in Section 3.

These paths are not exhaustive. A reader who has completed a path for their primary role may benefit from reading additional documents outside that path, particularly from adjacent tiers. The paths are entry points, not boundaries.

PATH A

Implementation Engineer

For engineers building a QODIQA-conformant enforcement layer. Prioritizes architectural specification, control requirements, cryptographic profile, failure behavior, and the reference implementation.

01 QODIQA-WP-2026-001: conceptual foundations and the Authorization/Consent distinction
02 QODIQA-CORE-2026-001: twelve control points, decision semantics, conformance profiles
03 QODIQA-68P-2026-001: full control framework, governance domains, cryptographic layer
04 QODIQA-ARCH-2026-001: reference architecture, component model, trust boundaries
05 QODIQA-SEC-2026-001: cryptographic requirements, key management, signature verification
06 QODIQA-GEI-2026-001: enforcement invariants that must hold across all deployments
07 QODIQA-SBT-2026-001: system boundary definition and trust model
08 QODIQA-IDC-2026-001: integration surfaces, anti-bypass controls, deployment constraints
09 QODIQA-RIS-2026-001: minimal reference implementation specification
10 QODIQA-IMPL-2026-001: 30/60-day integration playbook
11 QODIQA-FHS-2026-001: failure handling, recovery, and fail-closed guarantees
12 QODIQA-NCC-2026-001: non-compliance conditions and prohibited system behaviors
13 QODIQA-TM-2026-001: threat model and bypass scenario analysis
14 QODIQA-RISK-2026-001: residual risk disclosure, required acknowledgement

PATH B

Compliance and Audit

For compliance officers, internal auditors, and assurance teams evaluating whether a QODIQA implementation meets conformance requirements and produces adequate evidence artifacts.

01 QODIQA-WP-2026-001: conceptual model, consent gap analysis
02 QODIQA-CORE-2026-001: conformance profiles (CORE-MIN / CORE-VER / CORE-ENT)
03 QODIQA-CF-2026-001: certification framework, what certification validates and does not validate
04 QODIQA-CTS-2026-001: conformance test suite, test matrix, evidence packaging requirements
05 QODIQA-CVS-2026-001: conformance verification procedures
06 QODIQA-NCC-2026-001: hard and soft non-compliance conditions, severity levels
07 QODIQA-AEM-2026-001: audit and evidence generation model
08 QODIQA-ARP-2026-001: audit readiness and evidence pack structure
09 QODIQA-RISK-2026-001: residual risk register, assumption disclosures
10 QODIQA-GC-2026-001: governance charter, amendment process, authority structure

PATH C

Regulatory and Legal

For regulatory reviewers, legal counsel, and policy advisors evaluating the QODIQA framework against applicable legal and regulatory requirements, or assessing institutional liability exposure.

01 QODIQA-POS-2026-001: scope limitation, explicit non-claims, what QODIQA does not provide
02 QODIQA-WP-2026-001: conceptual foundations, consent gap analysis
03 QODIQA-CORE-2026-001: normative requirements, conformance definition
04 QODIQA-RAM-2026-001: GDPR, EU AI Act, NIST RMF, ISO 27001, ISO 42001 alignment matrix
05 QODIQA-RISK-2026-001: residual risk disclosure, no-guarantee clause, non-coverage declarations
06 QODIQA-PSLI-2026-001: public specification license terms
07 QODIQA-CF-2026-001: certification boundary, what certification does and does not establish
08 QODIQA-GC-2026-001: governance authority, amendment procedure, corpus integrity

PATH D

Executive and Strategic

For executives, board-level stakeholders, and strategic decision-makers evaluating QODIQA as an institutional infrastructure investment. Prioritizes problem framing, economic rationale, and strategic positioning.

01 QODIQA-EB-2026-001: executive brief, strategic case, adoption rationale
02 QODIQA-WP-2026-001: Section 1 (Executive Summary) and Section 3 (The Consent Gap)
03 QODIQA-EIA-2026-001: economic impact analysis, risk surface quantification, scenario modeling
04 QODIQA-POS-2026-001: scope limitations, non-claims, what QODIQA does not guarantee
05 QODIQA-RAM-2026-001: regulatory alignment summary, compliance coverage density
06 QODIQA-UCD-2026-001: use case dossiers for sector-specific deployment scenarios
07 QODIQA-RISK-2026-001: Section 0 (Executive Disclosure), residual risk register summary

SECTION 5

Cross-Reference Map

The following table identifies the primary dependency relationships between documents in the corpus. Column "Requires" lists documents that must be read before the listed document can be fully understood. Column "Referenced by" identifies documents that cite the listed document as a normative or informative source. This map is not exhaustive of all cross-references in the corpus; it identifies structural dependencies only.

TABLE 2. PRIMARY DOCUMENT DEPENDENCIES

DOCUMENT	REQUIRES (READ BEFORE)	REFERENCED BY
QODIQA-WP-2026-001 (Technical Whitepaper)	None (entry point)	All corpus documents
QODIQA-CORE-2026-001 (Core Standard)	QODIQA-WP-2026-001	All Tier II, III, IV documents
QODIQA-68P-2026-001 (68-Point Framework)	QODIQA-WP-2026-001, QODIQA-CORE-2026-001	ARCH, SEC, CTS, CVS, GEI, IMPL, RIS, CF
QODIQA-ARCH-2026-001 (Reference Architecture)	QODIQA-CORE-2026-001, QODIQA-68P-2026-001	RIS, IMPL, IDC, SBT, FHS
QODIQA-SEC-2026-001 (Security and Crypto Profile)	QODIQA-CORE-2026-001, QODIQA-68P-2026-001	RIS, IMPL, TM, ETM, IDC
QODIQA-TM-2026-001 (Threat Model)	QODIQA-CORE-2026-001, QODIQA-ARCH-2026-001, QODIQA-SEC-2026-001	ETM, NCC, FHS, RISK
QODIQA-CTS-2026-001 (Conformance Test Suite)	QODIQA-CORE-2026-001, QODIQA-68P-2026-001, QODIQA-ARCH-2026-001	CVS, CF, AEM, ARP
QODIQA-RISK-2026-001 (Residual Risk Annex)	QODIQA-CORE-2026-001, QODIQA-TM-2026-001	CF (required acknowledgement clause)
QODIQA-CF-2026-001 (Certification Framework)	QODIQA-CORE-2026-001, QODIQA-CTS-2026-001, QODIQA-RISK-2026-001	GC, ARP

DOCUMENT	REQUIRES (READ BEFORE)	REFERENCED BY
QODIQA-RAM-2026-001 (Regulatory Alignment Matrix)	QODIQA-CORE-2026-001, QODIQA-68P-2026-001	EB (informative), EIA (informative)

SECTION 6

Consolidated Glossary

The following glossary consolidates all defined terms used across the QODIQA corpus. Each definition is drawn from its authoritative source document, identified in the reference field. Where a term is defined in multiple documents, the definition from the highest-tier document is used. Terms are listed alphabetically.

This glossary is informative. In the event of apparent conflict between a definition here and the definition in the source document, the source document governs.

Audit Log

A tamper-evident, hash-chained sequence of Audit Records produced by the Audit Layer for every enforcement decision. The Audit Log must be written before execution proceeds on any ALLOW path and must be written on all DENY paths. Write failure triggers DENY.

Source: QODIQA-CORE-2026-001, Section 4.2

Audit Record

A single immutable entry in the Audit Log corresponding to one enforcement decision. An Audit Record contains the frozen evaluation context, the decision outcome, the policy version applied, and a cryptographic reference to the Decision Artifact.

Source: QODIQA-CORE-2026-001, Section 4.2

Authorization

Standing technical permission granted to an actor or system to access a resource or perform a class of operations. Expressed through roles, access control lists, API keys, or service identities. Generally static, coarse-grained, and context-independent. Authorization answers: *Is this actor permitted to perform this class of operation?* Authorization does not satisfy the Consent requirement and must not be used as a proxy for Consent evaluation.

Source: QODIQA-WP-2026-001, Section 3.3; QODIQA-CORE-2026-001, Section 3.1

Calling System

An application, service, or agent that requests an AI action by submitting a DecisionRequest to the Enforcement Gateway. The Calling System is responsible for providing a complete, explicit intent declaration. It must not submit partial, inferred, or defaulted intent fields.

Source: QODIQA-CORE-2026-001, Section 2.3

Conformance

The state of a system that satisfies all MUST-level requirements of the declared Conformance Profile. A system claiming Conformance must declare exactly one profile (CORE-MIN, CORE-VER, or CORE-ENT) and must satisfy all requirements of that profile. Declaration of a higher profile without satisfaction of its requirements constitutes non-Conformance.

Source: QODIQA-CORE-2026-001, Section 2.4

Consent

Contextual, conditional legitimacy for a specific AI action under explicitly declared purpose, scope, time constraints, jurisdictional binding, and current Revocation state. Consent answers: *Is this specific action legitimate under present conditions?* Consent is granular, time-bounded, and revocable. It is distinct from Authorization and must be evaluated independently of it.

Source: QODIQA-WP-2026-001, Section 3.3; QODIQA-CORE-2026-001, Section 3.1

Consent Registry

The authoritative system component that stores and resolves current Consent state, including grants, expiry timestamps, and Revocation status. The Consent Registry is the single source of truth for Consent state at enforcement time. Registry unavailability must trigger DENY; no system may default to ALLOW when the Registry is unreachable.

Source: QODIQA-CORE-2026-001, Section 4.2

Consent Token

A cryptographically signed, machine-readable data structure that encodes a Consent grant for a specific purpose, scope, data category set, jurisdictional binding, and time window. A Consent Token must be verified for cryptographic integrity and checked against the Consent Registry for current validity before any enforcement decision proceeds.

Source: QODIQA-CORE-2026-001, QCC-02; QODIQA-68P-2026-001

Decision Artifact

An immutable, cryptographically signed record of an enforcement decision, containing the frozen evaluation context, the policy version applied, the Consent state at evaluation time, and the resulting ALLOW or DENY outcome. Decision Artifacts are the primary evidence instrument for independent Replay Verification. Artifact generation failure must trigger DENY.

Source: QODIQA-CORE-2026-001, Section 4.3; QODIQA-WP-2026-001, Section 6.6

DecisionRequest

The explicit, machine-readable input contract submitted by the Calling System to the Enforcement Gateway. A DecisionRequest must contain all required fields: actor identity, requested action, purpose, data categories, and jurisdictional context: with no missing, inferred, or defaulted values. Unknown or missing fields must result in DENY without evaluation.

Source: QODIQA-WP-2026-001, Section 6.2; QODIQA-CORE-2026-001, QCC-01

Enforcement Gateway

The system component that intercepts all AI-bound requests on the critical path and implements the six-stage enforcement lifecycle. The Enforcement Gateway must be positioned on the critical path between the Calling System and the AI System; any architectural configuration that permits bypass of the Enforcement Gateway constitutes non-Conformance.

Source: QODIQA-CORE-2026-001, Section 4.2

Fail-Closed

The required failure mode of a Conformant QODIQA implementation. A system fails closed when it issues DENY on any missing, malformed, unavailable, or unresolvable input, rather than defaulting to ALLOW or attempting partial evaluation. No partial execution occurs on any DENY path. Fail-closed behavior is mandatory and cannot be disabled or bypassed.

Source: QODIQA-WP-2026-001, Section 4.6; QODIQA-CORE-2026-001, Section 4.3

Frozen Evaluation Context

The immutable snapshot of all inputs and state present at enforcement decision time, including the DecisionRequest, the resolved Consent state, the policy version, and the evaluation timestamp. The Frozen Evaluation Context is the basis for Decision Artifact generation and Replay Verification. No element of the Frozen Evaluation Context may be modified after the enforcement decision is made.

Source: QODIQA-CORE-2026-001, Section 6.1; QODIQA-WP-2026-001, Section 6.5

Intent Declaration

The explicit, machine-readable specification of who is acting, for what purpose, on what data, and under which constraints, provided as a mandatory component of every DecisionRequest. Intent must be declared, not inferred. The enforcement system must refuse to evaluate any request for which intent has not been explicitly declared.

Source: QODIQA-WP-2026-001, Section 5.2; QODIQA-CORE-2026-001, QCC-01

Policy Engine
The system component that executes deterministic policy evaluation as a pure function over a Frozen Evaluation Context. The Policy Engine must be free of side effects and must not make external calls during evaluation. Given identical inputs and the same policy version, the Policy Engine must always produce identical outputs.
Source: QODIQA-CORE-2026-001, Section 4.2

Purpose Binding
The enforcement requirement that a Consent grant be evaluated for its validity with respect to the specific declared purpose of an AI action. An action performed for a purpose not explicitly covered by the applicable Consent Token must be denied. Inference, approximation, or generalization of purpose is prohibited.
Source: QODIQA-CORE-2026-001, QCC-03

Replay Verification
The capability to reproduce a prior enforcement decision from its Frozen Evaluation Context and confirm that the same decision would be reached under the same inputs and policy version. Replay Verification is the primary mechanism for independent audit of QODIQA enforcement behavior. All Conformant implementations must produce Decision Artifacts sufficient to support Replay Verification.
Source: QODIQA-CORE-2026-001, Section 6.3; QODIQA-WP-2026-001, Section 7.3

Revocation
The withdrawal of a previously granted Consent, effective immediately upon issuance of the Revocation event. Revocation must propagate to all enforcement nodes within the required propagation window. Any Revocation event cancels all prior grants and supersedes any cached state or prior ALLOW decision. Revocation cannot be deferred, buffered, or ignored by any component of a Conformant implementation.
Source: QODIQA-CORE-2026-001, QCC-06, R-01

Runtime
The moment of AI system invocation, the execution boundary at which a model or inference system is called to perform an action. Runtime is the only boundary where purpose, actor identity, jurisdiction, and Revocation state are simultaneously present, evaluable, and current. QODIQA enforcement operates exclusively at this boundary.
Source: QODIQA-CORE-2026-001, Section 1.3; QODIQA-WP-2026-001, Section 1

Scope Binding
The enforcement requirement that a Consent grant be evaluated for its validity with respect to the specific data categories and action scope of a request. Where multiple applicable Consent grants exist, the most restrictive scope governs (Conflict Resolution Rule R-02). Inference or assumption of broader scope is prohibited.
Source: QODIQA-CORE-2026-001, QCC-04, R-02

This glossary contains the twenty most frequently referenced terms across the corpus. Terms used exclusively within a single document are defined in that document and are not reproduced here. For a complete term inventory of any individual document, refer to the Terminology and Normative Definitions section within that document, or to QODIQA-TERM-2026-001.

SECTION 7

Corpus Amendment History

This section records the amendment history of the Master Index itself. For the amendment history of the full corpus, refer to QODIQA-CCL-2026-001 (Corpus Change Log and Version History).

TABLE 3. MASTER INDEX REVISION HISTORY

VERSION	DATE	CHANGE DESCRIPTION	AUTHOR
1.0	2026	Initial issue. Full document index across five tiers, four reading paths, cross-reference map, and consolidated glossary for corpus Version 1.0.	QODIQA

DOCUMENT STATUS

Document Status

This document is issued as a companion navigational instrument to the QODIQA standard corpus at Version 1.0. It is classified as informative. It introduces no normative requirements and does not modify any existing document in the corpus. In the event of conflict between any statement in this document and any normative statement in a Tier I through Tier IV document, the normative statement governs without exception.

This document is intended for the following audiences:

- Engineers and architects beginning implementation of a QODIQA-conformant system
- Compliance and audit functions evaluating an existing or proposed implementation
- Regulatory, legal, and policy review bodies assessing the QODIQA corpus
- Executives and strategic decision-makers evaluating QODIQA as an institutional investment
- Any reader approaching the QODIQA corpus for the first time

This document should be updated whenever a new document is added to the corpus, an existing document is substantially revised, or a new defined term is introduced. Updates are subject to the amendment process defined in QODIQA-GC-2026-001.

INSTITUTIONAL CLOSING STATEMENT

The QODIQA corpus addresses a structural gap in artificial intelligence governance: the absence of deterministic enforcement of consent at the moment of model invocation. This Master Index is issued in the recognition that a corpus, however rigorous, is only as accessible as its navigational instruments allow. A standard that cannot be found cannot be adopted. A term that cannot be located cannot be consistently applied. A reading path that must be inferred imposes a cost that reduces the probability of correct implementation. This document exists to eliminate that cost.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dițescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-IDX-2026-001
TITLE	Master Index: Reader's Guide
SUBTITLE	Navigation Reference and Consolidated Glossary for the QODIQA Standard Corpus
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Informative: Navigation Instrument
DOCUMENT STATUS	Informative, Companion Document to QODIQA Corpus v1.0
GOVERNING AUTHORITY	QODIQA Governance Charter (QODIQA-GC-2026-001)
NORMATIVE FORCE	None. This document introduces no requirements and does not modify any normative instrument in the corpus.

CHAPTER 02 PART I · ORIENTATION AND FOUNDATIONS

Consent as Infrastructure for Artificial Intelligence

Conceptual and technical framework for machine-readable, deterministically enforceable consent in artificial intelligence systems.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-WP-2026-001	1.0	2026	Technical Whitepaper	Normative	NORMATIVE

ABSTRACT

Artificial intelligence systems increasingly operate as autonomous or semi-autonomous decision-making entities within economic, institutional, and social environments. While significant effort has been directed toward AI governance, regulation, and model oversight, the question of how consent is explicitly represented, validated, and enforced at runtime remains largely unresolved. In most contemporary systems, consent is implicit, static, or externalized to legal and policy frameworks that are not interpretable by machines and cannot be enforced during execution. This paper introduces QODIQA, a consent layer designed to formalize consent as a first-class, machine-readable primitive within AI-native systems. The proposed framework defines consent in terms of intent, scope, purpose, duration, revocability, and traceability, enabling AI systems to evaluate authorization conditions prior to executing actions. By embedding consent directly into system architecture rather than treating it as a post hoc or compliance-driven construct, QODIQA enables real-time enforcement, auditable decision flows, and verifiable accountability. The paper outlines the conceptual foundations, system architecture, and integration model of the QODIQA framework, and examines its implications for AI governance, risk reduction, and human agency in increasingly autonomous computational environments.

SECTION 1

Executive Summary

Artificial intelligence systems have moved beyond experimentation and are now embedded in the core execution paths of modern organizations. They generate outputs, trigger actions, and influence decisions with legal, financial, and societal consequences. While model capabilities have advanced rapidly, the mechanisms used to authorize, constrain, and justify AI behavior have remained largely static.

In most environments, consent and authorization exist outside the system that executes decisions: captured in policies, encoded inconsistently across services, or enforced only after the fact. This mismatch has created a growing structural risk.

Today's AI governance efforts focus primarily on documentation, compliance processes, and retrospective accountability. These approaches are necessary but insufficient. Risk does not materialize in policy documents or audit reports; it materializes at the moment an AI system acts. Yet most organizations cannot deterministically demonstrate that a specific AI action was authorized for a specific purpose, using specific data, under a valid consent state, at the exact time the action occurred. This inability is not a tooling problem. It is an infrastructure gap.

QODIQA addresses this gap by introducing an AI consent and decision enforcement layer that operates before execution. Rather than interpreting law, inferring intent, or applying probabilistic judgment, QODIQA formalizes consent as a machine-readable, enforceable primitive. Every AI action is evaluated at runtime against explicit intent, resolved consent state, and an immutable policy context. The result is a deterministic outcome: allow, deny, or allow with explicit restrictions, which downstream systems can rely on without ambiguity.

The defining property of QODIQA is determinism. Decisions are treated as artifacts, not transient responses. Given the same declared request, the same policy version, and the same evaluation rules, the system will always produce the same

decision. This behavior is enforced through closed request contracts, immutable and versioned policies, frozen evaluation contexts, and fail-closed semantics. If required inputs are missing, inconsistent, or unavailable, the system refuses to decide. This deliberate rigidity removes entire classes of governance failures caused by defaults, inference, and best-effort behavior.

By separating consent resolution from policy evaluation, QODIQA preserves human responsibility while ensuring mechanical enforcement. Consent is resolved explicitly as state: scoped, purpose-bound, time-limited, and revocable, before any policy logic is applied. Policies do not reason about consent; they declare the conditions they require.

This separation simplifies reasoning, prevents silent drift, and ensures that changes to consent or policy cannot retroactively alter past decisions. A critical consequence of this architecture is verifiability. Every decision can be replayed, independently validated, and traced back to a stable, immutable set of inputs. Audits move beyond interpretive narratives and become reproducible checks against deterministic decision artifacts. Accountability shifts from explanation to proof.

QODIQA is intentionally minimal. It is not a user interface, a compliance dashboard, or a model-level governance tool. It does not inspect raw data, perform semantic analysis, or assess risk probabilistically. Its scope ends at enforcement. More advanced reasoning, analytics, and reporting can exist downstream, but the core layer remains focused on a single responsibility: ensuring that AI execution is authorized, reproducible, and defensible by design.

As AI systems continue to scale in autonomy and reach, governance mechanisms that rely on documentation, manual review, or post hoc mitigation will become increasingly inadequate. Embedding consent enforcement directly into the execution path represents a structural shift: from policy-centric oversight to runtime control. By making authorization a prerequisite for execution, QODIQA transforms AI governance from an external obligation into an intrinsic property of the system itself.

SECTION 2

Introduction

Artificial intelligence has transitioned from a specialized capability to a general-purpose infrastructure embedded across modern software systems. AI models now operate inside customer support platforms, financial systems, healthcare workflows, internal decision tools, and public-sector services. In these environments, AI is no longer experimental or advisory; it executes actions, transforms data, and produces outcomes with real-world consequences.

This shift has exposed a fundamental asymmetry. While AI capabilities have become increasingly powerful and accessible, the mechanisms used to control their behavior remain fragmented, informal, and largely external to execution. Authorization is often implied rather than enforced. Consent is captured once and assumed to persist indefinitely. Policies are written for humans, not machines, and enforcement is deferred to manual review or post hoc remediation.

Much of the current discourse around AI governance focuses on regulation, ethical frameworks, and organizational processes. These efforts are necessary, but they operate at a different layer of abstraction. Regulation defines obligations, policies describe intent, and governance frameworks assign responsibility. None of these, by themselves, ensure that an AI system will act only within authorized boundaries at runtime. The gap between declared rules and executed behavior is where risk accumulates.

In practice, this gap manifests in predictable ways. AI features are delayed or constrained due to uncertainty around data usage. Engineering teams reimplement governance logic inconsistently across services. Legal and compliance functions rely on documentation rather than verifiable guarantees. When incidents occur, organizations struggle to reconstruct what happened, which rules applied, and whether consent was valid at the moment of execution. These failures are not primarily the result of poor intent, but of missing infrastructure.

This paper approaches AI governance from a system design perspective. Rather than asking how policies should be written or how models should behave, it asks how consent, intent, and authorization can be represented and enforced in a form that machines can reliably execute. Addressing this requires moving governance mechanisms closer to the execution path, where decisions are made and risk is realized.

The remainder of this paper examines the structural limitations of existing AI governance approaches, defines the properties required for enforceable consent in AI-native systems, and introduces a framework for embedding consent enforcement directly into runtime decision flows.

SECTION 3

The Consent Gap in Artificial Intelligence

Artificial intelligence systems increasingly operate as autonomous or semi-autonomous actors within complex organizational environments. They request access to data, perform transformations, generate outputs, and trigger downstream actions.

Yet the mechanisms that govern whether these actions are legitimately authorized remain largely implicit. Consent is often assumed, inherited, or approximated, rather than explicitly evaluated at the moment an AI action occurs. This creates a structural gap between intention and execution.

In most AI-enabled systems, authorization is treated as a static prerequisite rather than a dynamic condition. Access permissions are granted in advance, consent is captured once, and policies are assumed to apply uniformly across time, context, and use cases.

However, AI execution is inherently contextual. The same data, processed by the same model, can be legitimate for one purpose and invalid for another. When consent is not evaluated in relation to who is acting, why the action is performed,

what data is involved, and when the action occurs, control degrades into assumption.

This gap widens as AI systems scale. Large organizations operate multiple teams, services, models, and data sources, each evolving independently. Governance rules may exist, but their enforcement is fragmented across application code, middleware, and manual review processes.

Engineers are forced to reimplement authorization logic repeatedly. Legal and compliance teams rely on documentation and process controls that cannot be enforced programmatically. Over time, the system's actual behavior diverges from its declared constraints.

A common response to this problem is increased oversight: additional policies, review gates, approval workflows, or logging requirements. While these measures may improve visibility, they do not address the core issue. Oversight mechanisms observe behavior after execution; they do not constrain it at runtime. Logging explains what happened, but it does not prevent unauthorized actions. Reviews assign responsibility, but they do not provide machine-verifiable guarantees.

The limitations of implicit consent become most visible during failure. When an incident occurs, organizations attempt to reconstruct the chain of decisions that led to an outcome. They ask whether consent was valid, which policy applied, and whether the action should have been allowed. In many cases, these questions cannot be answered deterministically.

Policies may have changed, consent may have expired, or enforcement logic may have differed across services. The system cannot prove what rules were in effect at the moment of execution, only what rules exist now.

This inability to prove authorization is not an edge case; it is a systemic property of current AI architectures. Consent is treated as metadata rather than as an executable constraint. Authorization is enforced indirectly, if at all. As AI systems gain autonomy and operate at higher velocity, these assumptions become increasingly fragile.

The consent gap, therefore, is not a failure of intent or regulation. It is a failure of system design. Without a mechanism to evaluate consent explicitly and enforce it deterministically at runtime, AI systems will continue to act beyond the boundaries that organizations believe they have defined.

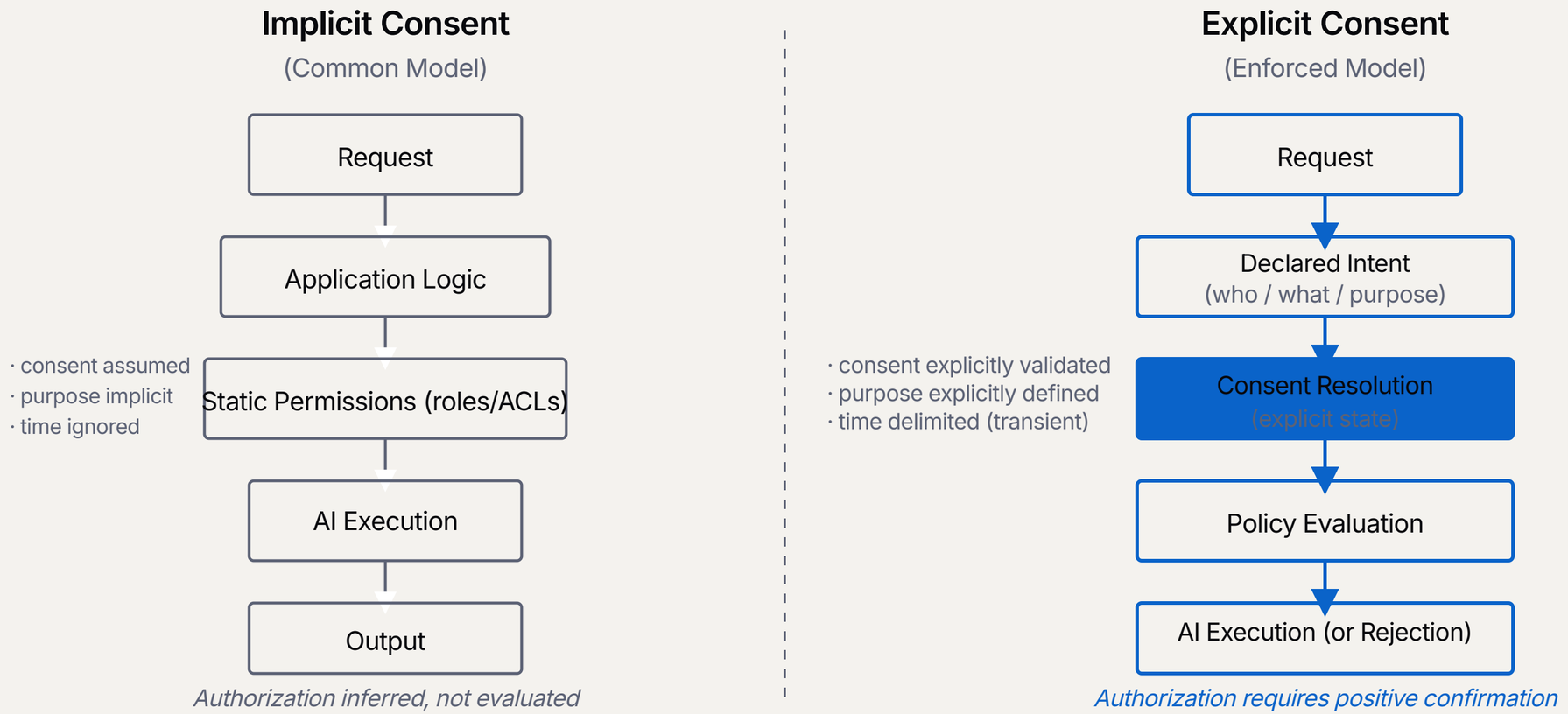


FIGURE 1. COMPARISON OF IMPLICIT CONSENT (COMMON MODEL) AND EXPLICIT CONSENT (ENFORCED MODEL) IN AI EXECUTION FLOWS.

1 Implicit consent refers to authorization derived from inherited permissions, prior access grants, or assumed legitimacy, without explicit validation at the moment of execution.

2 Declared intent denotes an explicit, machine-readable specification of who is acting, for what purpose, and under which constraints, provided as part of the execution request.

3 Consent resolution represents the evaluation of consent as explicit state: scoped, purpose-bound, time-limited, and revocable, prior to policy evaluation.
4 Policy evaluation is performed against an immutable snapshot of applicable rules, ensuring that authorization decisions are deterministic and reproducible.

3.1 Decision-Making Without Explicit Mandate

In many contemporary AI systems, decisions are executed without an explicit, verifiable mandate tied to the moment of action. Authorization is inferred from prior access grants, inherited permissions, or generalized system roles rather than evaluated as a concrete condition of execution. Once an AI component is permitted to operate within a system, its subsequent actions are often treated as implicitly authorized, regardless of changes in context, purpose, or consent state.

This pattern is not accidental. It is a consequence of how authorization has historically been modeled in software systems. Traditional access control mechanisms were designed for static actors and predictable operations. AI systems, by contrast, act dynamically. They reuse data across purposes, chain multiple operations, and adapt behavior based on upstream inputs. When authorization is not explicitly revalidated for each action, control shifts from deliberate permission to assumed legitimacy.

The absence of an explicit mandate creates ambiguity at multiple levels. From an engineering perspective, it becomes unclear which conditions must hold true for an action to be allowed. From a governance perspective, responsibility diffuses across teams and systems, making it difficult to determine who authorized what and under which constraints. From a legal and compliance perspective, the system cannot demonstrate that consent was valid for the specific action performed, only that some form of access existed at some point in time.

In practice, this leads to fragile control boundaries. AI systems may operate correctly under expected conditions, yet silently exceed their intended scope when inputs change or new use cases emerge. Because authorization is not evaluated as part of the execution path, violations are often discovered only after the fact, through audits, incident reviews, or external scrutiny. At that point, remediation is limited to explanation rather than prevention.

Attempts to compensate for this gap typically involve adding more process around AI usage: approval workflows, documentation requirements, or manual reviews. While these measures may reduce risk in isolated cases, they do not scale with system complexity or execution speed. As AI systems operate continuously and autonomously, control mechanisms that rely on human intervention or retrospective analysis become increasingly ineffective.

Decision-making without an explicit mandate is therefore not merely a compliance issue; it is a structural flaw. Without a mechanism that binds authorization directly to execution, AI systems are left to operate on assumptions rather than guarantees. As autonomy increases, these assumptions become harder to justify and impossible to prove.

3.2 Implicit Consent and Its Failure Modes

Implicit consent arises when authorization is assumed rather than evaluated. In AI-enabled systems, this typically occurs when access to data or system capabilities is granted once and then reused across multiple actions, purposes, or timeframes without revalidation. The system treats prior approval as a standing mandate, even as execution contexts change.

This approach may appear efficient, but it introduces several failure modes that become increasingly severe as AI systems scale. The first failure mode is **purpose drift**. Data collected or authorized for a specific use, such as customer support or internal analytics, is later reused for a different purpose, such as model training or behavioral inference. Because consent is not evaluated at the level of individual actions, the system lacks a mechanism to distinguish legitimate reuse from unauthorized expansion.

A second failure mode is **temporal decay**. Consent is rarely perpetual. It may expire, be revoked, or become invalid due to changes in regulation, contractual terms, or user preferences. In systems built on implicit consent, there is no reliable way to ensure that execution reflects the current validity of consent. AI actions continue to operate on outdated assumptions, and violations are detected only retrospectively, if at all.

A third failure mode emerges from **context collapse**. AI systems often serve multiple actors, applications, and data domains simultaneously. When consent is implicit, the system cannot reliably differentiate between actions initiated by different entities or for different objectives. Authorization becomes coarse-grained, and nuanced constraints, such as jurisdictional boundaries or data category restrictions, are flattened into generalized permissions.

These failure modes are compounded by system complexity. As organizations introduce additional models, services, and integrations, implicit consent logic fragments across codebases and teams. Each implementation encodes slightly different assumptions about what is allowed. Over time, this leads to inconsistent enforcement, unintended behavior, and an erosion of trust in governance controls.

Importantly, these failures are difficult to observe in real time. Logging and monitoring may capture that an action occurred, but not whether it should have been allowed under the precise conditions present at execution. When incidents are investigated, teams must reconstruct intent and authorization from partial evidence, often relying on interpretation rather than proof.

Implicit consent is therefore not merely an implementation shortcut; it is a structural liability. It substitutes assumption for verification and convenience for control. As AI systems become more autonomous and interconnected, this liability grows in proportion to their impact.

3.3 The Difference Between Authorization and Consent

In discussions about AI governance, the terms *authorization* and *consent* are often used interchangeably. In practice, they represent distinct concepts that operate at different levels of a system. Conflating them leads to ambiguous enforcement, inconsistent controls, and an overreliance on assumptions rather than guarantees.

Authorization refers to the technical permission granted to an actor or system to perform a class of actions. It is typically expressed through roles, access control lists, API keys, or service identities. Authorization answers the question: is this

actor allowed to perform this type of operation within the system? It is generally static, coarse-grained, and optimized for system access rather than contextual validity.

Consent, by contrast, is contextual and conditional. It answers a different question: is this specific action legitimate under the current purpose, scope, time, and constraints? Consent is not a standing permission. It is a state that can be granted, limited, revoked, or expired, and its validity depends on the circumstances of execution. In AI systems, consent must be evaluated in relation to what data is being used, for what purpose, and at the exact moment an action occurs.

Many AI systems rely on authorization as a proxy for consent. Once an application or service is authorized to access data or invoke a model, subsequent actions are assumed to be legitimate. This assumption breaks down as soon as context changes. The same authorization may cover multiple purposes, jurisdictions, or data categories, while consent may apply only to a narrow subset. When authorization is treated as sufficient, consent effectively disappears from the execution path.

This distinction becomes critical in AI-driven workflows because execution is dynamic. AI systems reuse data, chain operations, and adapt behavior across contexts. Authorization alone cannot encode purpose, temporal validity, or revocation. Consent, if not explicitly evaluated, degrades into an implicit assumption inherited from earlier states. At scale, this creates systems that are technically authorized but substantively unauthorized.

Separating authorization from consent is therefore not a semantic exercise; it is a structural requirement. Authorization establishes who may act within a system. Consent determines whether a specific action is allowed under current conditions. Treating these as interchangeable obscures responsibility and makes enforcement dependent on interpretation rather than execution. For AI governance to be enforceable, consent must become an explicit input to decision-making, evaluated independently of authorization and bound to execution. Without this separation, organizations cannot reliably demonstrate that AI actions were legitimate at the moment they occurred, only that access existed in general.

SECTION 4

Requirements for Enforceable AI Consent

For consent to be enforceable in AI-driven systems, it must satisfy a set of properties that go beyond policy definition or procedural compliance. These properties are not aspirational; they are technical requirements derived from the realities of distributed systems, autonomous execution, and regulatory accountability. If any of these requirements is missing, consent degrades from an enforceable constraint into an assumption. The following requirements define what enforceable AI consent must look like at the system level.

4.1 Explicitness

> **REQUIREMENT: EXPLICITNESS**
>
> Consent must be explicit. All relevant parameters: who is acting, what action is requested, for what purpose, under which constraints, must be declared in a machine-readable form as part of the execution request.

Implicit assumptions, inferred intent, or default values undermine enforceability by shifting responsibility from humans to the system. If intent is not explicitly declared, the system cannot reliably determine whether an action is legitimate. In enforceable systems, ambiguity is not resolved heuristically; it results in refusal to execute.

4.2 Contextual Binding

> **REQUIREMENT: CONTEXTUAL BINDING**
>
> Consent must be bound to context. Authorization decisions must account for purpose, data category, jurisdiction, and temporal validity. A consent grant that is valid in one context may be invalid in another, even if the same actor and data are involved. Without contextual binding, consent becomes overly permissive and loses its meaning as a constraint. Enforceable consent is not global; it is conditional.

4.3 Time-Bounded Validity

REQUIREMENT: TIME-BOUNDED VALIDITY

Consent must be time-aware. It must be possible to determine whether consent is valid at the exact moment an AI action occurs. Consent may expire, be revoked, or become invalid due to external changes. Systems that treat consent as perpetual are unable to enforce revocation and cannot demonstrate temporal correctness. Time-bounded validity ensures that execution reflects current authorization, not historical assumptions.

4.4 Deterministic Evaluation

REQUIREMENT: DETERMINISTIC EVALUATION

Consent evaluation must be deterministic. Given the same declared intent, consent state, and applicable rules, the system must always reach the same decision. Probabilistic or heuristic approaches are incompatible with enforceability, as they prevent independent verification and reproducibility. Determinism is a prerequisite for accountability. If decisions cannot be reproduced, they cannot be defended.

4.5 Separation of Responsibility and Enforcement

REQUIREMENT: SEPARATION OF RESPONSIBILITY

Human responsibility and machine enforcement must be clearly separated. Humans define policies, consent scopes, and rules. Systems enforce them mechanically. When systems are allowed to infer, reinterpret, or optimize consent, responsibility becomes ambiguous and accountability weakens. Enforceable consent requires that systems execute rules exactly as defined, without interpretation.

4.6 Fail-Closed Semantics

REQUIREMENT: FAIL-CLOSED SEMANTICS

Enforceable systems must fail closed. When required inputs are missing, inconsistent, or unavailable, execution must be denied. Silent fallbacks, retries, or best-effort behavior introduce undefined states that undermine trust and control. Refusal to act is safer than acting on incomplete authorization.

4.7 Auditability and Verifiability

> **REQUIREMENT: AUDITABILITY AND VERIFIABILITY**
>
> Consent enforcement must produce verifiable artifacts. It must be possible to demonstrate, after the fact, that a specific decision was made under specific conditions, using specific rules. Logs alone are insufficient if they cannot be tied to a reproducible evaluation context. Auditability is not an afterthought; it is a system property.

TABLE 1. PROPERTIES OF ENFORCEABLE AI CONSENT

REQUIREMENT	IMPLICIT CONSENT SYSTEMS	ENFORCEABLE CONSENT SYSTEMS
Explicit intent	Assumed or inferred	Declared per action
Context awareness	Limited or absent	Purpose- and scope-bound
Time-bounded validity	Rarely enforced	Enforced at execution time
Deterministic evaluation	No	Yes
Fail-closed behavior	Inconsistent	Mandatory
Revocability	Often complex or none	Granular and immediate

1 The requirements listed are cumulative and interdependent. Enforceable consent emerges only when all properties are satisfied at runtime; partial implementation results in implicit or assumption-based authorization.

SECTION 5

The QODIQA Framework

The requirements outlined in the previous section describe what enforceable AI consent must achieve at runtime.
Meeting these requirements cannot be accomplished through policy documents, application-level conventions, or post

hoc controls. It requires a dedicated system component designed to operate directly in the execution path of AI-driven actions.

QODIQA is introduced as such a component: a consent and decision enforcement layer that evaluates whether an AI action may proceed before execution occurs. Its role is not to replace governance, compliance, or legal interpretation, but to provide the missing execution-layer mechanism that allows those constructs to be applied deterministically and at scale. At a high level, the framework formalizes a simple but strict contract: no AI action is executed unless it is explicitly authorized under the current consent state and applicable rules. Authorization is not inferred from access, roles, or historical approvals. It is evaluated at runtime, for each action, using explicit inputs and immutable context.

5.1 Architectural Positioning

QODIQA operates as a thin middleware layer positioned between an application and the AI systems it invokes. All AI-bound requests pass through this layer prior to execution. The framework does not observe or analyze AI behavior after the fact; it acts as a gate that determines whether execution is permitted in the first place.

This positioning is intentional. By residing on the critical path, the framework ensures that enforcement cannot be bypassed without explicit architectural decisions. At the same time, it remains independent of specific AI models, vendors, or deployment environments. The layer is model-agnostic and does not depend on the internal workings of the AI systems it controls.

5.2 Input Contract

Every evaluation performed by the framework is driven by an explicit, machine-readable request. This request describes intent, not content. It specifies who is acting, what action is requested, for what purpose, under which constraints, and within which jurisdictional context. The contract is deliberately closed. All required fields must be present and valid.

Unknown or missing fields result in refusal to evaluate. This design choice eliminates ambiguity and prevents the system from compensating for incomplete or unclear intent through inference or defaults. By enforcing explicitness at the boundary, the framework ensures that responsibility for intent declaration remains human, while execution remains mechanical.

5.3 Consent Resolution

Within the framework, consent is treated as explicit state rather than embedded logic. Consent resolution occurs prior to policy evaluation and determines whether valid consent exists for the declared purpose, scope, and time of execution. This separation is critical. Policies do not reason about consent; they declare what consent conditions they require. Consent resolution answers whether those conditions are currently satisfied.

By decoupling these concerns, the framework avoids entangling policy logic with mutable consent state and maintains deterministic behavior. Consent may be revoked, expired, or constrained independently of policy changes. Once resolved, the resulting consent state becomes part of the immutable evaluation context for the decision.

5.4 Policy Evaluation

Policy evaluation is performed against an immutable snapshot of applicable rules. Policies are versioned, deterministic, and free of side effects. Given the same inputs and policy version, evaluation always produces the same outcome.

This immutability ensures that decisions remain reproducible even as policies evolve over time. Past decisions cannot be altered retroactively by policy changes, and future decisions are evaluated against explicitly defined rule sets. Policy evaluation does not inspect data content, apply probabilistic reasoning, or attempt to infer intent. Its sole function is to evaluate declared intent against resolved consent and defined constraints.

5.5 Decision Outcomes

Each evaluation produces one of a small, finite set of outcomes: execution allowed, execution denied, or execution allowed with explicit restrictions. Restrictions may include constraints on retention, logging, reuse, or downstream processing, but they are always declared explicitly as part of the decision.

Decisions are treated as artifacts. They are stable, identifiable, and traceable to the exact inputs and rules that produced them. This enables independent verification, replay, and audit without reliance on interpretation or narrative reconstruction.

5.6 Failure Semantics

The framework is designed to fail closed. If required inputs are missing, consent state cannot be resolved, or policy context is unavailable, execution is denied. There are no retries, silent fallbacks, or best-effort decisions. This behavior reflects a deliberate prioritization of correctness and predictability over convenience. In systems where AI actions have real-world consequences, refusal to act under uncertainty is safer than proceeding on assumption.

5.7 Scope and Boundaries

The QODIQA framework is intentionally minimal. Its responsibility ends at decision enforcement. It does not provide user interfaces, reporting dashboards, semantic analysis, or risk scoring. Such capabilities may exist downstream, but they are not part of the core framework.

By maintaining a narrow scope, the framework remains composable, auditable, and resilient to changes in AI models, vendors, and deployment patterns. It is designed to function as infrastructure: present, reliable, and largely invisible until it is missing.

1 All AI-bound actions pass through a single, deterministic execution path. Execution occurs only after explicit consent and policy evaluation.

SECTION 6

QODIQA System Architecture

This section describes the internal architecture of QODIQA as an execution-layer system. The goal is not to present an implementation tied to a specific stack, but to make the design concrete, inspectable, and reproducible. Every component described here exists to satisfy one or more of the enforceability requirements defined in Section 4. At its core, QODIQA is not a service that decides wisely, but a system that decides predictably.

6.1 Architectural Overview

QODIQA is composed of four core components, executed in a strict, linear sequence: Intent Interface, Consent Resolver, Policy Engine, and Decision Artifact Generator. Each component is deliberately simple in isolation. The innovation lies in their composition, ordering, and constraints.

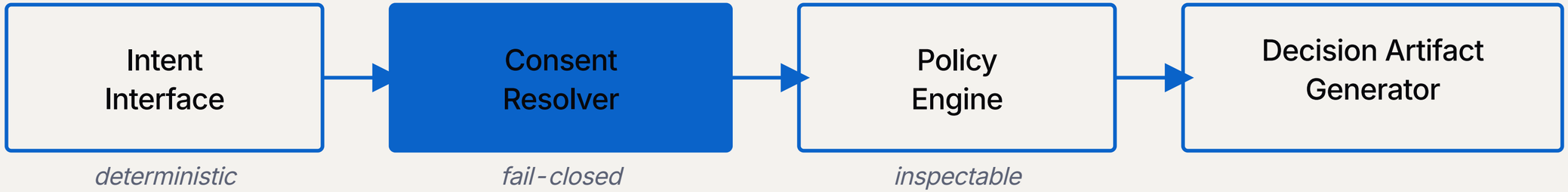


FIGURE 2. QODIQA CORE COMPONENT PIPELINE: STRICT LINEAR EXECUTION ENFORCING DETERMINISM AT EACH STAGE.

6.2 Intent Declaration: Closed by Design

Every interaction with QODIQA begins with an explicit intent declaration. This is not metadata. It is a hard contract.

EXAMPLE

```
class DecisionRequest:
    actor_id: str
    action: str
    purpose: str
    data_types: set[str]
    jurisdiction: str
```

VALIDATION LOGIC (SIMPLIFIED)

```
REQUIRED_FIELDS = {
    "actor_id",
    "action",
    "purpose",
    "data_types",
    "jurisdiction"
}

def validate_request(req: dict):
    missing = REQUIRED_FIELDS - req.keys()
    extra = set(req.keys()) - REQUIRED_FIELDS

    if missing:
        raise InvalidRequest(f"Missing fields: {missing}")
    if extra:
        raise InvalidRequest(f"Unknown fields: {extra}")
```

> DESIGN DECISION
>
> - No defaults
> - No inferred intent
> - No partial evaluation
>
> If intent is ambiguous, execution is refused. This single constraint removes an entire class of governance bugs.

1 Code snippets are illustrative and represent architectural intent rather than a specific implementation.

6.3 Consent Resolution as Explicit State

Consent in QODIQA is not embedded in policy logic. It is resolved before policy evaluation and treated as state.

CONCEPTUAL MODEL

```
class ConsentState:
    subject_id: str
    purpose: str
    valid_from: datetime
    valid_until: datetime
    scopes: set[str]
    revoked: bool
```

RESOLUTION STEP

```
def resolve_consent(request: DecisionRequest) → ConsentState:
    consent = consent_store.lookup(
        subject=request.actor_id,
        purpose=request.purpose
    )
```

```
    if not consent or consent.revoked:
        raise ConsentInvalid
    if not consent.is_valid_now():
        raise ConsentExpired
    return consent
```

Important: Once resolved, the consent reference becomes immutable input to the decision. Later changes do not retroactively affect the outcome.

6.4 Policy Evaluation as a Pure Function

Policies in QODIQA are: **versioned | immutable | deterministic | side-effect free**

POLICY EVALUATION SIGNATURE

```
def evaluate_policy(
    policy_version: str,
    request: DecisionRequest,
    consent: ConsentState
) → Decision:
    ...
```

EXAMPLE RULE (ILLUSTRATIVE)

```
if request.purpose == "training" and "personal_data" in request.data_types:
    return DENY

if request.jurisdiction == "EU" and consent.scopes ≠ {"inference"}:
    return DENY

return ALLOW_WITH_RESTRICTIONS(
    store_input=False,
    store_output=False,
```

```
    retention_seconds=0
)
```

> CONSTRAINTS
>
> There is no access to: system time (beyond frozen context); external services; mutable configuration.
>
> Same inputs produce the same outputs. Always.

6.5 Frozen Evaluation Context

At evaluation time, QODIQA constructs a frozen context that captures everything required to reproduce the decision.

CONTEXT CONSTRUCTION

```
EvaluationContext = {
    "request": canonicalize(request),
    "consent_ref": consent.id,
    "policy_version": policy.version,
    "ruleset_version": RULESET_VERSION
}

# Once frozen:
decision = evaluate(EvaluationContext)
assert decision == replay(EvaluationContext)
```

This removes temporal ambiguity, configuration drift, and environment dependence. Decisions become mathematical objects, not events.

6.6 Decision Artifacts and Identity

Every decision produces a stable, **deterministic identifier**.

```
def decision_id(context, outcome):
    payload = {
        "context": context,
        "outcome": outcome
    }
    raw = json.dumps(
        payload,
        sort_keys=True,
        separators=(",",":")
    ).encode()
    return hashlib.sha256(raw).hexdigest()[:24]
```

PROPERTIES

- Identical input produces identical ID
- No raw data stored
- Independently recomputable
- Suitable for audits and verification

Decisions stop being responses. They become artifacts.

6.7 Failure Semantics

QODIQA fails closed, always.

```
try:
    decision = evaluate(context)
except (
    InvalidRequest,
    ConsentInvalid,
```

```
    PolicyUnavailable
):
    decision = DENY
```

INVARIANTS

- No retries
- No best-effort evaluation
- No silent degradation

Predictability is prioritized over availability.

6.8 Why This Architecture Is Novel

Individually, none of these components are revolutionary. The innovation lies in the constraints:

- Consent as state, not logic
- Intent as a closed contract
- Policy as a pure function
- Decisions as immutable artifacts
- Execution gated before AI runs

Most systems optimize for flexibility. QODIQA optimizes for defensibility.

This section intentionally mixes conceptual architecture, executable logic, and enforceable constraints. This is deliberate. At this layer, the architecture is the product.

SECTION 7

Determinism, Auditability, and Proof

The architectural constraints described in the previous section are not stylistic choices. They exist to guarantee three properties that most AI systems fail to achieve simultaneously: determinism, auditability, and proof. Together, these properties shift AI governance from interpretation to verification, and from policy narratives to enforceable system behavior. This section explains how these properties emerge from the architecture itself and why they are essential for AI systems operating at scale.

7.1 Determinism as a First-Class Property

In many AI-enabled systems, decisions are inherently temporal. They depend on mutable configuration, dynamic policy resolution, external services, or implicit context. As a result, the same logical request may yield different outcomes depending on when, where, or how it is evaluated. This variability makes governance fragile and undermines accountability.

QODIQA treats determinism as a first-class system property. Every decision is defined as the output of a pure evaluation over an immutable set of inputs. Once intent, consent state, policy version, and ruleset are resolved, no additional factors may influence the outcome.

Determinism is enforced through strict architectural constraints: closed and explicit request contracts, immutable versioned policies, frozen evaluation contexts, absence of side effects, and absence of probabilistic or heuristic logic. Given the same inputs, the system will always produce the same decision. If this guarantee cannot be upheld, the system is considered incorrect.

7.2 Decisions as Stable Artifacts

In conventional systems, decisions are transient responses that are logged and forgotten. In QODIQA, decisions are treated as stable artifacts with well-defined identity. Each decision is derived from a canonical representation of its evaluation context, including: the declared intent (structure, not content), a reference to resolved consent state, the exact policy version, and the evaluation ruleset version.

Conceptually, a decision is defined as: `Decision = f(context)`, where *f* is a deterministic function and context is immutable.

This framing has a critical consequence: two independent systems evaluating the same context will produce the same outcome and the same decision identity, without coordination or shared state. Decisions cease to be events and become reproducible objects.

7.3 Replay as a Correctness Invariant

Replay is the strongest test of determinism. A decision that cannot be replayed cannot be proven. QODIQA is designed so that any past decision can be recomputed by re-evaluating its frozen context. Replay does not require access to production systems, live consent stores, or current policy state. It requires only the immutable inputs captured at evaluation time.

The system enforces the following invariant: `evaluate(context) == replay(context)`

If this invariant does not hold, the integrity of the system is compromised. Replay enables capabilities that are otherwise difficult or impossible to achieve: independent verification across environments, regression testing against historical decisions, detection of unintended behavioral drift, and audits that do not depend on trust or interpretation.

7.4 Auditability Without Interpretation

Traditional audit mechanisms rely on logs: partial records of execution interpreted after the fact. Logs can describe what happened, but they rarely prove why an action was allowed. They depend on completeness, consistency, and correct interpretation, all of which degrade as systems scale.

QODIQA replaces narrative auditability with reproducible verification. Each decision is backed by a frozen evaluation context, an immutable policy snapshot, a deterministic evaluation path, and a stable decision identifier.

An audit no longer asks, "What does the log say happened?" It asks, "Does this decision replay to the same result?" If the inputs match and the outcome is reproduced, the decision is valid. If not, the discrepancy is objective and detectable.

7.5 Proof Over Explanation

Many AI governance efforts emphasize explainability: generating human-readable justifications for decisions. While explanations can be useful, they are inherently subjective and difficult to validate independently.

QODIQA prioritizes proof over explanation. Proof does not persuade; it verifies. It does not rely on narratives, trust, or interpretation. It relies on reproducibility.

By enabling decisions to be replayed and independently verified, the system provides a stronger guarantee than explanation alone. It allows organizations to demonstrate that an AI action was authorized under specific conditions, using specific rules, at a specific point in time. This distinction is critical in regulatory, contractual, and legal contexts, where legitimacy must be demonstrated, not argued.

7.6 Intentional Limits

Determinism, auditability, and proof impose constraints. The system deliberately rejects flexibility that would compromise reproducibility. There is no adaptive behavior, no inference, no learning from past decisions, and no contextual

guesswork. These limitations are not temporary trade-offs. They are foundational design choices.

Intelligence exists elsewhere in the AI stack. Optimization exists elsewhere. Interpretation exists elsewhere. At this layer, authority matters more than adaptability.

This section does not introduce new components or diagrams by design. Its purpose is to formalize invariants, not flows. The next section addresses how these guarantees survive contact with real systems.

SECTION 8

Integration, Adoption, and Operational Reality

QODIQA is designed to integrate as a thin middleware layer that sits directly on the AI execution path. Integration does not require refactoring models, rewriting business logic, or adopting a specific AI vendor. Instead, it introduces a single, explicit decision point prior to AI invocation.

In practice, integration follows a consistent pattern: an application constructs an explicit intent declaration; the intent is evaluated by the enforcement layer; AI execution proceeds only if a positive decision is returned.

This pattern is compatible with centralized AI gateways, service-oriented architectures, event-driven systems, and synchronous and asynchronous execution models. Because the framework does not inspect content or model internals, it remains decoupled from the evolution of AI capabilities. Models can change, vendors can rotate, and deployment environments can shift without altering the enforcement logic.

8.2 Latency and Performance Constraints

Any system placed on the execution path must meet strict latency requirements. If enforcement introduces noticeable delays, teams will be incentivized to bypass it. QODIQA is designed with this reality as a first-order constraint. The evaluation path is single-hop, deterministic, free of synchronous external dependencies, and bounded in complexity.

Policy evaluation operates over in-memory, immutable structures. Consent resolution is explicit and resolved prior to evaluation. No network calls, probabilistic computation, or dynamic inference occur during the hot path.

As a result, enforcement latency is predictable and bounded. This predictability is more important than raw speed: teams can reason about performance impact and design accordingly.

8.3 Failure Modes and Fail-Closed Behavior

Operational systems fail. Networks degrade, dependencies become unavailable, and configuration errors occur. How a governance layer behaves under failure is as important as how it behaves under normal conditions.

QODIQA is designed to fail closed. When required inputs cannot be resolved, policies are unavailable, or evaluation context cannot be constructed, execution is denied. This behavior is intentional. Silent degradation, retries, or partial evaluation introduce undefined states that undermine trust. A denied execution is visible, traceable, and actionable. An unauthorized execution is not.

Fail-closed behavior shifts operational incentives. Instead of silently allowing risky behavior, the system forces explicit resolution of failures, making governance gaps visible rather than latent.

8.4 Adoption Across Teams and Organizations

Governance systems often fail not because they are incorrect, but because they are perceived as obstacles. QODIQA is designed to minimize friction while preserving authority.

For engineering teams, the framework provides a stable, explicit contract; deterministic behavior; and clear failure semantics. For legal and compliance teams, it provides enforceable constraints rather than guidelines; reproducible decision artifacts; and auditable proof instead of narrative logs. For organizations, this alignment reduces the need for ad hoc governance logic embedded across services. Responsibility remains human; enforcement becomes systemic.

8.5 Resistance to Bypass

Any control mechanism that can be bypassed easily will be bypassed eventually. QODIQA addresses this risk structurally.

Because enforcement occurs before AI execution and is positioned on the critical path, bypassing it requires an explicit architectural decision. Such decisions are visible, reviewable, and auditable. There is no implicit bypass through defaults, cached permissions, or inherited roles.

In effect, the framework shifts governance from convention to infrastructure. Control is no longer a matter of discipline; it is a property of the system.

8.6 Operational Transparency

QODIQA does not attempt to hide its decisions. Denials, restrictions, and failures are explicit. Each outcome produces a traceable artifact that can be surfaced to operators, developers, or auditors as needed. This transparency reduces operational ambiguity. When execution fails, the reason is deterministic and reproducible. Teams do not need to guess whether a failure is due to policy, consent, or system error.

This section addresses the practical conditions under which execution-layer governance either succeeds or fails. The design choices described here are not optimizations; they are prerequisites for adoption in real systems.

SECTION 9

Limitations, Trade-offs, and Non-Goals

Any system that aims to be enforceable at scale must make explicit trade-offs. QODIQA is no exception. Its design deliberately prioritizes determinism, verifiability, and execution-level control over flexibility, adaptiveness, or convenience. This section outlines the limitations and non-goals of the framework, not as weaknesses, but as intentional boundaries that preserve its core guarantees.

9.1 No Probabilistic or Adaptive Decision-Making

QODIQA does not employ probabilistic reasoning, machine learning, or adaptive logic in its decision process. Outcomes are not influenced by confidence scores, historical patterns, or inferred intent. This is a deliberate constraint. Probabilistic systems may optimize outcomes, but they undermine reproducibility. Adaptive behavior introduces statefulness that makes decisions difficult to replay and impossible to prove. At the enforcement layer, predictability outweighs optimization.

9.2 No Content Inspection or Semantic Understanding

The framework does not inspect raw data, analyze content semantics, or classify inputs based on meaning. It operates on declared intent, resolved consent state, and policy rules, not on the substance of the data itself. Content inspection and semantic analysis may be valuable in other parts of the AI stack, but embedding them into the consent enforcement layer would blur responsibility and reintroduce inference. QODIQA intentionally separates enforcement from interpretation.

9.3 No Legal Interpretation

QODIQA does not interpret laws, regulations, or contracts. Legal reasoning remains a human responsibility. Policies enforced by the system are assumed to be derived from legal, contractual, or organizational sources and translated into machine-readable rules elsewhere. This separation is essential. A system that claims to understand the law would implicitly assume responsibility for interpretation. QODIQA assumes responsibility only for execution.

9.4 Reduced Flexibility in Exchange for Authority

The framework rejects several forms of flexibility that are common in application-layer systems: default values for missing inputs, best-effort evaluation, silent fallbacks, and dynamic inference under uncertainty. These omissions may increase friction in the short term, but they are necessary to maintain authority and correctness. When enforcement systems attempt to be helpful, they often become unpredictable. QODIQA chooses refusal over ambiguity.

9.5 Not a User-Facing Governance Tool

QODIQA is not designed to be a dashboard, reporting platform, or user interface. While decision artifacts may be surfaced to other systems, visualization and reporting are explicitly out of scope for the core framework. This boundary ensures that the enforcement layer remains lightweight, composable, and resilient to changes in organizational workflows or tooling preferences.

9.6 No Retroactive Control

Decisions made by the system are immutable. Changes to policies, consent state, or rulesets do not retroactively alter past outcomes. While this may limit the ability to correct historical behavior, it preserves auditability and proof. Retroactive modification of decisions would compromise determinism and undermine trust in the system's outputs.

9.7 Scope as a Feature

The limitations described above are not temporary gaps or future roadmap items. They are structural choices that define what the framework is and is not. By narrowing its scope, QODIQA remains predictable under scale, defensible under scrutiny, and stable as AI systems evolve. Attempts to expand beyond these boundaries would erode the very guarantees the framework is designed to provide.

> This section exists to prevent misinterpretation and overextension. Clear non-goals are as important as defined capabilities when building infrastructure intended to outlast specific technologies or regulatory cycles.

SECTION 10

Conclusion and Outlook

Artificial intelligence has reached a level of capability where questions of governance can no longer be treated as external concerns. As AI systems increasingly operate inside critical workflows, the legitimacy of their actions depends not on intent or documentation, but on whether authorization can be enforced, verified, and proven at the moment of execution.

This paper has argued that many of the failures attributed to AI governance are not failures of regulation, ethics, or organizational process, but failures of system design. Consent and authorization are typically defined outside the execution path, expressed in human-readable form, and enforced retrospectively. In environments characterized by autonomy, scale, and speed, these approaches are structurally insufficient.

QODIQA introduces a different framing. Rather than attempting to make AI systems more cautious, interpretable, or compliant after the fact, it embeds consent enforcement directly into the execution path. By treating intent as an explicit

contract, consent as state, policies as immutable logic, and decisions as reproducible artifacts, the framework transforms authorization from an assumption into a verifiable system property.

The central contribution of this approach is not a new governance model, but a new layer of infrastructure. One that is deliberately narrow in scope, deterministic by design, and resistant to ambiguity. In doing so, it separates responsibility from enforcement: humans remain accountable for defining rules and consent, while systems execute those rules mechanically and without interpretation.

This shift has broader implications. It allows organizations to scale AI usage without scaling uncertainty. It enables audits that rely on proof rather than narrative. It reduces reliance on trust in systems whose behavior must instead be demonstrable.

Most importantly, it establishes a foundation on which legal, ethical, and organizational governance can be applied consistently, rather than aspirationally.

The approach outlined here does not claim to solve all challenges associated with AI governance. It intentionally avoids semantic interpretation, probabilistic reasoning, and adaptive behavior. These constraints are not limitations to be overcome, but boundaries that preserve correctness and defensibility. As AI systems continue to evolve, such boundaries will become increasingly valuable.

Execution-layer consent enforcement represents a necessary evolution in AI system architecture. As AI moves from experimentation to infrastructure, legitimacy must become a property of execution itself. QODIQA is proposed as a concrete step in that direction.

DOCUMENT STATUS

Document Status and Corpus Alignment

This document is the foundational whitepaper of the QODIQA specification corpus. It provides the conceptual and technical rationale for QODIQA: a deterministic runtime consent enforcement layer for artificial intelligence systems.

The whitepaper establishes the structural argument for treating consent as executable infrastructure rather than a policy obligation. It defines the properties that enforceable AI consent must satisfy, introduces the QODIQA enforcement framework, describes its internal architecture, and characterizes the determinism, auditability, and proof guarantees that emerge from this design.

This document should be read together with the following related specifications:

- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement
- QODIQA: Terminology and Normative Definitions
- QODIQA: Threat Model and Abuse Case Specification
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement

— QODIQA: Governance Charter for the QODIQA Standard Corpus

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dițescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-WP-2026-001
TITLE	Consent as Infrastructure for Artificial Intelligence: Technical Whitepaper
SUBTITLE	Deterministic Runtime Consent Enforcement for Artificial Intelligence Systems
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Technical Whitepaper
DOCUMENT STATUS	Normative
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 03 PART I · ORIENTATION AND FOUNDATIONS

Core Standard for Deterministic Runtime Consent Enforcement

A minimal runtime consent enforcement layer for artificial intelligence systems, forming the foundational conformance tier of an institutional governance framework.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-CORE-2026-001	1.0	2026	Normative Specification	Institutional	NORMATIVE

ABSTRACT

ABSTRACT

Artificial intelligence systems routinely operationalize data at scale while Consent remains external to runtime execution, captured in static terms, batch permissions, or policy documents that do not deterministically constrain model invocation at the moment of use. The resulting infrastructure gap is structural: Authorization governs system access, but no machine-verifiable mechanism evaluates, at the execution boundary, whether the specific action is legitimate under present Consent constraints.

This document specifies **QODIQA Core**: the minimal adoptable runtime Consent enforcement layer, derived from the QODIQA 68-Point Enforcement Framework. QODIQA Core defines twelve Control Points, a deterministic enforcement lifecycle, Fail-Closed semantics, and tamper-evident Decision Artifacts sufficient to make Consent enforceable as a runtime control plane without requiring model retraining or infrastructure replacement.

The standard is intended for adoption by enterprises, public institutions, and AI infrastructure providers within a 30–60 day integration horizon. It is the entry-level Conformance tier of a unified QODIQA institutional corpus and constitutes the foundational layer from which full 68-point Conformance is built.

SECTION 1

Executive Summary

1.1 Problem Statement: Unenforced Consent in AI Execution Paths

AI systems operationalize data at scale while Consent remains external to runtime execution, captured via static terms, batch permissions, or informal policy assertions that do not deterministically constrain model invocation at the moment of use. This produces an infrastructure gap: the technical act of calling an AI system proceeds under Authorization: API keys, role-based access controls, vendor entitlements, but without a machine-verifiable, context-bound evaluation of whether the specific action is legitimate under present Consent constraints encompassing purpose, scope, time, jurisdiction, and Revocation state.

QODIQA's foundational position is that **Consent is a first-class infrastructure primitive**: machine-readable, replayable, and enforced as a Gating Function before execution. The practical deficiency is not the absence of policy documents; it is the absence of Deterministic Enforcement at the decision boundary where an AI system is invoked.

1.2 The Systemic Infrastructure Gap

The absence of runtime Consent enforcement is not an edge-case deficiency. It is a structural failure class with predictable scaling properties.

Contemporary AI governance frameworks establish obligations: documentation requirements, data processing agreements, access control policies, audit procedures. These instruments are necessary and insufficient at the execution boundary. A policy document that describes permitted purposes does not prevent an AI system from acting on an impermissible purpose. An audit log that records what happened does not prevent its occurrence. Authorization that grants system access does not evaluate whether the specific action under the specific context is legitimate at the moment it is taken.

This is a missing runtime invariant, not a missing policy but a missing enforcement layer. Current AI governance architecture provides controls that observe or describe behavior. It lacks controls that deterministically gate it.

> The absence of Consent enforcement at runtime is structurally analogous to operating a cryptographic system without key verification: authentication exists at configuration time, but verification is absent at the moment it matters. In both cases, the gap is not philosophical; it is architectural.

This gap scales adversely. In single-model, single-purpose deployments, the risk is bounded. In multi-agent architectures where models call other models, where orchestration layers route requests across vendors, and where a single authorization event is reused across hundreds of downstream actions, the gap becomes a systemic risk amplifier.

The only architecturally sound response is to treat runtime Consent enforcement as infrastructure: present at every boundary, deterministic in behavior, and independent of the intelligence of the system it governs.

1.3 Runtime Enforcement as the Non-Substitutable Control Layer

Training-phase controls, including dataset audits, provenance records, pre-training permissions, are necessary and insufficient for operational governance because: (a) Consent changes over time through Revocation or expiry; (b) purpose drift occurs at inference time independent of training-phase Authorization basis; and (c) deployment patterns

route prompts and content across systems and vendors where retrospective governance cannot reconstruct what was permitted at the moment of use.

Runtime is the only boundary where purpose, actor identity, jurisdiction, and Revocation state are simultaneously present, evaluable, and current. Controls applied at any other lifecycle point cannot substitute for enforcement at execution. QODIQA Core positions its twelve Control Points at this boundary and at no other.

1.4 Why a Minimal Standard Is Required

Enterprises and public institutions require a practical, low-friction adoption layer that can be implemented within 30–60 days, does not require model retraining, and does not require rewriting existing AI infrastructure. A minimal standard is necessary to ensure: (i) interoperability across heterogeneous AI stacks; (ii) Deterministic Governance for regulated and contractual contexts; and (iii) a clear Conformance boundary that can be audited and independently certified.

1.5 Scope: What QODIQA Core Covers

- A closed, explicit intent declaration interface prohibiting inferred intent
- Cryptographically verifiable Consent Tokens and Deterministic validation logic
- Purpose- and scope-bound Policy Evaluation with Fail-Closed semantics
- Revocation and expiry enforcement at runtime with defined Conflict Resolution Rules
- Immutable Decision Artifacts and Audit Logging enabling independent Replay Verification

1.6 Explicit Exclusions

QODIQA Core does not define or require: model retraining or weight alteration; content inspection, semantic classification, or probabilistic risk scoring as part of the enforcement decision; economic rights or royalty distribution infrastructure; full lifecycle controls for training and fine-tuning governance; or user interfaces, dispute workflows, or appeals dashboards. The enforcement layer decides predictably, not interpretively.

1.7 Immediate Adoption Value

QODIQA Core makes Consent enforceable as a Deterministic runtime control plane, producing reproducible Decision Artifacts and tamper-evident Audit Logs that support independent Replay Verification. Adoption reduces ambiguity at the execution boundary, eliminates purpose drift, and establishes institutional defensibility without changes to deployed AI models.

SECTION 2

Normative Scope and Conformance

2.1 Normative Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC 2119] [RFC 8174]. Non-implementation of any MUST-level requirement constitutes non-Conformance with this standard across all profiles. Deviation from any SHOULD-level requirement MUST be explicitly documented with written justification and recorded in the Conformance Statement (Section 13).

2.2 Scope of Applicability

This standard applies to AI systems, including large language model systems, inference APIs, and tool-using agent runtimes, where an organization requires runtime enforcement of Consent constraints over AI actions that use, transform, generate, disclose, or otherwise operationalize data subject to Consent conditions. A Conformant implementation requires that a deterministic Enforcement Gateway be positioned on the request path before model invocation. This standard is model-agnostic and requires no access to model internals, weight parameters, or training artifacts.

2.3 System Boundary

A Conformant QODIQA Core implementation governs the decision boundary between a Calling System (application, service, or agent) requesting an AI action, and an AI System (model endpoint, orchestration service, or tool-using agent runtime) that would execute that action. The Enforcement Gateway MUST operate as a pre-execution gate and MUST be capable of failing closed on any missing, malformed, or unresolvable input.

2.4 Conformant Implementation

A Conformant implementation MUST: require explicit intent declarations via a closed DecisionRequest contract; validate Consent state at runtime including expiry and Revocation prior to execution; enforce purpose- and scope-bound Policy deterministically with Fail-Closed behavior on missing, malformed, or unavailable inputs; and emit Decision Artifacts and Audit Records sufficient for independent Replay Verification.

A non-Conformant system is any system that: infers or approximates intent in the absence of explicit declaration; treats Consent enforcement as an advisory check that can be bypassed; defaults to allow on incomplete, inconsistent, or unresolvable context; or cannot produce a tamper-evident Audit Trail traceable to a Frozen Evaluation Context.

2.5 Conformance Profiles

PROFILE	NAME	REQUIREMENTS
CORE-MIN	Minimal Runtime	All QCC-01 through QCC-10 at MUST level. QCC-11 and QCC-12 implemented or declared as deviations with written justification.
CORE-VER	Verifiable Runtime	CORE-MIN plus cryptographic Audit Log chaining and externally verifiable Decision Artifact export suitable for third-party audit.
CORE-ENT	Enterprise Runtime	CORE-VER plus defined operational SLAs for Revocation propagation, Audit Log retention, and a documented Control Effectiveness Testing cadence.

A Conformant implementation MUST declare exactly one profile. Declaration of a higher profile without satisfaction of all requirements of that profile constitutes non-Conformance.

SECTION 3

Threat and Risk Model

3.1 Authorization vs. Consent: Formal Distinction

DEFINED TERMS: Adopted from QODIQA Whitepaper Section 3.3

Authorization

Standing technical permission granted to an actor or system to access a resource or perform a class of operations. Expressed through roles, access control lists, API keys, or service identities. Generally static, coarse-grained, and context-independent. Answers: *Is this actor permitted to perform this class of operation?*

Consent

Contextual, conditional legitimacy for a specific action under explicitly declared purpose, scope, time constraints, jurisdictional binding, and current Revocation state. Granular, time-bounded, and revocable. Answers: *Is this specific action legitimate under present conditions?*

A system may hold Authorization for a class of operations and still lack valid Consent for a specific action within that class. Runtime enforcement MUST evaluate Consent independently of Authorization. The presence of Authorization does not satisfy the Consent requirement.

3.2 Consent Failure Taxonomy

TABLE 1: CONSENT FAILURE TAXONOMY

ID	THREAT CLASS	MECHANISM	PRIMARY CONTROL
T-01	Implicit Consent Assumption	Execution proceeds without machine-verifiable Consent proof at the moment of action	QCC-01, QCC-02
T-02	Purpose Drift	Capabilities operationalized for purposes not explicitly granted in the applicable Consent Token	QCC-03
T-03	Temporal Decay	Consent has expired or been revoked; downstream execution continues without re-evaluation	QCC-05, QCC-06
T-04	Context Collapse	Requests lack differentiating metadata; coarse permissions applied where granular evaluation is required	QCC-01, QCC-04
T-05	Consent Forgery and Replay	Fabricated or previously-captured tokens presented to obtain execution Authorization	QCC-02, QCC-06
T-06	Revocation Non-Propagation	Revocation event does not reach all enforcement nodes within the required propagation window	QCC-06, QCC-12
T-07	Audit Evasion	Actions occur without durable, tamper-evident records; enforcement cannot be independently verified	QCC-09, QCC-10
T-08	Registry Unavailability Bypass	System defaults to allow when Consent Registry is temporarily unavailable, converting transient failure into governance gap	QCC-06, QCC-08

3.3 Runtime vs. Training-Phase: Irreducible Distinction

Training-phase governance addresses: *What data was used to construct model capabilities, and under what conditions?* Runtime governance addresses: *What actions are performed now, by which actor, for which purpose, under which present Consent state?* These are non-substitutable control layers. Training-phase controls cannot substitute runtime enforcement because: (a) Revocation and expiry affect Consent validity independently of training provenance; (b) purpose drift emerges at inference time without correlation to the training-phase Authorization basis; and (c) multi-vendor and agentic deployment chains introduce runtime actors and data flows that were not present or knowable at training time.

3.4 Liability Surface and Evidentiary Posture

The institutional liability surface expands in direct proportion to the inability to demonstrate with objective, reproducible evidence: what the AI system was instructed to perform; the Policy Version under which the enforcement decision was made; the Consent state that applied at the precise moment of the decision; and whether Revocation and expiry were evaluated and enforced.

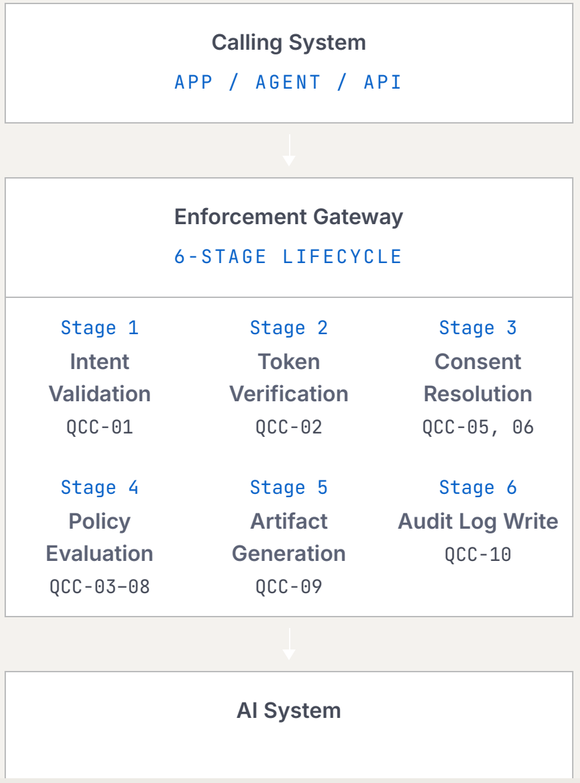
Without deterministic Decision Artifacts, an organization cannot objectively demonstrate lawful runtime behavior. Policy documents, intent statements, and log entries do not constitute proof that a specific action was authorized under a specific Consent state at a specific time. QODIQA's "proof over explanation" posture defines institutional defensibility as the ability to reproduce decisions from a Frozen Evaluation Context.

SECTION 4

Core Architecture

4.1 Minimal Enforceable Component Model

FIGURE 1: QODIQA CORE ENFORCEMENT ARCHITECTURE





4.2 Component Responsibilities

COMPONENT	RESPONSIBILITY	INVOCATION CONSTRAINT
Enforcement Gateway	Intercepts all AI-bound requests on the critical path; implements six-stage lifecycle	MUST be on the critical path; architectural bypass constitutes non-Conformance
Policy Engine	Executes Deterministic Evaluation as a pure function over a Frozen Evaluation Context	No side effects; no external calls during evaluation
Consent Registry	Stores and resolves current Consent state including grants, expiry, and Revocation status	Authoritative source for Consent state; unavailability triggers DENY
Decision Engine	Computes ALLOW or DENY; generates Decision Artifact	Artifact generation failure triggers DENY
Audit Layer	Writes tamper-evident, hash-chained Audit Log entries	Write-before-execute on ALLOW path; write failure triggers DENY

4.3 Allow and Deny Decision Semantics

ALLOW is issued if and only if all conditions are simultaneously satisfied: the intent declaration is explicit and structurally complete; Consent Token cryptographic integrity is verified; the Consent Token is ACTIVE for the requested purpose, scope, jurisdictional binding, and data categories; all applicable Policy conditions are satisfied; and the Decision Artifact and Audit Log entry are successfully generated and written.

DENY MUST be issued on any of: missing or malformed intent fields; invalid or unverifiable Consent Token; expired, revoked, or unresolvable Consent state; purpose mismatch; scope violation; jurisdictional constraint violation; Policy evaluation failure; Decision Artifact generation failure; or Audit Log write failure. No partial execution occurs on any DENY path.

4.4 Conflict Resolution Rules

RULE	NAME	BINDING EFFECT
R-01	Revocation Supersedes	Any Revocation event cancels all prior grants. No prior ALLOW decision or cached state overrides a REVOKED status.
R-02	Most Restrictive Scope	Where multiple applicable Consent grants exist, the narrowest defined scope governs.
R-03	Subject Jurisdiction Precedence	In cross-border conflicts, the subject's local jurisdiction requirements SHALL apply in full.
R-04	Explicit Override Required	Broader scope application requires explicit Authorization. Inference or assumption of broader scope is prohibited.

SECTION 5

The Twelve Core Control Points (QCC-01 through QCC-12)

The twelve Control Points defined in this section are the minimal, non-redundant enforcement nucleus of QODIQA Core. QCC-01 through QCC-10 are MUST-level; non-implementation of any constitutes non-Conformance across all profiles. QCC-11 and QCC-12 are SHOULD-level; non-implementation MUST be declared with written justification in the Conformance Statement.

TABLE 2: CONTROL POINT SUMMARY

ID	NAME	LEVEL	FRAMEWORK ALIGNMENT
QCC-01	Closed Intent Declaration	MUST	CP-01; Whitepaper Section 5.2
QCC-02	Consent Token Cryptographic Verification	MUST	CP-11, CP-63, CP-64; Gate 02
QCC-03	Purpose Binding	MUST	CP-02; Whitepaper Section 3.2
QCC-04	Scope Binding	MUST	CP-02, CP-16; Gate 03

ID	NAME	LEVEL	FRAMEWORK ALIGNMENT
QCC-05	Temporal Validity Enforcement	MUST	CP-43, CP-47, CP-48
QCC-06	Revocation Enforcement	MUST	CP-04, CP-45, CP-65; R-01
QCC-07	Deterministic Policy Evaluation	MUST	Whitepaper Sections 6.4–7.3
QCC-08	Fail-Closed Default Denial	MUST	Whitepaper Sections 5.6, 6.7, 9.4
QCC-09	Decision Artifact Generation	MUST	Whitepaper Sections 6.6, 7.2, 7.5
QCC-10	Tamper-Evident Audit Logging	MUST	CP-25, CP-49, CP-67; Section 13.3
QCC-11	Jurisdictional Contextual Binding	SHOULD	R-03; Whitepaper Section 4.2
QCC-12	Consent Withdrawal Operability	SHOULD	CP-04, CP-45, CP-65

QCC-01 Closed Intent Declaration

● MUST

OBJECTIVE	Enforce a closed, explicit intent declaration for every AI action request. Prohibit inferred, approximated, or partially-specified intent.
REQUIRED INPUTS	`actor_id`, `action` (enumerated), `purpose` (enumerated), `data_types` (non-empty array), `jurisdiction` (ISO 3166-1 alpha-2), `timestamp` (ISO 8601)

DECISION LOGIC

Decision Logic — NORMATIVE

```
// Any missing or null required field
if any_required_field_absent → DENY("INTENT_FIELD_MISSING")
// Purpose not in registered vocabulary
if purpose ∉ purpose_registry → DENY("PURPOSE_UNRECOGNIZED")
// Action not in registered vocabulary
if action ∉ action_registry → DENY("ACTION_UNRECOGNIZED")
// Empty or unrecognized data categories
if data_types = ∅ or any ∉ data_type_registry → DENY("DATA_TYPE_INVALID")
// Extra unknown fields
if extra_fields_present → DENY("SCHEMA_VIOLATION")
```

FAILURE STATE	HTTP 400 (malformed contract) or HTTP 403 (Consent required). MUST be Fail-Closed: no partial evaluation proceeds.
AUDIT REQUIREMENTS	Log canonicalized DecisionRequest hash, schema version, validation result, and denial reason code.
FRAMEWORK ALIGNMENT	CP-01 (Explicit Opt-In Gate); Whitepaper Section 5.2 (Input Contract: Closed by Design)

QCC-02 Consent Token Cryptographic Verification

MUST

OBJECTIVE	Verify cryptographic integrity and authenticity of the Consent Token before any further evaluation proceeds. Reject unsigned, malformed, or issuer-untrusted tokens.

REQUIRED INPUTS

`consent_token`, `issuer`, `verification_key_material` (trust anchor), `algorithm_identifier`. Ed25519 RECOMMENDED. ECDSA P-256 PERMITTED. RSA-PSS NOT RECOMMENDED. Algorithm negotiation MUST be server-authoritative.

DECISION LOGIC

Decision Logic — NORMATIVE

```
if consent_token = null → DENY("CONSENT_TOKEN_MISSING")
if signature absent or structurally malformed → DENY("TOKEN_SIGNATURE_ABSENT")
if verify_signature(token, trust_anchor) = false → DENY("CONSENT_SIGNATURE_INVALID")
if issuer ∉ trusted_issuer_registry → DENY("ISSUER_UNTRUSTED")
if algorithm_identifier ∉ approved_algorithms → DENY("ALGORITHM_UNSUPPORTED")
```

FAILURE STATE

HTTP 401 or HTTP 403. Bypass of this control by any means constitutes a critical-severity violation.

AUDIT REQUIREMENTS

Log Consent Token fingerprint (SHA-256), issuer identifier, algorithm identifier, and verification outcome.

FRAMEWORK ALIGNMENT

CP-11, CP-63, CP-64 (Cryptographic Consent Layer); Gate 02 (API Consent Handshake)

QCC-03 Purpose Binding

MUST

OBJECTIVE

Bind Consent to declared purpose at runtime. Prevent purpose drift by requiring exact match between requested purpose and purposes explicitly authorized in the Consent Token. No inference, approximation, or semantic similarity matching is permitted.

DECISION LOGIC

Decision Logic — NORMATIVE

```
// Exact string match required, no semantic approximation
if request.purpose ∉ token.purposes → DENY("PURPOSE_NOT_AUTHORIZED")
// Token-encoded restrictions contradicted by request
if purpose_restriction_violated → DENY("PURPOSE_RESTRICTION_VIOLATED")
```

FAILURE STATE

HTTP 403 (purpose not authorized).

FRAMEWORK ALIGNMENT

CP-02 (Granular Scope Binding); Whitepaper Section 3.2 (Purpose Drift as Primary Failure Mode)

QCC-04 Scope Binding

● MUST

OBJECTIVE

Enforce scope binding: requested action and data categories must fall within the scope explicitly authorized in the Consent Token. Conflict Rule R-02 (Most Restrictive Scope) applies.

DECISION LOGIC

Decision Logic — NORMATIVE

```
if request.action ∉ token.scope.actions → DENY("ACTION_OUT_OF_SCOPE")
if ∃ dt ∈ request.data_types : dt ∉ token.scope.data_types → DENY("DATA_TYPE_OUT_OF_SCOPE")
// Where multiple applicable scopes exist, apply the most restrictive (R-02)
applicable_scope ← min_restrictive(matching_scopes)
```

FAILURE STATE	HTTP 403 (scope violation). Partial scope matches do not satisfy this control.
FRAMEWORK ALIGNMENT	CP-02, CP-16 (Scope Validation Engine); Gate 03 (Scope Validation); R-02

QCC-05 Temporal Validity Enforcement

MUST

OBJECTIVE	Enforce temporal validity. Consent MUST be valid at the exact moment of execution. Consent that has expired MUST NOT authorize execution.
DECISION LOGIC	Decision Logic — NORMATIVE

```
if request_time > token.exp → DENY("CONSENT_EXPIRED")
if token.nbf present and request_time < token.nbf → DENY("CONSENT_NOT_YET_VALID")
if |request_time - token.iat| > clock_tolerance_seconds → DENY("CLOCK_SKEW_EXCEEDED")
if clock_source unavailable or unsynchronized → DENY("CLOCK_UNRESOLVABLE")
```

FAILURE STATE	HTTP 403 (consent expired / not yet valid). `clock_tolerance_seconds` MUST NOT exceed 300s without documented justification.
FRAMEWORK ALIGNMENT	CP-43 (Consent Expiry Enforcement), CP-47 (Stale Consent Invalidation), CP-48 (Time-Bound Use Restriction)

QCC-06 Revocation Enforcement

MUST

OBJECTIVE	Enforce Revocation at runtime. Revocation supersedes all prior grants (R-01). A revoked Consent Token MUST NOT authorize execution regardless of its temporal validity or scope.
DECISION LOGIC	

Decision Logic — NORMATIVE

```
if registry.lookup(token_id) = REVOKED → DENY("CONSENT_REVOKED")  // R-01
if registry_state = UNKNOWN or registry_unavailable → DENY("REVOCATION_UNRESOLVED")
if cache_age > cache_ttl_seconds and registry_unavailable → DENY("REVOCATION_CACHE_STALE")
// Revocation supersedes all prior grants, no exception
```

FAILURE STATE	HTTP 403 (revoked) or HTTP 503/403 (registry unavailable). MUST be Fail-Closed in all registry-unavailable scenarios. CORE-ENT revocation propagation SLA: MUST NOT exceed 300 seconds.
FRAMEWORK ALIGNMENT	CP-04 (Revocation Propagation), CP-45 (Retroactive Revocation Handling), CP-65 (Revocation Registry Sync); R-01

QCC-07 Deterministic Policy Evaluation

MUST

OBJECTIVE	Policy Evaluation MUST be deterministic: given identical Frozen Evaluation Contexts, the decision MUST be identical across any Conformant implementation instance, time, or environment. See Section 6 (Formal Determinism) for the formal invariant.

DECISION LOGIC

Decision Logic — NORMATIVE

```
frozen ← canonicalize(request, consent_ref, policy_version, ruleset_version)
decision ← policy_engine.evaluate(frozen)  // pure function, no side effects
// Invariant: evaluate(frozen) = replay(frozen) for all valid frozen contexts
if policy_engine cannot guarantee determinism → DENY("POLICY_ENGINE_FAULT") + SYSTEM_FAULT
```

FAILURE STATE

SYSTEM_FAULT with DENY outcome. The fault condition MUST be logged and escalated. No execution proceeds under a non-deterministic or faulted policy engine.

FRAMEWORK ALIGNMENT

Whitepaper Sections 6.4–6.5, 7.1–7.3 (Determinism as First-Class Property; Replay as Correctness Invariant)

QCC-08 Fail-Closed Default Denial

MUST

OBJECTIVE

Enforce Fail-Closed behavior across all Enforcement Gateway stages. Ambiguity, missing data, unresolvable state, or any processing exception MUST result in DENY. No partial evaluation, best-effort evaluation, or exception-to-allow path is permissible.

DECISION LOGIC

Decision Logic — NORMATIVE

```
// Default state of the Policy Engine MUST be DENY
default_effect = DENY
// ALLOW requires all conditions satisfied; any unsatisfied → DENY
if evaluation_exception → DENY("POLICY_ENGINE_FAULT")
if partial_evaluation_required → DENY("EVALUATION_INCOMPLETE")
// No "best-effort" or "optimistic" evaluation paths permitted
```

FAILURE STATE	HTTP 403 (default deny). Any ALLOW path must be explicitly computed, not assumed.
FRAMEWORK ALIGNMENT	Whitepaper Sections 5.6, 6.7, 9.4 (Fail-Safe Defaults; Conservative Execution Model)

QCC-09 Decision Artifact Generation

MUST

OBJECTIVE	Generate a stable, tamper-evident Decision Artifact for every evaluation, covering both ALLOW and DENY outcomes. The artifact must contain sufficient information to replay and verify the decision independently.
REQUIRED FIELDS	`artifact_id` (deterministic hash), `decision` (ALLOW|DENY), `reason_code`, `frozen_context_hash`, `policy_version`, `ruleset_version`, `timestamp` (ISO 8601, NTP-verified)

DECISION LOGIC

Artifact Identity — NORMATIVE

```
artifact_id ← H(canonical(R̂ ‖ S ‖ V_p ‖ V_r ‖ outcome))

// where H = SHA-256 (minimum); SHA-512 RECOMMENDED
// identical FEC + identical outcome → identical artifact_id
// independently recomputable by any party with access to FEC inputs
if artifact_generation_fails → DENY("ARTIFACT_GENERATION_FAILED")
```

FRAMEWORK ALIGNMENT

Whitepaper Sections 6.6, 7.2, 7.5 (Decision as Artifact; Proof Over Explanation)

QCC-10 Tamper-Evident Audit Logging

MUST

OBJECTIVE

Write a tamper-evident, hash-chained Audit Log entry for every enforcement decision. Write MUST precede execution on the ALLOW path. Write failure MUST trigger DENY (integrity over availability).

DECISION LOGIC

Write-Before-Execute Constraint — NORMATIVE

```
// ALLOW path: write audit entry BEFORE execution signal
if audit_write(artifact, ALLOW) = FAILURE → DENY("AUDIT_WRITE_FAILED")
// integrity > availability, audit failure blocks execution
// Hash chaining: entry_n.hash = H(entry_n.content ‖ entry_{n-1}.hash)
// Append-only; deletion before minimum retention = non-Conformance event
```

RETENTION	Declared minimum retention period in Conformance Statement. CORE-ENT minimum: 365 days. Durable, append-only or write-once storage required.
FRAMEWORK ALIGNMENT	CP-25 (Audit Trail Mandate), CP-49 (Temporal Audit Sequencing), CP-67 (Immutable Consent Ledger)

QCC-11 Jurisdictional Contextual Binding

SHOULD

OBJECTIVE	Apply jurisdictional contextual binding to Consent decisions when jurisdiction is declared. Enforce Conflict Rule R-03 when subject jurisdiction and processing jurisdiction differ.

DECISION LOGIC

Decision Logic — NORMATIVE

```
if jurisdictions_conflict → apply subject_jurisdiction precedence (R-03)
if jurisdiction declared but unrecognized → DENY("JURISDICTION_UNRECOGNIZED")
if jurisdiction required by profile and absent → DENY("JURISDICTION_REQUIRED")
```

DEVIATION	Non-implementation MUST be declared in Conformance Statement with written justification and named risk acceptance authority.
FRAMEWORK ALIGNMENT	R-03 (Subject Jurisdiction Precedence); Whitepaper Section 4.2 (Contextual Binding)

QCC-12 Consent Withdrawal Operability

SHOULD

OBJECTIVE	Ensure Consent withdrawal, once accepted, propagates to enforcement within a declared propagation SLA and is operationally effective. The Revocation Registry MUST reflect the withdrawn state before the SLA expires.
DECISION LOGIC	Decision Logic — NORMATIVE

```
on accepted_revocation:
  registry.mark_revoked(token_ref)
  next_evaluation(token_ref) → DENY("CONSENT_REVOKED")  // guaranteed
if propagation_time > propagation_sla → CONTROL_FAILURE
  log_and_escalate(REVOCATION_SLA_BREACH)
```

DEVIATION	Non-implementation MUST be declared in Conformance Statement with written justification and named risk acceptance authority.
FRAMEWORK ALIGNMENT	CP-04 (Revocation Propagation), CP-45, CP-65; 68-point Section 13.2 (Revocation SLA Windows)

SECTION 6

Formal Determinism and Replay Invariants

6.1 Frozen Evaluation Context: Formal Definition

The Frozen Evaluation Context is the minimal immutable input set from which any Conformant enforcement decision can be independently reproduced.

DEFINITION: Frozen Evaluation Context (FEC)

FEC = $(\hat{R}, S, V_p, V_r)$

$\hat{R}$ = canonicalize(R): canonical serialization of the DecisionRequest R, deterministically produced from its required fields using a specified canonicalization algorithm (e.g., canonical JSON, RFC 8785).

S = consent_state_ref: an immutable reference to the resolved Consent state snapshot at evaluation time. S is not the live Consent store; it is the state as resolved and frozen at the moment of evaluation.

V_p = policy_version: the version identifier of the Policy document in effect at evaluation time. References an immutable, versioned policy artifact.

V_r = ruleset_version: the version identifier of the Ruleset applied during Policy Evaluation.

6.2 Formal Evaluation Function

Policy Evaluation is defined as a total, deterministic function over the Frozen Evaluation Context:

DEFINITION: Evaluation Function

$f : FEC \rightarrow \{ALLOW, DENY\}$

$decision = f(\hat{R}, S, V_p, V_r)$

where f is **Total** (defined for all well-formed FEC inputs), **Deterministic** (f(C) = f(C) for all C), **Pure** (produces no observable side effects), and **Time-independent** (does not take the current time as an input beyond the frozen context).

6.3 Determinism Invariant

Replay Invariant: For all frozen evaluation contexts C and for all Conformant implementations I_1, I_2 operating over the same policy and ruleset versions: I_1.evaluate(C) = I_2.evaluate(C) = I_1.replay(C). A decision that cannot be replayed to the same outcome cannot be proven. A Conformant implementation that fails this invariant is incorrect.

6.4 Non-Determinism Exclusion Clause

The following inputs and behaviors are explicitly prohibited during Policy Evaluation:

PROHIBITED INPUT / BEHAVIOR	REQUIREMENT
Randomness	No calls to random number generators, probabilistic functions, or stochastic sampling
External service calls	No network calls, database reads, cache lookups, or API invocations during evaluation
Current time access	No access to wall-clock time or system time during evaluation. Temporal bounds were resolved in QCC-05 and are encoded in S
Mutable configuration	No reads from mutable configuration stores, feature flags, or dynamic policy sources

PROHIBITED INPUT / BEHAVIOR	REQUIREMENT
Environment-dependent behavior	Evaluation outcome MUST NOT depend on hardware, OS, locale, timezone, or execution environment state
Side effects	f MUST NOT write to any external store, emit events, or mutate any shared state during execution

6.5 Decision Identity and Artifact Stability

Artifact Identity — NORMATIVE

```
artifact_id ← H(canonical(R̂ ‖ S ‖ V_p ‖ V_r ‖ outcome))

// where H = SHA-256 (minimum); SHA-512 RECOMMENDED
// Properties:
//   · identical FEC + identical outcome → identical artifact_id
//   · no raw sensitive data stored in artifact
//   · independently recomputable by any party with access to FEC inputs
//   · suitable for audit, litigation, and regulatory evidence production
```

SECTION 7

Implementation Model

7.1 API Enforcement Pattern

Every AI-bound request MUST pass through a preflight evaluation by the Enforcement Gateway before reaching the AI System. The Calling System MUST NOT invoke the AI System without a valid ALLOW decision accompanied by a Decision Artifact. The AI System MUST NOT be reachable via any path that bypasses the Enforcement Gateway. Architectural bypass constitutes non-Conformance.

Enforcement Interaction Pattern — NORMATIVE

```
Calling System → (1) DecisionRequest + ConsentToken → Enforcement Gateway
Enforcement Gateway → (2) ALLOW + DecisionArtifact → Calling System
Calling System → (3) AI Request + ArtifactReference → AI System

// AI System MUST NOT be reached without a valid ALLOW + ArtifactReference
// DENY path: Enforcement Gateway → DenyDecision + reason_code → Calling System
// AI System is not invoked on DENY path
```

7.2 Reference Data Structures

DecisionRequest: Closed Contract — JSON

```json
{
  "request_id":    "b9d4f0c0-7f7c-4c0b-9b2c-2d14f6b0df53",
  "actor_id":      "did:example:actor-123",       // DID-format RECOMMENDED
  "action":        "inference.generate_text",      // enumerated action code
  "purpose":       "customer_support",             // enumerated purpose code
  "data_types":    ["user_provided_text"],         // non-empty; enumerated
  "jurisdiction":  "EU",                          // ISO 3166-1 alpha-2
  "target_system": "llm.vendorX.chat",
  "timestamp":     "2026-02-26T12:00:00Z"
}
```

Consent Token: Minimal Normative Schema — JSON

```json
{
  "token_id":           "consent:01HZZ...9K",
  "subject_id":         "did:example:subject-456",
```

```
  "issuer":              "did:example:consent-authority",
  "issued_at":           "2026-01-01T00:00:00Z",
  "expires_at":          "2026-12-31T23:59:59Z",
  "purposes":            ["customer_support"],
  "scope": {
    "actions":           ["inference.generate_text"],
    "data_types":        ["user_provided_text"],
    "systems":           ["llm.vendorX.chat"]
  },
  "jurisdiction":        "EU",
  "revocation_registry": "https://consent.example.org/revocations",
  "signature": {
    "alg": "Ed25519",
    "value": "BASE64URL_ENCODED_SIGNATURE"
  }
}
```

7.3 Normative Enforcement Pseudocode

Enforcement Decision Function: Normative Reference — PSEUDOCODE

```
function decide(request, token):
  now ← gateway_synchronized_time()  // NTP-verified source

  // QCC-01 · Closed intent contract
  if ¬validate_intent_schema(request):
    return deny("INTENT_INVALID", null)

  // QCC-02 · Cryptographic token verification
  if token = null:    return deny("CONSENT_TOKEN_MISSING", null)
  if ¬verify_signature(token, trust_anchor):
    return deny("CONSENT_SIGNATURE_INVALID", null)
```

```
// QCC-05 · Temporal validity
if now > token.expires_at:  return deny("CONSENT_EXPIRED", null)
if token.nbf ∧ now < token.nbf:  return deny("CONSENT_NOT_YET_VALID", null)

// QCC-06 · Revocation: fail-closed on registry unavailability (R-01)
rev ← revocation_registry.lookup(token.token_id)
if rev = REVOKED:         return deny("CONSENT_REVOKED", null)
if rev = UNKNOWN ∨ registry_unavailable():
  return deny("REVOCATION_UNRESOLVED", null)

// QCC-03 · Purpose binding, exact match
if request.purpose ∉ token.purposes:
  return deny("PURPOSE_NOT_AUTHORIZED", null)

// QCC-04 · Scope binding, strict set containment
if request.action ∉ token.scope.actions:
  return deny("ACTION_OUT_OF_SCOPE", null)

// QCC-07 + QCC-08 · Deterministic policy evaluation, fail-closed
frozen ← freeze(request, token.token_id, policy_version(), ruleset_version())
try:
  decision ← policy_engine.evaluate(frozen)
except:
  return deny("POLICY_ENGINE_FAULT", null)

// ALLOW path, artifact and audit write MUST precede execution
artifact ← generate_artifact(frozen, ALLOW)  // QCC-09
if artifact = null: return deny("ARTIFACT_GENERATION_FAILED", null)

if audit_write(artifact, ALLOW) = FAILURE:   // QCC-10, write-before-execute
  return deny("AUDIT_WRITE_FAILED", null)
```

```
    return allow(artifact)
```

7.4 Deployment Topologies

TOPOLOGY	DESCRIPTION	KEY CONSTRAINT
SaaS Gateway	Hosted enforcement; organizations integrate via API	Contractual evidence-integrity guarantees from service provider; key management and data minimization in transit
On-Premises Gateway	Runs within organization boundary; Consent Registry may be fully internal	Suitable where data locality constraints or regulatory requirements preclude external processing
Edge / API Plugin	Lightweight Policy and token checks at edge with Registry sync via cache	Revocation TTL strictly enforced; cache-stale denial operationally accepted

SECTION 8

Interoperability and Ecosystem Compatibility

8.1 Interoperability Posture

QODIQA Core is implementable by independent vendors, open-source projects, and institutional operators without weakening enforcement semantics. Interoperability does not require shared infrastructure; it requires semantic equivalence of enforcement behavior under the same Frozen Evaluation Context.

Two independent implementations are interoperable at the enforcement layer if and only if: for any valid DecisionRequest and Consent Token pair, both reach the same decision outcome; both produce artifact IDs recomputable from the same frozen context; and both reject the same classes of invalid or incomplete inputs under Fail-Closed semantics.

8.2 Vocabulary Registry Governance

Interoperability at the intent layer depends on shared enumerated vocabularies for purpose codes, action codes, and data type codes. The QODIQA governance authority MUST maintain versioned, published registries for these vocabularies. Implementations MUST declare the vocabulary version in use in their Conformance Statement.

REGISTRY	DESCRIPTION
Purpose Code Registry	Versioned enumeration of permitted purpose identifiers
Action Code Registry	Versioned enumeration of permitted action identifiers; structured as a hierarchical namespace (e.g., `inference.generate_text`)
Data Type Code Registry	Versioned enumeration of data category identifiers
Reason Code Registry	Versioned enumeration of denial reason codes; implementations MUST map internal error states to registered reason codes

8.3 Third-Party Implementation Model

Any party may implement a Conformant QODIQA Core enforcement gateway without license from QODIQA, provided the implementation satisfies all MUST-level requirements and declares a Conformance Profile in accordance with Section 13. Third-party implementations are subject to the same Conformance test suite as any other implementation.

8.4 Compatibility with Existing AI Stacks

QODIQA Core is compatible with vendor LLM APIs, open-source inference servers, agent frameworks, retrieval-augmented generation stacks, and tool-calling runtimes, provided a deterministic Enforcement Gateway can be positioned before model invocation on the critical path. QODIQA Core does not require access to model weights, internals, or training artifacts.

SECTION 9

Adoption Model

9.1 30-Day Integration Path to CORE-MIN

WEEK	ACTIVITIES	CONTROLS ADDRESSED
Week 1	Identify AI call boundaries and insertion point. Define enumerated purpose, action, and data type vocabularies. Establish trust anchor for Consent Token verification.	QCC-01 (vocabulary definition)
Week 2	Implement intent validation and fail-closed handler. Implement Consent Token signature verification with trust anchor. Establish Policy Engine with `default_effect: DENY`.	QCC-01, QCC-02, QCC-08
Week 3	Implement Consent Registry integration and Revocation checks. Implement purpose and scope binding with enumerated code comparison.	QCC-05, QCC-06, QCC-03, QCC-04
Week 4	Implement Deterministic Policy Evaluation and Frozen Evaluation Context. Implement Decision Artifact generation and Audit Log writing. Execute Conformance tests. Produce initial Conformance Statement.	QCC-07, QCC-09, QCC-10

9.2 60-Day Path to CORE-ENT Readiness

Days 31–45: Formalize Policy versioning, Audit Log retention, and Revocation propagation SLAs. Implement jurisdictional overlays if applicable (QCC-11). Define and test the Audit evidence export mechanism.

Days 46–60: Independent replay verification audit of a sample of Decision Artifacts. Document all SHOULD-control deviations. Finalize Conformance Statement signed by the organizational authority. Publish or retain for audit.

SECTION 10

Regulatory Alignment

This section describes technical mechanisms provided by QODIQA Core that may support demonstrability of alignment with existing regulatory frameworks. This section does not interpret law, does not assert legal compliance, and does not constitute legal advice. Organizations are solely responsible for their own compliance assessments.

10.1 GDPR Consent Principles

The General Data Protection Regulation requires Consent to be freely given, specific, informed, and unambiguous; demonstrability of Consent is required; and withdrawal must be as easy as giving Consent. QODIQA Core provides technical mechanisms that support demonstrability of these requirements:

GDPR REQUIREMENT	QODIQA CORE MECHANISM
Purpose specificity	Explicit purpose binding (QCC-03) and scope enforcement (QCC-04) provide technical mechanisms supporting demonstrability of specific Consent and function-creep constraint
Demonstrability	Immutable Decision Artifacts and Replay Verification (QCC-09, QCC-10) provide reproducible evidence that a specific action was authorized under a specific Consent state at a specific time
Withdrawal operability	Revocation enforcement and propagation (QCC-06, QCC-12) ensure that withdrawal takes practical effect within a declared propagation window

10.2 EU AI Act

The EU AI Act imposes record-keeping and logging requirements for high-risk AI systems and deployer obligations related to log retention, transparency, and instructions for use. QODIQA Core provides technical mechanisms supporting demonstrability of alignment: deterministic logs and tamper-evident Audit Trails (QCC-10) provide structured evidence supporting traceability and monitoring obligations; structured Decision Artifacts (QCC-09) provide context supporting deployer documentation requirements.

SECTION 11

Economic Model Overview

11.1 Infrastructure Certification Layer

QODIQA Core functions as an infrastructure certification boundary: organizations that implement it can demonstrate, through reproducible Decision Artifacts and independent Conformance testing, that their AI execution paths operate within deterministic Consent constraints. Conformance is an assurance asset: evidence of operational discipline producible to regulators, auditors, counterparties, and institutional customers.

11.2 Implementation and Certification Operating Models

Licensing: Implementers may license reference implementations, Consent Token libraries, and Conformance testing tooling. Licensing terms are external to this standard and do not affect Conformance obligations.

Certification: Organizations may seek third-party attestation of CORE-VER or CORE-ENT Conformance, focusing on Deterministic Enforcement, Revocation handling, and Audit Replay Verification.

Enforcement-as-a-Service: A hosted gateway may be operated as a shared Enforcement Gateway provided it preserves Fail-Closed semantics, evidence integrity, and the contractual basis for the customer's Audit evidence chain. SaaS operation does not alter Conformance obligations; it transfers operational responsibility under a defined contractual model.

This section is descriptive. It does not project markets, revenues, or adoption timelines, and does not alter the technical requirements of Conformance.

SECTION 12

Governance and Versioning

12.1 Amendment Process

Proposed amendments MUST be submitted as written change requests containing: technical rationale, security implications, interoperability impact, and backward compatibility analysis. Amendments that weaken Fail-Closed semantics, reduce Determinism guarantees, or reduce the normative level of any MUST-level Control Point MUST be treated as breaking changes requiring a MAJOR version increment.

12.2 Version Numbering

INCREMENT	TRIGGER
MAJOR	Breaking changes to Control Point semantics, required interface contracts, or Conformance requirements. Any change weakening Fail-Closed semantics or Determinism guarantees.
MINOR	Backward-compatible additions (new optional fields, new SHOULD-level controls). MUST NOT affect existing MUST-level behavior.
PATCH	Clarifications without behavioral change. No impact on Decision Artifact computation or enforcement logic.

12.3 Governance Authority Responsibilities

- Maintain versioned registries of purpose codes, action codes, data type codes, and reason codes
- Publish reference test vectors for Consent Token signature verification and Decision Artifact hashing
- Publish Conformance test suites covering all MUST-level Control Points with deterministic pass/fail criteria

- Maintain a public attestation registry for CORE-VER and CORE-ENT declarations
- Publish a trust anchor registry at `/.well-known/qodiqa-trust` for federation and third-party verification

SECTION 13

Conformance Statement Template

Conformance Statement: Normative Template — VERBATIM

```
Organization Name:          _______________________________
System Name / Product:      _______________________________
Conformance Profile:        CORE-MIN  /  CORE-VER  /  CORE-ENT
Standard Version:           QODIQA Core v1.0
Standard Identifier:        QODIQA-CORE-1.0-202603
Deployment Topology:        SaaS / On-Premises / Gateway Plugin / Other
Date of Declaration:        _______________________________
Responsible Authority:      _______________________________  (Title / Role)

_____________________________________________
CONFORMANCE DECLARATION
_____________________________________________
We hereby declare that the above-identified system implements the QODIQA Core Standard
for Deterministic Runtime Consent Enforcement (QODIQA-CORE-2026-001) in accordance
with the Conformance Profile specified above, including all MUST-level Control Points
QCC-01 through QCC-10 and the profile-appropriate treatment of QCC-11 and QCC-12.

We affirm that:
  · Consent is enforced deterministically at runtime with Fail-Closed behavior.
  · Consent Token cryptographic verification is performed before every decision.
```

```
  · Revocation state is evaluated at runtime before every decision.
  · Decision Artifacts and Audit Log entries are generated and retained per declared policy.
  · evaluate(context) = replay(context) for all Frozen Evaluation Contexts.

_____________________________________________

DEVIATION DECLARATIONS (SHOULD-level controls)
_____________________________________________

QCC-11 (Jurisdictional Binding):
  [ ] Implemented  [ ] Not Implemented
  Justification: ______________________________________
  Risk Accepted By: _____________ (Title / Role / Date)

QCC-12 (Withdrawal Operability):
  [ ] Implemented  [ ] Not Implemented
  Justification: ______________________________________
  Risk Accepted By: _____________ (Title / Role / Date)

_____________________________________________

OPERATIONAL PARAMETERS
_____________________________________________

  Revocation Propagation SLA (seconds):   ______
  Audit Log Minimum Retention (days):     ______
  Clock Tolerance (seconds):              ______
  Revocation Cache TTL (seconds):         ______
  Policy Version in Effect:               ______
  Ruleset Version in Effect:              ______
  Signature Algorithm:                    ______
  Audit Hash Algorithm:                   ______

_____________________________________________

Authorized Signature:   ______________________________
```

```
Title / Role:          _______________________________
Date:                  _______________________________
```

13.1 Machine-Readable Declaration (RECOMMENDED)

Machine-Readable Conformance Declaration — JSON

```json
{
  "standard":                   "QODIQA-CORE-1.0-202603",
  "org":                        "Example Institution",
  "system":                     "Example AI Enforcement Gateway",
  "profile":                    "CORE-VER",
  "declaration_date":           "2026-03-01T00:00:00Z",
  "implemented_controls": [
    "QCC-01","QCC-02","QCC-03","QCC-04","QCC-05",
    "QCC-06","QCC-07","QCC-08","QCC-09","QCC-10",
    "QCC-11","QCC-12"
  ],
  "deviations": [],
  "policy_version":             "1.0.0",
  "ruleset_version":            "2026-03-01",
  "audit_retention_days":       365,
  "revocation_propagation_sla_s": 300,
  "clock_tolerance_seconds":    60,
  "signature_algorithm":        "Ed25519",
  "audit_hash_algorithm":       "SHA-256",
  "attesting_authority": {
    "role": "CISO",
    "signed_date": "2026-03-01T00:00:00Z"
  }
}
```

13.2 Normative References

REFERENCE	CITATION
[RFC2119]	Bradner, S. "Key words for use in RFCs." BCP 14, RFC 2119, March 1997.
[RFC8174]	Leiba, B. "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words." BCP 14, RFC 8174, May 2017.
[RFC8032]	Josefsson, S., Liusvaara, I. "Edwards-Curve Digital Signature Algorithm (EdDSA)." RFC 8032, January 2017.
[RFC8785]	Rundgren, A. et al. "JSON Canonicalization Scheme (JCS)." RFC 8785, June 2020.
[JSON-LD]	Sporny, M. et al. "JSON-LD 1.1." W3C Recommendation, July 2020.
[DID-CORE]	Sporny, M. et al. "Decentralized Identifiers (DIDs) v1.0." W3C Recommendation, July 2022.

13.3 Informative References

REFERENCE	CITATION
[QODIQA-WP]	QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0
[QODIQA-68]	QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0
[ISO-27001]	ISO/IEC 27001:2013. Information security management systems: Requirements.
[NIST-AIRM]	NIST AI Risk Management Framework 1.0. January 2023.
[GDPR]	Regulation (EU) 2016/679. General Data Protection Regulation. April 2016.

INSTITUTIONAL CLOSING STATEMENT

Institutional Closing Statement

QODIQA Core defines a minimal Consent enforcement layer that is intentionally narrow in scope. It does not interpret law, inspect content semantics, or modify AI model behavior through probabilistic governance. Its function is infrastructural: placing Deterministic Consent enforcement on the runtime execution path as a precondition to AI invocation, and producing reproducible Decision Artifacts that support independent Replay Verification.

This minimalism is not a reduction of the underlying thesis. It is the operational boundary required to make Consent enforceable in complex, heterogeneous AI systems without relying on retrospective narrative, probabilistic judgment, or assumption about prior state. The Frozen Evaluation Context requirement, the Replay Invariant, the Fail-Closed semantics, and the tamper-evident audit architecture constitute the minimum viable infrastructure layer through which Consent graduates from declaration to proof.

WHERE CONSENT MUST BE DEMONSTRABLE, DETERMINISTIC, AND AUDITABLE:

not a statement of intent, but a verifiable runtime fact; a Consent Enforcement Gateway is a necessary technical instrument at every AI execution boundary.

The QODIQA Whitepaper establishes the conceptual and philosophical foundation. The 68-Point Enforcement Framework establishes the full governance architecture. QODIQA Core is the adoptable, testable, and certifiable entry

point into that architecture, the first layer any organization can deploy, and the foundation upon which full institutional Conformance is built.

The runtime Consent enforcement gap is not a problem that resolves through policy refinement, documentation improvements, or organizational discipline. It is a structural defect in AI infrastructure. QODIQA Core is the structural remedy: deterministic, verifiable, and enforceable by design.

DOCUMENT STATUS

Document Status and Classification

This document defines the minimal normative core of the QODIQA framework. It specifies the smallest enforceable set of requirements required to implement deterministic runtime consent enforcement in AI systems, and serves as the foundational Conformance tier from which full 68-Point compliance is constructed.

QODIQA Core is not presented as a theoretical framework, but as an implementable infrastructure standard designed to introduce enforceable authorization, deterministic policy execution, runtime verifiability, and auditable proof mechanisms into AI-native systems without requiring model retraining or infrastructure replacement.

The material contained herein is intended for:

- Systems architects and infrastructure engineers
- Enterprise AI platform designers
- Security and governance officers
- Regulatory and policy stakeholders
- Organizations building or operating autonomous AI systems

All technical principles described in this document are designed to be production-oriented, cryptographically anchored, and architected for integration within large-scale AI ecosystems. No model retraining, weight access, or infrastructure replacement is required for Conformance.

This document should be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Implementation Playbook for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0
- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dușescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-CORE-2026-001
TITLE	Core Standard for Deterministic Runtime Consent Enforcement
SUBTITLE	Minimal Adoptable Runtime Consent Enforcement Layer for AI Systems
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Normative Specification
DOCUMENT STATUS	Normative, Core Standard
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

PART II

II

Normative technical specifications

8 documents

04	68-Point Enforcement Framework	121
05	Reference Architecture	154
06	Security and Cryptographic Profile	202
07	Conformance Test Suite	263
08	Conformance Verification	321
09	Threat Model and Abuse Cases	350
10	Extended Adversarial Threat Model	387
11	Global Enforcement Invariants	422

CHAPTER 04 PART II · NORMATIVE TECHNICAL SPECIFICATIONS

68-Point Enforcement Framework for Runtime Consent Governance

Normative governance framework defining 68 control points for deterministic, machine-verifiable consent enforcement across the Artificial Intelligence system lifecycle.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-FRAMEWORK-2026-003	1.0	2026	Specification	Normative	NORMATIVE

ABSTRACT

The QODIQA 68-Point Enforcement Framework is a normative governance infrastructure specification designed to provide machine-verifiable consent enforcement across the Artificial Intelligence system lifecycle. This document defines a comprehensive set of 68 control points spanning ten governance domains, establishing a protocol-layer mechanism for asserting, verifying, and enforcing data sovereignty, creator rights, and safety constraints.

Unlike voluntary guidelines, this specification provides a rigorous, testable technical standard suitable for implementation in enterprise, research, and open infrastructure environments. It relies on cryptographic tokenization, immutable ledger auditing, and standardized API gateways to ensure that AI systems, from training through runtime inference, operate within explicitly granted consent scopes.

SECTION 1

Introduction

As Artificial Intelligence systems scale in complexity and autonomy, the separation between data provenance, model capability, and output generation has created a governance gap. Traditional legal frameworks rely on post-hoc litigation, which is insufficient for real-time, algorithmic decision-making.

The QODIQA Framework resolves this by introducing a protocol-layer consent infrastructure. Analogous to how TLS (Transport Layer Security) provides a universal security layer for the web, QODIQA provides a universal consent layer for AI. It operates independently of model architecture (Transformer, SSM, Diffusion) and deployment topology (Cloud, Edge, Federated).

> **CORE OBJECTIVE**
>
> The core objective is to render consent machine-verifiable. A system conformant with this specification does not merely "promise" to respect data rights; it is cryptographically unable to process data or generate inference without a valid, signed consent token that explicitly authorizes the specific operation, scope, and timeframe.

SECTION 2

Terminology and Definitions

2.1 Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals.

KEYWORD	NORMATIVE MEANING
MUST / SHALL	Absolute requirement. Failure to satisfy constitutes non-conformance.
MUST NOT / SHALL NOT	Absolute prohibition.
SHOULD / SHOULD NOT	Recommended behavior. Deviations MUST be documented with justification.
MAY / OPTIONAL	Truly optional behavior with no conformance impact.

2.2 Definitions

TERM	DEFINITION
Consent Enforcement Engine	The architectural component responsible for validating consent tokens against the request context and policy constraints at every pipeline stage.
Consent Token	A cryptographically signed data structure (JSON-LD format) containing the subject identifier, scope, expiry, commercial flag, and conditions of the consent grant.
Control Point	A discrete, testable enforcement mechanism identified by a unique identifier (CP-01 through CP-68).
Governance Domain	A logical grouping of related control points (D01 through D10).
Lifecycle Stage	One of six temporal phases of AI system operation: Training, Fine-Tuning, Deployment, Runtime, Output, or Retirement.
Maturity Level	A designation (L0 through L4) indicating the depth and rigor of the governance implementation. L4 represents full conformance.
Subject ID	A W3C Decentralized Identifier (DID) representing the entity granting consent.
Zero-Knowledge Proof	A cryptographic mechanism enabling consent verification without revealing the subject's identity.
Immutable Ledger	A tamper-evident distributed ledger recording all token lifecycle events using hash-chained audit entries and Merkle tree verification.

2.3 Abbreviations

ABBREVIATION	DEFINITION	ABBREVIATION	DEFINITION
DID	Decentralized Identifier	PKI	Public Key Infrastructure
DPA	Data Processing Agreement	TLS	Transport Layer Security
HSM	Hardware Security Module	TTL	Time To Live
JSON-LD	JSON for Linked Data	W3C	World Wide Web Consortium
ZK	Zero-Knowledge		

SECTION 3

Architectural Overview

The QODIQA architecture is defined as a layered system stack where enforcement occurs at multiple boundaries. The system is designed to intercept control flow at critical junctures, including data ingestion, training initiation, API request, and output generation, to verify consent status at each stage.

The architecture consists of three primary layers:



FIGURE 1: LAYERED ARCHITECTURE MODEL: GOVERNANCE DOMAINS DRIVE LIFECYCLE ENFORCEMENT, BACKED BY A CRYPTOGRAPHIC CORE PROVIDING PKI, TOKEN SIGNING, AND IMMUTABLE LEDGER FUNCTIONS.

SECTION 4

Governance Domains (D01–D10)

This section defines the 68 normative control points categorized into ten governance domains. Each control point is discrete, testable, and assigned a unique identifier (CP-01 through CP-68).

D01: Individual and Creator Protection

DOMAIN 01

Individual and Creator Protection

Control Points CP-01 through CP-07 · 7 control points

This domain governs the fundamental rights of data subjects and creators, ensuring that no data enters the ecosystem without explicit authorization.

CP-01: EXPLICIT OPT-IN GATE MUST

Requirement: Systems MUST require verifiable, explicit opt-in for all data ingestion. Default-on, pre-checked, or passive collection mechanisms are prohibited.

Implementation: The system must record a positive affirmation action (e.g., checkbox selection, API consent grant) with timestamp, IP address if applicable, and user agent. Consent MUST be granular and unbundled.

Verification: Audit logs show a timestamped opt-in event linked to each data object. Automated testing MUST verify that data ingestion fails in the absence of an opt-in record.

Failure: If data is processed without a corresponding opt-in event, the system MUST generate a **critical severity** alert.

CP-02: GRANULAR SCOPE BINDING MUST

Requirement: Consent grants MUST be bound to specific scopes (e.g., "training only", "non-commercial inference", "research purposes"). Blanket or ambiguous consent is prohibited.

Implementation: The consent token MUST include a `scope` array with explicit use-case identifiers. Systems MUST validate that the requested operation matches at least one scope. Cross-scope operations require separate consent grants.

Verification: Token inspection reveals distinct scope arrays; system blocks operations outside these scopes with HTTP 403 (Scope Violation).

Failure: Processing data for a purpose not listed in the scope array constitutes a **medium-severity** violation.

CP-03: CREATOR ATTRIBUTION TOKEN MUST

Requirement: A cryptographic token carrying attribution metadata MUST be associated with every creative work ingested into training pipelines.

Implementation: Attribution tokens MUST include creator identifier (DID recommended), work identifier (e.g., DOI, URI), license terms, and attribution text. Systems MUST preserve attribution chains through derivative works.

Verification: Each training record is linked to a valid, signature-verified attribution token. Audit queries MUST trace any output back to source creators.

Failure: Training on creative works without attribution tokens is a **high-severity** violation and may constitute copyright infringement.

TABLE: D01 CONTROL POINT SUMMARY (CP-04 THROUGH CP-07)

ID	CONTROL POINT	LEVEL
CP-04	Revocation Propagation: MUST propagate to all nodes within defined SLA window.	MUST
CP-05	Derivative Use Boundary: MUST enforce constraints on derivative works as defined in the consent token.	MUST
CP-06	Minor Data Prohibition: Processing data from subjects under 18 MUST verify guardian consent or be blocked.	MUST
CP-07	Sensitive Attribute Firewall: Sensitive data categories (biometric, health) MUST require elevated privilege and distinct consent scopes.	MUST

D02: Malicious Use Prevention

DOMAIN 02

Malicious Use Prevention

Control Points CP-08 through CP-14 · 7 control points

Controls designed to mitigate adversarial attacks and prevent the AI system from being used to generate harmful content or deceive users.

ID	CONTROL POINT	LEVEL
CP-08	Intent Classification Filter: Incoming prompts MUST be classified for malicious intent before processing.	MUST
CP-09	Adversarial Prompt Firewall: A firewall MUST intercept known jailbreak or injection patterns.	MUST
CP-10	Deepfake Output Block: Generation of non-consensual synthetic likenesses MUST be blocked at the output layer.	MUST
CP-11	Consent Forgery Detection: The system MUST validate cryptographic signatures on all consent tokens to detect forgery.	MUST
CP-12	Shadow Model Detection: Systems SHOULD detect and block patterns indicative of model distillation or extraction attacks.	SHOULD

ID	CONTROL POINT	LEVEL
CP-13	Cross-Platform Aggregation Block: Data aggregation across platforms without composite consent MUST be prevented.	MUST
CP-14	Model Distillation Abuse Control: Rate limits and output noise SHOULD be applied to mitigate unauthorized distillation.	SHOULD

D03: Infrastructure and Model-Level Enforcement

DOMAIN 03

Infrastructure and Model-Level Enforcement

Control Points CP-15 through CP-21 · 7 control points

Technical controls embedded within the model serving infrastructure and the API layer.

ID	CONTROL POINT	LEVEL
CP-15	API Consent Handshake Protocol: All API sessions MUST initiate with a verified consent handshake.	MUST
CP-16	Scope Validation Engine: A dedicated engine MUST validate request parameters against the token's allowed scope.	MUST
CP-17	Runtime Inference Gate: Inference execution MUST be blocked if a valid, active token is not present.	MUST
CP-18	Dataset Provenance Registry: All training data MUST be logged in an immutable provenance registry.	MUST
CP-19	Model Fingerprinting: Models MUST embed verifiable fingerprints to trace lineage.	MUST
CP-20	Pipeline Integrity Check: The CI/CD pipeline MUST verify consent compliance before deployment.	MUST
CP-21	Compute Isolation Boundary: Tenant data MUST be processed in isolated compute environments.	MUST

D04: Institutional and Enterprise Controls

DOMAIN 04

Institutional and Enterprise Controls

Control Points CP-22 through CP-28 · 7 control points

Organizational policies and mechanisms for compliance, auditing, and third-party management.

ID	CONTROL POINT	LEVEL
CP-22	Organisational Consent Policy Engine: Enterprises MUST implement a central policy engine to manage organizational consent settings.	MUST
CP-23	Role-Based Access Control: Administrative access to consent settings MUST be governed by RBAC.	MUST
CP-24	Third-Party Vendor Consent Chain: Consent obligations MUST flow down to third-party vendors via technical enforcement.	MUST
CP-25	Audit Trail Mandate: All consent decisions MUST generate an immutable, hash-chained audit log.	MUST
CP-26	Data Processing Agreement Linkage: Technical controls MUST be linked to active legal DPAs.	MUST
CP-27	Breach Notification Trigger: Detected consent breaches MUST automatically trigger notification workflows.	MUST
CP-28	Enterprise Consent Dashboard: A centralized view of consent posture SHOULD be provided to administrators.	SHOULD

D05: Model Lifecycle Governance

DOMAIN 05

Model Lifecycle Governance

Control Points CP-29 through CP-35 · 7 control points

Controls specific to the stages of model development, from pre-training through retirement.

ID	CONTROL POINT	LEVEL
CP-29	Pre-Training Consent Audit: A comprehensive audit MUST be passed before training runs initiate.	MUST
CP-30	Fine-Tuning Consent Rebinding: Fine-tuning on new data MUST trigger a re-verification of consent.	MUST
CP-31	Deployment Consent Gate: Deployment pipelines MUST fail if consent criteria are not met.	MUST
CP-32	Runtime Consent Monitoring: Active models MUST be continuously monitored for consent compliance.	MUST
CP-33	Model Retirement Consent Purge: Decommissioned models MUST undergo a verified data purge.	MUST
CP-34	Version Control Consent Binding: Model versions MUST be immutably bound to the consent state of their training data.	MUST
CP-35	Consent Drift Detection: Systems MUST detect and alert on divergence between current consent status and model training basis.	MUST

D06: Output-Layer Enforcement

DOMAIN 06

Output-Layer Enforcement

Control Points CP-36 through CP-42 · 7 control points

Controls applied to the generated artifacts (text, image, audio) produced by the AI system.

ID	CONTROL POINT	LEVEL
CP-36	Output Compliance Filter: Outputs MUST be filtered against active consent constraints prior to delivery.	MUST
CP-37	Attribution Preservation: Attribution metadata MUST be preserved in the output payload.	MUST
CP-38	Commercial Use Flag: Outputs MUST be tagged with a commercial use flag derived from input consents.	MUST

ID	CONTROL POINT	LEVEL
CP-39	Watermarking Enforcement: AI-generated content MUST include robust watermarking.	MUST
CP-40	Synthetic Data Labelling: Synthetic data outputs MUST be explicitly labeled.	MUST
CP-41	Output Redaction Gate: Content violating consent scopes (e.g., PII) MUST be redacted.	MUST
CP-42	Downstream Use Restriction: Restrictions on downstream use SHOULD be embedded in output metadata.	SHOULD

D07: Temporal Governance

DOMAIN 07

Temporal Governance

Control Points CP-43 through CP-49 · 7 control points

Management of consent over time, including expiry, renewal, and retroactive revocation.

ID	CONTROL POINT	LEVEL
CP-43	Consent Expiry Enforcement: Tokens MUST expire automatically upon reaching their `expiry_timestamp`.	MUST
CP-44	Right to Erasure Pipeline: Valid erasure requests MUST trigger a verified deletion pipeline.	MUST
CP-45	Retroactive Revocation Handling: Revocation MUST apply retroactively to active processing sessions.	MUST
CP-46	Consent Renewal Notification: Systems SHOULD notify subjects prior to consent expiry.	SHOULD
CP-47	Stale Consent Invalidation: Consent not refreshed within defined periods MUST be invalidated.	MUST
CP-48	Time-Bound Use Restriction: Usage rights MUST be strictly limited to the time bounds defined in the token.	MUST
CP-49	Temporal Audit Sequencing: Audit logs MUST maintain strict temporal sequencing via hash chains.	MUST

D08: Economic Rights Layer

DOMAIN 08

Economic Rights Layer

Control Points CP-50 through CP-56 · 7 control points

Mechanisms to track value generation and enforce compensation or royalty distribution.

ID	CONTROL POINT	LEVEL
CP-50	Revenue Attribution Tracking: Systems SHOULD track revenue generation events linked to consented data.	SHOULD
CP-51	Royalty Distribution Engine: Automated distribution mechanisms MAY be integrated for royalty payments.	MAY
CP-52	Commercial Licence Enforcement: Commercial use requires a distinct license token; otherwise, it MUST be blocked.	MUST
CP-53	Data Valuation Transparency: Metrics used for data valuation SHOULD be transparent to the data subject.	SHOULD
CP-54	Creator Compensation Gate: Access to monetization features MUST be gated by valid compensation agreements.	MUST
CP-55	Economic Right Revocation: Revocation of economic rights MUST cease future revenue generation.	MUST
CP-56	Compensation Audit Trail: Financial transactions related to data use MUST be audited.	MUST

D09: AI-to-AI Consent Control

DOMAIN 09

AI-to-AI Consent Control

Control Points CP-57 through CP-62 · 6 control points

Governance for autonomous agents and multi-model systems interacting without human intervention.

ID	CONTROL POINT	LEVEL
CP-57	Agent Consent Delegation Protocol: Agents MUST use a standardized protocol to delegate consent in chains.	MUST
CP-58	Inter-Model Data Transfer Gate: Data transfer between models MUST pass a consent gate.	MUST
CP-59	Autonomous Agent Scope Boundary: Agents MUST NOT exceed the scope of their delegated authority.	MUST
CP-60	AI Chain Consent Propagation: Consent context MUST propagate through the entire agent chain.	MUST
CP-61	Model-to-Model Attribution: Source models MUST be attributed in downstream model outputs.	MUST
CP-62	Orchestration Layer Consent Audit: The orchestration layer MUST log all inter-agent consent transactions.	MUST

D10: Cryptographic Consent Layer

DOMAIN 10

Cryptographic Consent Layer

Control Points CP-63 through CP-68 · 6 control points

The foundational cryptographic primitives ensuring the integrity and non-repudiation of the system.

ID	CONTROL POINT	LEVEL
CP-63	Consent Token Issuance: Tokens MUST be issued by a verified authority using standard schemas.	MUST
CP-64	Token Signature Verification: Signatures MUST be verified using Ed25519 (or compliant algorithm) keys.	MUST
CP-65	Revocation Registry Sync: Implementations MUST sync with the revocation registry in real-time or near real-time.	MUST
CP-66	Zero-Knowledge Consent Proof: Systems SHOULD support ZK-proofs to verify consent without revealing identity.	SHOULD

ID	CONTROL POINT	LEVEL
CP-67	Immutable Consent Ledger: A tamper-evident ledger MUST record all token lifecycle events.	MUST
CP-68	Cryptographic Compliance Export: The system MUST be able to export a cryptographically verifiable compliance report.	MUST

SECTION 5

Lifecycle Model

The QODIQA Framework enforces governance across six distinct stages of the AI system lifecycle, from initial training data ingestion through model decommissioning. Each stage is associated with one or more critical control points that MUST be satisfied before the stage may proceed.

STAGE	DESCRIPTION	CRITICAL CONTROL POINT
Training	Initial model development.	CP-29: Pre-Training Consent Audit
Fine-Tuning	Refinement on specific datasets.	CP-30: Consent Rebinding
Deployment	Integration into production.	CP-31: Deployment Consent Gate
Runtime	Active inference.	CP-17 / CP-32: Inference Gate and Monitoring
Output	Generation of results.	CP-36: Output Compliance Filter
Retirement	Decommissioning.	CP-33: Model Retirement Consent Purge

SECTION 6

Cryptographic Layer and Protocols

The integrity of the QODIQA Framework relies on robust cryptographic primitives applied consistently across all enforcement boundaries. This section specifies the mandatory token format, approved algorithm suite, and algorithm status classifications.

6.1 Token Format

Consent tokens **MUST** use the JSON-LD format. The `subject_id` **SHOULD** be a W3C Decentralized Identifier (DID).

CONSENT TOKEN: JSON-LD SCHEMA

```
{
  "@context":   "spec/qodiqa/1.0/consent-context.jsonld",
  "spec":       "QODIQA/1.0",
  "subject_id": "did:web:subject.example.org",
  "scope":      ["training", "non-commercial"],
  "expiry":     "2026-12-31T23:59:59Z",
  "commercial": false,
  "signature":  "<Base64url-encoded Ed25519 signature>"
}
```

6.2 Algorithms

ALGORITHM	USE CASE	STATUS
Ed25519	Token signing (primary)	RECOMMENDED
ECDSA P-256	Token signing (alternate)	PERMITTED

ALGORITHM	USE CASE	STATUS
RSA-PSS	Token signing (legacy)	NOT RECOMMENDED
SHA-256	Audit chain hashing	REQUIRED
SHA-512	Audit chain hashing	RECOMMENDED

SECTION 7

Protocol Flow (Six-Gate Sequence)

The enforcement logic follows a linear sequence of six gates. Each gate represents a discrete enforcement boundary where consent validation occurs. Failure at any gate MUST produce the defined failure response and MUST prevent progression to subsequent gates.

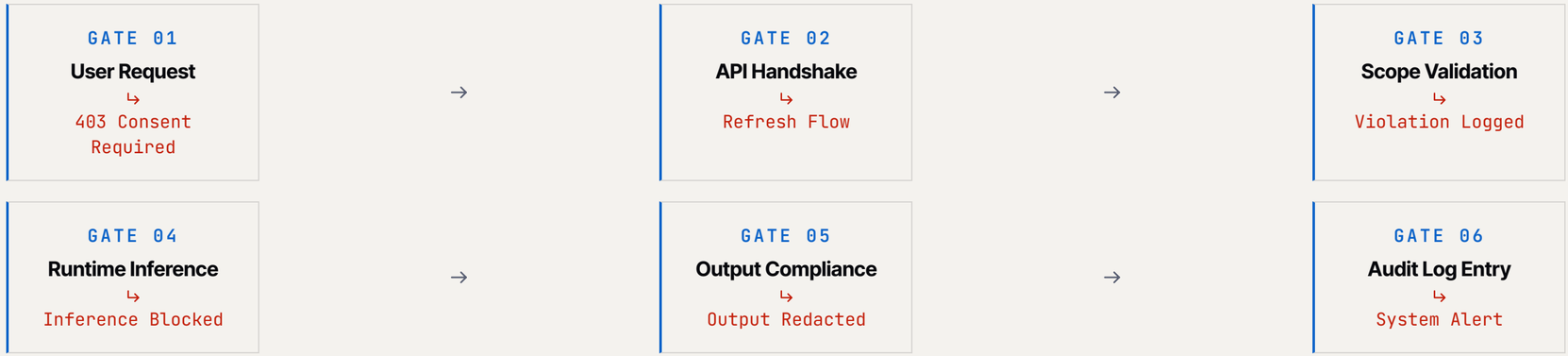


FIGURE 2: SIX-GATE ENFORCEMENT SEQUENCE: EACH GATE CONSTITUTES AN INDEPENDENT ENFORCEMENT BOUNDARY. FAILURE AT ANY GATE BLOCKS PROGRESSION AND TRIGGERS THE DEFINED FAILURE RESPONSE.

TABLE: GATE REFERENCE

GATE	NAME	FUNCTION	FAILURE STATE
Gate 01	User Request	Inbound API call arrival	403 Consent Required
Gate 02	API Consent Handshake	Token exchange and signature verification	Refresh Flow Initiated
Gate 03	Scope Validation	Check request against token scope	Violation Logged
Gate 04	Runtime Inference	Execution of model inference	Inference Blocked
Gate 05	Output Compliance	Filter generated content	Output Redacted
Gate 06	Audit Log Entry	Write event to ledger	System Error / Alert

SECTION 8

Threat Model

8.1 Threat Assumptions

- Attackers may have access to API endpoints and attempt to forge consent tokens.
- Legitimate tokens may be captured and replayed by adversaries.
- Model operators may intentionally or inadvertently bypass consent controls.
- Cross-jurisdictional deployments may create enforcement gaps.
- Cryptographic keys may be compromised through social engineering or technical exploits.

8.2 Threat Severity Classification

Critical Threats: Fundamental consent violations requiring immediate enforcement.

THREAT	DESCRIPTION	INTERCEPTION	CONTROL POINTS
Consent Forgery	Fraudulent token generation or signature spoofing	Gate 02	CP-11, CP-63, CP-64
Biometric Identity Abuse	Synthetic biometric generation without explicit consent	Gates 04–05	CP-10
Minor Data Processing	Processing data from subjects under 18 without verified guardian consent	Gates 01–03	CP-06

High Severity: MUST be logged and escalated within 1 hour.

THREAT	DESCRIPTION	INTERCEPTION	CONTROL POINTS
Token Replay Attack	Reuse of captured tokens beyond their validity period	Gate 02	CP-63 CP-64
Shadow Model Replication	Distillation-based reproduction of model capabilities	Gate 04	CP-12 CP-19
Model Distillation Abuse	Systematic capability extraction via query harvesting	Gate 05	CP-14 CP-36
Cross-Platform Aggregation	Combining consented data across platforms without composite consent	Gates 01–03	CP-13
Revocation Propagation Failure	Failure to sync revocation events across enforcement nodes	All Gates	CP-04 CP-65

Medium Severity: SHOULD trigger automated remediation within 24 hours.

THREAT	DESCRIPTION	CONTROL POINTS
Stale Consent Usage	Consent applied beyond defined expiry or inactivity threshold	CP-43, CP-47
Derivative Use Boundary Breach	Outputs used outside of the consented scope	CP-05

THREAT	DESCRIPTION	CONTROL POINTS
Economic Rights Non-Compliance	Commercial deployment without creator compensation plan	CP-54

8.3 Known Attack Classes

ATTACK VECTOR	INTERCEPTION GATE	PRIMARY CONTROL POINTS
Prompt Injection	Gate 03 (Scope Validation)	CP-09
Consent Forgery	Gate 02 (API Handshake)	CP-11, CP-63, CP-64
Shadow Models	Gate 04 (Runtime Inference)	CP-12, CP-19
Distillation Abuse	Gate 05 (Output Compliance)	CP-14, CP-36

8.4 Mitigations

MITIGATION	REQUIREMENT	CONTROL LEVEL
Cryptographic Binding	All tokens MUST be signed using Ed25519 or equivalent algorithms.	MUST
Nonce / Sequence Enforcement	Replay protection MUST use monotonic sequence numbers or time-bound nonces.	MUST
Real-Time Registry Sync	Revocation registries MUST be synchronized across all nodes within defined SLA windows.	MUST
Hardware-Backed Key Storage	Signing keys SHOULD be protected using Hardware Security Modules (HSMs).	SHOULD
Zero-Knowledge Proofs	ZK mechanisms SHOULD be deployed to minimize identity exposure (required for L3 and above).	SHOULD

8.5 Residual Risks

The following risks remain outside the scope of this framework and are acknowledged as conditions that cannot be eliminated by architectural or cryptographic means alone:

- Compromise of the root cryptographic authority
- Nation-state attacks on the underlying cryptographic primitives
- Legal jurisdiction conflicts that cannot be resolved by technical means
- Social engineering attacks targeting individual consent capture mechanisms

SECTION 9

Conformance and Profiles

9.1 Conformance Requirements

Implementations claiming conformance to this specification MUST satisfy all MUST-level requirements within their declared conformance profile. Conforming implementations MUST declare: profile adherence, maturity level, supported cryptographic algorithms, and SLA windows for revocation propagation.

9.2 Conformance Profiles

PROFILE	CONTEXT	REQUIRED MATURITY	MUST DOMAINS
PRF-ENT	Commercial, regulated, enterprise	Level 3–4	D01–D07
PRF-RES	Academic, research, non-commercial	Level 2 minimum	D01–D05 (partial)
PRF-OPN	Open-source, public APIs, federated	Level 2–3	D01–D06

9.3 Conformance Statement Format

MACHINE-READABLE CONFORMANCE STATEMENT: JSON-LD

```
{
  "@context":      "spec/qodiqa/1.0/conformance-context.jsonld",
  "specification": "QODIQA-FRAMEWORK-2026-003",
  "profile":       "PRF-ENT",
  "maturity_level":"L4",
  "control_points": {
    "implemented": ["CP-01", "CP-02", ..., "CP-68"],
    "deviations":  []
  },
  "signature_algorithm": "Ed25519",
  "revocation_sla":    "300s",
  "attestation": {
    "authority": "did:web:attestor.example.org",
    "issued":    "2026-03-01T00:00:00Z",
    "expires":   "2027-03-01T00:00:00Z"
  }
}
```

SECTION 10

Maturity Model

Governance capability is measured on a five-point scale (L0 through L4). Organizations SHOULD progress through levels sequentially, with each level building on the enforcement controls of the preceding level.

L0
NO CONSENT
Non-conformant. Zero control points implemented.

L1
LOGGED
Informal logging; manual revocation. CP-01, CP-25.

L2
ENFORCED
API-layer enforcement; automated revocation. CP-01 to CP-36.

L3
VERIFIABLE
Signed tokens, ZK-proofs, immutable ledger. Minimum for PRF-ENT.

L4
ATTESTED
Full 68-point conformance; independent certification. CP-01 to CP-68.

TABLE: MATURITY LEVEL IMPLEMENTATION TIMELINE

LEVEL	DOMAINS COVERED	CONTROL POINTS	TYPICAL TIMELINE
L0	None	0	N/A
L1	D01–D04 (partial)	CP-01, CP-25	1–3 months
L2	D01–D06	CP-01 to CP-36	6–12 months
L3	D01–D09	CP-01 to CP-62	12–24 months
L4	D01–D10 (all)	CP-01 to CP-68	24–36 months

SECTIONS 11-12

Interoperability and Attestation

11. Interoperability Framework

To ensure cross-system compatibility, implementations MUST support the QODIQA Federation API. Trust anchors MUST be published at `/.well-known/qodiqa-trust`. In conflict scenarios, the Most Restrictive Scope rule applies: the narrowest defined scope governs, and broader scope application requires explicit additional authorization.

12. Attestation Model

Independent verification is required for L4 maturity. Attestations MUST be recorded in a public attestation registry. Systems MUST generate machine-readable compliance exports (JSON-LD) to facilitate external audit.

> Attestation bodies MUST be independent of the implementation vendor, verify all 68 control points for full conformance, conduct verification without accessing personal data, and publish attestation records in the public registry. Re-certification is required annually.

SECTION 13

Security Considerations

13.1 Token Security

Critical: Improper token validation enables consent forgery. Implementations MUST verify Ed25519 (or equivalent) signatures against the issuing authority PKI before every processing decision (CP-11, CP-63, CP-64). Signature verification MUST include cryptographic signature validation, certificate chain verification to trusted root, certificate revocation status check, and timestamp validity window verification.

High: Tokens captured in transit MUST NOT be reusable. Implementations MUST enforce a nonce or monotonic sequence number field and verify freshness within a defined tolerance window (typically 300 seconds).

High: Consent token signing keys SHOULD be protected via Hardware Security Modules. Key rotation MUST be performed on a documented schedule (recommended: quarterly). Compromised keys MUST trigger immediate revocation of all tokens issued under that key within 1 hour from detection.

13.2 Revocation and Propagation

Critical: Failure to propagate revocation events enables continued unauthorized processing. Revocation registries MUST be synchronized across all enforcement nodes within defined SLA windows (CP-65).

SYSTEM TYPE	SLA WINDOW
Critical infrastructure	30 seconds
Enterprise systems	300 seconds (5 minutes)
Research systems	3600 seconds (1 hour)

13.3 Audit Integrity

Audit logs MUST be tamper-evident and hash-chained. Each audit entry MUST contain: previous entry hash (forming a chain), timestamp in ISO 8601 format with timezone, event type and associated control point, subject identifier (pseudonymized), and cryptographic signature of the entry.

13.4 Federation Security

Federation trust anchors MUST be rotated periodically (recommended: annually). Federated consent tokens MUST carry a cryptographic binding to the originating authority. Systems MUST reject federated tokens that cannot be verified against a published trust anchor at `/.well-known/qodiqa-trust`.

13.5 Lifecycle Security

High: Partial lifecycle enforcement weakens runtime guarantees. Implementing only deployment and runtime gates without training-stage controls (CP-29) is a non-conformant posture. CP-33 (Model Retirement Consent Purge) MUST be treated as a MUST-level control regardless of profile.

13.6 Implementation Security Notes

- TLS 1.3 or higher MUST be used for all API communications
- Rate limiting SHOULD be implemented to prevent denial-of-service attacks
- Token storage MUST NOT use plaintext formats; encryption at rest is REQUIRED
- Logging infrastructure SHOULD be segregated from primary application infrastructure

SECTION 14

Privacy Considerations

14.1 Data Minimization

The QODIQA consent token structure is designed to minimize personal data exposure. Tokens MUST NOT contain raw personal data. The `subject_id` SHOULD reference a pseudonymous or DID-based identifier. Audit log exports MUST exclude raw personal data unless explicitly required by applicable law. The default retention period is 7 years.

14.2 Identity and Linkability

Implementations SHOULD support zero-knowledge proof mechanisms (CP-66) to enable consent verification without exposing subject identity. ZK-proof support is REQUIRED for implementations operating at Maturity Level 3 or above. Implementations MUST NOT use consent tokens or audit records to construct cross-system identity profiles of data subjects.

14.3 Cross-Border Obligations

Cross-border federation MUST respect subject jurisdiction precedence as defined in Conflict Resolution Rule R-05. Where the subject's jurisdiction imposes stricter data protection requirements than the processing jurisdiction, the

subject's jurisdiction requirements SHALL apply in full.

14.4 Erasure and Portability

Consent revocation triggers the right-to-erasure pipeline (CP-44). Erasure MUST be completed within the applicable regulatory timeframe (typically 30 days for GDPR-covered entities). Systems MUST provide cryptographic attestation of erasure completion (CP-33). Machine-readable compliance exports (CP-68) MUST support GDPR Article 20 data portability where applicable.

14.5 Privacy-by-Design Principles

PRINCIPLE	IMPLEMENTATION
Proactive not reactive	Consent enforcement occurs before processing, not after
Privacy as default	No processing without explicit token (CP-01)
Embedded into design	Consent layer integrated at protocol level
End-to-end security	Protection across all lifecycle stages
Visibility and transparency	Audit logs and compliance exports
Respect for user privacy	ZK-proofs and pseudonymization

SECTION 15

Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997. https://www.rfc-editor.org/info/rfc2119

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, May 2017. https://www.rfc-editor.org/info/rfc8174

[DID-CORE] Sporny, M., et al., "Decentralized Identifiers (DIDs) v1.0", W3C Recommendation, 19 July 2022. https://www.w3.org/TR/did-core/

[JSON-LD] Sporny, M., et al., "JSON-LD 1.1", W3C Recommendation, 16 July 2020. https://www.w3.org/TR/json-ld11/

[RFC8032] Josefsson, S., Liusvaara, I., "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, January 2017. https://www.rfc-editor.org/info/rfc8032

[RFC6979] Pornin, T., "Deterministic Usage of DSA and ECDSA", RFC 6979, August 2013. https://www.rfc-editor.org/info/rfc6979

Informative References

[GDPR] European Union, "General Data Protection Regulation (GDPR)", Regulation (EU) 2016/679, 27 April 2016.

[ISO-27001] ISO/IEC 27001:2013, "Information technology · Security techniques · Information security management systems · Requirements."

[NIST-AI-RMF] NIST, "Artificial Intelligence Risk Management Framework (AI RMF 1.0)", January 2023.

[W3C-VC] Sporny, M., et al., "Verifiable Credentials Data Model v1.1", W3C Recommendation, 3 March 2022. https://www.w3.org/TR/vc-data-model/

APPENDICES A–E

Normative Annexes

Annex A: Consent Lifecycle State Machine

STATE	DESCRIPTION	ALLOWED TRANSITIONS
ISSUED	Token generated and PKI signed	→ ACTIVE
ACTIVE	Validated and in effect	→ RESTRICTED, → REVOKED, → EXPIRED
RESTRICTED	Scope narrowed by override	→ REVOKED
REVOKED	Explicit withdrawal (terminal)	→ ARCHIVED
EXPIRED	Expiry timestamp exceeded	→ ARCHIVED
ARCHIVED	Retained for audit (minimum 7 years)	(terminal state)

All state transitions MUST generate immutable, hash-chained audit entries. No transition may be reversed; all revocations are permanent.

Annex B: Attack Interception Overlay

ATTACK VECTOR	INTERCEPTION GATE	PRIMARY CONTROL POINTS
Prompt Injection	Gate 03 (Scope Validation)	CP-09
Consent Forgery	Gate 02 (API Handshake)	CP-11, CP-63, CP-64
Shadow Model Replication	Gate 04 (Runtime Inference)	CP-12, CP-19

ATTACK VECTOR	INTERCEPTION GATE	PRIMARY CONTROL POINTS
Distillation Abuse	Gate 05 (Output Compliance)	CP-14, CP-36

Annex C: Conflict Resolution Rules

- **R-01 (Revocation Supersedes):** Any revocation event cancels all prior grants.
- **R-02 (Most Restrictive Scope):** The narrowest defined scope governs.
- **R-03 (Subject Jurisdiction Precedence):** Subject's local requirements take precedence in cross-border conflicts.
- **R-04 (Explicit Override Required):** Broader scope application requires explicit authorization.

Annex D: Cryptographic Algorithms Reference

ALGORITHM	USE CASE	STATUS
Ed25519	Token signing (primary)	RECOMMENDED
ECDSA P-256	Token signing (alternate)	PERMITTED
RSA-PSS	Token signing (legacy)	NOT RECOMMENDED
SHA-256	Audit chain hashing	REQUIRED
SHA-512	Audit chain hashing	RECOMMENDED

Annex E: Scope Boundaries

In Scope: Machine-verifiable consent enforcement infrastructure, cryptographic guarantees of consent validity, protocol-layer consent mechanisms independent of model architecture, cross-jurisdictional consent management.

TABLE: EXPLICIT EXCLUSIONS

ID	EXCLUSION
OUT-01	Model Intelligence Regulation: Does not regulate cognitive capabilities, reasoning quality, or emergent intelligence properties.
OUT-02	Ideological Governance: Does not prescribe political, ethical, philosophical, or values-based constraints on AI behavior.
OUT-03	Regulatory Authority Replacement: Does not replace, supersede, or substitute for national or supranational regulatory frameworks.
OUT-04	Output Quality Guarantee: Does not guarantee accuracy, reliability, completeness, or fitness for purpose of AI-generated outputs.
OUT-05	Bias Elimination: Does not claim to detect, mitigate, measure, or eliminate statistical or representational bias in model outputs.
OUT-06	Sovereign Legal Override: Does not override or take precedence over applicable sovereign legal frameworks in any jurisdiction.

This document presents a normative governance framework and does not constitute legal, regulatory, or operational security advice. Implementations claiming conformance must conduct deployment-specific assessments accounting for their specific topology, operational context, and applicable regulatory obligations.

DOCUMENT STATUS

Document Status and Corpus Alignment

This document is part of the QODIQA specification corpus and defines the normative enforcement framework governing deterministic runtime consent enforcement systems. It specifies the complete set of 68 control points, conformance profiles, lifecycle model, cryptographic requirements, and governance obligations required of implementations claiming QODIQA conformance.

The framework is intended as a machine-implementable technical standard rather than a voluntary guideline. It operates at the protocol layer, independently of model architecture and deployment topology, providing deterministic enforcement mechanisms across all phases of the AI system lifecycle, from training data ingestion through runtime inference and model retirement.

This document should be read together with the following related specifications:

- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement
- QODIQA: Terminology and Normative Definitions
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement

- QODIQA: Threat Model and Abuse Case Specification
- QODIQA: Residual Risk and Assumption Disclosure Annex

Version 1.0 represents the initial corpus-normalized release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Duțescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-FRAMEWORK-2026-003
TITLE	68-Point Enforcement Framework for Runtime Consent Governance
SUBTITLE	Deterministic Runtime Consent Enforcement for Artificial Intelligence Systems
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Normative Governance Framework
DOCUMENT STATUS	Normative
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 05 PART II · NORMATIVE TECHNICAL SPECIFICATIONS

Reference Architecture for Deterministic Runtime Consent Enforcement

Canonical, vendor-neutral architectural topology for deterministic runtime consent enforcement.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-ARCH-2026-001	1.0	2026	Reference Architecture	Institutional	NORMATIVE

ABSTRACT

This document defines the canonical, vendor-neutral architectural topology for deterministic runtime consent enforcement as specified in the QODIQA corpus. It translates the normative requirements of the Core Standard, the 68-Point Enforcement Framework, the Security and Cryptographic Profile, and the Implementation Playbook into a structurally coherent reference model of components, data flows, trust boundaries, failure modes, and scaling properties.

This document does not prescribe implementation technology, vendor choice, or product design. It establishes the logical architecture (the arrangement, isolation, and interaction of components) within which any conformant implementation must operate. **The governing constraint is structural: the architecture is the enforcement boundary, and the enforcement boundary must be deterministic, fail-closed, and cryptographically anchored or it is not a conformant QODIQA architecture.**

> AI systems do not enforce consent. This architecture introduces a deterministic enforcement boundary before inference execution. Every request is evaluated, cryptographically bound, and audit-committed prior to execution. In the absence of a valid enforcement decision, execution does not occur.

SECTION 1

Purpose and Scope

The QODIQA corpus defines what must be enforced and how conformance is measured. This Reference Architecture defines the logical structure within which enforcement occurs. It is the structural complement to the normative corpus: where the Core Standard defines required behavior, this document defines the architectural form within which that behavior is realized.

A reference architecture is a vendor-neutral, technology-neutral canonical model. It defines the minimum structural elements, their responsibilities, their relationships, and the invariants that must hold across all of them, regardless of how any individual element is implemented. Implementations may vary in technology, topology, and scale; they may not vary in the structural properties this architecture specifies. AI systems operating without deterministic consent enforcement lack a verifiable enforcement boundary and cannot produce auditable guarantees of consent compliance.

1.1 Relationship to the Core Standard

The Core Standard is normative over enforcement semantics (what decisions are made and when). This Reference Architecture is normative over the structural properties that make those decisions possible (how components are arranged, isolated, and connected). The Core Standard governs the decision function; this architecture governs the decision infrastructure.

1.2 Relationship to the Implementation Playbook

The Implementation Playbook defines how to build a conformant deployment from this architecture. This document defines what must be built. The Playbook is operationally specific; this document is logically abstract. A deployment that satisfies the Playbook necessarily satisfies this architecture; a deployment that satisfies this architecture but not the Playbook may be architecturally valid but operationally non-conformant.

1.3 Relationship to the Security Profile

The Security and Cryptographic Profile defines the cryptographic properties of the key hierarchy, artifact generation, and audit chain integrity. This architecture defines where those cryptographic properties are applied: at which component boundaries, in which directions, and with which isolation requirements. The Security Profile governs the cryptographic specification; this architecture governs the cryptographic topology.

1.4 Non-Product Positioning Statement

> THIS REFERENCE ARCHITECTURE IS: AND IS NOT
>
> - A logical component model defining canonical responsibilities and boundaries
> - A structural reference against which implementations are assessed for architectural conformance
> - A technology-neutral specification applicable to any conforming implementation
> - A normative reference for component interaction, trust assumptions, and failure behavior
> - Not a product design, API specification, or deployment blueprint
> - Not a prescriptive technology stack or vendor reference
> - Not a commercial architecture, SaaS design, or infrastructure product
> - Not a replacement for enterprise IAM, data governance, or model management infrastructure
>
> This document is a logical architecture model published as part of the QODIQA technical standards corpus.

SECTION 2

Architectural Principles

Eight architectural principles govern the QODIQA reference architecture. Each is stated as a structural invariant: a property the architecture must satisfy at all times, not a design aspiration. An architecture that violates any of these principles is not a conformant QODIQA architecture, regardless of the correctness of individual components.

AP-01: DETERMINISTIC DECISION BOUNDARY

The enforcement decision function is a pure function of its inputs: frozen context, token state at evaluation time, and frozen policy version. For identical inputs, the function must return an identical output on every invocation, across every Policy Engine instance, after every restart, and across all time. No implementation detail (execution order, cache population, thread scheduling, or network timing) may cause the decision function to return different results for identical inputs.

Structural requirement: evaluation isolation; no shared mutable state in decision path.

AP-02: FAIL-CLOSED EXECUTION

The default state of the enforcement system is denial. Any condition that prevents the system from producing a verified, audit-committed enforcement decision (component unavailability, network partition, ledger write failure, signing key unavailability, timeout expiry) must result in denial of the request and must not result in inference execution. There is no degraded mode in which enforcement is relaxed. The absence of a successful enforcement decision is itself an enforcement outcome: deny.

Structural requirement: circuit breakers at all external component interfaces; AL write before inference release.

AP-03: RELEASE SEPARATION OF CONSENT STATE AND POLICY LOGIC

Consent state (the current validity of a specific consent grant) and policy logic (the rules that determine what a consent grant authorizes) must be managed by structurally separate components with independent storage, independent lifecycle, and independent deployment. A change to consent state must not require a policy redeployment. A change to policy logic must not alter existing consent state. The two domains must be logically decoupled and can only be combined at evaluation time within the Policy Engine.

Structural requirement: Consent Registry and Policy Engine are separate components; no shared persistent storage.

AP-04: CRYPTOGRAPHIC ANCHORING

Every enforcement decision artifact must be cryptographically bound to the specific enforcement context that produced it. The artifact must be infeasible to generate without access to the current signing key and must be verifiable against the context without re-executing the enforcement decision. The cryptographic binding must make it structurally impossible for the inference system to execute on an artifact that was not produced by an authorized enforcement decision for the specific request context.

Structural requirement: HMAC-SHA256 artifact generation; three-tier key hierarchy; artifact verification at inference boundary.

Security implication: prevents artifact replay across contexts; an artifact issued for one enforcement context cannot authorize execution under any other.

AP-05: IMMUTABLE AUDIT TRACE

Every enforcement decision must produce a durable, tamper-evident audit record before the corresponding inference execution may proceed. The audit record is committed before the artifact is released; the artifact is released before inference execution begins. This sequence is mandatory and non-negotiable. An audit record that cannot be verified for integrity is equivalent to no audit record.

Structural requirement: write-before-execute invariant; append-only ledger; hash chain integrity.

Security implication: the write-before-execute invariant ensures no inference execution occurs without a prior durable audit record.

AP-06: REPLAY VERIFIABILITY

Any enforcement decision recorded in the audit ledger must be independently reproducible from the archived frozen context and the corresponding frozen policy snapshot, without re-executing the original request. Replay verifiability enables post-hoc audit, determinism certification, and dispute resolution. The architecture must preserve all information required for deterministic replay for the duration of the declared retention period.

Structural requirement: frozen context archived in AL; policy snapshot archived and retrievable; determinism test protocol operable against historical records.

AP-07: ZERO IMPLICIT TRUST BETWEEN COMPONENTS

No component in the enforcement architecture implicitly trusts any other component. All inter-component communication must be authenticated. All data received from another component must be verified before use. A component that has been compromised must not be able to cause another component to produce an incorrect enforcement outcome by sending it crafted data. Trust is established through cryptographic authentication at every interface, not through network position or service identity alone.

Structural requirement: mTLS on all enforcement-path interfaces; signed payloads; verification before use.

AP-08: CONSENT STATE SUPERSESSION

A revocation event, once committed to the Consent Registry, supersedes all prior consent grants for the affected consent_id. No cached state, in-flight decision, or previously issued artifact may authorize inference execution for a revoked consent after the revocation has propagated within the declared TTL. The architecture must ensure that revocation is structurally irreversible within the enforcement path: no component may restore a revoked grant without an explicit re-grant operation.

Structural requirement: linearizable revocation reads; bounded TTL propagation; conservative mode on propagation delay.

SECTION 3

Canonical Component Model

The QODIQA reference architecture defines nine logically separate components. Each component has a distinct responsibility domain, a defined trust boundary, specified inputs and outputs, explicit security assumptions, and a defined failure behavior. No two components may share persistent state. Components may communicate only through defined interfaces; side channels are non-conformant.

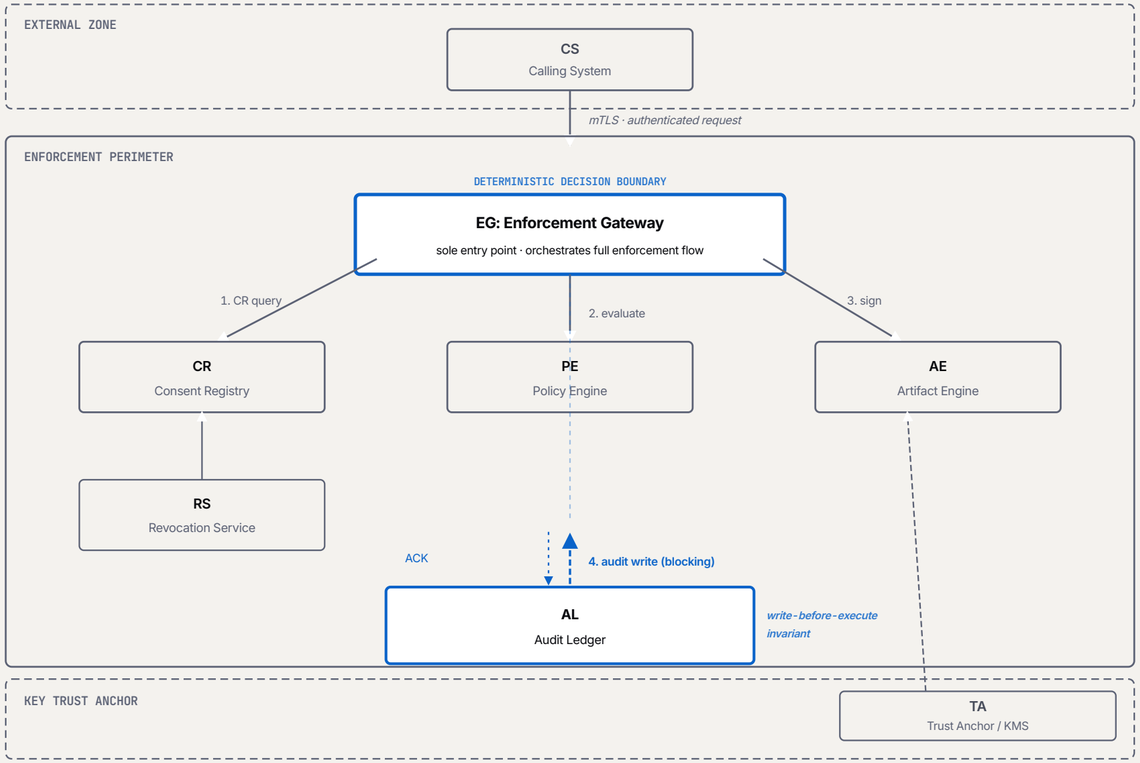


FIGURE 3.1: QODIQA CANONICAL ENFORCEMENT ARCHITECTURE. THE ENFORCEMENT GATEWAY IS THE SOLE ENTRY POINT AND DETERMINISTIC DECISION BOUNDARY; ALL DOWNSTREAM COMPONENTS OPERATE WITHIN THE TRUSTED ENFORCEMENT PERIMETER. THE AUDIT LEDGER WRITE IS BLOCKING: NO ARTIFACT IS RELEASED UNTIL ACKNOWLEDGED.

CS: CALLING SYSTEM

PROPERTY	DETAIL
Responsibilities	Initiates inference requests; presents consent_id; provides frozen context hash; receives enforcement decision artifact; presents artifact to inference system boundary. Does not evaluate policy. Does not access Consent Registry directly.
Trust Boundary	External to the enforcement perimeter. Treated as untrusted by all internal components. Must authenticate to Enforcement Gateway via mTLS or equivalent. Cannot be assumed to have received an uncorrupted enforcement artifact.
Inputs	Inference request payload; consent_id; context parameters used to construct frozen context hash.
Outputs	Authenticated enforcement request to EG; artifact to inference system boundary (if PERMIT returned).
Security Assumptions	CS is not trusted with enforcement logic. A compromised CS may attempt to replay artifacts, fabricate consent_ids, or bypass the EG. The architecture must be secure against all three attack vectors at the EG layer.
Failure Behavior	CS failure has no effect on enforcement infrastructure. A CS that becomes unavailable cannot affect the enforcement state of any consent_id. Crashed or compromised CS instances must not be able to preserve or replay prior PERMIT decisions.

EG: ENFORCEMENT GATEWAY

PROPERTY	DETAIL
Responsibilities	Single structural entry point for all inference requests. Extracts and validates consent_id and context_hash. Orchestrates the enforcement flow: CR query → PE evaluation → AE artifact generation → AL write → artifact release. Enforces rate limits. Operates circuit breakers on all downstream components.
Trust Boundary	Internal enforcement perimeter. Trusted by PE, AE, CR, and AL to present authenticated, well-formed enforcement requests. Does not trust CS. Does not expose any internal component interface to CS. Is the sole structural chokepoint between the external domain and the enforcement infrastructure.
Inputs	Authenticated inference request from CS; CR revocation state (query response); PE decision (PERMIT or DENY); AE artifact (if PERMIT); AL write confirmation.
Outputs	CR query; PE evaluation request; AE signing request; AL write request; enforcement decision (PERMIT and artifact, or DENY) to CS.
Security Assumptions	EG is the only component that communicates with CS. All other components can assume they receive only EG-originated requests. EG operates in a trust-verified context. A compromised EG is a complete breach of the enforcement perimeter; redundant EG instances do not reduce single-instance trust requirements.

PROPERTY	DETAIL
Failure Behavior	EG failure: all requests denied at the transport layer. No fallback path. EG restart must re-verify all component connectivity before resuming enforcement. EG must not resume after restart without a fresh CR read for any in-flight consent_id.

PE: POLICY ENGINE

PROPERTY	DETAIL
Responsibilities	Evaluates the enforcement decision: given a frozen context and current token state, applies the frozen policy snapshot to produce a deterministic PERMIT or DENY. Holds a verified, immutable copy of the active policy snapshot. Does not query external state during evaluation. Does not write to any persistent store.
Trust Boundary	Internal enforcement perimeter; trusted only by EG. Does not trust data from any source other than the EG enforcement request (which carries the frozen context hash) and its own local policy snapshot (which is verified at load time). Stateless at evaluation time; all evaluation state is confined to the single request scope.
Inputs	Enforcement request from EG: (consent_id, context_hash, token_state, policy_version); verified local policy snapshot.
Outputs	Deterministic enforcement decision: PERMIT or DENY, with decision reason code and policy_version binding, returned to EG.
Security Assumptions	PE must not be reachable by any component other than EG. PE must not make network calls during evaluation. PE policy snapshot must be verified against the AL POLICY_FROZEN record at load time. A compromised PE that produces incorrect decisions is a structural breach; multiple PE instances do not eliminate this risk: each instance must independently verify its snapshot.
Failure Behavior	PE unavailability: EG circuit breaker opens; all subsequent requests denied with PE_UNAVAILABLE until PE recovers and snapshot re-verifies. PE must not enter a partially initialized state from which it accepts evaluation requests without a verified policy snapshot.

CR: CONSENT REGISTRY

PROPERTY	DETAIL
Responsibilities	Authoritative store of consent grant state. Responds to EG queries with the current state of a specified consent_id: ACTIVE, REVOKED, EXPIRED, or NOT_FOUND. Accepts revocation writes from the Revocation Service. Provides linearizable reads for revocation state. Maintains replication consistency across all deployment domains within the declared TTL.
Trust Boundary	Internal enforcement perimeter. Accepts queries only from authenticated EG instances. Accepts revocation writes only from the authenticated Revocation Service. Rejects all other write attempts. Read responses are considered trusted by EG within the declared cache TTL.

PROPERTY	DETAIL
Inputs	Consent_id query from EG; revocation event from RS.
Outputs	Consent state response to EG; replication events to peer CR instances (in replicated deployments).
Security Assumptions	CR holds the authoritative revocation state. A CR that serves stale or incorrect state is a safety failure. CR must use linearizable reads for revocation checks: eventually consistent reads are non-conformant for revocation state. A CR compromised to serve fabricated ACTIVE states is a complete consent enforcement breach.
Failure Behavior	CR unavailability: EG opens circuit breaker; all requests denied. Cached state may be served for the duration of the cache TTL, after which CR unavailability causes denial. No request may be PERMIT-ted based on cache state after TTL expiry without a fresh CR read.

RS: REVOCATION SERVICE

PROPERTY	DETAIL
Responsibilities	Receives revocation events from authorized principals; propagates revocation state to the Consent Registry within the declared TTL; emits revocation notifications to all EG instances. Holds the sole credential authorized to write revocation flags to the CR. Provides durable delivery guarantee for revocation events.
Trust Boundary	Partially external (receives revocation requests from authorized principals, which may be external to the enforcement perimeter) and partially internal (writes to CR). The RS must authenticate all revocation event sources. The RS signing credential for CR writes must be isolated and not derivable from any other system credential.
Inputs	Authenticated revocation event from authorized principal; acknowledgment from CR on write commit.
Outputs	Signed revocation write to CR; revocation notification broadcast to EG instances (for cache invalidation); AL REVOCATION_COMMITTED record.
Security Assumptions	RS is the only authorized writer of revocation state. Any component that can write REVOKED flags to CR without RS mediation is a structural violation. RS must guarantee delivery of revocation events: loss of a revocation event is a consent enforcement failure.
Failure Behavior	RS unavailability: revocation events queue for delivery on RS recovery. EG instances enter conservative mode for affected consent_ids if RS delivery is delayed beyond TTL/2. A revocation event that cannot be delivered within TTL causes conservative denial for all in-flight requests for the affected consent_id.

AE: ARTIFACT ENGINE

PROPERTY	DETAIL
Responsibilities	Generates the cryptographic enforcement artifact that binds a PERMIT decision to a specific enforcement context. Computes HMAC-SHA256 over (consent_id, context_hash, policy_version, decision, nonce) using the current active signing key. Provides artifact verification capability for the inference system. Key operations may be delegated to a hardware security module.
Trust Boundary	Internal enforcement perimeter; trusted only by EG. Receives signing requests only from authenticated EG instances carrying valid PERMIT decisions. Does not independently evaluate policy. Does not access CR or PE. The signing key must be accessible only to AE; no other component holds the signing key or a derivation thereof.
Inputs	Signing request from EG: (consent_id, context_hash, policy_version, decision=PERMIT, nonce); active signing key (from key store).
Outputs	HMAC-SHA256 enforcement artifact returned to EG; artifact verification result when queried by inference system.
Security Assumptions	AE is the cryptographic boundary of the enforcement decision. A compromised AE that produces artifacts for fabricated decisions represents a complete bypass of enforcement logic. AE must be isolated from all external network access. AE signing operations must be deterministic and reproducible under the same key and nonce. *Security implication: AE isolation from external networks is mandatory; any external access path to AE represents a complete bypass vector.*
Failure Behavior	AE unavailability: EG cannot release PERMIT decisions; all PERMIT-eligible requests denied. No artifact generated means no inference execution. AE must not emit partial artifacts. A signing operation that fails must produce no artifact output visible to EG.

AL: AUDIT LEDGER

PROPERTY	DETAIL
Responsibilities	Provides append-only, hash-chained, tamper-evident storage for all enforcement events. Records every enforcement decision (PERMIT and DENY), every policy lifecycle event, every key rotation event, every revocation event, and every system integrity event. Is the authoritative accountability record for all enforcement activity. Writes must be acknowledged before artifacts are released.
Trust Boundary	Internal enforcement perimeter. Accepts write operations only from authenticated EG and RS instances. Accepts read operations from authenticated assessors and audit processes. Rejects DELETE, UPDATE, and TRUNCATE operations at the storage interface. All write operations must extend the hash chain.
Inputs	Enforcement decision records from EG; revocation event records from RS; key rotation records from key management; policy lifecycle records from deployment process.
Outputs	Hash-chained records readable by audit processes; signed exports for external archival; chain integrity verification results.

PROPERTY	DETAIL
Security Assumptions	AL integrity is a precondition for certification. A tampered AL makes it impossible to verify enforcement accountability. AL must be designed so that any modification, deletion, or reordering of records is detectable through chain verification without access to the original write clients.
Failure Behavior	AL unavailability: EG cannot commit audit records; PERMIT decisions not released; all requests denied. An AL write that fails must not be retried silently: EG must treat an unacknowledged write as a write failure and deny. Partial writes must be detectable and must not corrupt the existing chain.

ML: MONITORING AND TELEMETRY LAYER

PROPERTY	DETAIL
Responsibilities	Collects enforcement-path metrics, component health signals, revocation propagation measurements, and integrity heartbeat results. Dispatches alerts for Class I and Class 0 conditions. Provides the observability surface for operational monitoring and certification evidence. Is not in the enforcement decision path.
Trust Boundary	Partially internal (receives metrics from all enforcement components) and partially external (exports metrics to monitoring infrastructure). ML compromise does not affect enforcement decisions. ML must not be used as a data plane for enforcement decisions. ML availability is not a condition for enforcement operation.
Inputs	Metrics from EG, PE, CR, RS, AE, AL; component health probes; integrity heartbeat results.
Outputs	Alert dispatches (Class I and Class 0); metric time-series for dashboards and certification evidence; integrity heartbeat records.
Security Assumptions	ML is a read-only observer of the enforcement plane. It must not issue commands to any enforcement component. Metric data must not contain sensitive enforcement payloads. ML compromise may suppress alerts but must not affect enforcement outcomes.
Failure Behavior	ML failure has no effect on enforcement decisions. Enforcement continues in the absence of ML telemetry. Alert suppression due to ML failure is a monitoring gap, not an enforcement failure. Components must not block enforcement operations waiting for ML write acknowledgment.

TA: TRUST ANCHOR AND KEY MANAGEMENT LAYER

PROPERTY	DETAIL
Responsibilities	Manages the three-tier key hierarchy: root key (offline), intermediate key, AE signing key. Performs key provisioning to AE; manages key rotation lifecycle; maintains the key rotation audit trail in AL. Root key never leaves the offline trust anchor. Intermediate key rotation events trigger dual-key window management in AE.

PROPERTY	DETAIL
Trust Boundary	The highest trust boundary in the enforcement architecture. The root key trust anchor is isolated from all operational systems. Key provisioning to AE occurs over an authenticated channel with explicit key delivery confirmation. No other component in the architecture holds key material at the root or intermediate level.
Inputs	Key rotation schedule; human authorization for root key operations; AE key provisioning requests.
Outputs	Provisioned AE signing key (to AE); key rotation records (to AL); key rotation notifications (to EG for artifact verification window management).
Security Assumptions	Compromise of the root key trust anchor represents a complete collapse of the enforcement architecture's cryptographic guarantees. Root key operations require out-of-band human authorization. Intermediate key compromise requires immediate rotation and notification of all artifact-verifying systems.
Failure Behavior	TA unavailability does not immediately affect enforcement (AE holds a valid current key). Key rotation cannot proceed during TA unavailability; if the current AE signing key expires during TA unavailability, AE enters conservative mode and enforcement becomes deny-only until key provisioning resumes.

SECTION 4

Logical Data Flow Model

The enforcement data flow is strictly sequential. No step may be executed concurrently with a step that depends on its output. No step may be skipped. No step may be reordered. The sequence is the enforcement invariant: any implementation that satisfies this sequence, regardless of the technology used to realize each component, satisfies the structural data flow requirement.

4.1 Pre-Execution Gating Flow

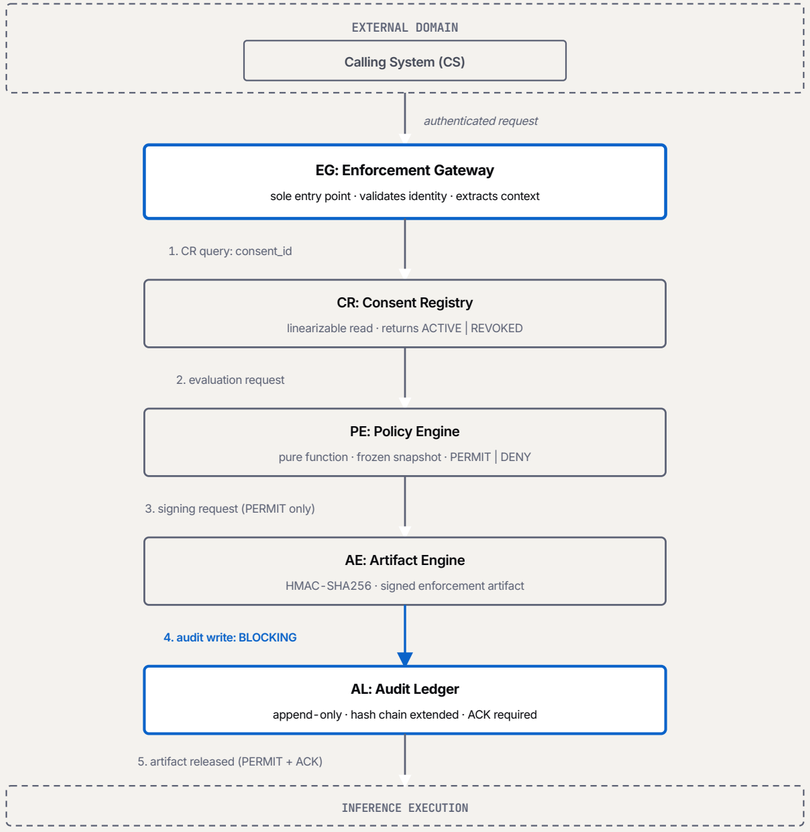


FIGURE 4.1: PRE-EXECUTION GATING FLOW. STEPS ARE STRICTLY SEQUENTIAL; NO STEP MAY BE SKIPPED OR REORDERED. THE AL WRITE AT STEP 4 IS BLOCKING; THE ARTIFACT IS NOT RELEASED UNTIL THE WRITE IS ACKNOWLEDGED.

4.2 Decision Artifact Binding

The enforcement artifact is the cryptographic binding between the enforcement decision and the inference execution it authorizes. The artifact is computed over a fixed set of fields that unambiguously identify the enforcement decision: the consent_id that was evaluated, the context_hash that was evaluated against, the policy version that governed the evaluation, the decision outcome, and a single-use nonce that prevents artifact replay. Any execution that does not present a valid artifact for its exact context is structurally unauthorized.

> **Artifact Binding Invariant:** An enforcement artifact is valid if and only if `HMAC-SHA256(K, consent_id || context_hash || policy_version || decision || nonce)` verifies correctly and the nonce has not been previously consumed. The artifact is not valid for any other consent_id, context, policy version, or decision outcome.

4.3 Write-Before-Execute Invariant

The write-before-execute invariant is the strongest sequencing constraint in the architecture. It states: no inference execution may begin until the EG has received an acknowledged write from the AL for the enforcement decision authorizing that execution. This invariant has no exceptions. A network failure between the AL write and the artifact release means the artifact is not released. A timeout between the AL write and the acknowledgment means the artifact is not released. An AL that acknowledges without actually writing is a structural integrity failure.

STEP	DESCRIPTION
`W-01: PE Decision Complete`	PE returns PERMIT or DENY to EG. Decision is in EG memory only; no external record exists yet. No artifact is generated. No inference execution is possible.

4.4 Revocation Propagation Flow

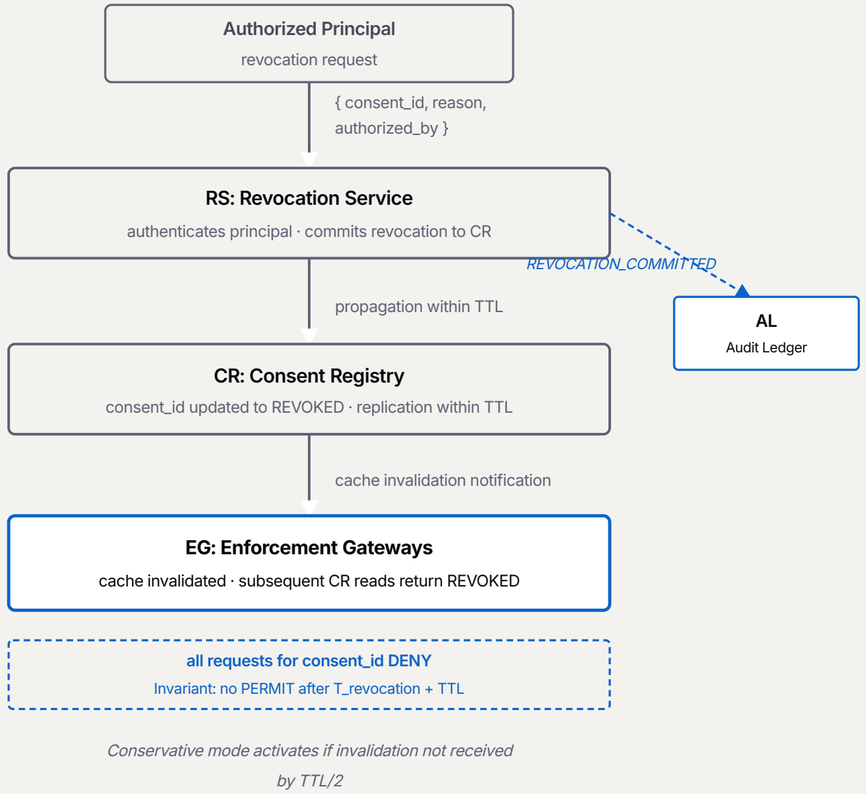


FIGURE 4.4: REVOCATION PROPAGATION FLOW. THE RS COMMITS REVOCATION TO CR AND RECORDS THE EVENT IN AL CONCURRENTLY. CACHE INVALIDATION NOTIFICATIONS ARE DISPATCHED TO ALL EG INSTANCES; CONSERVATIVE DENIAL ACTIVATES IF PROPAGATION LAGS BEYOND TTL/2.

SECTION 5

Trust Boundary Definition

Trust boundaries are structural properties of the architecture, not configuration choices. A boundary defines where trust transitions: where one domain ends and another begins, and what authentication or verification must occur at the crossing. Every component interface is a boundary crossing; no crossing is implicit.

5.1 External Boundary

The external boundary separates the Calling System (and all external agents) from the enforcement perimeter. The sole authorized crossing point is the Enforcement Gateway. No other enforcement component (PE, CR, RS, AE, AL) has a network interface accessible from outside the enforcement perimeter. Any architecture in which PE, CR, or AL is reachable from the external domain is non-conformant regardless of access control configuration.

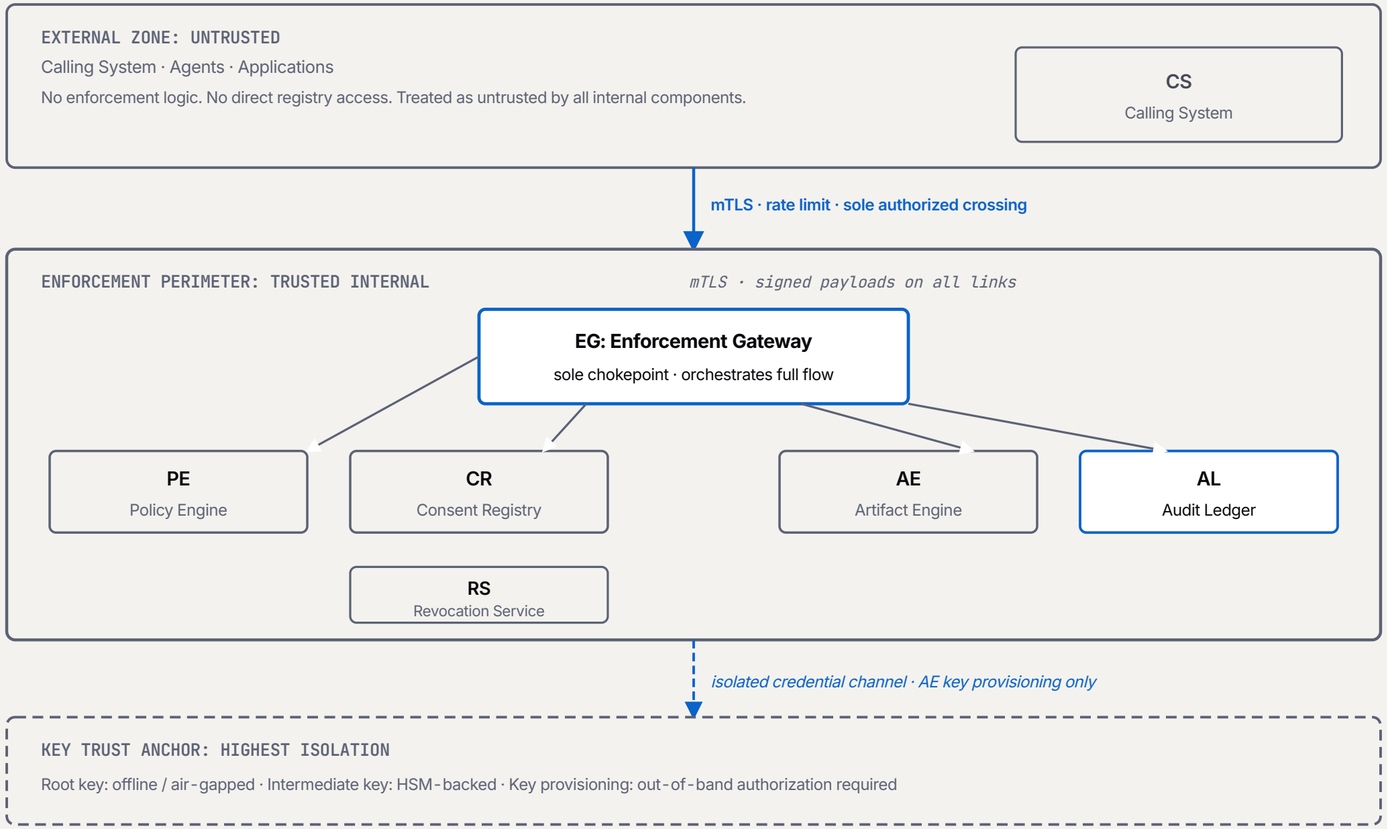


FIGURE 5.1: TRUST BOUNDARY ZONES. THE ENFORCEMENT GATEWAY IS THE SOLE AUTHORIZED CROSSING POINT BETWEEN THE EXTERNAL AND INTERNAL ZONES. ALL INTER-COMPONENT COMMUNICATION WITHIN THE PERIMETER USES MTLS WITH SIGNED PAYLOADS.

5.2 Internal Trust Segments

Within the enforcement perimeter, trust is not uniform. Inter-component trust is directional and purpose-bounded. EG trusts CR to return accurate consent state but does not trust CR to make enforcement decisions. PE trusts the EG to provide an authentic enforcement request but does not trust the EG to interpret policy. AL trusts EG to write accurately but does not trust EG to determine what constitutes a complete enforcement record. These directional trust properties must be enforced by design, not by convention.

TABLE 5.1: INTERNAL COMPONENT TRUST MATRIX

SOURCE COMPONENT	TARGET COMPONENT	TRUST GRANTED	TRUST WITHHELD
EG	CR	Consent state response accuracy within TTL	Enforcement decision authority; policy interpretation
EG	PE	Decision output for the provided input tuple	Context construction; consent state verification
EG	AE	Artifact generation for EG-provided PERMIT decision	Independent policy evaluation; consent verification
EG	AL	Write durability and hash-chain integrity	Decision correctness determination; audit record interpretation
PE	EG	Authenticity of the enforcement request; context_hash accuracy	Policy evaluation; decision outcome; audit record
AE	TA	Key provisioning correctness; key hierarchy integrity	Enforcement decision; consent state; policy evaluation
CR	RS	Authorization to write revocation flags	Enforcement decisions; policy evaluation; audit records
ML	All	Metric data accuracy (read-only, non-authoritative)	No enforcement authority; no write authority on any component

5.3 Cross-Region Boundary

In multi-region deployments, each region constitutes a distinct enforcement domain. Enforcement decisions are made locally within each region; cross-region communication is limited to consent state synchronization and revocation propagation. A region must be able to enforce correctly, including fail-closed, while cross-region communication is

unavailable. Loss of cross-region connectivity triggers conservative mode for affected consent_ids until synchronization is confirmed.

5.4 Registry Synchronization Boundary

The Registry Synchronization Boundary is the interface between CR instances in a replicated deployment. Writes committed at one CR instance must propagate to all CR instances within the declared TTL. The synchronization protocol must preserve linearizability for revocation writes: once a revocation is committed at any CR instance, that instance must not serve a non-revoked state for the affected consent_id, and all other instances must converge to the revoked state within TTL.

5.5 Key Material Isolation Boundary

Key material boundaries define where cryptographic credentials are permitted to exist. The root key must exist only within the offline Trust Anchor. The intermediate key must exist only within HSM-backed storage, never in software or on a general-purpose filesystem. The AE signing key must exist only within the AE signing context, provisioned by the TA over an authenticated channel. No other component may possess, derive, or observe any key material at any tier of the hierarchy. Violation of this boundary nullifies the cryptographic guarantees of the entire enforcement architecture.

SECTION 6

Deployment Topology Variants

The canonical component model supports five reference topology variants. Each variant satisfies the architectural principles and component model defined in this document. The variants differ in how components are distributed, replicated, and isolated across failure domains, not in what the components do or what invariants they must satisfy. All

variants enforce identical invariants; topology determines resilience, latency, and failure containment, not enforcement semantics.

DT-01: CENTRALIZED CONTROL PLANE

TOPOLOGY PROPERTIES

- All components deployed within a single datacenter or availability zone
- Single EG cluster; single PE cluster; single CR instance; single AL partition
- RS operates as an internal service within the enforcement perimeter
- TA root key stored offline in same physical facility
- ML collocated with enforcement components
- Simplest deployment topology; lowest operational complexity

CHARACTERISTICS

- Revocation propagation: single-instance CR → near-zero propagation delay for local EG instances
- Failure containment: datacenter boundary; total failure on datacenter loss
- Latency: lowest enforcement latency; all components on low-latency network fabric
- AL durability: single-region; requires external replication for durability guarantees
- Suitable for: single-domain deployments, development and pre-production environments, low-criticality enforcement workloads

PROPERTY	VALUE
Max Certification Level	CL-01 (Declared or Verified)
Failure Domain	Single datacenter
Revocation TTL	Declared; 300s minimum

DT-02: FEDERATED MULTI-DOMAIN ARCHITECTURE

TOPOLOGY PROPERTIES

- Multiple enforcement domains, each with independent EG, PE, and local CR replica
- Shared Consent Registry with per-domain replicas synchronized via RS
- Single authoritative AL with domain-specific write partitions
- RS operates as a shared propagation broker across domains
- Cross-domain revocation propagation bound by inter-domain TTL
- Each domain independently fail-closed; domain failure does not affect other domains

CHARACTERISTICS

- Revocation propagation: within-domain immediate; cross-domain within inter-domain TTL
- Failure containment: per-domain; domain isolation is a design requirement, not an option
- Data flow: CS in Domain A communicates only with EG in Domain A; no cross-domain enforcement path
- Conservative mode: domain enters conservative mode if cross-domain sync exceeds TTL/2
- Suitable for: multi-tenant enforcement infrastructure, federated AI governance, regulatory boundary enforcement

PROPERTY	VALUE
Max Certification Level	CL-02 (Verified)
Failure Domain	Per enforcement domain
Revocation TTL	Declared per domain; 60s intra-domain

DT-03: ON-PREMISES ENTERPRISE DEPLOYMENT

TOPOLOGY PROPERTIES

- All enforcement components deployed within the enterprise network boundary
- No enforcement component reachable from external networks
- EG is the sole interface between enterprise AI workloads and enforcement infrastructure
- TA root key in on-premises HSM or offline secure element
- AL stored in enterprise-controlled immutable storage
- ML integrated with enterprise monitoring infrastructure; no external telemetry

CHARACTERISTICS

- Revocation propagation: within enterprise network; bounded by internal network latency
- Failure containment: enterprise network boundary; air-gap protects against external interference
- Key material: fully on-premises; no cloud key management services required
- AL durability: enterprise storage; backup and replication per enterprise policy
- Suitable for: regulated industries, government, financial services, high-confidentiality AI workloads

PROPERTY	VALUE
Max Certification Level	CL-03 (Certified) if SLA evidence available
Failure Domain	Enterprise network
Revocation TTL	Operator-declared; 60s recommended

DT-04: EDGE ENFORCEMENT MODEL

TOPOLOGY PROPERTIES

- EG and PE deployed at network edge, co-located with inference workloads
- CR replica at edge; synchronizes with central CR within declared TTL
- AL writes forwarded to central AL over authenticated channel; local buffering only
- AE at edge with signing key provisioned from central TA; rotation via secure channel
- RS at central location; propagation to edge CR via RS propagation channel
- Enforcement decision made at edge without requiring WAN round-trip for CR query

CHARACTERISTICS

- Revocation propagation: central RS → edge CR within TTL; edge must enter conservative mode if sync lapses
- Failure containment: edge enforcement isolated from central failure; central failure does not affect edge decisions for cached consent state within TTL
- AL durability: central AL; edge buffers locally until write is confirmed; enforcement blocked if buffer exceeds threshold without central acknowledgment
- Latency: lowest enforcement latency; WAN latency not in enforcement critical path
- Suitable for: latency-sensitive AI workloads, geographically distributed inference, air-gap tolerant deployments

PROPERTY	VALUE
Max Certification Level	CL-02; CL-03 requires central AL durability evidence
Failure Domain	Edge site (enforcement); central (audit durability)
Revocation TTL	Declared; must account for WAN sync latency

DT-05: MULTI-REGION ACTIVE-ACTIVE

TOPOLOGY PROPERTIES

- Full enforcement stack (EG, PE, CR replica, AE, AL partition) deployed in each region
- All regions simultaneously active; no primary/secondary distinction
- Revocation propagates to all regions within inter-region TTL via RS
- Cross-region consensus required for revocation commitment; divergence triggers conservative mode
- AL partitioned by region; cross-region chain verification for export integrity
- TA root key offline; intermediate keys per-region or shared via secure channel

CHARACTERISTICS

- Revocation propagation: within-region immediate; cross-region bounded by TTL (maximum 15s for CL-04)
- Failure containment: regional; full regional failure handled by other regions; no enforcement continuity in failed region
- Consensus: cross-region revocation commits require quorum; unilateral revocation in one region is not propagated without quorum confirmation
- Clock skew: cross-region timestamp comparisons subject to skew tolerance; requests within skew window for TTL boundary are denied conservatively
- Suitable for: global-scale AI governance, high-availability enforcement infrastructure, regulated multi-jurisdictional deployments

PROPERTY	VALUE
Max Certification Level	CL-04 (High-Assurance; accredited ACB required)
Failure Domain	Per region; cross-region consensus boundary
Revocation TTL	15s cross-region (CL-04); 60s (CL-03)

SECTION 7

Deterministic Decision Boundary Model

The deterministic decision boundary is the structural isolation that makes QODIQA enforcement verifiable. It is not a quality property of the evaluation logic; it is an architectural constraint on what the evaluation function is permitted to access. The decision function must be a pure function: it produces output from inputs, and nothing else. Any access to state not present in the input tuple corrupts the determinism guarantee.

7.1 Evaluation Isolation

The Policy Engine evaluation function is isolated from all external state at the moment of evaluation. The evaluation is presented with exactly three inputs: the frozen context (including token state as read from CR before the evaluation begins), the frozen policy snapshot (loaded and verified at PE startup), and a representation of the declared policy version. No other information may enter the evaluation function. In particular, the evaluation function must not: make network calls of any kind; read from any mutable shared store; access a real-time clock; generate or consume randomness; or depend on the thread-local state of the evaluation process beyond the input tuple.

> **Evaluation Isolation Invariant:** Let `f` be the PE evaluation function, `C` be the frozen context, and `P` be the frozen policy snapshot. The invariant is: `f(C, P)` is a pure total function. For all `C1, C2` such that `C1 = C2`, and all snapshots `P1, P2` such that `P1 = P2`: `f(C1, P1) = f(C2, P2)`. No implementation artifact (execution order, cache state, memory layout, or timing) may cause this equality to fail.

7.2 Frozen Context Binding

The frozen context is the canonical, version-tagged serialization of all enforcement-relevant attributes of a request at the moment of EG evaluation. Once the EG has computed the frozen context and its hash, neither the context nor the

hash may be modified before delivery to PE. The context_hash delivered to PE by EG is the caller's commitment to a specific context; the PE evaluates that commitment, not the raw request.

STRUCTURE: FROZEN CONTEXT · CANONICAL STRUCTURE

```
// Frozen context: all fields that govern the enforcement decision
// Must be canonically serialized before hashing
// Version prefix ensures hash namespace isolation across schema revisions

frozen_context = {
  "version":        "qodiqa-ctx-v1",
  "consent_id":     "<opaque string>",
  "policy_version": "<declared frozen policy version>",
  "caller_id":      "<authenticated caller identifier>",
  "token_state":    "ACTIVE" | "REVOKED" | "EXPIRED" | "NOT_FOUND",
  "token_scope":    ["scope_a", "scope_b"],  // canonical-sorted
  "issued_at":      1700000000,              // integer seconds
  "expires_at":     1700003600,              // integer seconds; no float
  "nonce":          "<unique per request; not included in context_hash>"
}

// Canonicalization invariants (from Playbook §8.1)
CS-INV-01: Keys sorted lexicographically by UTF-8 byte sequence
CS-INV-02: No whitespace in serialized output
CS-INV-03: UTF-8 encoding, no BOM, no surrogates
CS-INV-04: context_hash = "sha256:qodiqa-ctx-v1:" + hex(SHA256(serialize(context)))
CS-INV-05: Semantic equivalence implies hash equality
CS-INV-06: No floating-point values permitted
```

7.3 Artifact Generation Atomicity

Artifact generation is an atomic operation from the perspective of the EG. The AE either produces a complete, valid artifact or it produces nothing. There is no partial artifact state observable to EG. If the AE signing operation fails for any reason (key unavailable, hardware fault, signing algorithm rejection of malformed input) the AE returns an error, EG receives no artifact, and the enforcement flow terminates with a DENY decision. EG must not retry signing with a different nonce in response to a signing failure; a new enforcement evaluation must be initiated from the beginning by CS.

7.4 Decision Independence from External Network

At the moment of PE evaluation, the enforcement decision must not depend on any external network call. The token state has already been read from CR by EG and included in the enforcement request. The policy snapshot is already loaded in PE local memory. No additional network access is required or permitted. An implementation that makes external calls during PE evaluation is structurally non-conformant regardless of whether those calls succeed.

- The enforcement decision must not depend on any call to an external service during evaluation
- The enforcement decision must not depend on the current wall clock time; token expiry is checked against the clock before context freezing, not during evaluation
- The enforcement decision must not depend on randomness or entropy from any source
- The enforcement decision must not depend on mutable shared state accessible from multiple PE evaluation threads simultaneously
- The enforcement decision must produce the same result regardless of how many parallel evaluations are in progress in the PE at the same moment

SECTION 8

Scaling and Performance Model

The reference architecture is designed for horizontal scalability of stateless components and vertical or replicated scalability of stateful components. Scalability must not relax enforcement invariants. An architecture that achieves higher throughput by skipping enforcement steps, relaxing audit write guarantees, or introducing eventual consistency into the revocation check path is not a conformant scale-out architecture; it is a non-conformant architecture with increased throughput.

8.1 Horizontal Scaling of Enforcement Gateway

EG instances are structurally stateless at the enforcement decision level. Each EG instance independently maintains its local revocation cache (bounded by the declared TTL), its circuit breaker state (per downstream component), and its rate limit counters (per consent_id, per instance). Because rate limit counters are per-instance, distributed rate limiting is not a structural requirement; per-instance rate limits are a conformant implementation. Multiple EG instances behind a load balancer constitute a horizontally scalable enforcement perimeter with linear throughput scaling.

MODEL: EG HORIZONTAL SCALING · INVARIANTS

```
// EG is stateless at the enforcement decision level
// Each instance independently correct; no coordination required

throughput(EG, n_instances) = n x throughput(EG, 1)   // linear

// Cache invalidation cost: bounded
// On revocation notification: O(1) per EG instance
// Total cost per revocation: O(n_EG_instances)
// Must complete within TTL; provision EG pool size accordingly
```

```
// Saturation behavior:
// EG throughput ceiling = min(EG capacity, CR read capacity, AL write capacity)
// Saturation triggers fail-closed, NOT graceful degradation
// Provision all components at ≤ 70% of single-instance ceiling

// Enforcement latency model (no-cache, upper bound):
L_enforcement ≤ L_CR + L_PE + L_AE + L_AL + L_EG_overhead
              ≤ 30  + 10  + 5   + 20  + 3   = 68ms
```

8.2 Stateless Enforcement Node Properties

EG and PE are structurally stateless enforcement nodes. Statelessness means: no enforcement decision depends on the history of prior decisions on the same instance; adding or removing an instance does not change the enforcement outcome for any specific request; and an instance restart does not require state recovery (beyond snapshot re-verification for PE and fresh CR connectivity for EG). These properties are architectural requirements, not implementation preferences.

8.3 Registry Caching with TTL

EG instances may cache CR query results for the duration of the declared TTL. Caching is a latency optimization, not a semantic change. The cache TTL is identical to the revocation propagation TTL: a consent_id may be served from cache until the cache entry expires, after which a fresh CR query is required. Cache invalidation on revocation notification must complete within the TTL. An EG instance that cannot invalidate a cache entry for a revoked consent_id within TTL must enter conservative mode and deny all requests for that consent_id until CR confirms revocation.

8.4 Ledger Write Throughput Model

The AL is the throughput ceiling of the enforcement system. Every PERMIT and every DENY requires an AL write. The AL write throughput must exceed the enforcement decision rate; otherwise, the enforcement path saturates at the AL and

the circuit breaker opens, causing system-wide denial. The AL must be provisioned for the expected peak decision rate with a minimum 40% headroom. AL partitioning by consent_id or by EG instance is a valid scaling strategy; the hash chain must remain verifiable across partition boundaries via the partition-level chain structure defined in the Implementation Playbook.

8.5 Artifact Hashing Cost Envelope

The HMAC-SHA256 computation for artifact generation is `O(n)` in the byte length of the serialized context. For typical enforcement contexts of 200–500 bytes, this operation completes in microseconds on general-purpose hardware. The cost is bounded and deterministic; it does not depend on the complexity of the policy evaluation or the number of concurrent evaluations. AE throughput scales linearly with the number of AE instances, subject to the signing key being accessible to each instance independently.

TABLE 8.1: SCALING PROPERTIES BY COMPONENT

COMPONENT	SCALING AXIS	STATE MODEL	THROUGHPUT SCALING	SATURATION BEHAVIOR
EG	Horizontal (instances)	Stateless per decision	Linear	Circuit breaker → deny-all
PE	Horizontal (instances)	Stateless; snapshot pinned	Linear	EG circuit breaker → deny
CR	Read: horizontal (replicas); Write: vertical	Stateful; linearizable writes	Read: near-linear; Write: bounded	EG circuit breaker → deny-all
RS	Vertical; message broker replicas	Stateful; durable delivery	Event-driven; not in hot enforcement path	Queue overflow → revocation delayed; conservative mode
AE	Horizontal (instances)	Stateless; key-bound	Linear per key	EG circuit breaker → deny PERMIT
AL	Horizontal (partitions)	Stateful; append-only	Partition-linear	EG circuit breaker → deny-all

SECTION 9

Failure and Degradation Model

The QODIQA reference architecture does not have a graceful degradation mode. Degradation is not a permitted operational state. The architecture has two modes: full enforcement (all components operational, all invariants satisfied) and fail-closed (some components unavailable, all requests denied). Any intermediate state (partial enforcement, selective bypass, or probabilistic enforcement) is structurally non-conformant. Every failure condition described below resolves to one of these two modes.

> **Degradation Invariant:** There is no failure condition in the QODIQA reference architecture that results in enforcement being weakened, bypassed, or probabilistically applied. All ambiguity resolves to denial. The only permitted operational states are: full enforcement (PERMIT or DENY decisions for all requests) and fail-closed enforcement (DENY decisions for all requests due to component unavailability). There is no third state.

9.1 Registry Unavailability

STEP	DESCRIPTION
F-01a: CR Query Timeout	EG issues CR query; response not received within timeout ceiling. EG increments CR failure counter. If counter exceeds circuit breaker threshold within window W: circuit OPEN. All subsequent requests: DENY with CR_UNAVAILABLE. No inference execution. AL receives DENY records.

9.2 Policy Engine Failure

PE failure is architecturally equivalent to CR failure from the EG perspective: no PE decision can be obtained → no PERMIT can be issued → all requests DENY. PE failure is surfaced by EG as PE_UNAVAILABLE and recorded in AL. PE recovery requires: (a) process restart; (b) snapshot re-load; (c) snapshot hash verification against AL POLICY_FROZEN record; (d) EG circuit breaker probe success. PE may not resume evaluation before all four steps complete. A PE that

bypasses snapshot verification during recovery is non-conformant regardless of whether it happens to load the correct snapshot.

9.3 Audit Ledger Failure

AL failure has the most severe impact in the architecture: a PERMIT-eligible request that receives a PERMIT decision from PE and a valid artifact from AE cannot be released because the AL write cannot be committed. The request is denied. The artifact is discarded. The request must be retried by CS from the beginning after AL recovery. Under no circumstances may EG release an artifact without an AL write acknowledgment. An AL that acknowledges writes without durably committing them (acknowledging before fsync, for example) is a silent data integrity failure and is structurally non-conformant at all certification levels.

9.4 Clock Drift Detection

Clock drift affects TTL calculations, token expiry checks, and cross-region timestamp comparisons. The reference architecture defines a maximum clock skew tolerance of 30 seconds between any two enforcement components. If clock skew between an EG instance and its time reference exceeds the tolerance: the EG instance must enter conservative mode, deny all requests where token expiry falls within the skew window, and alert ML with a CLOCK_SKEW_EXCEEDED condition. EG instances must not make enforcement decisions based on timestamps they cannot verify for consistency.

9.5 Key Revocation Event

A key revocation event (distinct from a consent token revocation event) is the retirement of an AE signing key. Key revocation triggers the dual-key window protocol: both the outgoing and incoming keys are valid for artifact verification during the window duration. After the window closes, the outgoing key is retired from all verification contexts. An artifact presented to the inference system after the window closes that was signed by the retired key is no longer valid,

regardless of when the original enforcement decision was made. Deployments must ensure that no artifact with a validity period extending beyond the window duration is issued.

9.6 Partial System Partition

A partial system partition is a network condition in which some enforcement components can reach each other but others cannot. Partial partitions are among the most operationally complex failure conditions because they can produce inconsistent behavior across EG instances. The reference architecture handles all partition scenarios through the same mechanism: if a required component is unreachable from EG, the circuit breaker opens for that component at that EG instance, and all requests handled by that EG instance are denied until connectivity is restored. EG instances that can reach all components continue to enforce normally. The partition does not affect enforcement correctness at any individual EG instance; it affects enforcement availability for requests routed to affected EG instances.

SECTION 10

Interoperability and Extensibility Model

The reference architecture is designed for structural stability. Interoperability means that different conformant implementations of the same component can be substituted without affecting enforcement semantics. Extensibility means that new control profiles or audit record types can be added to the architecture without violating existing invariants. Neither interoperability nor extensibility permits relaxation of enforcement invariants or introduction of probabilistic enforcement. This architecture is deployment-agnostic and may be positioned in front of any AI inference system without modification to model internals.

10.1 Version Binding

Every enforcement interaction is bound to an explicit version: the context schema version (embedded in the context_hash prefix), the policy version (referenced in every enforcement request), and the artifact schema version (embedded in the artifact structure). Version binding ensures that a receiver can unambiguously determine the schema under which a context, decision, or artifact was produced. Mixed-version deployments (where different EG instances run different corpus versions) are non-conformant; all instances within a single deployment must run the same corpus version. Version binding is compatible with real-time inference APIs, batch processing systems, streaming inference pipelines, and edge deployments; the binding mechanism is transport-independent.

10.2 Backward Compatibility Handling

Backward compatibility is defined at the interface level. A new version of the corpus that changes the frozen context schema is a breaking change; all components that produce or consume frozen contexts must be updated together. A new version that adds new AL record types is non-breaking; existing components that do not produce the new record types continue to be conformant unless the new record type is declared normatively required. No corpus version may relax enforcement semantics as a backward compatibility measure. The enforcement boundary defined by this architecture applies equivalently to synchronous real-time inference APIs, asynchronous batch processing systems, and streaming inference pipelines; backward compatibility obligations hold across all deployment paradigms.

10.3 Profile Layering

The reference architecture supports two compliance profiles: Core Profile and Full Framework Profile. The Core Profile requires only the controls specified in the Core Standard and the component model defined in this document. The Full Framework Profile additionally requires all 68 controls from the 68-Point Enforcement Framework. Both profiles satisfy all invariants in this architecture; the Full Framework Profile adds additional control surface area above the structural

minimum. A deployment may declare conformance at the Core Profile or Full Framework Profile; the claimed profile must be disclosed in the conformance claim.

TABLE 10.1 · PROFILE LAYERING · COMPONENT REQUIREMENTS

COMPONENT	CORE PROFILE	FULL FRAMEWORK PROFILE	DIFFERENCE
EG	Required; single instance minimum	Required; 2 or more instances, independent failure domains	Redundancy and failure domain isolation
PE	Required; single instance; snapshot verified	Required; 2 or more instances; cross-instance snapshot hash agreement verified	Policy drift detection across instances
CR	Required; linearizable revocation reads	Required; replicated; cross-domain sync within TTL 60s	Replication and TTL tightening
RS	Required; durable delivery to CR	Required; durable; cross-domain; conservative mode on lag	Cross-domain propagation and conservative mode
AE	Required; HMAC-SHA256; key from hierarchy	Required; HSM-backed; attestation on restart	Hardware-backed key operations
AL	Required; append-only; weekly chain verify	Required; per-write chain verify; daily external export	Verification frequency and durability cadence
ML	Required; core metrics; Class I alerts	Required; full drift matrix; Class 0 alerts; heartbeat	Observability completeness
TA	Required; root key offline; hierarchy documented	Required; intermediate key HSM-backed; rotation log in AL	Intermediate key protection and rotation evidence

10.4 Future Control Extension

New control profiles may be defined as annexes to the QODIQA corpus in future revisions. Any new control profile must be additive to the existing architecture: it may require additional component behavior, additional audit record types, or additional test procedures, but it may not modify the enforcement semantics of existing components, relax existing invariants, or introduce probabilistic enforcement. A new control profile that contradicts any invariant in this reference architecture is architecturally incompatible with the QODIQA corpus and cannot be adopted without a corpus version revision.

SECTION 11

Reference Invariants Summary

The following invariants constitute the structural commitments of the QODIQA reference architecture. Each invariant is stated as an unconditional property that any conformant implementation must satisfy. These invariants align with and extend the global system invariants defined in the Implementation Playbook and are the canonical reference for architectural conformance assessment.

INV-A: DETERMINISTIC REPLAY INVARIANT

Statement: For all frozen contexts `C` and frozen policy snapshots `P`: `evaluate(C, P) = replay(C, P)`. The enforcement function is a pure total function. Identical inputs produce identical outputs on every invocation, on every PE instance, after every restart, and across every time interval, subject only to the constraint that the frozen context and frozen policy snapshot are identical.

This invariant is violated if: the PE makes any external network call during evaluation; the PE reads from any mutable shared state during evaluation; the PE uses real-time clock values in the evaluation; or any implementation artifact causes different PE instances to produce different decisions for the same input.

Corpus reference: Playbook INV-07; Certification Framework §6; AP-01.

INV-B: FAIL-CLOSED INVARIANT

Statement: For all enforcement requests `R`: if any component required to complete the enforcement flow (CR, PE, AE, AL) is unavailable or returns an error, then `outcome(R) = DENY`. There exists no operational state of the enforcement architecture in which a request receives a PERMIT outcome without all required components having successfully participated in the enforcement flow for that specific request.

This invariant is violated if: a PERMIT is issued when CR is unavailable; a PERMIT is issued without an AL write acknowledgment; a PERMIT is issued when AE cannot generate an artifact; or any enforcement component failure results in an outcome other than DENY.

Corpus reference: Playbook INV-02; Certification Framework §5.3; AP-02.

INV-C: ARTIFACT IMMUTABILITY INVARIANT

Statement: For all enforcement artifacts `A` produced by the AE: `A` is cryptographically bound to exactly one enforcement context (consent_id, context_hash, policy_version, decision, nonce). `A` cannot authorize inference execution for any context other than the one for which it was produced. `A` is valid for exactly one use (nonce consumed on first verification). `A` is invalid after the current AE signing key is retired.

This invariant is violated if: an artifact verifies successfully for a different context_hash than the one used to produce it; a nonce can be reused to authorize a second inference execution with the same artifact; or a retired key continues to be accepted for artifact verification.

Corpus reference: Playbook INV-04; Certification Framework §6.4; AP-04.

INV-D: REVOCATION SUPERSESSION INVARIANT

Statement: For all consent_ids `C` and all revocation events `V` committed at time `T_rev`: for all enforcement requests for `C` arriving after `T_rev + TTL` at any EG instance, `outcome = DENY`. No cached state, in-flight artifact, or previously issued decision permits inference execution for `C` after `T_rev + TTL` at any EG instance in the deployment.

This invariant is violated if: any EG instance issues a PERMIT for `C` after `T_rev + TTL`; any artifact issued before `T_rev` remains valid for verification after `T_rev + TTL`; or conservative mode is not activated when propagation lag exceeds `TTL/2`.

Corpus reference: Playbook INV-03; Certification Framework §8; AP-08.

INV-E: POLICY VERSION BINDING INVARIANT

Statement: For all enforcement decisions `D` : `D` is produced by evaluating the frozen context against the declared frozen policy snapshot at the declared policy version. No enforcement decision is produced by evaluating against a policy snapshot that has not been frozen (hash committed to AL), by evaluating against a policy version that does not match the declared policy_version field in the enforcement request, or by evaluating against a modified version of a previously frozen snapshot.

This invariant is violated if: PE evaluates against a policy snapshot that has not been verified against a POLICY_FROZEN AL record at load time; PE accepts evaluation requests for a policy version that does not correspond to the loaded snapshot; or any mechanism allows modification of the loaded policy snapshot after it has been verified.

Corpus reference: Playbook INV-06; Certification Framework §6.3; AP-03.

INV-F: AUDIT COMPLETENESS INVARIANT

Statement: For all enforcement requests `R` that produce a decision: there exists a record `r` in AL such that `r.consent_id = R.consent_id`, `r.context_hash = R.context_hash`, `r.decision = outcome(R)`, and `r.committed_at < inference_execution_time(R)`. There is no inference execution without a prior durable AL record. The absence of an AL record for a request that reached PE is an audit completeness failure, more severe than a chain integrity failure.

This invariant is violated if: inference execution begins before the corresponding AL record is acknowledged; an AL record is lost due to storage failure without detection; or any enforcement decision is produced without a corresponding AL record within the declared retention period.

Corpus reference: Playbook INV-01, INV-05; Certification Framework §7; AP-05.

SECTION 12

Non-Coverage Disclosure

This Reference Architecture defines the canonical component model, data flows, trust boundaries, and invariants for QODIQA consent enforcement. It does not define everything relevant to a production AI infrastructure deployment. The

following domains are explicitly outside the scope of this document and must not be inferred from its contents.

12.1 Product APIs and Protocol Specifications

This document does not define product APIs, wire protocols, message formats beyond what is needed to specify the enforcement flow, or inter-component communication standards. API specification is outside the scope of a reference architecture. Conformant implementations may use any API design, message encoding, or transport protocol that satisfies the structural requirements of this document.

12.2 Specific Technology Stacks

This document does not mandate, recommend, or imply the suitability of any specific technology stack, programming language, database system, message broker, or infrastructure platform for any component. References to architectural properties (linearizability, append-only, distributed consensus) describe requirements, not implementations. Any technology that satisfies the structural requirement is equally valid.

12.3 Enterprise IAM and Identity Systems

This document does not replace, extend, or integrate with enterprise identity and access management systems. The Calling System authenticates to the Enforcement Gateway; the mechanism by which the calling system's identity is established is outside the scope of this architecture. QODIQA enforces consent at the inference boundary, not authentication at organizational or user-identity boundaries. Enterprise IAM and QODIQA serve complementary but structurally distinct functions.

12.4 AI Model Internals

This document does not govern the internal architecture, training process, serving infrastructure, or behavioral properties of AI inference systems. QODIQA enforcement operates at the boundary between the request and the inference system;

it does not inspect, modify, or constrain what happens within the inference system. The inference system is treated as a black box that accepts or rejects execution requests based on artifact validity.

12.5 Data Storage Encryption

This document does not define encryption standards for data at rest in any enforcement component. Storage encryption for the Consent Registry, Audit Ledger, or any other component is a deployment-specific security control that must be satisfied according to the applicable regulatory and organizational requirements. This architecture specifies the tamper-evidence properties of the Audit Ledger through hash-chaining; storage encryption is orthogonal to hash-chain integrity and is not specified here.

12.6 Consent Grant Origination

This document does not govern the process by which consent grants are created, the user experience through which data principals express consent, or the legal basis on which consent is established. QODIQA enforcement begins at the point of inference-time enforcement; the processes upstream of consent grant creation (consent collection, consent validity determination, consent storage in authoritative systems of record) are outside the scope of this architecture. The Consent Registry as defined here stores the enforcement-relevant state of consent grants; it is not a system of record for consent governance.

NON-COVERAGE SUMMARY: WHAT THIS ARCHITECTURE DOES NOT DEFINE

- Product APIs, wire protocols, or specific inter-component message formats
- Specific technology stack, database, or infrastructure platform choices
- Enterprise IAM, user authentication, or organizational identity management
- AI model internals, training infrastructure, or inference system architecture
- Data storage encryption standards for any enforcement component
- Consent grant origination, data principal UX, or legal basis for consent
- Network infrastructure, physical security, or datacenter design
- Billing, access control to enforcement configuration, or operational tooling

This architecture defines the logical component model only. All implementation choices not constrained by the invariants in this document are at the discretion of the implementing organization.

SECTION 13

Institutional Closing Statement

QODIQA is a runtime control plane. It does not govern what AI systems may do in the abstract; it governs what inference executions may occur in specific operational contexts, against specific consent states, evaluated by specific frozen policy versions, at specific instants in time. The architecture defined in this document is the structural form through which that governance is exercised. It is not a description of intent; it is a specification of a runtime boundary.

> The enforcement architecture is the consent boundary. Not a policy document about the consent boundary. Not a monitoring system that observes the consent boundary. The architecture itself is the boundary. When the EG denies a request, when the AL records the denial, when the circuit breaker opens under component failure, these are the consent boundary operating correctly. The boundary does not exist outside the architecture that implements it.

Determinism is structural in this architecture, not aspirational. The architectural constraint that the PE evaluation function be pure (no network calls, no shared mutable state, no real-time clock, no randomness) is not a performance optimization. It is the mechanism by which enforcement decisions become auditable, replayable, and certifiable. An enforcement system that makes probabilistic or state-dependent decisions cannot be audited; it cannot be replayed; it cannot be certified. Determinism is the precondition for accountability.

> **Infrastructure precedes policy interpretation.** A consent policy is not enforceable by declaration. It is enforceable only when there exists an infrastructure (a runtime control plane) that gates inference execution against the policy, records every decision with cryptographic evidence, and fails closed when any component required for verified enforcement is unavailable. **This reference architecture is the specification of that infrastructure.** Where it is implemented correctly, consent enforcement is structural. Where it is not, consent governance remains advisory.

13.1 Corpus Alignment Statement

This Reference Architecture (QODIQA-ARCH-2026-001 v1.0) is normatively complementary to and must be read in conjunction with the following corpus documents: QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0; QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0; QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0; QODIQA: Implementation Playbook for Deterministic Runtime Consent Enforcement, Version 1.0; QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0. This document is authoritative for the canonical component

model, trust boundary definitions, and architectural invariants. For normative enforcement semantics, defer to the Core Standard. For operational deployment guidance, defer to the Implementation Playbook. For certification procedures, defer to the Certification Framework.

DOCUMENT STATUS

Document Status and Classification

This document is the Reference Architecture of the QODIQA specification corpus. It defines the canonical, vendor-neutral architectural topology for deterministic runtime consent enforcement. It is issued as a Reference Architecture Specification and is not a product specification, an implementation guide, or a commercial deployment blueprint. It is a logical architectural model against which conformant implementations may be assessed for structural conformance.

The governing constraint is structural: the architecture is the enforcement boundary, and the enforcement boundary must be deterministic, fail-closed, and cryptographically anchored or it is not a conformant QODIQA architecture.

The material contained herein is intended for:

- Principal architects designing QODIQA-conformant enforcement infrastructure
- Distributed systems engineers implementing canonical enforcement components
- Security architects defining trust boundaries and isolation properties for AI inference infrastructure
- Standards body members and regulatory reviewers evaluating the architectural basis of QODIQA enforcement
- Independent assessors and certification bodies conducting architectural conformance evaluation
- Technical governance leads responsible for AI consent enforcement infrastructure decisions

All architectural principles described in this document are production-oriented, deterministically anchored, and designed for integration within large-scale AI ecosystems.

This document should be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Implementation Playbook for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0
- QODIQA: Threat Model and Abuse Case Specification, Version 1.0
- QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dițescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-ARCH-2026-001
TITLE	Reference Architecture for Deterministic Runtime Consent Enforcement
SUBTITLE	Canonical Topology and Component Interaction Model for QODIQA Enforcement Infrastructure
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Reference Architecture Specification
DOCUMENT STATUS	Normative: Reference Architecture Specification
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 06 PART II · NORMATIVE TECHNICAL SPECIFICATIONS

Security and Cryptographic Profile for Runtime Consent Enforcement

Mandatory and recommended cryptographic, integrity, and security mechanisms for high-assurance implementations of the QODIQA enforcement infrastructure.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-SEC-2026-001	1.0	2026	Security Specification	Institutional	NORMATIVE

ABSTRACT

This document defines the **mandatory and recommended cryptographic, integrity, and security mechanisms** required for high-assurance implementations of QODIQA Core v1.0, the 68-Point Enforcement Framework v3.0, and the associated Conformance Test Suite.

This profile is not general security guidance. It is not a cybersecurity best-practices document. It is not a compliance advisory. It is a **deterministic security hard profile** for runtime consent enforcement infrastructure, specifying precisely which algorithms are permitted, which are forbidden, how consent tokens are signed, how trust anchors are managed, how revocation registries are protected, and how audit integrity is preserved across the full enforcement lifecycle.

All requirements expressed herein use normative language. Implementations claiming conformance with QODIQA Core v1.0 at Hardened Enterprise Profile or above SHALL satisfy all MUST requirements in this document.

SECTION 1

Purpose and Security Scope

QODIQA Core v1.0 defines a deterministic consent enforcement layer for AI runtime systems. This Security and Cryptographic Profile extends that standard by specifying the cryptographic substrate required to make QODIQA's deterministic properties tamper-resistant, verifiable, and auditable under adversarial conditions.

1.1 Scope of This Profile

This profile governs the security requirements applicable to the QODIQA enforcement layer, specifically the runtime components responsible for consent token issuance, artifact production, policy evaluation, registry consultation, and audit ledger maintenance. It applies to all software, hardware, and operational components that implement or host QODIQA enforcement functions.

> **Boundary Delimitation.** This profile governs enforcement layer security. It does not govern the security of AI model weights, training pipelines, inference parameters, or any component internal to the model under enforcement. The System Under Protection (SUP) is the enforcement layer itself.

1.2 Relationship to Core Standard

QODIQA Core v1.0 defines the logical architecture of enforcement: consent gates, frozen evaluation contexts, decision artifacts, replay verification, and fail-closed semantics. This profile defines the cryptographic mechanisms that make those logical properties provably enforceable. Where Core defines *what* must be enforced, this profile defines *how* that enforcement must be anchored.

1.3 Relationship to Conformance Test Suite

The Conformance Test Suite references this profile for cryptographic verification test cases. Implementations that pass all Core conformance tests but fail cryptographic profile requirements SHALL be classified as non-conformant at any profile level above Baseline Security.

1.4 System Under Protection

DEFINED BOUNDARY: System Under Protection (SUP)

SUP.CONSENT-GATE

The runtime enforcement gate evaluating consent before AI invocation. Boundary includes token ingestion, policy lookup, and decision production.

SUP.ARTIFACT-STORE

The immutable store of Decision Artifacts produced by the enforcement gate. Boundary includes write path, hash chain, and replay verification interface.

SUP.REGISTRY

The revocation and policy version registry consulted during enforcement evaluation. Boundary includes read path, integrity verification, and TTL management.

SUP.AUDIT-LEDGER

The append-only audit log recording all enforcement decisions. Boundary includes write-before-execute path, hash chain, and archival interface.

SUP.KEY-STORE

The secure store of signing keys, trust anchors, and root certificates. Boundary includes HSM or secure enclave where applicable.

Components external to the above SUP definitions, including model weights, inference engines, application-layer logic, and data stores, are out of scope for this profile's cryptographic requirements.

SECTION 2

Threat Model Definition

The following threat categories represent realistic, high-impact adversarial scenarios against the QODIQA enforcement layer. All are grounded in known attack classes against cryptographic infrastructure, distributed systems, and audit mechanisms. Speculative or theoretical threats are excluded.

THR-01

Consent Token Forgery

An adversary constructs a syntactically valid consent token without possessing the issuing authority's private key. Mitigated by mandatory signature verification at every enforcement gate.

THR-02

Key Compromise

An adversary obtains a private signing key, enabling forgery of valid consent tokens. Mitigated by HSM key storage, rotation intervals, revocation propagation, and key compromise response procedures.

THR-03

Registry Tampering

An adversary modifies the revocation registry to remove revocation records or insert false active-key entries. Mitigated by signed snapshots, hash chain integrity, and anti-downgrade sequence number enforcement.

THR-04

Replay Manipulation

An adversary replays a previously valid consent token against a different context or after revocation. Mitigated by nonce binding, timestamp validation, registry consultation, and artifact hash verification.

THR-05

Audit Log Deletion and Modification

An adversary removes or alters audit entries to conceal enforcement decisions. Mitigated by append-only storage, hash chain integrity, and write-before-execute sequencing.

THR-06

Clock Manipulation

An adversary skews system time to extend token lifetime, suppress TTL expiry, or produce false temporal audit records. Mitigated by NTP synchronization requirements, monotonic clock enforcement, and clock drift bounds.

THR-07

Policy Version Substitution

An adversary substitutes an outdated or permissive policy version during evaluation. Mitigated by inclusion of policy version in artifact hash and mandatory registry version verification.

THR-08

Artifact Collision Attempt

An adversary engineers a second execution context that produces the same artifact hash as a legitimately authorized context. Mitigated by SHA-256 minimum, full canonical context inclusion, and collision-resistance requirements.

THR-09

Network Partition Abuse

An adversary induces a network partition to prevent registry consultation, forcing a fallback to a permissive state. Mitigated by fail-closed semantics: enforcement MUST deny when registry is unreachable.

THR-10

Insider Tampering

A privileged internal actor modifies keys, artifacts, or audit records. Mitigated by HSM-based key storage, dual-control key management procedures, append-only audit architecture, and independent log archival.

SECTION 2.1

Security Invariants

The following invariants are formal properties that MUST hold at all times across all conformant implementations of the QODIQA enforcement layer. An invariant violation is not a configuration error; it is an integrity failure. Any condition that causes an invariant to be false MUST trigger an integrity violation event, a fail-closed enforcement state, and an audit ledger entry recording the breach.

These invariants are not derived from policy preferences. They are derived from the structural requirements of deterministic, verifiable, tamper-resistant consent enforcement. They hold regardless of deployment profile tier.

FORMAL SECURITY INVARIANTS — SEC-INV · V1.0

```
// INV-01: Artifact Signature Invariant
∀ artifact ∈ ARTIFACT_STORE:
  artifact.signature IS VALID
```

```
  ∧ artifact.signing_key ∈ TRUST_REGISTRY
  ∧ artifact.signing_key.status = ACTIVE
  → INVARIANT: No unsigned or unverifiable artifact SHALL exist

// INV-02: Write-Before-Execute Invariant
∀ decision ∈ {PERMIT, DENY}:
  LEDGER.contains(decision.ledger_entry) = TRUE
  ∧ LEDGER.committed(decision.ledger_entry) = TRUE
  ∧ decision.ledger_entry.timestamp ≤ decision.response_timestamp
  → INVARIANT: No PERMIT response SHALL be returned before its ledger entry is committed

// INV-03: Registry Validation Invariant
∀ token ∈ PRESENTED_TOKENS:
  REGISTRY.consulted(token.kid) = TRUE
  ∧ REGISTRY.snapshot_age ≤ CONFIGURED_TTL
  → INVARIANT: No valid token SHALL bypass live registry validation

// INV-04: Revocation Invariant
∀ key ∈ REVOKED_KEYS:
  ¬∃ token: VERIFY(token.sig, key) = ACCEPTED
  → INVARIANT: No revoked key SHALL produce an accepted signature

// INV-05: Clock Certainty Invariant
∀ enforcement_decision:
  CLOCK.sync_error ≤ PROFILE.drift_tolerance
  ∧ CLOCK.source = AUTHENTICATED_NTP | GPS | PTP
  → INVARIANT: No enforcement decision SHALL execute under clock uncertainty exceeding tolerance

// INV-06: Deterministic Evaluation Invariant
∀ evaluation_invocation:
  evaluation.external_callouts = ∅
  ∧ evaluation.random_in_path = FALSE
  ∧ evaluation.context = FROZEN
```

```
  → INVARIANT: No enforcement decision SHALL execute without a deterministic evaluation path

// INV-07: Artifact Chain Invariant
∀ artifact_n ∈ ARTIFACT_STORE:
  artifact_n.prev_artifact_id = SHA256(artifact_{n-1})
  ∨ (n = 0 ∧ artifact_n.prev_artifact_id = ZERO_BYTES_32)
  → INVARIANT: The artifact hash chain MUST be unbroken from genesis
```

SEC-INV-01 **Invariant Violation Response** ● MUST

REQUIREMENT	Any runtime condition that causes one or more of the above invariants to evaluate as false MUST trigger: (1) an immediate fail-closed DENY-ALL enforcement state; (2) a signed integrity violation entry written to the audit ledger; (3) an alert to the designated security operations contact. Invariant violations MUST NOT be silently absorbed, corrected without logging, or treated as recoverable operational events without explicit documented escalation.
VERIFICATION	Conformance Test Suite cases SHALL include invariant violation injection tests. An implementation that does not enter fail-closed state upon induced invariant violation fails High-Assurance and Hardened profile conformance.

SECTION 2.2

Attack Surface Mapping

Each SUP component defined in Section 1.4 presents a discrete attack surface. The following table maps each component to its primary attack vectors, associated threat class, normative mitigation control, and the enforcement point at which the control is applied. This mapping is structural: it does not add requirements beyond those stated in the corresponding normative sections. It exists to provide a deterministic traceability reference between threat classes and controls.

TABLE 2.2-A: ENFORCEMENT LAYER ATTACK SURFACE MAPPING

COMPONENT	ATTACK SURFACE	THREAT CLASS	MITIGATION CONTROL	ENFORCEMENT POINT
SUP.CONSENT-GATE	Token ingestion interface; policy evaluation path; decision output channel	Token forgery (THR-01); replay manipulation (THR-04); non-deterministic evaluation (INV-06)	Signature verification (SEC-ALG-01); deterministic verification sequence (SEC-TOK-VER); anti-replay JTI cache (SEC-TOK-REPLAY-CLUSTER)	Runtime enforcement gate on every invocation
SUP.REGISTRY	Snapshot read interface; sequence number state; cached state window	Registry tampering (THR-03); downgrade substitution; stale-state exploitation	Signed snapshots (SEC-REG-01); fail-closed on unreachability (SEC-REG-02); anti-downgrade sequence enforcement (Section 6.1)	Registry consultation at each enforcement gate invocation
SUP.AUDIT-LEDGER	Ledger write path; hash chain continuity; archival interface	Audit log deletion and modification (THR-05); hash chain break; write-before-execute violation (INV-02)	Write-before-execute requirement (SEC-AUDIT-01); append-only storage (Section 8.2); tamper detection (Section 8.3)	Ledger write before PERMIT response is issued
SUP.ARTIFACT-STORE	Artifact write interface; hash chain linkage; artifact read interface for replay verification	Artifact collision (THR-08); artifact tampering; hash chain break (INV-07)	Artifact immutability (SEC-ART-01); model binary integrity binding (SEC-ART-02); canonical hash construction (Section 7.1)	Artifact production at enforcement decision time; replay verification on demand
SUP.KEY-STORE	Private key material; key export interface; key activation and rotation path	Key compromise (THR-02); insider tampering (THR-10)	HSM storage requirement (SEC-KEY-01); prohibited storage patterns (SEC-KEY-02); hardware key protection and dual-control (SEC-KEY-03)	Key generation, storage, rotation, and activation procedures

SECTION 2.3

Failure Mode Enumeration and Deterministic Response

Failure modes are conditions under which one or more security invariants defined in Section 2.1 cannot be satisfied. For each failure mode, the system response MUST be deterministic and fail-closed. No failure mode permits a PERMIT decision to be issued when the condition rendering the enforcement environment unverifiable remains unresolved. The

following table enumerates the primary failure conditions, their detection mechanisms, required system responses, enforcement outcomes, and audit obligations.

TABLE 2.3-A: FAILURE MODE AND DETERMINISTIC RESPONSE MATRIX

FAILURE CONDITION	DETECTION MECHANISM	REQUIRED SYSTEM RESPONSE	ENFORCEMENT OUTCOME	AUDIT REQUIREMENT	SEVERITY
Registry unavailable	Registry connection timeout or network error on consultation attempt	Enforcement gate MUST enter fail-closed state. No cached state may be used if TTL has expired. Operations MUST be suspended until registry is reachable and a valid signed snapshot is confirmed.	DENY-ALL until registry restored	MUST write registry unavailability event to audit ledger with timestamp and gate identifier	CRITICAL
Registry stale snapshot	Snapshot age exceeds configured TTL for the applicable profile (Baseline: 300 s; Hardened: 60 s; High-Assurance: 15 s)	Cached state MUST NOT be used beyond TTL expiry. Enforcement gate MUST deny all requests until a fresh snapshot is obtained and verified.	DENY until fresh signed snapshot obtained	MUST write stale-snapshot event to audit ledger	HIGH
Key identifier not resolvable	Registry lookup of token `kid` returns null or status REVOKED	Enforcement gate MUST immediately produce a DENY decision. No exception path or fallback key resolution is permitted.	DENY: immediate	MUST write key resolution failure event including presented `kid` and resolution outcome	CRITICAL
Signature verification failure	VERIFY(token.sig, canonical_payload, public_key) returns FALSE	Enforcement gate MUST produce a DENY decision. The token MUST be treated as potentially forged. No retry or alternative verification path is permitted.	DENY: immediate	MUST write signature failure event to audit ledger	CRITICAL
Clock drift violation	Measured clock synchronization error exceeds the configured drift tolerance for the applicable profile	Enforcement gate MUST enter fail-closed state for all decisions requiring temporal validation. Gate MUST remain in fail-closed state until clock synchronization is restored and verified within tolerance.	DENY-ALL temporal decisions until clock restored	MUST write clock drift violation event with measured error and tolerance threshold	CRITICAL

FAILURE CONDITION	DETECTION MECHANISM	REQUIRED SYSTEM RESPONSE	ENFORCEMENT OUTCOME	AUDIT REQUIREMENT	SEVERITY
Audit ledger write failure	Ledger write returns error or acknowledgment timeout before PERMIT response is issued	Enforcement gate MUST return DENY. The write-before-execute invariant (INV-02) MUST NOT be relaxed under any operational condition, including high load or storage failure.	DENY: ledger write failed	MUST attempt to write failure event to secondary log sink; if unavailable, alert security operations	CRITICAL
Partial artifact persistence	Artifact hash chain verification detects a gap: i.e., a missing or unlinked artifact in the sequence	Enforcement gate MUST enter fail-closed state. The artifact store MUST be quarantined pending forensic review. No new artifacts may be written until the chain is validated or restored from a verified backup.	DENY-ALL until artifact chain integrity confirmed	MUST write integrity violation event recording the last valid artifact identifier and the gap location	HIGH

TABLE 2.3-B: CONTROL TRACEABILITY INDEX

CONTROL ID	SECTION REFERENCE	THREAT COVERAGE	COMPONENT SCOPE
SEC-ALG-01	Section 3.1	THR-01 (Token Forgery); algorithm substitution	SUP.CONSENT-GATE
SEC-ALG-AGILITY-01	Section 3.4	Algorithm obsolescence; PQC transition risk	SUP.CONSENT-GATE; SUP.KEY-STORE
SEC-INV-01	Section 2.1	All invariant violations (INV-01 through INV-07)	All SUP components
SEC-TOK-01	Section 4.1	THR-01 (Token Forgery); signature malleability	SUP.CONSENT-GATE
SEC-TOK-REPLAY-CLUSTER	Section 4.5	THR-04 (Replay Manipulation); distributed replay	SUP.CONSENT-GATE
SEC-KEY-01	Section 5.1	THR-02 (Key Compromise); unresolvable trust chain	SUP.KEY-STORE; SUP.CONSENT-GATE
SEC-KEY-02	Section 5.4	THR-02 (Key Compromise); THR-10 (Insider Tampering)	SUP.KEY-STORE
SEC-KEY-03	Section 5.4	THR-02 (Key Compromise); THR-10 (Insider Tampering)	SUP.KEY-STORE
SEC-REG-01	Section 6	THR-03 (Registry Tampering); hash chain break	SUP.REGISTRY

CONTROL ID	SECTION REFERENCE	THREAT COVERAGE	COMPONENT SCOPE
SEC-REG-02	Section 6	THR-09 (Network Partition Abuse); stale state exploitation	SUP.REGISTRY; SUP.CONSENT-GATE
SEC-ART-01	Section 7.1	THR-08 (Artifact Collision); artifact tampering (INV-07)	SUP.ARTIFACT-STORE
SEC-ART-02	Section 7.1	THR-07 (Policy Version Substitution); silent model substitution	SUP.ARTIFACT-STORE; SUP.CONSENT-GATE
SEC-AUDIT-01	Section 8.1	THR-05 (Audit Log Modification); write-before-execute violation (INV-02)	SUP.AUDIT-LEDGER
SEC-CLK-01	Section 9	THR-06 (Clock Manipulation); clock drift violation (INV-05)	SUP.CONSENT-GATE; SUP.AUDIT-LEDGER
SEC-NET-01	Section 10	Network interception; plaintext transport exposure	All SUP components (network interfaces)
SEC-NET-PFS	Section 10	Retrospective decryption; non-PFS cipher suite exploitation	All SUP components (TLS connections)
SEC-DET-01	Section 11	Non-deterministic evaluation (INV-06); evaluation path contamination	SUP.CONSENT-GATE
SEC-SCOPE-01	Section 14	Misrepresentation of security boundary; SUP boundary violations	All SUP components (documentation and claims)

SECTION 3

Cryptographic Algorithm Requirements

3.1 Approved Signature Algorithms

Implementations SHALL use only the following signature algorithms for consent token signing, registry snapshot signing, and audit chain anchoring. Algorithm selection MUST be recorded in the implementation's cryptographic configuration manifest.

TABLE 3.1: APPROVED SIGNATURE ALGORITHMS

ALGORITHM	STANDARD	REQUIREMENT	NOTES
Ed25519	RFC 8032	MUST	Primary recommended algorithm. Deterministic signature generation. 128-bit security level.
ECDSA P-256	NIST FIPS 186-4	SHOULD	Optional interoperability profile. MUST use deterministic nonce per RFC 6979. SHA-256 digest required.
ECDSA P-384	NIST FIPS 186-4	SHOULD	High-Assurance profile only. SHA-384 digest required. Justified by extended security margin requirement.
RSA-PSS ≥ 3072-bit	RFC 8017	SHOULD	Legacy integration only. SHA-256 or SHA-384 digest required. SHALL NOT be used for new implementations.

SEC-ALG-01 **Primary Signature Algorithm Requirement** ● MUST

REQUIREMENT: All conformant implementations SHALL support Ed25519 as the primary signing algorithm. Non-NIST-approved algorithms, including those not listed in Table 3.1, SHALL NOT be used under any circumstance, including experimental or prototype deployments.

RATIONALE: Ed25519 provides deterministic signature generation, eliminating the nonce reuse vulnerability class inherent to non-deterministic ECDSA. Its compactness and performance characteristics are appropriate for inline enforcement gate use.

3.2 Hashing Algorithms

TABLE 3.2: APPROVED HASHING ALGORITHMS

ALGORITHM	STANDARD	REQUIREMENT	USE
SHA-256	NIST FIPS 180-4	MUST	Artifact hashing, audit chain, token payload digest. Minimum acceptable.
SHA-384	NIST FIPS 180-4	SHOULD	High-Assurance profile artifact hashing. Required when ECDSA P-384 is the signing algorithm.

ALGORITHM	STANDARD	REQUIREMENT	USE
SHA-512	NIST FIPS 180-4	SHOULD	Merkle tree leaf hashing in optional audit ledger variant. Acceptable at all profiles.
BLAKE3	IETF Draft	SHOULD	Internal high-throughput hashing only. MUST NOT be used as the sole artifact hash without SHA-256 co-digest.

3.3 Forbidden Algorithms

The following algorithms are **unconditionally forbidden** in all QODIQA implementations regardless of deployment context, legacy compatibility claims, or performance justifications. No conformance exception is available for these prohibitions.

TABLE 3.3: FORBIDDEN ALGORITHMS

ALGORITHM	STATUS	REASON	PROHIBITION SCOPE
MD5	FORBIDDEN	Cryptographically broken. Collision attacks demonstrated.	MUST NOT be used for any purpose.
SHA-1	FORBIDDEN	Collision resistance compromised. SHAttered attack demonstrated in 2017.	MUST NOT be used for any purpose.
RSA < 2048-bit	FORBIDDEN	Insufficient key length. Factorization attacks feasible.	MUST NOT be used in any context.
ECDSA non-deterministic nonce	FORBIDDEN	Nonce reuse yields private key recovery. Pre-RFC 6979 implementations affected.	MUST NOT be deployed.
DES / 3DES	FORBIDDEN	Block cipher with insufficient security margin.	MUST NOT be used for any symmetric operation.
RC4	FORBIDDEN	Stream cipher with known statistical biases.	MUST NOT be used in any context.
MD4, MD2, RIPEMD-128	FORBIDDEN	Broken hash functions. No collision resistance.	MUST NOT be used for any purpose.
Non-standard / proprietary hash functions	FORBIDDEN	Unaudited algorithms without public cryptanalysis history.	MUST NOT be used.

3.4 Cryptographic Agility and Deprecation Model

Cryptographic algorithms are subject to continuous cryptanalytic pressure. A static algorithm selection, however conservative at time of adoption, requires a structured lifecycle to remain valid over time. This section defines the mandatory review, deprecation, and migration framework for all algorithms approved under Section 3.

SEC-ALG-AGILITY-01 **Mandatory Cryptographic Review Interval** ● MUST

REQUIREMENT	All conformant implementations MUST conduct a formal cryptographic algorithm review at an interval not exceeding 12 months. The review SHALL assess each approved algorithm against current NIST guidance, published cryptanalytic literature, and the NIST Post-Quantum Cryptography (PQC) migration roadmap. Review results MUST be documented and retained as part of the implementation's security configuration record.
PQC READINESS	All implementations targeting active deployment beyond 2030 MUST include a documented PQC migration readiness plan, identifying candidate post-quantum signature algorithms (CRYSTALS-Dilithium, FALCON, SPHINCS+ per NIST FIPS 204/205/206) and a transition timeline. Migration readiness SHALL be verified as part of High-Assurance profile recertification cycles.

Algorithm Deprecation Lifecycle

TABLE 3.4.1: ALGORITHM DEPRECATION LIFECYCLE STAGES

STAGE	DESIGNATION	OPERATIONAL EFFECT	DURATION LIMIT	MIGRATION ACTION REQUIRED
Stage 1	WARNING	Algorithm remains approved. Advisory notice issued. New deployments SHOULD prefer alternatives.	≤ 12 months	Migration plan MUST be documented.
Stage 2	RESTRICTED USE	Algorithm permitted for existing deployments only. MUST NOT be used in new implementations or new key generations.	≤ 18 months	Active migration to replacement MUST begin. Dual-signature required (Section 3.4.2).

STAGE	DESIGNATION	OPERATIONAL EFFECT	DURATION LIMIT	MIGRATION ACTION REQUIRED
Stage 3	PROHIBITED	Algorithm added to Forbidden Algorithms list (Section 3.3). All uses unconditionally cease.	Permanent	Migration MUST be complete before Stage 3 entry. No exceptions.

3.4.1 Emergency Algorithm Withdrawal

In the event of a critical cryptographic break, defined as a publicly demonstrated attack reducing effective security below 80-bit equivalent, the affected algorithm SHALL be withdrawn immediately, bypassing the staged deprecation lifecycle. Emergency withdrawal SHALL: (1) add the algorithm to the forbidden list within 48 hours of NIST or equivalent advisory publication; (2) trigger the Key Compromise Response procedure (Section 13.1) for all keys using the affected algorithm; (3) require immediate dual-signature transition for any artifacts produced under the withdrawn algorithm.

3.4.2 Dual-Signature Transition Model

During algorithm migration from a Stage 2 (Restricted) algorithm to an approved replacement, implementations MUST employ a dual-signature transition period. During this period, consent tokens, registry snapshots, and audit ledger anchors SHALL carry signatures from both the outgoing algorithm and the replacement algorithm. A verifying enforcement gate MUST accept a token as valid only if BOTH signatures verify correctly against their respective keys. This period SHALL terminate when all enforcement gates in the deployment have confirmed support for the replacement algorithm exclusively.

DUAL-SIGNATURE MIGRATION FLOW SEC-ALG-MIGRATION

```
// Phase 0: Detection, algorithm enters Stage 1 (WARNING)
ADVISORY_ISSUED(algorithm_id)
→ document_migration_plan()
→ identify_replacement_algorithm()  // MUST be from approved list Section 3.1
```

```
// Phase 1: Preparation, algorithm enters Stage 2 (RESTRICTED)
generate_replacement_keypair()      // in HSM
publish_replacement_key_to_registry()
configure_dual_sig_mode()

// Phase 2: Dual-Signature Active
for each token | snapshot | anchor:
  sig_legacy    = SIGN(payload, outgoing_key)
  sig_new       = SIGN(payload, replacement_key)
  token.signatures = [sig_legacy, sig_new]

// Phase 2 verification requirement
assert VERIFY(sig_legacy, payload, outgoing_pubkey) = TRUE
assert VERIFY(sig_new,    payload, replacement_pubkey) = TRUE
if either fails → DENY

// Phase 3: Cutover · all gates confirm replacement support
when ALL_GATES.replacement_verified() = TRUE:
  disable_legacy_sig_generation()
  outgoing_key → REVOKE
  algorithm_id → PROHIBITED (Stage 3)

// Phase 4: Post-migration validation
assert ARTIFACT_STORE.no_active_outgoing_sigs() = TRUE
run_full_conformance_suite()
```

SECTION 4

Consent Token Structure Security

Consent tokens are the primary credential presented to the QODIQA enforcement gate. Their cryptographic integrity is the first line of defense against consent forgery and replay manipulation. The following requirements define the minimum structural and cryptographic properties for all conformant consent tokens.

4.1 Canonical Serialization

Token payloads MUST be canonically serialized prior to signing. Canonical serialization eliminates signature malleability arising from equivalent representations of the same semantic content. Implementations SHALL use JSON Canonicalization Scheme (JCS, RFC 8785) or a formally specified binary serialization format documented in the implementation's cryptographic configuration manifest.

SEC-TOK-01 **Token Canonical Serialization** — MUST

REQUIREMENT	Token payloads MUST be canonically serialized (RFC 8785 or equivalent formal specification) before signing. The enforcement gate MUST re-canonicalize the received token before signature verification. Signatures over non-canonical representations SHALL NOT be accepted.
VERIFICATION STEP	`canonical_payload = JCS(token_claims)` → `verify(signature, canonical_payload, public_key)` → `accept | deny`

4.2 Required Token Claims

MANDATORY TOKEN CLAIMS: Normative

iss

Issuer identifier. MUST be a stable, verifiable URI or DID resolvable to a public key set. MUST be cryptographically bound to the signing key via the key registry.

sub

Subject of consent. MUST uniquely identify the data subject or principal. MUST NOT be inferred or defaulted by the enforcement gate.

scope

Enumerated consent scope. MUST be an explicit list of permitted operations. Wildcards MUST NOT be accepted unless the enforcement gate's policy explicitly permits wildcard scope resolution.

iat

Issued-at time (Unix epoch, integer). MUST be present. MUST be verified against current time within configured clock drift tolerance.

exp

Expiration time (Unix epoch, integer). MUST be present. Maximum lifetime: 86400 seconds (24 hours) for Baseline; 3600 seconds (1 hour) for Hardened; 900 seconds (15 minutes) for High-Assurance.

kid

Key identifier referencing the signing key in the trust anchor registry. MUST be present. MUST be resolved against the live registry before signature verification.

jti

Unique token identifier (UUID v4 or cryptographic random, 128-bit minimum). MUST be present for Hardened and High-Assurance profiles. Used for anti-replay enforcement.

ctx_hash

SHA-256 hash of the canonical execution context at the time of consent grant. MUST be verified against the frozen evaluation context presented at enforcement time.

4.3 Deterministic Verification Sequence

The enforcement gate MUST execute token verification in the following deterministic sequence. Deviation from this sequence SHALL be treated as a verification failure, producing a DENY decision.

TOKEN VERIFICATION SEQUENCE SEC-TOK-VER

```
// Step 1: Structural validation
assert token.iss IS PRESENT AND RESOLVES
assert token.sub IS PRESENT
assert token.scope IS PRESENT AND NON-EMPTY
assert token.iat, token.exp, token.kid, token.jti ARE PRESENT

// Step 2: Key resolution
public_key = REGISTRY.resolve_key(token.kid)
if public_key IS NULL OR REVOKED → DENY

// Step 3: Signature verification
canonical = JCS(token.claims)
if not VERIFY(token.sig, canonical, public_key) → DENY

// Step 4: Temporal validation
```

```
now = MONOTONIC_CLOCK.now()
if now < token.iat - DRIFT_TOLERANCE → DENY
if now > token.exp → DENY

// Step 5: Anti-replay (Hardened + High-Assurance)
if JTI_CACHE.contains(token.jti) → DENY
JTI_CACHE.add(token.jti, ttl=token.exp)

// Step 6: Context hash verification
expected_ctx = SHA256(FROZEN_CONTEXT.canonical())
if token.ctx_hash ≠ expected_ctx → DENY

// Step 7: Scope authorization
if not POLICY.permits(token.scope, REQUEST.operation) → DENY

// All checks passed
→ PERMIT
```

4.4 Maximum Token Lifetime by Profile

TABLE 4.4: TOKEN LIFETIME LIMITS

PROFILE	MAXIMUM EXP - IAT	JTI REQUIRED	CTX_HASH REQUIRED
Baseline Security	86400 s (24 h)	SHOULD	SHOULD
Hardened Enterprise	3600 s (1 h)	MUST	MUST
High-Assurance	900 s (15 min)	MUST	MUST

4.5 Distributed Replay Protection Model

In deployments where multiple enforcement gate nodes operate concurrently, whether in active-active clusters, federated deployments, or geographically distributed configurations, single-node JTI cache semantics are insufficient. A token accepted by one node and rejected by another creates an inconsistent enforcement posture exploitable for replay. The following requirements define cluster-wide replay protection semantics.

SEC-TOK-REPLAY-CLUSTER **Cluster-Wide Replay Protection** ● MUST

CLUSTER SEMANTICS	In any deployment with two or more enforcement gate nodes, the JTI cache MUST be shared across all nodes through a distributed cache with strong consistency guarantees. Eventually consistent caches do not satisfy this requirement. Enforcement nodes MUST NOT make independent PERMIT decisions without confirming JTI uniqueness at the cluster level.
REPLAY WINDOW	The maximum permitted replay window, the interval during which a presented JTI MUST be detectable as previously used, SHALL equal the lesser of: the token's `exp` minus `iat` value, or 2× the configured registry TTL for the applicable profile. Replay windows exceeding the profile's registry TTL create an exploitable gap between registry state and replay detection.
CONSISTENCY REQUIREMENT	JTI cache writes MUST achieve linearizable consistency before the PERMIT decision is returned to the caller. Under no circumstance SHALL a PERMIT be returned while the JTI cache write is pending acknowledgment. The JTI write and the ledger write (Section 8.1 SEC-AUDIT-01) MUST both be durably committed before the PERMIT response is issued.
PARTITION BEHAVIOR	When the distributed JTI cache is unreachable from an enforcement gate node, due to network partition, cache node failure, or connectivity loss, the affected enforcement gate MUST enter fail-closed DENY-ALL state. No PERMIT decision SHALL be issued by any gate that cannot confirm JTI uniqueness at the cluster level. Cache partition events MUST be recorded in the audit ledger.
REPLICATION BOUND	JTI replication lag between enforcement nodes MUST be bounded to a value no greater than the configured registry TTL for the applicable profile (Baseline: ≤ 300 s; Hardened: ≤ 60 s; High-Assurance: ≤ 15 s). Any detected replication lag exceeding this bound MUST generate an integrity alert and SHOULD trigger enforcement gate isolation pending replication restoration.

DISTRIBUTED REPLAY CHECK: CLUSTER FLOW SEC-TOK-REPLAY-CLUSTER

```
// Single-node flow (Baseline only)
if CLUSTER_SIZE = 1:
  if LOCAL_JTI_CACHE.contains(token.jti) → DENY
  LOCAL_JTI_CACHE.add(token.jti, ttl=token.exp)

// Cluster flow (Hardened and High-Assurance, MUST)
if CLUSTER_SIZE > 1:

  // Step 1: Verify cache reachability
  if not CLUSTER_JTI_CACHE.reachable() → DENY + ALERT

  // Step 2: Atomic check-and-set with linearizable consistency
  result = CLUSTER_JTI_CACHE.set_if_absent(
    key   = token.jti,
    value = { gate_id, timestamp: NOW() },
    ttl   = min(token.exp - token.iat, 2 * PROFILE.registry_ttl),
    consistency = LINEARIZABLE
  )
  if result = ALREADY_EXISTS → DENY   // replay detected
  if result = WRITE_FAILED   → DENY   // fail-closed on uncertainty

  // Step 3: Verify replication lag is within bound
  if CLUSTER_JTI_CACHE.replication_lag() > PROFILE.registry_ttl:
    ALERT(severity=HIGH, event="JTI_REPLICATION_LAG_EXCEEDED")
    → DENY   // SHOULD for Hardened; MUST for High-Assurance

  // Step 4: Commit ledger entry (linearizable, before PERMIT)
  LEDGER.write_commit(decision_entry)   // fail-closed if write fails

→ PERMIT   // only after both JTI cache and ledger are committed
```

SECTION 5

Trust Anchor and Key Management Model

5.1 Root Trust Anchor Model

The QODIQA trust model is anchored in a verifiable public key registry. Each signing key used to issue consent tokens MUST be traceable to a root trust anchor through a verifiable chain. Self-signed tokens with no trust chain are unconditionally rejected.

SEC-KEY-01 **Trust Chain Requirement** ● MUST

REQUIREMENT	Every signing key presented for token verification MUST be resolvable through a documented trust chain to a root trust anchor registered in the implementation's key registry. Unresolvable keys SHALL produce an immediate DENY decision. Trust chain validation MUST occur at every enforcement gate invocation, not only at session establishment.
KEY STORAGE	Private signing keys MUST NOT be stored in plaintext on disk, in environment variables, in source code, or in configuration files. Hardened and High-Assurance profiles MUST use HSM or OS-provided secure enclave for private key storage.

5.2 Key Rotation Requirements

TABLE 5.2: MANDATORY KEY ROTATION INTERVALS

PROFILE	SIGNING KEY ROTATION	ROOT ANCHOR ROTATION	EMERGENCY ROTATION SLA
Baseline Security	≤ 365 days	≤ 730 days	≤ 72 hours
Hardened Enterprise	≤ 90 days	≤ 365 days	≤ 24 hours

PROFILE	SIGNING KEY ROTATION	ROOT ANCHOR ROTATION	EMERGENCY ROTATION SLA
High-Assurance	≤ 30 days	≤ 180 days	≤ 4 hours

5.3 Key Compromise Response

In the event of confirmed or suspected private key compromise, the following actions MUST be executed in order. No deviation from this sequence is permitted.

TABLE 5.3: KEY COMPROMISE RESPONSE SEQUENCE

#	ACTION	REQUIREMENT
1	Immediate Revocation	The compromised key SHALL be added to the revocation registry within the Emergency Rotation SLA for the applicable profile.
2	Registry Propagation	Revocation SHALL be propagated to all enforcement gates before any new token issuance with a replacement key.
3	Artifact Invalidation	All Decision Artifacts signed with the compromised key SHALL be flagged as potentially invalid and subject to manual review.
4	Replacement Key Generation	A replacement key MUST be generated in an HSM or secure enclave. The previous key's metadata MUST NOT be reused for the replacement key entry.
5	Certification Notification	If applicable, the QODIQA certification authority MUST be notified within 24 hours of confirmed compromise.

5.4 Public Key Discovery

Public keys MUST be discoverable through a signed key manifest served over TLS 1.3. The key manifest MUST include each key's `kid`, algorithm identifier, active status, rotation schedule, and validity period. Key manifests MUST themselves be signed by the root trust anchor.

SEC-KEY-02

Prohibited Key Storage Patterns

● MUST NOT

FORBIDDEN	Implementations MUST NOT store private keys in: plaintext files, environment variables, version control systems, application configuration files, database fields without HSM-backed encryption, or memory regions accessible to untrusted processes.
RATIONALE	Any private key accessible outside a hardware security boundary is effectively a compromised key. The enforcement layer's integrity guarantee is void if signing keys are recoverable through software-layer attacks.

SEC-KEY-03

High-Assurance Hardware Key Protection

● MUST

CERTIFICATION REQUIREMENT	For implementations targeting the High-Assurance deployment profile (Section 12), all signing keys used in consent token issuance, registry snapshot signing, and audit ledger anchoring MUST be generated and stored within an HSM achieving FIPS 140-2 Level 3 certification or equivalent (e.g., Common Criteria EAL 4+). Software-only key storage does not satisfy this requirement under any interpretation.
NON-EXPORTABILITY	Private signing keys in High-Assurance deployments MUST be configured as non-exportable within the HSM. Any HSM configuration that permits plaintext key export, including administrative export modes, backup-in-plaintext features, or wrapped export without separate authorization controls, MUST be explicitly disabled and the disabled state MUST be verified as part of deployment validation. This requirement applies permanently for the lifetime of the key.
KEY USAGE AUDIT LOGGING	The HSM MUST be configured to produce a tamper-evident audit log of all key usage events. Logged events MUST include: key identifier, requesting process identity, timestamp (synchronized with the implementation's authenticated time source), operation type (sign / verify / wrap / unwrap), and outcome. HSM audit logs MUST be archived externally to the HSM within the retention window specified by Section 8.4 (minimum 7 years) and MUST be protected against deletion and modification using the same hash-chain integrity model specified in Section 8.
DUAL-CONTROL KEY ACTIVATION	Root trust anchor keys and key-encryption keys in High-Assurance deployments MUST require dual-control activation, the two-person rule. No single operator SHALL be capable of activating, rotating, or destroying a root or key-encryption key unilaterally. The dual-control mechanism MUST be enforced at the HSM hardware level, not solely by organizational policy or software access controls. Dual-control key activation events MUST be recorded in both the HSM audit log and the implementation's audit ledger.

TAMPER-EVIDENT HSM LOGGING	The HSM's internal audit log MUST be tamper-evident through cryptographic means. If the HSM does not natively support tamper-evident log output, the implementation MUST immediately forward HSM events to an external, append-only log sink. The latency between HSM event occurrence and external log commit MUST not exceed 5 seconds in High-Assurance deployments. Any gap in HSM log continuity MUST be treated as a potential tampering event and MUST trigger the incident response procedure defined in Section 13.
APPLICABILITY	SEC-KEY-03 applies exclusively to the High-Assurance deployment profile. Hardened Enterprise profile implementations SHOULD meet the FIPS 140-2 Level 3 and non-exportability requirements; key usage audit logging MUST be met at the Hardened tier. Dual-control activation MUST at the Hardened tier only when root key operations are performed. Baseline profile implementations are exempt from SEC-KEY-03.

SECTION 5.5

High-Assurance Key Management Model

At the High-Assurance deployment profile, key management is not a configuration decision; it is a structural requirement. The following subsection defines the mandatory key hierarchy, offline root anchor requirement, quorum activation model, key ceremony obligation, and blast radius containment model that collectively constitute the High-Assurance key management architecture. These requirements extend and make concrete the normative controls established in SEC-KEY-01 through SEC-KEY-03.

5.5.1 Key Hierarchy: Root, Intermediate, and Issuer Separation

The QODIQA High-Assurance key hierarchy MUST be structured as a three-tier chain. The Root Trust Anchor is the ultimate trust source and MUST NOT be used for day-to-day signing operations. The Intermediate Signing Authority is an HSM-resident key used to authorize issuer keys and to sign key manifests. The Consent Token Issuer is the operational signing key whose public key is published in the trust registry and whose private counterpart signs consent tokens at runtime. This tier separation ensures that compromise of any operational key does not transitively compromise the root, and that blast radius is bounded to the scope of the compromised tier.

5.5.2 Offline Root Key Requirement

The Root Trust Anchor private key MUST be stored offline. An offline root key is one that is not connected to any network, is not accessible from any running system, and is only activated through a formal key ceremony procedure under dual-control conditions. The offline root key MUST be used only to: sign the intermediate key at initial deployment, rotate the intermediate key on its scheduled rotation interval, and respond to confirmed intermediate key compromise. All other signing operations MUST use the intermediate or issuer tier keys.

5.5.3 Dual Control and Quorum Requirement

All root key operations, including generation, activation, signing, rotation, and destruction, MUST require a minimum quorum of two authorized personnel, each presenting independent authentication credentials. The quorum MUST be enforced at the HSM hardware level. No software-layer access control satisfies this requirement in substitution for hardware-enforced quorum. The quorum size MUST be documented in the implementation's key management policy and verified as part of conformance assessment.

5.5.4 Key Ceremony Requirement

Initial generation of the Root Trust Anchor key and each new intermediate key MUST be performed as a formal key ceremony. A key ceremony is a documented, witnessed procedure in which: the HSM is in a known-clean state verified before the ceremony begins; all ceremony participants are authenticated and their identities are recorded; the generation procedure is executed according to a pre-approved script; the generated key is verified to be non-exportable; and a signed ceremony record is produced and retained as part of the implementation's security configuration record. The ceremony record MUST be retained for the lifetime of the key plus 7 years.

5.5.5 Blast Radius Containment

The three-tier key hierarchy is designed to contain the operational impact of key compromise. Compromise of an issuer key requires immediate revocation of that key and rotation to a new issuer key, but does not invalidate the intermediate or root tier. Compromise of the intermediate key requires rotation of all issuer keys authorized under it, but does not require root key replacement provided the root key remains uncompromised and offline. Compromise of the root key constitutes a full trust anchor failure requiring a complete key hierarchy rebuild under new ceremonies. Implementations MUST document the blast radius for each tier in their security configuration record.

TABLE 5.5-A: KEY HIERARCHY AND CONTROL REQUIREMENTS

KEY TYPE	STORAGE REQUIREMENT	ACCESS CONTROL MODEL	ROTATION REQUIREMENT	COMPROMISE IMPACT SCOPE
`Root Trust Anchor`	Offline HSM; air-gapped; activated only during key ceremony	Dual-control quorum (minimum 2 authorized personnel); hardware-enforced; ceremony procedure required	≤ 180 days (High-Assurance); rotation via key ceremony only	Full trust hierarchy failure; all issuer and intermediate keys invalidated; complete rebuild required
`Intermediate Signing Authority`	Online HSM; FIPS 140-2 Level 3 minimum; non-exportable; not directly accessible from application layer	Dual-control for root operations; single authorized operator for scheduled manifest signing under audit	≤ 365 days (High-Assurance); triggered by root rotation or confirmed compromise	All issuer keys authorized under this intermediate MUST be rotated; root unaffected if offline root is uncompromised
`Consent Token Issuer`	Online HSM; FIPS 140-2 Level 3 minimum; non-exportable; accessible to enforcement infrastructure under audit	Single authorized operator with HSM authentication; key usage audit logging MUST be active	≤ 30 days (High-Assurance); emergency rotation ≤ 4 hours on confirmed compromise	All tokens signed under compromised key MUST be flagged; revocation propagated before replacement key is active

SECTION 6

Revocation Registry Security

The revocation registry is a critical component of the QODIQA enforcement layer. Its integrity directly determines whether revoked consents can be re-presented fraudulently. The following requirements govern its protection.

SEC-REG-01 Signed Registry Snapshots

● MUST

REQUIREMENT	The revocation registry MUST produce cryptographically signed snapshots at each update. Each snapshot MUST include a monotonically increasing sequence number, a timestamp, the SHA-256 hash of the previous snapshot (forming a hash chain), and a digital signature from the registry's root signing key.
SNAPSHOT FORMAT	`snapshot_hash = SHA-256(prev_hash || seq_num || timestamp || registry_body)` Signed with registry root key (Ed25519 or ECDSA P-256).

SEC-REG-02 Fail-Closed on Registry Unreachability

● MUST

REQUIREMENT	When the revocation registry is unreachable, the enforcement gate MUST deny the request. No cached registry state SHALL be used as a substitute for a live registry response beyond the configured TTL. Fail-open behavior under registry unavailability is a conformance violation at all profile levels.
CACHED STATE TTL	Baseline: ≤ 300 s. Hardened: ≤ 60 s. High-Assurance: ≤ 15 s. After TTL expiry, the cached registry state MUST NOT be used and the gate MUST deny.

6.1 Anti-Downgrade Protection

The enforcement gate MUST verify that the registry snapshot sequence number presented is equal to or greater than the highest sequence number previously observed. A snapshot with a lower sequence number than the locally stored high-water mark SHALL be rejected as a potential downgrade attack.

6.2 Registry Authenticity Verification

On each registry read, the enforcement gate MUST: (1) verify the snapshot signature against the registry root public key; (2) verify hash chain continuity from the previous accepted snapshot; (3) verify sequence number monotonicity; (4) verify timestamp recency within configured TTL bounds. Failure of any verification step MUST produce a registry consultation failure, triggering fail-closed behavior.

SECTION 7

Artifact Hashing and Integrity Model

Decision Artifacts are the cryptographically bound records of enforcement decisions produced by the QODIQA consent gate. Their integrity guarantees the verifiability of the Replay Invariant defined in QODIQA Core v1.0. The following specifies the canonical artifact hash model.

7.1 Formal Artifact Hash Model

CANONICAL ARTIFACT HASH FORMULA — SEC-ART-HASH · V1.0

```
// Canonical artifact hash construction
artifact_id = SHA-256(
    canonical_context  // JCS-serialized frozen evaluation context
  || decision          // "PERMIT" | "DENY" (UTF-8 encoded)
```

```
  || policy_version    // policy document hash at evaluation time
  || consent_ref       // token jti or stable consent identifier
  || timestamp         // enforcement timestamp (monotonic, Unix epoch, 8 bytes)
  || prev_artifact_id  // SHA-256 of preceding artifact (hash chain)
  || model_binary_hash // SHA-256 of model binary at enforcement time
                       // MUST: High-Assurance | SHOULD: Hardened | OPTIONAL: Baseline
                       // When absent in Baseline/Hardened: substitute ZERO_BYTES_32
)

// Policy version component
policy_version = SHA-256(policy_document_canonical)

// Canonical context component
canonical_context = SHA-256(JCS({
  model_id, model_version, operation_id,
  subject_id, scope, data_classification,
  enforcement_gate_id, environment_fingerprint
}))

// Model binary hash component
model_binary_hash = SHA-256(model_binary_content)
// model_binary_content = the exact byte sequence of the model file(s)
// at the time enforcement is invoked. Multi-file models: SHA-256(SHA-256(file_1)
// || SHA-256(file_2) || ... || SHA-256(file_n)) in deterministic sort order.
// Implementations MUST pre-compute and cache this value at model load time.
// model_binary_hash MUST be recomputed upon any model reload or version change.
```

SEC-ART-01 **Artifact Immutability Guarantee** ● MUST

REQUIREMENT	Once written to the artifact store, a Decision Artifact MUST NOT be modified, deleted, or overwritten under any operational circumstance. The artifact store MUST enforce write-once semantics at the storage layer. Any attempt to modify an artifact MUST produce an integrity violation event in the audit ledger.
HASH CHAIN	Each artifact MUST include the SHA-256 hash of the immediately preceding artifact. The initial artifact in a chain uses a zero-value predecessor hash (32 zero bytes). Any break in the hash chain constitutes a tamper event.
COLLISION RESISTANCE	The artifact hash construction MUST include all listed components without exception. Omission of any component reduces the collision resistance of the hash space against adversarial context engineering (THR-08).

SEC-ART-02 **Model Binary Integrity Binding** ● MUST

PROFILE APPLICABILITY	The `model_binary_hash` component of the canonical artifact hash formula MUST be included in High-Assurance profile deployments. It SHOULD be included in Hardened Enterprise profile deployments. It is OPTIONAL in Baseline profile deployments. In profiles where `model_binary_hash` is not included, the artifact hash construction MUST substitute ZERO_BYTES_32 in the position of `model_binary_hash` to maintain a fixed-structure hash input and prevent hash construction ambiguity.
HASH COMPUTATION	The `model_binary_hash` SHALL be computed as SHA-256 of the model binary content. For models represented as multiple files, the hash SHALL be computed as SHA-256 of the concatenated SHA-256 hashes of each file, taken in a deterministic sort order (e.g., lexicographic by file path relative to the model root). Implementations MUST pre-compute and cache the `model_binary_hash` at model load time. The cached value MUST be invalidated and recomputed upon any model reload, model version change, or detected modification of model binary content.

INTEGRITY GUARANTEE	The inclusion of `model_binary_hash` in the artifact hash ensures that enforcement decisions are cryptographically bound to a specific, verifiable model binary state. A Decision Artifact produced under model binary state M is computationally infeasible to reinterpret as having been produced under any model binary state M' where M' ≠ M. This binding prevents silent model substitution (THR-07) from producing artifacts that are indistinguishable from legitimately authorized artifacts.
MODEL STATE CHANGE RESPONSE	When a model binary change is detected at runtime, including modifications to model weights, model configuration files, tokenizer files, or any other file included in the model binary hash computation, the enforcement gate MUST: (1) suspend enforcement operations; (2) recompute the `model_binary_hash` from the updated binary content; (3) write a model state change event to the audit ledger; (4) resume enforcement only after the new `model_binary_hash` has been verified against an expected reference value or authorized via an explicit change approval record. Unannounced model binary changes MUST be treated as potential integrity violations.
NON-COVERAGE REMINDER	`model_binary_hash` binds enforcement decisions to model binary state. It does not constitute a guarantee that the model weights are safe, unbiased, or compliant with any behavioral standard. Verification that the bound binary state is the intended and authorized model version remains the responsibility of the operating organization. See Section 14 (Non-Coverage Disclosure).

7.2 Replay Verification via Artifact Hash

To verify that a historical enforcement decision is reproducible under the same conditions, a verifier SHALL: (1) reconstruct the canonical context from available records; (2) recompute the artifact hash using the same formula; (3) compare against the stored artifact ID. A mismatch indicates either context drift, context manipulation, or artifact tampering, all of which constitute enforcement integrity violations.

SECTION 8

Audit Ledger Integrity Requirements

8.1 Hash Chain Format

The audit ledger MUST be maintained as a hash chain. Each ledger entry MUST include the SHA-256 hash of the preceding entry, forming an unbroken sequence from ledger initialization. The ledger chain provides tamper evidence: any modification to a historical entry invalidates all subsequent entry hashes.

AUDIT LEDGER ENTRY STRUCTURE — SEC-AUDIT-CHAIN

```
ledger_entry = {
  seq_num:        uint64,          // monotonically increasing
  timestamp:      unix_epoch_ns,   // nanosecond precision, monotonic clock
  event_type:     "PERMIT" | "DENY" | "REVOCATION" | "KEY_ROTATION" | "INTEGRITY_VIOLATION",
  artifact_id:    bytes32,         // SHA-256 of Decision Artifact
  gate_id:        string,          // enforcement gate identifier
  subject_hash:   bytes32,         // SHA-256 of subject_id (privacy-preserving)
  prev_entry_hash:bytes32,         // SHA-256 of preceding ledger entry
  entry_sig:      bytes            // Ed25519 signature over canonical entry
}

entry_hash = SHA-256(canonical(ledger_entry_without_entry_sig))
```

SEC-AUDIT-01 **Write-Before-Execute Requirement** ● MUST

REQUIREMENT	The audit ledger entry for an enforcement decision MUST be durably committed before the enforcement gate returns any response to the calling system. A PERMIT decision that is returned before its ledger entry is committed violates the write-before-execute invariant. On ledger write failure, the enforcement gate MUST return DENY.
DURABILITY	Durable commitment requires fsync or equivalent at the storage layer, or acknowledgment from a distributed consensus system. In-memory buffering without persistence does not satisfy this requirement.

8.2 Append-Only Storage

The audit ledger storage backend MUST enforce append-only semantics. The storage system MUST prevent modification or deletion of any existing ledger entry through all available interfaces. At the Hardened and High-Assurance profiles, the ledger MUST be replicated to a minimum of two geographically or logically separated storage backends.

8.3 Tamper Detection

A tamper detection procedure MUST be executable on demand and MUST run automatically at a frequency no less than the Ledger Integrity Verification interval for the applicable profile (Baseline: 24h; Hardened: 1h; High-Assurance: 15min). The procedure SHALL verify: hash chain continuity, signature validity for each entry, sequence number monotonicity, and timestamp monotonicity.

8.4 Archival Requirements

Audit ledger entries MUST be retained for a minimum of 7 years. Archived entries MUST retain their hash chain continuity with active entries. Archival storage MUST maintain integrity verification capability equivalent to active storage.

Informative: Merkle Tree Variant. Implementations MAY construct a Merkle tree over ledger entry batches to enable efficient range-proof verification of large audit archives. The Merkle root MUST be anchored to a signed, timestamped manifest. This variant does not reduce the hash chain requirement; it supplements it.

SECTION 9

Time and Clock Security

Temporal integrity is a foundational property of the QODIQA enforcement layer. Token expiry, artifact timestamps, audit ledger ordering, and TTL enforcement all depend on reliable, manipulation-resistant time. The following requirements govern clock security at all profile levels.

TABLE 9.1: CLOCK SECURITY REQUIREMENTS BY PROFILE

REQUIREMENT	BASELINE	HARDENED	HIGH-ASSURANCE
NTP Synchronization	MUST	MUST (authenticated NTP / NTPsec)	MUST (authenticated NTP + GPS/PTP secondary)
Maximum Clock Drift Tolerance	±30 s	±5 s	±1 s
Monotonic Clock for Artifacts	MUST	MUST	MUST
Anti-Rollback Protection	SHOULD	MUST	MUST
Fail-Closed on Clock Uncertainty	MUST	MUST	MUST

SEC-CLK-01 **Monotonic Clock Requirement** ● MUST

REQUIREMENT	All timestamps recorded in Decision Artifacts and audit ledger entries MUST be sourced from a monotonic clock. Wall-clock adjustments (NTP step corrections, DST transitions, leap seconds) MUST NOT retroactively modify artifact or ledger timestamps. If a monotonic clock source is unavailable, the enforcement gate MUST enter a DENY-ALL fail-closed state until a valid monotonic source is restored.
ANTI-ROLLBACK	The enforcement gate MUST store the highest previously observed monotonic timestamp. Any new timestamp that is less than this high-water mark MUST be treated as a clock rollback event and SHALL trigger a fail-closed state and an integrity violation audit entry.

When clock synchronization error exceeds the configured drift tolerance for the applicable profile, the enforcement gate MUST enter fail-closed state for all decisions requiring temporal validation. The gate MUST remain in fail-closed state until synchronization is restored and verified.

SECTION 10

Network and Transport Security Requirements

SEC-NET-01 **Minimum Transport Layer Security** ● MUST

MINIMUM VERSION	TLS 1.2. TLS 1.0 and TLS 1.1 MUST NOT be used. Plaintext HTTP transport for any QODIQA enforcement component is unconditionally forbidden at all profile levels.
PREFERRED VERSION	TLS 1.3. All Hardened and High-Assurance implementations MUST use TLS 1.3 exclusively. TLS 1.2 support may be retained for legacy registry compatibility at Baseline only.

CIPHER SUITES	TLS 1.3: TLS_AES_256_GCM_SHA384, TLS_CHACHA20_POLY1305_SHA256. TLS 1.2 (Baseline only): TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384, TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384. All RC4, 3DES, and export cipher suites MUST NOT be negotiated.

10.1 Certificate Validation

All TLS connections to registry, audit log, and key management endpoints MUST perform full certificate chain validation including revocation checking (OCSP or CRL). Certificate pinning SHOULD be implemented for registry and key management endpoints at Hardened and High-Assurance profiles. Self-signed certificates MUST NOT be trusted in production enforcement environments.

10.2 Mutual TLS

Mutual TLS (mTLS) authentication SHOULD be implemented for all inter-component communications within the QODIQA enforcement layer at Hardened profile, and MUST be implemented at High-Assurance profile. Client certificate validation MUST follow the same chain validation and revocation checking requirements as server certificate validation.

SEC-NET-02 **Plaintext Transport Prohibition** ● MUST NOT

PROHIBITION	No QODIQA enforcement component SHALL transmit consent tokens, decision artifacts, revocation data, key material, or audit records over plaintext (unencrypted) transport. This prohibition applies to all network interfaces including loopback, container-internal, and service-mesh communications at Hardened and High-Assurance profiles.

SEC-NET-PFS

Perfect Forward Secrecy Requirement

MUST

PFS MANDATE	All TLS connections used by QODIQA enforcement components MUST provide Perfect Forward Secrecy (PFS). PFS ensures that compromise of long-term server private keys does not enable retrospective decryption of captured session traffic. This requirement applies to all profiles, including Baseline.
TLS 1.2 KEY EXCHANGE RESTRICTION	Where TLS 1.2 is permitted (Baseline profile only, per SEC-NET-01), ONLY ephemeral Elliptic Curve Diffie-Hellman (ECDHE) key exchange cipher suites SHALL be used. TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 and TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 are approved. Static RSA key exchange (cipher suites of the form TLS_RSA_WITH_*) MUST NOT be negotiated under any circumstances. DHE with finite-field groups SHOULD NOT be used; if used, the minimum Diffie-Hellman group size MUST be 2048-bit.
STATIC RSA PROHIBITION	Static RSA key exchange cipher suites MUST NOT be configured, negotiated, or accepted on any QODIQA enforcement component in any profile. This prohibition is unconditional. The use of RSA certificates for server authentication is permitted; only the use of RSA for key encipherment (as in TLS_RSA_WITH_* suites) is prohibited. TLS server configurations MUST explicitly remove all TLS_RSA_WITH_* cipher suite entries from the server's cipher suite list.
NON-PFS CIPHER SUITE PROHIBITION	The following cipher suite categories MUST NOT be configured or negotiated: (1) Static RSA key exchange (TLS_RSA_WITH_*); (2) Anonymous key exchange (TLS_DH_anon_*, TLS_ECDH_anon_*); (3) NULL encryption or NULL authentication suites; (4) Export-grade cipher suites (EXP_*); (5) Fixed Diffie-Hellman with server key reuse (TLS_DH_DSS_*, TLS_DH_RSA_*). Cipher suites MUST be configured using an explicit allowlist; default cipher suite sets provided by TLS libraries SHALL NOT be accepted without review against this prohibition list.
TLS 1.3 INHERENT PFS	TLS 1.3 provides PFS for all cipher suites defined in its specification. Implementations using TLS 1.3 exclusively (Hardened and High-Assurance profiles) satisfy the key exchange requirements of this normative block inherently. However, the cipher suite allowlist and non-PFS prohibition requirements MUST still be enforced to prevent downgrade negotiation on connections that may accept TLS 1.2 as a fallback.
VERIFICATION	Deployment validation MUST include a TLS cipher suite scan of all enforcement component endpoints. Any discovered endpoint negotiating a non-PFS cipher suite under any TLS version SHALL be treated as a configuration violation requiring immediate remediation. Cipher suite scans MUST be repeated after any TLS library upgrade or TLS configuration change.

SECTION 10.3

Performance and Latency Constraints

The enforcement operations defined in this profile introduce measurable latency at each stage of the consent evaluation path. This section enumerates the primary latency contributors at each SUP component. It does not provide optimization guidance and does not authorize any weakening of normative requirements on the basis of performance considerations. Performance constraints are enumerated here to ensure that implementations plan for and account for the operational cost of enforcement, not to create grounds for bypassing enforcement controls.

Every operation listed in the table below is a normative requirement of this profile or of QODIQA Core v1.0. The latency it introduces is an inherent property of verifiable enforcement. No conformant implementation may eliminate any listed operation in the name of performance without simultaneously creating a conformance failure.

TABLE 10.3-A: ENFORCEMENT LATENCY CONTRIBUTORS

COMPONENT	OPERATION	LATENCY SENSITIVITY	SCALING CONSIDERATION	FAIL-CLOSED IMPACT
SUP.CONSENT-GATE	Ed25519 signature verification of consent token (SEC-ALG-01)	Low absolute latency (sub-millisecond on modern hardware); high sensitivity to HSM round-trip if verification is HSM-delegated	Verification is stateless and parallelizable per gate instance; not a horizontal scaling bottleneck under normal load	Verification failure produces immediate DENY; no retry path; failure adds zero marginal latency to the enforcement decision
SUP.REGISTRY	Registry snapshot consultation and signature verification (SEC-REG-01, SEC-REG-02)	High sensitivity; network round-trip to registry on each invocation when TTL is expired; TTL cache reduces frequency at cost of staleness risk	Registry response latency is bounded by the configured TTL; lower TTL increases consultation frequency and latency exposure; TTL MUST NOT be increased beyond profile limits to reduce latency	Registry unreachability MUST trigger fail-closed DENY-ALL; enforcement gate blocked until registry restored; no fallback to stale cache beyond TTL
SUP.AUDIT-LEDGER	Durably committed audit ledger write before PERMIT response (SEC-AUDIT-01)	Highest latency contributor; blocking write to persistent storage with fsync or distributed consensus acknowledgment is on the critical path of every PERMIT decision	Ledger write throughput bounds enforcement gate PERMIT throughput; distributed consensus adds latency proportional to quorum round-trip; replication required at Hardened and High-Assurance profiles	Ledger write failure MUST produce DENY; enforcement gate cannot issue PERMIT until write is confirmed; storage degradation directly degrades enforcement throughput

COMPONENT	OPERATION	LATENCY SENSITIVITY	SCALING CONSIDERATION	FAIL-CLOSED IMPACT
SUP.ARTIFACT-STORE	Artifact hash chain computation and write at enforcement decision time (SEC-ART-01)	Moderate latency; SHA-256 computation is fast; serialization of artifact and storage write add bounded overhead; hash chain linkage requires reading the previous artifact hash before writing	Sequential artifact write with hash chain linkage serializes artifact production per gate; parallel gate deployments require coordination to maintain chain integrity across nodes	Artifact write failure MUST trigger fail-closed state; enforcement gate cannot proceed until artifact chain integrity is confirmed
SUP.KEY-STORE	HSM-delegated signing operation for issuer key operations (SEC-KEY-03)	Moderate-to-high latency when signing is HSM-delegated; HSM round-trip adds 1–50 ms depending on HSM model and interface; applies to token issuance, not to per-invocation verification	Token issuance latency is not on the per-request critical path; verification uses public key and does not require HSM; HSM latency affects token issuance throughput, not enforcement gate throughput	HSM unavailability blocks new token issuance; does not directly block enforcement gate verification of existing tokens within their validity period

This profile defines deterministic enforcement constraints that SHALL NOT be bypassed, relaxed, or conditionally disabled for performance optimization under any deployment scenario.

SECTION 11

Side-Channel and Non-Deterministic Risk Mitigation

The QODIQA enforcement gate is a deterministic decision function. Its determinism is a specification requirement, not an implementation preference. The following requirements protect this property from erosion through side-channels, non-deterministic dependencies, and evaluation contamination.

SEC-DET-01 **Prohibition of Randomness in Decision Path** ● MUST

REQUIREMENT The enforcement gate's decision function MUST be fully deterministic. No random number generation, probabilistic sampling, stochastic policy resolution, or AI-model inference SHALL occur within the evaluation path. Randomness is permissible only in key generation, nonce generation for anti-replay, and JTI generation, all outside the decision path.

SECURE RANDOM	Where random bytes are required (key generation, nonce generation), implementations MUST use a cryptographically secure pseudorandom number generator (CSPRNG) seeded from the operating system entropy source (e.g., `/dev/urandom`, `getrandom()`, or platform HSM). Userspace PRNGs without OS entropy seeding MUST NOT be used.

11.1 Evaluation Function Isolation

The enforcement gate's policy evaluation function MUST be isolated from external network callouts, file system reads, database queries, and inter-process communication during the evaluation window. Any external dependency required for evaluation (policy document, registry state) MUST be resolved before the evaluation window opens and frozen in the evaluation context.

11.2 Dependency Isolation Rules

- No outbound network calls during the enforcement evaluation window.
- No reads from mutable file system paths during the evaluation window.
- No shared mutable state between concurrent evaluation invocations.
- No JIT compilation or dynamic code loading within the evaluation function.
- No access to environment variables within the evaluation function.
- Evaluation function dependencies MUST be resolved, validated, and immutably bound before evaluation begins.

11.3 Constant-Time Comparison

All cryptographic comparisons within the enforcement gate, including signature verification results, hash comparisons, and token claim validation, MUST use constant-time comparison functions where the compared values could reveal information through timing side-channels. Variable-time string comparison functions (e.g., `strcmp`, operator `==` on byte arrays) MUST NOT be used for security-sensitive comparisons.

SECTION 12

Secure Deployment Profiles

Three tiered security deployment profiles are defined. Each tier inherits all requirements of the preceding tier and adds incremental cryptographic and operational constraints. Profile selection MUST be documented in the implementation's security configuration manifest and verified by conformance testing.

PROFILE I

Baseline Security

Minimum conformant deployment for internal or low-risk AI systems. Satisfies all MUST requirements in this profile. TLS 1.2+, SHA-256, Ed25519 or ECDSA P-256, software key storage acceptable, 24h token lifetime, 5-minute registry TTL.

PROFILE II

Hardened Enterprise

Required for externally exposed or regulated deployments. TLS 1.3 mandatory, HSM key storage required, mTLS SHOULD, 1-hour token lifetime, 60s registry TTL, authenticated NTP, anti-replay JTI cache required, 1-hour ledger integrity scan.

PROFILE III

High-Assurance

Required for safety-critical, high-sensitivity, or regulatory-classified AI deployments. TLS 1.3 exclusive, HSM mandatory, mTLS MUST, 15-minute token lifetime, 15s registry TTL, GPS/PTP time source, geographically replicated audit ledger, 15-minute integrity scan, formal key management procedures.

TABLE 12.1: SECURITY PROFILE COMPARISON MATRIX

CONTROL	BASELINE	HARDENED	HIGH-ASSURANCE
Signing Algorithm	Ed25519 or ECDSA P-256	Ed25519: MUST	Ed25519: MUST
Hash Algorithm	SHA-256	SHA-256 or SHA-384	SHA-384: MUST
TLS Version	≥ 1.2	1.3 exclusive	1.3 exclusive
Key Storage	Software acceptable	HSM required	HSM + dual-control
mTLS	OPTIONAL	SHOULD	MUST
Token Lifetime Max	24 hours	1 hour	15 minutes
JTI Anti-Replay	SHOULD	MUST	MUST
Registry TTL	300 s	60 s	15 s
Ledger Replication	Single	Dual	Dual + geographic
Key Rotation	≤ 365 days	≤ 90 days	≤ 30 days
Integrity Scan Interval	24 hours	1 hour	15 minutes

SECTION 13

Compromise and Incident Response Model

The following response procedures are mandatory for the categories of security events most likely to affect QODIQA enforcement layer integrity. These procedures are procedural requirements, not legal guidance.

13.1 Key Compromise Response

- Immediately revoke the compromised key in the registry within the Emergency Rotation SLA for the applicable profile.
- Propagate revocation to all enforcement gates before issuing replacement key. Verify propagation through active monitoring.
- Flag all Decision Artifacts produced under the compromised key as requiring manual integrity review.
- Generate replacement key in HSM or secure enclave. Assign a new, distinct key identifier. Do not reuse any attribute of the compromised key entry.
- Notify QODIQA certification authority within 24 hours if the implementation holds a QODIQA certification.
- Produce a Post-Incident Report (PIR) within 72 hours identifying root cause, timeline, and remediation actions.

13.2 Registry Compromise Response

- Immediately suspend enforcement gate operations. Enter fail-closed DENY-ALL state.
- Restore registry from the most recent verified signed snapshot. Validate hash chain continuity from restoration point.
- Verify snapshot signature authenticity before restoration. Do not restore from an unsigned or unverifiable snapshot.
- Audit all enforcement decisions made between the last verified snapshot and the compromise detection point. Flag as requiring manual review.

- Rotate registry signing keys as per the Key Compromise Response procedure above.

13.3 Artifact Mismatch Detection Response

When replay verification produces a hash mismatch, i.e., the recomputed artifact hash does not match the stored artifact ID, the following response SHALL be initiated: (1) the mismatched artifact is quarantined and flagged for investigation; (2) an integrity violation event is written to the audit ledger; (3) any active enforcement session associated with the artifact is terminated; (4) the incident is escalated for forensic review. Mismatches SHALL NOT be silently discarded or treated as benign data drift.

13.4 Audit Chain Corruption Procedure

If tamper detection identifies a broken audit chain (hash discontinuity), the following procedure applies: (1) isolate the enforcement gate to prevent further audit writes; (2) preserve a forensic copy of the current ledger state; (3) identify the last valid hash chain segment; (4) restore from the last verified archive checkpoint; (5) recover entries between the checkpoint and compromise point from secondary replicas if available; (6) document the corrupted segment in the restoration metadata. Under no circumstances SHALL a repaired audit chain suppress evidence of the tampering event itself.

SECTION 14

Non-Coverage Disclosure

This profile explicitly delimits its scope. The following classes of security concern are outside the boundary of this document. Transparency on these exclusions is essential to prevent misrepresentation of QODIQA's security posture.

This profile does NOT: secure AI model weights or training data; prevent model bias, adversarial prompt injection, or inference-time manipulation; replace an organization's enterprise security architecture or information security management system; eliminate insider risk beyond the enforcement layer boundary defined in Section 1.4; provide guidance on data encryption at rest beyond the enforcement layer; govern application-layer authentication or authorization of end users; secure the network infrastructure underlying the enforcement layer beyond TLS transport requirements; guarantee the correctness of consent records entered by human operators; or provide legal compliance certification of any kind.

Organizations deploying QODIQA enforcement infrastructure MUST treat this profile as one component of a broader security architecture. The enforcement layer's cryptographic guarantees apply only within the System Under Protection boundary defined in Section 1.4. Security properties outside that boundary require independent architectural controls.

SEC-SCOPE-01 **Boundary Acknowledgment Requirement** ● MUST

REQUIREMENT

Implementations claiming conformance with this Security Profile MUST include a documented acknowledgment that the profile's guarantees are bounded to the SUP definitions in Section 1.4. Marketing materials, certification claims, or technical documentation MUST NOT represent QODIQA security profile conformance as providing guarantees outside the defined SUP boundary.

14.1 Additional AI-Specific Exclusions

The following classes of concern are explicitly outside the scope of this Security and Cryptographic Profile. They are enumerated separately because they arise specifically in the context of AI systems and may be incorrectly attributed to this profile's scope in architectural or compliance contexts. No requirement in this document addresses, mitigates, or characterizes any of the following.

EXPLICIT OUT-OF-SCOPE: AI SYSTEM BEHAVIOR

EXCLUSION CATEGORY	SCOPE STATEMENT
Model hallucination behavior.	This profile does not govern, detect, or mitigate outputs produced by an AI model that are factually incorrect, fabricated, or internally inconsistent. The enforcement layer operates on consent and authorization, not on output semantic correctness.
Prompt injection attacks.	This profile does not govern adversarial inputs designed to override model behavior, hijack instruction following, or exfiltrate context through model output. Prompt injection operates within the model's inference path, which is outside the SUP boundary defined in Section 1.4.
Training data governance.	This profile does not govern the composition, provenance, licensing, or safety properties of training data used to produce AI model weights. Training data governance is upstream of the enforcement layer boundary.
Model alignment and safety tuning.	This profile does not govern whether a model has been trained to follow safety guidelines, refuse harmful requests, or exhibit aligned behavior. RLHF, constitutional AI approaches, or other alignment methods are not within the scope of runtime consent enforcement.
Output semantic correctness.	This profile does not guarantee, verify, or audit whether AI model outputs are accurate, appropriate, safe, or compliant with any content policy. The enforcement layer determines whether an invocation was authorized under a valid consent token, not whether the resulting output is correct or acceptable.

These exclusions are permanent and apply at all deployment profile tiers. Conformance with this profile does not imply any guarantee regarding the above properties.

CLOSING STATEMENT

Institutional Closing Statement

Deterministic consent enforcement is not a statement of policy. It is a runtime guarantee. A guarantee is only as strong as the cryptographic substrate that anchors it. Without signature binding, hash-chained audit integrity, revocation registry protection, and clock security, the QODIQA enforcement model remains logically sound but operationally falsifiable.

This profile does not introduce new enforcement semantics. It specifies the minimum cryptographic conditions under which QODIQA Core v1.0's enforcement semantics (determinism, fail-closed behavior, replay verifiability, audit immutability) hold in adversarial conditions. An implementation that satisfies QODIQA Core's logical requirements but omits the cryptographic requirements herein provides a weaker guarantee than its certification level implies.

The threat categories in Section 2 are not hypothetical. They are operational attack classes observed against deployed runtime enforcement infrastructure. Each maps to a specific cryptographic control defined in this profile. Omission of any required control creates a specific exploitable gap, not a theoretical weakness.

WHERE ENFORCEMENT MUST BE MORE THAN INTENT:

determinism must be cryptographically anchored, fail-closed behavior requires integrity guarantees, and runtime consent enforcement without strong cryptography is structurally incomplete.

The algorithms defined as approved in this profile reflect current NIST guidance and cryptographic best practice as of the document date. Implementations MUST maintain an active cryptographic configuration review process and update algorithm selections as NIST guidance evolves. This profile will be versioned in parallel with QODIQA Core Standard revisions.

† All normative requirements in this document use RFC 2119 / RFC 8174 language: MUST, SHALL, SHOULD, MAY, MUST NOT, SHALL NOT, SHOULD NOT. Lowercase usage denotes non-normative text.

**This profile is classified as a companion specification within the QODIQA standard corpus. It does not supersede the Core Standard but supplements its security requirements.

* Algorithm approval status is current as of March 2026. Organizations MUST monitor NIST Post-Quantum Cryptography (PQC) standardization developments and prepare migration plans for quantum-resistant signature algorithms for deployments extending beyond 2030.

ANNEX X: NORMATIVE REFERENCE

Cryptographic Trust Model

This annex specifies the hierarchical trust structure that underpins all cryptographic operations in a conformant QODIQA enforcement deployment. The trust model is not a policy construct; it is a structural requirement. Every cryptographic verification operation performed at a QODIQA enforcement gate traces to this chain. The chain MUST be complete, verifiable, and non-circular at every point during enforcement operation.

X.1 Hierarchical Trust Chain

The QODIQA trust hierarchy follows a three-tier model in which each tier signs the public key material of the tier below it. A verification at any tier is valid only if the complete chain from that tier to the Root Trust Anchor can be constructed and each link in the chain passes signature verification against the tier above it. There are no lateral trust relationships in this model. Trust flows strictly downward from root to enforcement gate.

QODIQA CRYPTOGRAPHIC TRUST CHAIN — TRUST-CHAIN · V1.0

```
ROOT TRUST ANCHOR
(offline HSM; dual-control; air-gapped)
│
│  Signs: Intermediate Signing Authority public key
│  Verified by: enforcement gate on manifest load
↓
INTERMEDIATE SIGNING AUTHORITY
(online HSM; FIPS 140-2 Level 3; non-exportable)
```



X.2 Deterministic Validation Chain

At runtime, the enforcement gate executes the following trust chain validation sequence for every presented consent token. This sequence MUST be completed in the order specified. No step may be skipped or reordered. Failure at any step produces an immediate DENY decision and an audit ledger entry recording the step at which validation failed.

TABLE X.2-A: DETERMINISTIC VALIDATION CHAIN STEPS

STEP	VALIDATION OPERATION	REQUIREMENT
1	**Root manifest verification.**	The enforcement gate verifies that the loaded key manifest is signed by the Root Trust Anchor public key held in the gate's trust store. This step is performed at manifest load time and on each manifest refresh.

STEP	VALIDATION OPERATION	REQUIREMENT
2	**Intermediate key binding verification.**	The enforcement gate verifies that the Intermediate Signing Authority key listed in the manifest is signed by the Root Trust Anchor. A manifest that does not carry a valid root signature for the intermediate key MUST NOT be trusted.
3	**Issuer key binding verification.**	The enforcement gate verifies that the Consent Token Issuer key referenced by the token's `kid` claim is signed by the Intermediate Signing Authority. An issuer key not bound through a valid intermediate signature MUST NOT be used for token verification.
4	**Token signature verification.**	The enforcement gate verifies the consent token signature against the resolved Consent Token Issuer public key using the algorithm specified by the key manifest. The canonical payload is reconstructed before verification per SEC-TOK-01.
5	**Revocation status verification.**	The enforcement gate confirms that the issuer key is not listed in the revocation registry. This step occurs after signature verification to avoid unnecessary registry calls on structurally invalid tokens.

This five-step chain constitutes the complete trust validation path. An enforcement gate that short-circuits any step, including by caching intermediate verification results across invocations without re-verifying the chain on manifest refresh, is non-conformant. The deterministic validation chain is not optional at any deployment profile tier.

ANNEX A: NON-NORMATIVE

Informative Cryptographic Reference Mapping

INFORMATIVE: NON-NORMATIVE	This annex does not introduce compliance claims. It provides reference mapping between QODIQA cryptographic requirements and corresponding NIST and FIPS standards. No additional requirements are created herein.

The normative requirements in Sections 3–12 of this document are derived from, and consistent with, a body of published cryptographic standards. This annex provides an informative mapping to those standards for implementers, auditors, and reviewers who wish to cross-reference QODIQA requirements against established standards bodies. Mapping to a standard does not constitute a claim of certification or compliance under that standard's audit regime.

A.1 FIPS 180-4: Secure Hash Standard

FIPS 180-4 (Secure Hash Standard, SHS), published by NIST, specifies the SHA family of hash functions including SHA-256, SHA-384, and SHA-512. All hash algorithm requirements in Section 3.2 and Section 7 of this document are consistent with FIPS 180-4 approved algorithms. SHA-256 is the minimum approved hash for artifact construction in Section 7. SHA-384 is required at the High-Assurance profile in certain contexts. SHA-512 is approved for use. MD5 and SHA-1 prohibitions in Section 3.3 align with NIST's withdrawal of these algorithms from approved use.

TABLE A.1: FIPS 180-4 ALGORITHM MAPPING

QODIQA SECTION	REQUIREMENT	FIPS 180-4 BASIS	STATUS IN STANDARD
Section 3.2	SHA-256 minimum for hashing	FIPS 180-4 Section 6.2	Approved
Section 3.2	SHA-384 for High-Assurance	FIPS 180-4 Section 6.3	Approved
Section 3.2	SHA-512 approved	FIPS 180-4 Section 6.4	Approved
Section 3.3	MD5 prohibited	Not in FIPS 180-4	Withdrawn / Not Approved
Section 3.3	SHA-1 prohibited	FIPS 180-4 (deprecated)	Deprecated per NIST SP 800-131A Rev.2
Section 7	Artifact hash: SHA-256	FIPS 180-4 Section 6.2	Approved
Section 7	model_binary_hash: SHA-256	FIPS 180-4 Section 6.2	Approved

A.2 FIPS 186-4 / FIPS 186-5: Digital Signature Standard

FIPS 186-4 (Digital Signature Standard) and the updated **FIPS 186-5** specify the approved digital signature algorithms for federal use. ECDSA P-256 and ECDSA P-384 approved in Section 3.1 are directly specified in FIPS 186-4 Section 6.1 (curves P-256 and P-384). The RSA-PSS algorithm approved in Section 3.1 is specified in FIPS 186-4 Section 5.5. The prohibition on RSA < 2048-bit keys in Section 3.3 aligns with FIPS 186-4 Section 5.1 minimum key size guidance.

The deterministic nonce requirement for ECDSA (RFC 6979) supplements FIPS 186-4 Section 6.3, which does not require deterministic nonce generation but permits it. Ed25519 is specified in FIPS 186-5 Section 5.1 as an approved signature algorithm.

TABLE A.2: FIPS 186-4/186-5 ALGORITHM MAPPING

QODIQA SECTION	ALGORITHM	FIPS REFERENCE	APPROVAL STATUS
Section 3.1 (MUST)	Ed25519	FIPS 186-5 Section 5.1	Approved
Section 3.1 (SHOULD)	ECDSA P-256	FIPS 186-4 Section 6.1 / FIPS 186-5 Section 6.1	Approved
Section 3.1 (SHOULD)	ECDSA P-384	FIPS 186-4 Section 6.1 / FIPS 186-5 Section 6.1	Approved
Section 3.3	RSA < 2048-bit (prohibited)	FIPS 186-4 Section 5.1	Non-compliant (key size)
Section 3.3	Non-deterministic ECDSA (prohibited)	RFC 6979 / FIPS 186-5 Section 6.3	Not recommended
Section 10 SEC-NET-PFS	ECDHE only for TLS 1.2	FIPS 186-4 Section 6.1	Approved key agreement

A.3 NIST SP 800-57: Key Management

NIST SP 800-57 (Recommendation for Key Management, Parts 1–3) provides comprehensive guidance on cryptographic key management including key generation, distribution, storage, use, and destruction. The QODIQA key management requirements in Section 5, including key rotation intervals, HSM storage requirements, key compromise response, and prohibited storage patterns, are consistent with and informed by NIST SP 800-57 Part 1 (General) and Part 2 (Best Practices for Key Management Organizations). The dual-control key activation requirement in SEC-KEY-03 corresponds to NIST SP 800-57 Part 2 guidance on separation of duties for sensitive key operations. Key rotation intervals in Section 5.2 are consistent with NIST SP 800-57 Part 1 Section 5.6 recommendations for cryptoperiods.

TABLE A.3: NIST SP 800-57 KEY MANAGEMENT MAPPING

QODIQA SECTION	REQUIREMENT	SP 800-57 REFERENCE
Section 5.1 SEC-KEY-01	HSM storage for Hardened / High-Assurance	Part 1 Section 8.2.1, Part 2 Section 3.3
Section 5.1 SEC-KEY-02	Prohibited key storage patterns	Part 1 Section 8.2.3
Section 5.1 SEC-KEY-03	FIPS 140-2 Level 3, dual-control	Part 2 Section 2.3 (Separation of Duties)
Section 5.2	Key rotation intervals (365d / 90d / 30d)	Part 1 Section 5.6 (Cryptoperiods)
Section 13.1	Key compromise response	Part 1 Section 8.3 (Key Compromise)
Section 3.4 SEC-ALG-AGILITY-01	12-month algorithm review	Part 1 Section 5.2 (Transition)

A.4 NIST SP 800-63: Identity and Authentication

NIST SP 800-63 (Digital Identity Guidelines) provides the federal standard for digital identity assurance, authentication, and federation. QODIQA's consent enforcement layer operates at the boundary between identity verification and authorization. While QODIQA does not implement identity proofing or authentication (those are upstream of the SUP boundary), the consent token structure defined in Section 4 draws from principles consistent with SP 800-63B (Authentication and Lifecycle Management) regarding token lifetime limits, session management, and revocation requirements. The registry consultation requirement (Section 6) is structurally consistent with SP 800-63C guidance on federation assertion validity and revocation. Implementers integrating QODIQA within a broader digital identity infrastructure SHOULD refer to SP 800-63 for identity and authentication controls upstream of the SUP enforcement boundary.

A.5 NIST Post-Quantum Cryptography Standardization

The NIST Post-Quantum Cryptography (PQC) standardization process has yielded initial standards in 2024. The primary PQC signature algorithms now standardized are:

TABLE A.5: NIST PQC STANDARDIZED SIGNATURE ALGORITHMS

ALGORITHM	NIST STANDARD	BASIS	SIGNATURE SIZE	RELEVANCE TO QODIQA SECTION 3.4
CRYSTALS-Dilithium (ML-DSA)	FIPS 204	Module lattice	2.4 KB – 4.6 KB	Primary PQC candidate for Ed25519 migration path
FALCON (FN-DSA)	FIPS 206	NTRU lattice	0.6 KB – 1.3 KB	Compact signatures; constrained environments
SPHINCS+ (SLH-DSA)	FIPS 205	Hash-based stateless	7.8 KB – 49.8 KB	Hash-based; no algebraic assumptions

The cryptographic agility requirements in Section 3.4 (SEC-ALG-AGILITY-01) directly address the PQC transition. Implementations targeting active deployment beyond 2030 MUST include a documented PQC migration readiness plan, as required by SEC-ALG-AGILITY-01. The NIST PQC migration guidance documents (NIST IR 8413, NIST SP 800-131A Rev.2, and NIST PQC Migration FAQ) provide additional implementation guidance beyond the scope of this document.

The dual-signature transition model defined in Section 3.4.2 is directly applicable to the PQC migration scenario: an implementation may produce artifacts signed under both Ed25519 and ML-DSA (Dilithium) during the transition period, satisfying both classical and post-quantum verifiers simultaneously until the deployment is fully migrated.

A.6 FIPS 140-2 / FIPS 140-3: Security Requirements for Cryptographic Modules

FIPS 140-2 (Security Requirements for Cryptographic Modules) and its successor **FIPS 140-3** (aligned with ISO/IEC 19790:2012) define four levels of cryptographic module security. QODIQA SEC-KEY-03 requires FIPS 140-2 Level 3 or equivalent for High-Assurance profile HSMs. The levels are summarized for reference:

TABLE A.6: FIPS 140-2 SECURITY LEVEL SUMMARY (INFORMATIVE)

LEVEL	PHYSICAL SECURITY	KEY EXPORT	AUTHENTICATION	QODIQA APPLICABILITY
Level 1	No physical requirements	Not controlled	Not required	Not sufficient for any QODIQA profile requiring HSM
Level 2	Tamper-evident coatings	Controlled	Role-based	Minimum for Hardened profile HSM use
Level 3	Tamper-detection/response; zeroization on tamper	Encrypted only; private key non-exportable	Identity-based; multi-factor	**Required for High-Assurance profile (SEC-KEY-03)**
Level 4	Complete physical protection envelope; environmental failure protection	Encrypted only; private key non-exportable	Identity-based; multi-factor	Exceeds minimum; acceptable for High-Assurance

FIPS 140-3 replaces FIPS 140-2 for new certifications initiated after September 22, 2021. FIPS 140-2 certificates remain valid until their expiration. Implementations procuring new HSM hardware SHOULD prefer FIPS 140-3 certified modules at the equivalent security level. FIPS 140-3 Level 3 satisfies the FIPS 140-2 Level 3 requirement in SEC-KEY-03.

> **Annex A Non-Normative Confirmation.** Nothing in this annex supersedes, modifies, weakens, or extends the normative requirements of Sections 1–14 of this document. The mappings provided are informative and for cross-reference purposes only. Conformance to this Security and Cryptographic Profile is defined exclusively by the normative blocks in Sections 1–14. Reference to a NIST or FIPS standard in this annex does not constitute a claim that an implementation of this document achieves certification, validation, or compliance status under any external audit regime without separate independent evaluation.

DOCUMENT STATUS

Document Status and Classification

This document defines the mandatory cryptographic, key management, and integrity requirements for high-assurance implementations of QODIQA: a runtime consent enforcement framework for artificial intelligence systems. The Security and Cryptographic Profile is not presented as a general security advisory, but as a deterministic, implementable security hard profile specifying precisely which algorithms are permitted, how consent tokens are secured, how trust anchors are managed, and how audit integrity is preserved across the full enforcement lifecycle.

The material contained herein is intended for:

- Systems architects and cryptographic infrastructure engineers
- Enterprise AI platform designers
- Security and governance officers
- Regulatory and compliance stakeholders
- Organizations building or operating high-assurance AI enforcement systems

This document should be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0

- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Threat Model and Abuse Case Specification, Version 1.0
- QODIQA: Implementation Playbook for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dușescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-SEC-2026-001
TITLE	Security and Cryptographic Profile for Runtime Consent Enforcement
SUBTITLE	Cryptographic, Key Management, and Integrity Requirements for Deterministic Runtime Consent Enforcement
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	High-Assurance Security Specification
DOCUMENT STATUS	Normative: High-Assurance Security Specification
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 07 PART II · NORMATIVE TECHNICAL SPECIFICATIONS

Conformance Test Suite Specification

The authoritative testing methodology by which a System Under Test is assessed for compliance with the normative control corpus.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-CTSS-2026-001	1.0	2026	Test Specification	Institutional	NORMATIVE

Status note. No certification body or program exists yet. This is a conformance framework for self-assessment; any reference to certification or certified status describes a planned process, not an existing one.

ABSTRACT

This document is the formally complete revision of the QODIQA Conformance Test Suite Specification. Version 1.0 elevates the specification from laboratory-grade test protocol to hyperscaler-grade deterministic verification standard. All structural gaps present in prior drafts have been closed: QCC-05 through QCC-12 now carry full test card granularity; cross-architecture and runtime determinism validation is formally specified; a deterministic gateway state machine model is defined; a complete control-to-test coverage mapping matrix is provided; performance benchmark methodology is specified to deterministic standards; and canonical serialization requirements are normatively defined.

All constraints are preserved without modification. No existing control semantics have been altered. No probabilistic tolerance has been introduced. No fail-closed requirements have been relaxed. This version adds specified coverage of all twelve control points; completeness has not been independently demonstrated, and it does not substitute for rigor.

Conformance assessed under this specification requires satisfaction of all mandatory test cases across all twelve control points, with reproducible evidence artifacts, architecture-pair determinism validation, and a sealed evidence bundle verifiable by an independent auditing body operating under ISO/IEC 17025 principles.

SECTION 1

Purpose and Scope

1.1 Purpose

This Conformance Test Suite Specification (CTSS) defines the authoritative testing methodology by which a System Under Test (SUT) is assessed for compliance with the QODIQA normative control corpus. The objective of this specification is to transform QODIQA from a normatively defined standard into a deterministically testable and independently verifiable system. Each test defined herein corresponds to one or more normative requirements drawn from the QODIQA Core Standard and the 68-Point Enforcement Framework. Tests yield binary pass/fail determinations from reproducible, documented procedures. No subjective evaluation is permitted.

1.2 Relationship to Normative Standards

SOURCE STANDARD	IDENTIFIER	ROLE IN THIS SPECIFICATION
QODIQA: Core Standard for Deterministic Runtime Consent Enforcement Version 1.0	QODIQA-CORE-v1.0	Primary normative source. Defines QCC-01 through QCC-12.
QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement Version 1.0	QODIQA-68P-v3.0	Supplementary normative source. Enforcement points EP-01 through EP-68.
QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement Version 1.0	QODIQA-CERT-v1.0	Defines certification tiers and evidence requirements.
QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement Version 1.0	QODIQA-SCP-v1.1	Cryptographic algorithm requirements for Sections 8, 9, 12, and 15.

1.3 System Under Test Boundary

The SUT boundary is strictly the **QODIQA Gateway Enforcement Layer**: the runtime component responsible for evaluating consent tokens, enforcing policy decisions, generating execution artifacts, and writing to the audit ledger.

- Consent token validation and signature verification logic
- Purpose and intent field parsing and enforcement
- Policy engine evaluation and DENY/PERMIT decision emission
- Revocation registry query and response handling
- Execution artifact generation and hash computation
- Audit ledger write operations and hash chain maintenance
- Fail-closed behavior under all error and ambiguity conditions

1.4 Explicit Exclusions

- × Model internals, weights, or probabilistic inference behaviors
- × Training data pipelines or dataset provenance systems
- × Application-layer orchestration above the gateway

- × Network infrastructure external to the SUT boundary
- × End-user consent collection interfaces
- × Governance policy authoring workflows

SECTION 2

Conformance Testing Principles

Principle DefinitionsTESTING AXIOMS · QTS-P-01 THROUGH QTS-P-06
QTS-P-01 · Determinism Invariant For any frozen, canonically serialized context C, the SUT MUST produce identical output across all executions. A single divergent output across N replay iterations constitutes non-conformance.
QTS-P-02 · Fail-Closed Invariant In all conditions of ambiguity, error, incomplete context, or resource unavailability, the SUT MUST emit a DENY decision. Any PERMIT decision under ambiguous or degraded conditions constitutes critical non-conformance.
QTS-P-03 · Reproducibility Requirement Every test MUST be independently reproducible by an external auditor from the documented procedure and captured evidence artifacts alone. Tests that cannot be reproduced from artifacts are invalid.
QTS-P-04 · Negative Test Priority Negative test cases (those designed to elicit DENY, rejection, or error behavior) are considered primary conformance indicators. Positive path validation is secondary to negative boundary validation.
QTS-P-05 · No Probabilistic Allowance No statistical pass rate is acceptable. Conformance is binary. A SUT that passes 99 of 100 iterations of a determinism test is non-conformant.
QTS-P-06 · Evidence Capture Requirement Every test execution MUST capture structured evidence artifacts as defined in Section 16. Test results unsupported by captured evidence SHALL be treated as inconclusive.

2.1 Formal Determinism Invariant

FORMAL INVARIANT — QTS-P-01 · DETERMINISM CONSTRAINT

```
-- Determinism Invariant: QTS-P-01
INVARIANT Determinism:
  FORALL C : FrozenConsentContext,
  FORALL i, j : ExecutionRun,
    evaluate(C, i) == evaluate(C, j)

-- Failure condition:
--   EXISTS i, j such that evaluate(C, i) ≠ evaluate(C, j)
--   ⇒ SUT is NON-CONFORMANT under QTS-P-01
```

SECTION 3

Test Environment Requirements

TABLE 3.1: ENVIRONMENT SPECIFICATION

PARAMETER	REQUIREMENT	TOLERANCE	OBLIGATION
Clock Source	NTP-synchronized or hardware atomic reference	≤ 50ms drift during test window	MUST
Clock Drift During Execution	Monotonically non-decreasing within each test run	Zero backward jumps permitted	MUST
Test Harness Isolation	SUT isolated from production traffic during test window	Zero concurrent production requests	MUST
Registry Simulation	Dedicated test registry with seeded deterministic revocation state	Deterministic seed required	MUST
Key Material	Test-only key material via documented key injection interface	Production keys MUST NOT be present	MUST
Log Capture	Full structured log output captured to append-only local store	Zero log loss during execution	MUST

PARAMETER	REQUIREMENT	TOLERANCE	OBLIGATION
Network Partition Simulation	Capability to block or delay registry and external endpoints	Latency injection to ≥ 5000ms required	MUST
Hardware Neutrality	Tests MUST produce identical results across x86-64 and ARM64	Architecture divergence = non-conformance	MUST
Audit Write Simulation	Fault injection capability for audit write failures	Deterministic fault injection required	MUST

3.1 Registry Simulation States

- **VALID**: Token present, not revoked, within TTL
- **REVOKED_FRESH**: Token revoked within the last 10 seconds
- **REVOKED_STALE**: Token revoked but within TTL stale window
- **ABSENT**: Token not present (unknown)
- **TIMEOUT**: Registry query exceeds timeout threshold
- **PARTITIONED**: Registry unreachable due to network partition
- **TTL_BOUNDARY**: Token at exact TTL expiry boundary (±1 second)

SECTION 4

Control Point Test Matrix

All twelve QCC control points are specified at full test card granularity in this section. Negative cases are mandatory. A control point is considered tested only when all listed test cases have been executed and evidence captured. No test case may reference external annexes; all procedures are self-contained herein.

QCC-01: Consent Token Presence and Structural Validity

TABLE 4.1: QCC-01 TEST CASES

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-01-001	Valid token, all fields present	Well-formed token, valid signature, unexpired	
	EXPECTED PERMIT (proceed to QCC-02)	FAILURE DENY on valid token	EVIDENCE Decision log, token hash NORMAL
QTS-01-002	Token absent: no header	No Authorization header	
	EXPECTED DENY · immediate	FAILURE PERMIT without token	EVIDENCE Decision log, request log CRITICAL
QTS-01-003	Token truncated at 50%	Authorization header present, token body truncated	
	EXPECTED DENY · parse failure	FAILURE Any non-DENY response	EVIDENCE Decision log, error log, raw input CRITICAL

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-01-004	Missing `intent` field	Well-formed JWT, `intent` claim omitted	
	EXPECTED DENY · missing field	FAILURE PERMIT without intent	EVIDENCE Decision log, field validation trace CRITICAL
QTS-01-005	Missing `purpose` field	Well-formed JWT, `purpose` claim omitted	
	EXPECTED DENY · missing field	FAILURE PERMIT without purpose	EVIDENCE Decision log, field validation trace CRITICAL
QTS-01-006	Unknown `purpose` code	Purpose field contains unregistered code string	
	EXPECTED DENY · unknown purpose	FAILURE PERMIT on unknown purpose	EVIDENCE Decision log, registry query log CRITICAL
QTS-01-007	Empty string field values	Required fields present, value `""`	
	EXPECTED DENY · empty field	FAILURE PERMIT on empty values	EVIDENCE Decision log, field validation trace NORMAL

QCC-02: Cryptographic Signature Verification

TABLE 4.2: QCC-02 TEST CASES

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-02-001	Valid signature, registered key	Token signed with registered trust anchor key	
	EXPECTED Verification PASS	FAILURE Rejection on valid signature	EVIDENCE Verification trace, key ID log NORMAL
QTS-02-002	Payload tampered post-signing	One byte modified in signed payload	
	EXPECTED DENY · signature invalid	FAILURE Any non-DENY response	EVIDENCE Decision log, tamper evidence, byte diff CRITICAL
QTS-02-003	Unknown key ID	Token signed with key not in trust anchor registry	
	EXPECTED DENY · unknown key	FAILURE PERMIT on unknown key ID	EVIDENCE Decision log, key lookup trace CRITICAL
QTS-02-004	Revoked signing key	Token signed with key marked revoked in registry	
	EXPECTED DENY · revoked key	FAILURE PERMIT with revoked key	EVIDENCE Decision log, key revocation record CRITICAL

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-02-005	Weak algorithm: SHA-1/RSA-1024	Token signed with algorithm below minimum threshold	
	EXPECTED DENY · algorithm weak	FAILURE PERMIT on weak algorithm	EVIDENCE Decision log, algorithm policy trace CRITICAL
QTS-02-006	Algorithm set to `none`	JWT `alg` header explicitly `none`	
	EXPECTED DENY · alg:none	FAILURE Any non-DENY response	EVIDENCE Decision log, algorithm rejection log CRITICAL

QCC-03: Token Expiry and Temporal Validity

TABLE 4.3: QCC-03 TEST CASES

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-03-001	Token within valid time window	`exp` future, `nbf` past	
	EXPECTED Temporal PASS	FAILURE DENY on valid window	EVIDENCE Decision log, timestamp comparison NORMAL
QTS-03-002	Expired by exactly 1 second	`exp` = evaluation time − 1s	
	EXPECTED DENY · expired	FAILURE PERMIT on expired token	EVIDENCE Decision log, timestamp delta record CRITICAL

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-03-003	`nbf` in future (60s)	`nbf` = evaluation time + 60s	
	EXPECTED DENY · not yet valid	FAILURE PERMIT before nbf	EVIDENCE Decision log, nbf evaluation trace CRITICAL
QTS-03-004	Missing `exp` field	Token with `exp` claim absent	
	EXPECTED DENY · missing expiry	FAILURE PERMIT without expiry	EVIDENCE Decision log, field validation trace CRITICAL
QTS-03-005	Lifetime exceeds policy maximum	`exp - iat` > policy maximum (e.g., >24h)	
	EXPECTED DENY · lifetime exceeded	FAILURE PERMIT on overlong lifetime	EVIDENCE Decision log, lifetime policy trace NORMAL

QCC-04: Revocation Registry Verification

TABLE 4.4: QCC-04 TEST CASES

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-04-001	Token valid in registry	Registry state: VALID; response within timeout	
	EXPECTED Revocation PASS	FAILURE DENY on non-revoked token	EVIDENCE Decision log, registry response NORMAL

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-04-002	Token revoked	Registry state: REVOKED_FRESH	
	EXPECTED DENY · revoked	FAILURE PERMIT on revoked token	EVIDENCE Decision log, revocation record CRITICAL
QTS-04-003	Registry timeout	Registry delayed beyond timeout threshold	
	EXPECTED DENY · registry unavailable	FAILURE PERMIT when registry unreachable	EVIDENCE Decision log, timeout trace CRITICAL
QTS-04-004	Malformed registry response	Registry returns non-parseable body	
	EXPECTED DENY · registry response invalid	FAILURE PERMIT on malformed response	EVIDENCE Decision log, raw response CRITICAL
QTS-04-005	Token absent in registry	Registry state: ABSENT	
	EXPECTED DENY · not registered	FAILURE PERMIT on unknown token	EVIDENCE Decision log, registry 404 CRITICAL

QCC-05: Policy Engine Evaluation and Decision Emission

TABLE 4.5: QCC-05 TEST CASES

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-05-001	Valid request matches explicit PERMIT rule	Policy loaded; request purpose matches configured PERMIT entry	
	EXPECTED PERMIT emitted with matching rule ID	FAILURE DENY on explicit PERMIT match	EVIDENCE Decision log, matched rule ID, policy version hash NORMAL
QTS-05-002	Request purpose not in policy: default-deny fires	Policy loaded; purpose code absent from all rules	
	EXPECTED DENY · default-deny applied	FAILURE PERMIT on unmatched purpose	EVIDENCE Decision log, default rule activation trace CRITICAL
QTS-05-003	Empty policy ruleset loaded	Policy engine initialized with zero rules	
	EXPECTED DENY · no applicable rule	FAILURE PERMIT under empty policy	EVIDENCE Decision log, policy load confirmation, empty ruleset hash CRITICAL
QTS-05-004	Policy engine crash mid-evaluation	Policy engine process terminated after intake, before decision	
	EXPECTED DENY · policy engine failure	FAILURE Any non-DENY response or silent hang	EVIDENCE Decision log, crash report, fault injection record CRITICAL

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-05-005	Explicit DENY rule matched	Request purpose matches explicit DENY rule	
	EXPECTED DENY · explicit rule matched	FAILURE PERMIT when DENY rule applies	EVIDENCE Decision log, matched rule ID CRITICAL
QTS-05-006	Policy evaluation determinism: replay	Identical frozen context replayed N=10 times against same policy version	
	EXPECTED All 10 runs emit identical decision and matched rule ID	FAILURE Any run diverges from baseline decision	EVIDENCE Replay bundle, 10x decision logs, frozen context ID CRITICAL
QTS-05-007	Conflicting rules: DENY wins	Policy contains overlapping PERMIT and DENY rules for same purpose scope	
	EXPECTED DENY · conflict resolution enforces DENY	FAILURE PERMIT when DENY rule also present	EVIDENCE Decision log, conflict resolution trace CRITICAL
QTS-05-008	Policy engine returns ambiguous/null decision	Policy engine configured to return null or undefined state	
	EXPECTED DENY · ambiguous decision treated as DENY	FAILURE Any non-DENY on null/undefined decision	EVIDENCE Decision log, engine output capture CRITICAL

QCC-06: Context Hash Binding and Canonical Serialization

TABLE 4.6: QCC-06 TEST CASES

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-06-001	Hash matches computed value	Valid context; SUT-computed hash equals independently computed reference hash	
	EXPECTED Hash binding PASS; decision proceeds	FAILURE DENY on valid hash match	EVIDENCE Decision log, hash comparison record, reference hash NORMAL
QTS-06-002	Context field modified after hash computation	One context field value altered between hash compute and policy evaluation step	
	EXPECTED DENY · context hash mismatch	FAILURE PERMIT without re-verification	EVIDENCE Decision log, tamper trace, before/after hash CRITICAL
QTS-06-003	Serialization key order varies	Identical context serialized with different JSON key order	
	EXPECTED Both serializations produce identical hash (canonical ordering enforced)	FAILURE Different hashes for identical content	EVIDENCE Decision log, both serializations, hash outputs CRITICAL
QTS-06-004	Whitespace variation in serialization	Context serialized with and without insignificant whitespace	
	EXPECTED Identical hash produced regardless of whitespace	FAILURE Hash diverges due to whitespace	EVIDENCE Both serialized forms, both hashes NORMAL

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-06-005	Hash engine failure	Hash computation function returns error	
	EXPECTED DENY · hash computation failure	FAILURE PERMIT without verified hash	EVIDENCE Decision log, fault injection record CRITICAL
QTS-06-006	Cross-architecture hash equivalence	Identical context hashed on x86-64 and ARM64	
	EXPECTED Byte-identical hash output on both architectures	FAILURE Hash diverges across architectures	EVIDENCE Hashes from both architectures, frozen context record CRITICAL

QCC-07: Policy Version Binding

TABLE 4.7: QCC-07 TEST CASES

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-07-001	Policy version correctly bound at intake	Policy version ID captured at request intake; same ID referenced in artifact and decision log	
	EXPECTED Policy version binding PASS	FAILURE Version ID absent from decision log or artifact	EVIDENCE Decision log, artifact, policy version record NORMAL
QTS-07-002	Policy updated during evaluation	Policy version incremented between request intake and policy evaluation step	
	EXPECTED Request evaluated against version pinned at intake; no silent drift to new version	FAILURE Request evaluated against new policy version without rebinding	EVIDENCE Decision log, version pinning trace, policy update event log CRITICAL

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-07-003	Referenced policy version absent	Request bound to policy version ID that no longer exists in policy store	
	EXPECTED DENY · policy version not found	FAILURE PERMIT or evaluation against different version	EVIDENCE Decision log, policy store lookup trace CRITICAL
QTS-07-004	Policy version field absent from artifact	Artifact generated without `policy_version_id` field	
	EXPECTED Artifact rejected; treated as invalid at audit verification	FAILURE Artifact without version ID accepted as valid	EVIDENCE Artifact inspection log, validation failure record NORMAL
QTS-07-005	Concurrent version drift under load	Policy updated while N=100 concurrent requests are in-flight	
	EXPECTED All in-flight requests maintain their pinned version; zero silent drift events	FAILURE Any request evaluated against different version than pinned at intake	EVIDENCE Version pin log per request, policy update event timestamp CRITICAL

QCC-08: Execution Artifact Generation

TABLE 4.8: QCC-08 TEST CASES

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-08-001	Artifact generated on PERMIT	Valid PERMIT decision issued; artifact generator runs	
	EXPECTED Artifact produced with all required fields present	FAILURE Artifact absent or missing required fields after PERMIT	EVIDENCE Artifact, field completeness check log NORMAL

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-08-002	Artifact generator fails after PERMIT	PERMIT decision issued; artifact generator returns persistent error	
	EXPECTED `Execution blocked; DENY emitted or execution voided`	FAILURE Execution proceeds without artifact	EVIDENCE Decision log, fault injection record, execution block confirmation CRITICAL
QTS-08-003	Artifact hash computation failure	SHA-256 of artifact body returns error during generation	
	EXPECTED `DENY · artifact hash failure`	FAILURE Artifact issued without embedded hash	EVIDENCE Decision log, fault injection record CRITICAL
QTS-08-004	Artifact hash recomputation: integrity	Independently recompute SHA-256 of artifact body; compare to embedded `artifact_hash`	
	EXPECTED `Computed hash equals embedded hash`	FAILURE Hash mismatch	EVIDENCE Artifact, recomputed hash, comparison record CRITICAL
QTS-08-005	Artifact not generated on DENY	DENY decision emitted for any reason	
	EXPECTED `No artifact produced`	FAILURE Artifact produced on DENY decision	EVIDENCE Decision log, artifact store check confirming absence NORMAL
QTS-08-006	Artifact determinism: replay	Identical frozen context replayed N=10 times under PERMIT	
	EXPECTED `All 10 artifacts byte-identical (excluding timestamp field)`	FAILURE Any artifact content divergence across replays	EVIDENCE 10x artifact corpus, diff log CRITICAL

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-08-007	Artifact timestamp within tolerance	Verify `artifact_timestamp` is within ±1 second of decision timestamp	
	EXPECTED Timestamps within tolerance	FAILURE Timestamp divergence exceeds tolerance	EVIDENCE Decision log, artifact, timestamp comparison NORMAL

QCC-09: Audit Ledger Write

TABLE 4.9: QCC-09 TEST CASES

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-09-001	Ledger write succeeds on PERMIT	Valid PERMIT decision; audit write returns confirmed	
	EXPECTED Ledger entry present; chain intact	FAILURE Execution proceeds without ledger entry	EVIDENCE Ledger extract, chain validation report NORMAL
QTS-09-002	Audit write fails: persistent error	Audit ledger write returns persistent error after PERMIT decision	
	EXPECTED DENY · audit unavailable; execution blocked	FAILURE Execution permitted without audit record	EVIDENCE Decision log, fault injection record, execution block confirmation CRITICAL
QTS-09-003	DENY decision written to ledger	DENY decision emitted for any reason	
	EXPECTED DENY recorded in ledger with denial reason code	FAILURE DENY events absent from ledger	EVIDENCE Ledger extract confirming DENY entry presence NORMAL

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-09-004	Hash chain integrity maintained across writes	N=100 sequential decisions written; chain validator runs	
	EXPECTED `Chain validates without error across all N entries`	FAILURE Chain break at any entry	EVIDENCE Full ledger extract, chain validation report CRITICAL
QTS-09-005	Concurrent write race: chain serialization	N=50 concurrent PERMIT decisions; chain write attempted simultaneously	
	EXPECTED `All entries present; chain intact; no collision or gap`	FAILURE Chain broken or entry lost due to concurrent write	EVIDENCE Ledger extract, chain validation report, concurrency log CRITICAL
QTS-09-006	Ledger write timeout	Audit write operation times out after configured threshold	
	EXPECTED `DENY · audit write timeout; execution blocked`	FAILURE Execution proceeds after audit write timeout	EVIDENCE Decision log, timeout trace CRITICAL

QCC-10: Data Minimisation in Artifact Construction

TABLE 4.10: QCC-10 TEST CASES

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-10-001	No raw PII in artifact	Request contains known PII fields; artifact generated	
	EXPECTED `Artifact contains no raw PII values: only hashes or minimised references`	FAILURE Raw PII present in artifact body	EVIDENCE Artifact, PII field scan report CRITICAL

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-10-002	Raw request body absent from artifact	Request body contains arbitrary payload; artifact generated	
	EXPECTED Raw request body not reproduced in artifact	FAILURE Raw request body present in artifact	EVIDENCE Artifact, request body comparison CRITICAL
QTS-10-003	Consent token raw value absent from artifact	Full consent token presented; artifact generated	
	EXPECTED Artifact contains token hash reference only, not raw token	FAILURE Raw token value present in artifact	EVIDENCE Artifact, token hash reference check CRITICAL
QTS-10-004	Required structural fields present	Valid PERMIT; artifact generated	
	EXPECTED Artifact contains all required structural fields per specification	FAILURE Required structural fields absent	EVIDENCE Artifact, field completeness check NORMAL
QTS-10-005	Artifact replay without sensitive reconstruction	Artifact submitted for independent replay verification	
	EXPECTED Replay verification succeeds from artifact alone without raw input data	FAILURE Replay requires raw sensitive data not present in artifact	EVIDENCE Artifact, replay verification output log NORMAL

QCC-11: Replay Detection and Nonce Validation

TABLE 4.11: QCC-11 TEST CASES

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-11-001	Unique nonce: first submission accepted	Token with fresh, unregistered nonce submitted	
	EXPECTED PERMIT; nonce registered in nonce store	FAILURE DENY on fresh unique nonce	EVIDENCE Decision log, nonce store entry NORMAL
QTS-11-002	Nonce replayed immediately	Identical token resubmitted within 1 second of first acceptance	
	EXPECTED DENY · nonce replay detected	FAILURE PERMIT on immediate replay	EVIDENCE Decision log, nonce store lookup trace CRITICAL
QTS-11-003	Nonce replayed after TTL expiry	Nonce resubmitted after nonce store TTL window has expired	
	EXPECTED DENY · nonce replay (TTL expiry does not permit re-use)	FAILURE PERMIT after nonce TTL expiry	EVIDENCE Decision log, nonce TTL policy, nonce store state CRITICAL
QTS-11-004	Missing nonce field	Token submitted without nonce claim	
	EXPECTED DENY · missing nonce	FAILURE PERMIT without nonce	EVIDENCE Decision log, field validation trace CRITICAL

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-11-005	Nonce store unavailable	Nonce store query returns error or timeout	
	EXPECTED DENY · nonce store unavailable	FAILURE PERMIT when nonce uniqueness unverifiable	EVIDENCE Decision log, fault injection record CRITICAL
QTS-11-006	Concurrent nonce submission: race	Identical nonce submitted N=20 times simultaneously	
	EXPECTED Exactly one request PERMIT; all others DENY · nonce replay	FAILURE More than one PERMIT issued for same nonce under concurrency	EVIDENCE All 20 decision logs, nonce store serialization log CRITICAL

QCC-12: Concurrent Request Isolation

TABLE 4.12: QCC-12 TEST CASES

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-12-001	Isolated context per concurrent request	N=100 concurrent requests with distinct tokens and contexts	
	EXPECTED Each request evaluated against its own context; no cross-contamination	FAILURE Any request evaluated against another request's context fields	EVIDENCE 100x decision logs, context ID per request CRITICAL
QTS-12-002	Context bleed: shared memory inspection	N=2 concurrent requests: one PERMIT, one DENY; inspect context fields post-evaluation	
	EXPECTED PERMIT context fields are absent from DENY evaluation path and vice versa	FAILURE Any context field from one request visible in another's decision path	EVIDENCE Decision logs, context field isolation audit CRITICAL

TEST ID	SCENARIO	INPUT CONDITION	EXPECTED OUTCOME
QTS-12-003	DENY decision under high concurrency	N=500 concurrent requests, 50% with invalid tokens	
	EXPECTED `All invalid token requests DENY; valid token requests correctly PERMIT/DENY per policy`	FAILURE Any invalid request returns PERMIT under concurrent load	EVIDENCE All decision logs, pass/fail ratio, load profile CRITICAL
QTS-12-004	Policy evaluation serialization under concurrency	N=100 concurrent requests referencing same policy version	
	EXPECTED `All requests evaluated against pinned policy; no version bleed`	FAILURE Any request evaluated against a different policy version under concurrency	EVIDENCE Policy version per decision log, concurrency timeline NORMAL
QTS-12-005	Audit write serialization under concurrency	N=200 concurrent decisions; all write to audit ledger	
	EXPECTED `All entries present; chain intact; no write collision or lost entry`	FAILURE Any entry lost or chain broken under concurrent audit writes	EVIDENCE Ledger extract, chain validation report CRITICAL

SECTION 5

Determinism Verification Protocol

5.1 Frozen Context Capture Procedure

Normative Frozen Context Structure · QTS-FC-01

CONSENT_TOKEN_HASH	SHA-256 of raw serialized consent token. Computed before any field parsing.
REQUEST_CONTEXT_HASH	SHA-256 of canonically serialized request context per QCC-06 and Section 15.
POLICY_VERSION_ID	Exact policy version string active at time of evaluation. MUST include patch version component.
REGISTRY_STATE_SNAPSHOT	Deterministic fixture identifier for the registry state seeded for this test run.
EVALUATION_TIMESTAMP	Unix epoch timestamp at moment of evaluation initiation, millisecond precision.
FROZEN_CONTEXT_ID	SHA-256 of concatenation of all above fields in canonical order. Used as replay identifier.

5.2 Replay Testing Procedure

— Capture frozen context record and assign `frozen_context_id`

- Execute evaluation against frozen context. Record decision D_1 and full output log.
- Reset SUT to clean state. Verify registry fixture re-seeded to identical deterministic state.
- Repeat for minimum N = 10 independent iterations yielding D_1 through D_{10}.
- If $\text{EXISTS } i, j: D_i \neq D_j$, the SUT is non-conformant under QTS-P-01.
- Package all iteration logs as a replay evidence bundle per Section 16.

5.3 Non-Deterministic Dependency Detection

TABLE 5.1: NON-DETERMINISM INDICATORS

INDICATOR	DETECTION METHOD	REQUIRED RESPONSE
Wall-clock dependency in evaluation	Decision varies with replay timestamp drift	Root cause isolation. Mark SUT non-conformant.
Random salt or nonce injected at evaluation time	Context hash differs across replays on identical input	Root cause isolation. Mark SUT non-conformant.
External cache affecting policy resolution	Decision differs when cache is warm vs. cold	Root cause isolation. Cache must not influence decision.
Thread scheduling affecting decision path	Concurrent execution produces divergent outcomes	Root cause isolation. Concurrency must not affect correctness.
Registry TTL boundary ambiguity	Decision oscillates at TTL boundary across replays	Root cause isolation. TTL boundary MUST be deterministically defined.

SECTION 6

Fail-Closed Enforcement Tests

Fail-closed behavior is the most critical conformance property in this specification. The SUT MUST emit DENY in every condition where the correct outcome cannot be positively and completely verified. Failure of any single case below constitutes critical non-conformance.

TABLE 6.1: FAIL-CLOSED TEST CASES

TEST ID	FAULT CONDITION	INJECTED FAULT	REQUIRED RESPONSE	FAILURE CONDITION
QTS-FC-001	Missing consent token	Request with no Authorization header	DENY · immediate	Any non-DENY response
QTS-FC-002	Registry timeout	Registry delayed to 5x defined timeout value	DENY · registry unavailable	PERMIT while awaiting registry
QTS-FC-003	Policy engine failure	Policy engine terminated mid-evaluation	DENY · policy engine failure	Execution permitted without policy decision
QTS-FC-004	Hash engine failure	Context hash computation returns error	DENY · hash failure	PERMIT without verified context hash
QTS-FC-005	Audit write failure	Audit ledger write returns persistent error	DENY · audit unavailable	Execution permitted without audit record
QTS-FC-006	Partial request context	Request context missing mandatory fields	DENY · incomplete context	PERMIT on partial context
QTS-FC-007	Artifact generation failure	Artifact generator returns error after PERMIT	DENY · artifact failure	Execution proceeds without artifact
QTS-FC-008	Policy default absent	Policy engine loaded with no default rule	DENY · no applicable rule	PERMIT under empty policy
QTS-FC-009	Trust anchor unavailable	Trust anchor registry query fails on startup	DENY · trust anchor unavailable	Signature verification attempted without anchors
QTS-FC-010	Total network partition	All external dependencies blocked via firewall rule	DENY on all requests	Any PERMIT under total partition

Pass Definition: A SUT passes fail-closed testing if and only if it emits DENY for every test case in Table 6.1 with a corresponding structured log entry capturing the fault condition, decision outcome, timestamp, and context identifier. Any PERMIT under any fault condition is an automatic critical non-conformance finding.

SECTION 7

Revocation Propagation Race Condition Tests

Fail-open behavior under any revocation race condition constitutes automatic non-conformance. The only acceptable behavior under any race scenario is DENY with a structured log entry specifying the race condition type and revocation status at time of decision.

TABLE 7.1: REVOCATION RACE CONDITION TESTS

TEST ID	RACE SCENARIO	TIMING CONDITION	REQUIRED OUTCOME	FAILURE CONDITION
QTS-RC-001	Revocation immediately before evaluation	Token revoked at T−10ms; evaluation at T	DENY · token revoked	PERMIT after revocation event
QTS-RC-002	Revocation during registry query	Revocation fires while registry query is in-flight	DENY on any ambiguity	PERMIT when revocation timing uncertain
QTS-RC-003	Stale registry cache	Cache holds VALID; actual state is REVOKED	DENY · status unverifiable	PERMIT based on stale cache
QTS-RC-004	TTL boundary: exact expiry	Evaluation at exactly T = `exp`	DENY · ambiguous boundary	PERMIT at exact expiry
QTS-RC-005	N=20 concurrent requests: same token under revocation	Token revoked mid-burst	DENY on all requests once revoked	Any PERMIT after first revocation confirmation
QTS-RC-006	Network partition during revocation query	Registry unreachable; local state shows VALID	DENY · status unverifiable	PERMIT based on local state under partition

FORMAL CONSTRAINT — QTS-RC · REVOCATION SAFETY PROPERTY

```
INVARIANT RevocationSafety:
  FORALL r : Request,
  LET   status = RevocationRegistry.query(r.token) AT decision_time,
  IF    status ∉ { CONFIRMED_NOT_REVOKED }
  THEN  decision(r) == DENY

-- States that MUST trigger DENY:
--   REVOKED, STALE, UNKNOWN, TIMEOUT, ABSENT, PARTITION, AMBIGUOUS, CACHE_ONLY
```

SECTION 8

Artifact Integrity Validation

TABLE 8.1: ARTIFACT INTEGRITY TEST CASES

TEST ID	VALIDATION TARGET	PROCEDURE
QTS-AI-001	Artifact hash recomputation	Recompute SHA-256 of artifact body; compare to embedded `artifact_hash`
	PASS CONDITION Computed equals embedded hash	FAILURE Hash mismatch
QTS-AI-002	Context binding validation	Verify `context_hash` in artifact matches frozen context record
	PASS CONDITION Hashes match exactly	FAILURE Context hash mismatch

TEST ID	VALIDATION TARGET	PROCEDURE
QTS-AI-003	Policy version inclusion	Verify `policy_version_id` present and matches active policy at evaluation
	PASS CONDITION Present and correct	FAILURE Missing or incorrect version
QTS-AI-004	Raw sensitive data exclusion	Scan artifact for raw PII fields from original request
	PASS CONDITION No raw sensitive fields present	FAILURE Raw sensitive data present
QTS-AI-005	Tamper detection	Modify one field in artifact; resubmit for integrity verification
	PASS CONDITION Tamper detected; verification returns INVALID	FAILURE Tampered artifact passes verification
QTS-AI-006	Generation failure handling	Inject artifact generator failure after PERMIT
	PASS CONDITION Execution blocked or voided	FAILURE Execution proceeds without artifact
QTS-AI-007	Timestamp binding	Verify `artifact_timestamp` within ±1s of decision timestamp
	PASS CONDITION Within tolerance	FAILURE Timestamp divergence exceeds tolerance

SECTION 9

Audit Ledger Integrity Tests

Silent modification of any ledger entry in any form constitutes critical non-conformance.

TABLE 9.1: AUDIT LEDGER INTEGRITY TEST CASES

TEST ID	ATTACK SIMULATION	PROCEDURE	PASS CONDITION	FAILURE CLASSIFICATION
QTS-AL-001	Hash chain validation: intact	Run chain validator on N=1000 sequential entries	Chain validates without error	Validation failure on intact chain
QTS-AL-002	Field tampering	Modify `decision` field in entry N; re-run chain validator	Break detected at entry N	Tamper undetected
QTS-AL-003	Entry deletion	Remove entry N from chain; re-run validator	Gap detected at entry N	Deletion undetected
QTS-AL-004	Entry reordering	Swap entries N and N+1; re-run validator	Ordering violation detected	Reordering undetected
QTS-AL-005	Retention verification	Verify all entries within retention window present and retrievable	All entries present; chain intact	Any entry missing within window
QTS-AL-006	Concurrent write race	N=50 concurrent PERMIT decisions; chain serialization	All entries present; no collision	Chain broken by concurrent write

SECTION 10

Negative and Adversarial Testing

No adversarial input shall produce a PERMIT decision. All adversarial inputs must be rejected without exception.

TABLE 10.1: ADVERSARIAL TEST CASES

TEST ID	ATTACK VECTOR	INJECTED PAYLOAD
QTS-ADV-001	Forged consent token	Token signed with self-generated key absent from trust anchors
	REQUIRED RESPONSE **DENY · signature invalid**	FAILURE CONDITION PERMIT on forged token

TEST ID	ATTACK VECTOR	INJECTED PAYLOAD
QTS-ADV-002	Malformed JSON payload	Request body with invalid JSON and nested control characters
	REQUIRED RESPONSE **DENY · parse failure**	FAILURE CONDITION Exception not safely handled
QTS-ADV-003	Future timestamp: clock skew	Token `iat` set to T+3600
	REQUIRED RESPONSE **DENY · invalid issuance time**	FAILURE CONDITION PERMIT on future-dated token
QTS-ADV-004	Aged token replay	Valid token resubmitted 25h after original issuance
	REQUIRED RESPONSE **DENY · expired / replay**	FAILURE CONDITION PERMIT on replayed aged token
QTS-ADV-005	Hash collision simulation	Two payloads with identical MD5 submitted; SHA-256 path verified independent
	REQUIRED RESPONSE SHA-256 collision path unaffected; both evaluated on SHA-256	FAILURE CONDITION SUT uses MD5 or collision affects evaluation
QTS-ADV-006	Unknown action type injection	Action type not in registered action registry
	REQUIRED RESPONSE **DENY · unknown action**	FAILURE CONDITION PERMIT on unregistered action
QTS-ADV-007	Concurrent policy version drift	Policy updated during N=100 concurrent requests
	REQUIRED RESPONSE All in-flight pinned to intake version; no silent drift	FAILURE CONDITION Any request evaluated against different version

TEST ID	ATTACK VECTOR	INJECTED PAYLOAD
QTS-ADV-008	Oversized token: 10MB	Token body padded to 10MB
	REQUIRED RESPONSE DENY · payload exceeds maximum	FAILURE CONDITION Processing or memory fault
QTS-ADV-009	Unicode normalization attack	Purpose field with visually identical but byte-distinct Unicode
	REQUIRED RESPONSE DENY · normalization failure or unknown purpose	FAILURE CONDITION PERMIT due to normalization inconsistency
QTS-ADV-010	Null byte injection	Token field values contain embedded null bytes
	REQUIRED RESPONSE DENY · invalid field content	FAILURE CONDITION Null byte causes truncation or bypass

SECTION 11

Performance Boundary Tests and Deterministic Benchmark Methodology

11.1 Benchmark Harness Requirements

TABLE 11.1: BENCHMARK HARNESS SPECIFICATION

PARAMETER	REQUIREMENT	OBLIGATION
Load Generation Seed	Deterministic PRNG seed. Identical seed MUST produce identical request sequence across runs.	MUST
Fixed Payload Corpus	Pre-generated corpus of N=10,000 requests used for all benchmark runs without regeneration.	MUST
Registry Fixture Consistency	Registry seeded to identical deterministic state before each benchmark run.	MUST

PARAMETER	REQUIREMENT	OBLIGATION
Network Latency Injection	Controlled baseline latency applied to all registry calls: defined as constant value, not random.	MUST
Concurrency Model	Fixed thread pool or goroutine count; no dynamic scaling during benchmark window.	MUST
Benchmark Duration	Sustained window: 30 minutes nominal, 5 minutes burst. Both MUST be executed.	MUST
Warm-up Phase	Minimum 60-second warm-up before measurement window opens. Warm-up results excluded.	MUST

11.2 Measurement Methodology

Normative Latency Measurement Definition · QTS-PM-01

MEASUREMENT POINTS	T_0 = timestamp at token bytes fully received by gateway. T_1 = timestamp at DENY/PERMIT emitted to caller. Latency = $T_1 - T_0$.
CLOCK SOURCE	Same NTP-synchronized reference as test environment clock. Monotonic clock MUST be used for within-process delta measurement. Wall clock for external correlation only.
PERCENTILE COMPUTATION	HDR histogram (High Dynamic Range). All samples retained. No binning until final percentile extraction. Percentiles: p50, p90, p99, p99.9.
OUTLIER HANDLING	No statistical trimming permitted. All samples including maximum values MUST be retained and reported. Trimmed benchmarks are invalid evidence.
SCOPE	End-to-end latency inclusive of registry lookup, hash computation, policy evaluation, artifact generation, and audit write. Partial scope benchmarks are invalid.

11.3 Performance Thresholds

TABLE 11.2: PERFORMANCE BOUNDARY DEFINITIONS

METRIC	PASS THRESHOLD	CONDITIONAL THRESHOLD	FAIL THRESHOLD
Decision Latency p50	≤ 20ms	20ms – 50ms	> 50ms
Decision Latency p99	≤ 80ms	80ms – 200ms	> 200ms
Decision Latency p99.9	≤ 250ms	250ms – 500ms	> 500ms
Revocation Lookup p99	≤ 150ms	150ms – 300ms	> 300ms
Artifact Generation Overhead	≤ 10ms	10ms – 30ms	> 30ms
Audit Write Latency p99	≤ 25ms	25ms – 75ms	> 75ms
Throughput: Nominal (30-min)	≥ 500 req/s	250–500 req/s	< 250 req/s
Throughput: Peak (5-min burst)	≥ 1000 req/s	500–1000 req/s	< 500 req/s

11.4 Cold vs. Warm Start Behavior

TABLE 11.3: START STATE LATENCY BOUNDS

STATE	DEFINITION	LATENCY BOUND (P99)	OBLIGATION
Cold Start	First request after process initialization; no cached state	≤ 500ms for first decision	MUST
Warm Steady-State	After 60-second warm-up window; all caches and connections established	Satisfies Table 11.2 p99 thresholds	MUST
Policy Reload	Decision latency during and immediately after policy version update	≤ 200ms p99 during reload window	MUST
Registry Reconnect	First request after registry connection re-established post-partition	≤ 300ms for first decision post-reconnect	MUST

SECTION 12

Cross-Architecture and Runtime Determinism Validation

Determinism invariant QTS-P-01 applies across hardware architectures and runtime environments without exception. A SUT that produces conformant results on one architecture but divergent results on another is non-conformant. This section defines the mandatory validation procedures for cross-architecture and cross-runtime determinism.

12.1 Formal Cross-Architecture Invariant

FORMAL INVARIANT — QTS-CA-01 · CROSS-ARCHITECTURE DETERMINISM

```
-- Cross-Architecture Determinism Invariant: QTS-CA-01
-- For identical FrozenContext C across any two architecture targets A and B:

INVARIANT CrossArchitectureDeterminism:
  FORALL C  : FrozenConsentContext,
  FORALL A, B : Architecture -- e.g. x86_64, ARM64, ARMv7, RISC-V
    evaluate_A(C) == evaluate_B(C)

-- Where evaluate_X(C) returns:
--   { decision: PERMIT|DENY, artifact_hash: SHA-256, context_hash: SHA-256 }
--   All three fields MUST be byte-identical across architectures

-- Failure condition:
--   ANY field diverges across architecture pair ⇒ NON-CONFORMANT
```

12.2 Hardware Determinism Test Cases

TABLE 12.1: HARDWARE DETERMINISM TESTS

TEST ID	VALIDATION TARGET	ARCHITECTURE PAIR
QTS-CA-001	Decision outcome equivalence	x86-64 ↔ ARM64
	PROCEDURE `Execute identical frozen context on both architectures; compare decision field`	PASS CONDITION decision field byte-identical FAILURE Decision diverges across architectures
QTS-CA-002	Context hash byte equivalence	x86-64 ↔ ARM64
	PROCEDURE `Compute SHA-256 of identical canonical context on both; compare raw bytes`	PASS CONDITION Byte-identical hash output FAILURE Hash diverges (endianness or encoding dependency detected)
QTS-CA-003	Endianness validation	Any little-endian ↔ big-endian pair
	PROCEDURE `Verify integer fields serialized to canonical byte order before hashing`	PASS CONDITION Identical hash regardless of native endianness FAILURE Hash diverges due to endianness of integer representation
QTS-CA-004	Floating-point determinism	x86-64 ↔ ARM64
	PROCEDURE `If any numeric computation in policy evaluation produces floating-point result: verify identical bit pattern across architectures`	PASS CONDITION Bit-identical floating-point output or floating-point entirely absent from decision path FAILURE Floating-point divergence in decision path

TEST ID	VALIDATION TARGET	ARCHITECTURE PAIR
QTS-CA-005	Serialization key ordering	x86-64 ↔ ARM64
	PROCEDURE Serialize identical context object on both architectures; compare byte-for-byte	PASS CONDITION Byte-identical serialized form FAILURE Serialization order differs across architectures
QTS-CA-006	Artifact hash equivalence	x86-64 ↔ ARM64
	PROCEDURE Generate artifact on both architectures from same context; compare `artifact_hash` fields	PASS CONDITION Identical artifact hash FAILURE Artifact hash diverges across architectures

12.3 Runtime Environment Determinism Tests

TABLE 12.2: RUNTIME DETERMINISM TESTS

TEST ID	VALIDATION TARGET	RUNTIME VARIATION	PROCEDURE	PASS CONDITION	FAILURE CONDITION
QTS-RT-001	OS kernel version independence	Linux kernel 5.x vs. 6.x	Execute identical frozen context on both kernel versions; compare decision and hash outputs	All outputs byte-identical	Any output diverges across kernel versions
QTS-RT-002	Container runtime independence	containerd vs. Docker runtime	Execute identical frozen context in both container runtimes; compare outputs	All outputs byte-identical	Output diverges between container runtimes
QTS-RT-003	Cold start vs. warm start decision equivalence	Cold (first request) vs. Warm (post-warmup)	Execute identical frozen context in cold and warm states; compare decision outcome (not latency)	Decision outcome identical; latency may differ within bounds of Section 11.4	Decision outcome differs between cold and warm state

TEST ID	VALIDATION TARGET	RUNTIME VARIATION	PROCEDURE	PASS CONDITION	FAILURE CONDITION
QTS-RT-004	Multi-thread scheduling non-influence	Single-thread vs. N=8 worker threads	Execute identical frozen context under single-thread and 8-thread configurations; compare decision and hash	All outputs byte-identical regardless of thread count	Decision or hash diverges due to thread scheduling
QTS-RT-005	Garbage collector non-determinism exclusion	GC forced during evaluation vs. GC idle	Trigger GC cycle mid-evaluation; verify decision output unchanged	Decision output unaffected by GC event	GC event alters decision or introduces timing-dependent path divergence
QTS-RT-006	Locale and timezone independence	UTC vs. non-UTC timezone; various locale settings	Execute identical frozen context with different system timezone and locale; compare outputs	All outputs byte-identical; no locale-dependent behavior	Timestamp parsing or string comparison affected by locale

SECTION 13

Formal Gateway State Machine Model

The QODIQA Gateway Enforcement Layer operates as a deterministic finite state machine. Every request transitions through a fixed sequence of states. Any failure or ambiguity at any state MUST transition to the DENY_EMIT terminal state. No state transition may be skipped. Each transition has defined pre-conditions, post-conditions, failure paths, determinism requirements, and mandatory log entries.

13.1 State Transition Formal Invariant

FORMAL MODEL — QTS-SM-01 · GATEWAY STATE SEQUENCE

```
STATES := [
  INTAKE, STRUCTURAL_VALIDATE, SIGNATURE_VERIFY,
  TEMPORAL_VALIDATE, REVOCATION_CHECK, POLICY_EVALUATE,
```

```
  ARTIFACT_GENERATE, AUDIT_WRITE, DECISION_EMIT
]

INVARIANT StateMonotonicity:
  -- Normal path: each state reached at most once, in defined order
  FORALL r : Request, s ∈ STATES :
    reached(r, s) ⇒ ¬revisited(r, s)

INVARIANT FailClosed:
  -- Any failure at any state MUST produce DENY
  FORALL r : Request, s ∈ STATES :
    failure_at(r, s) ⇒ terminal_decision(r) == DENY

INVARIANT NoSkip:
  -- States MUST be traversed in sequence; no skipping
  FORALL r : Request :
    sequence(r) ⊑ ordered_prefix(STATES)
```

13.2 State Transition Definitions

INTAKE	
PRE-CONDITION	Request received by gateway process; bytes fully buffered
ACTION	Extract Authorization header. Bind `policy_version_id` at intake timestamp. Assign `request_id`.
POST-CONDITION	Token bytes available. Policy version pinned. Request ID assigned.
FAILURE PATH	→ DENY_EMIT: missing Authorization header, oversized payload, parse error
LOG ENTRY	`{ state: INTAKE, request_id, policy_version_id, timestamp }`

STRUCTURAL_VALIDATE	
PRE-CONDITION	Token bytes available from INTAKE
ACTION	Parse token. Verify all required claims present and non-empty. Verify purpose code is registered. Reject unknown fields if strict mode.
POST-CONDITION	All required fields validated. Claim values available for subsequent states.
FAILURE PATH	→ DENY_EMIT: malformed token, missing claim, empty claim value, unknown purpose code
LOG ENTRY	`{ state: STRUCTURAL_VALIDATE, result: PASS|FAIL, failure_reason? }`
SIGNATURE_VERIFY	
PRE-CONDITION	Structurally valid token available. Trust anchor registry accessible.
ACTION	Resolve key ID against trust anchor registry. Verify algorithm meets minimum policy. Verify cryptographic signature.
POST-CONDITION	Signature verified. Key ID confirmed as non-revoked trust anchor.
FAILURE PATH	→ DENY_EMIT: signature invalid, unknown key, revoked key, weak algorithm, alg:none
LOG ENTRY	`{ state: SIGNATURE_VERIFY, key_id, algorithm, result: PASS|FAIL }`
TEMPORAL_VALIDATE	
PRE-CONDITION	Signature verified. Clock source available.
ACTION	Verify `exp` is future. Verify `nbf` is past. Verify `iat` is not future. Verify lifetime does not exceed policy maximum.
POST-CONDITION	Token confirmed valid within temporal window.
FAILURE PATH	→ DENY_EMIT: expired, not-yet-valid, missing exp, lifetime exceeded, future iat
LOG ENTRY	`{ state: TEMPORAL_VALIDATE, exp, nbf, iat, result: PASS|FAIL }`

REVOCATION_CHECK	
PRE-CONDITION	Temporally valid token. Revocation registry reachable within timeout.
ACTION	Query revocation registry for token ID. Await confirmed response within timeout. Validate response authenticity.
POST-CONDITION	Registry returned CONFIRMED_NOT_REVOKED within timeout.
FAILURE PATH	→ DENY_EMIT: revoked, absent, timeout, malformed response, stale cache, partition
LOG ENTRY	`{ state: REVOCATION_CHECK, registry_status, latency_ms, result: PASS|FAIL }`
POLICY_EVALUATE	
PRE-CONDITION	Token not revoked. Context hash computed and bound. Policy version pinned from INTAKE.
ACTION	Evaluate request against pinned policy version. Apply conflict resolution (DENY wins). Emit PERMIT or DENY with matched rule ID.
POST-CONDITION	Deterministic PERMIT or DENY with matched rule ID and policy version reference.
FAILURE PATH	→ DENY_EMIT: engine failure, null decision, no rule matched (default-deny), DENY rule matched
LOG ENTRY	`{ state: POLICY_EVALUATE, decision, matched_rule_id, policy_version_id }`
ARTIFACT_GENERATE	
PRE-CONDITION	PERMIT decision from POLICY_EVALUATE. Hash engine available.
ACTION	Construct artifact with minimised fields. Compute and embed `artifact_hash`. Bind `context_hash` and `policy_version_id`.
POST-CONDITION	Artifact produced with verified hash. No raw sensitive data present.
FAILURE PATH	→ DENY_EMIT: generator failure, hash failure, missing required field in artifact
LOG ENTRY	`{ state: ARTIFACT_GENERATE, artifact_hash, context_hash, result: PASS|FAIL }`

AUDIT_WRITE	
PRE-CONDITION	Decision (PERMIT or DENY) available. Artifact available (PERMIT only). Audit ledger writable.
ACTION	Append decision record to hash-chained ledger. Await write confirmation. Verify chain integrity for new entry.
POST-CONDITION	Decision record durably written. Hash chain intact.
FAILURE PATH	→ DENY_EMIT (if PERMIT pending): write failure, timeout, chain verification failure
LOG ENTRY	`{ state: AUDIT_WRITE, ledger_entry_id, chain_hash, result: PASS|FAIL }`
DECISION_EMIT	
PRE-CONDITION	AUDIT_WRITE confirmed. Decision finalized.
ACTION	Emit PERMIT or DENY response to caller with decision metadata.
POST-CONDITION	Response delivered to caller. Terminal state reached.
FAILURE PATH	None: terminal state. DENY emitted on any unresolved failure from prior states.
LOG ENTRY	`{ state: DECISION_EMIT, final_decision, request_id, total_latency_ms }`

13.3 Test ID to State Transition Mapping

TABLE 13.1: TEST ID TO STATE MACHINE MAPPING

STATE	PRIMARY TEST IDS	FAILURE PATH TESTS
INTAKE	QTS-01-002, QTS-ADV-008	QTS-FC-001, QTS-ADV-002
STRUCTURAL_VALIDATE	QTS-01-003, QTS-01-004, QTS-01-005, QTS-01-006, QTS-01-007	QTS-ADV-009, QTS-ADV-010
SIGNATURE_VERIFY	QTS-02-001, QTS-02-002, QTS-02-003, QTS-02-004, QTS-02-005, QTS-02-006	QTS-ADV-001, QTS-FC-009
TEMPORAL_VALIDATE	QTS-03-001 through QTS-03-005	QTS-ADV-003, QTS-ADV-004

STATE	PRIMARY TEST IDS	FAILURE PATH TESTS
REVOCATION_CHECK	QTS-04-001 through QTS-04-005	QTS-RC-001 through QTS-RC-006, QTS-FC-002
POLICY_EVALUATE	QTS-05-001 through QTS-05-008	QTS-FC-003, QTS-FC-008, QTS-ADV-006, QTS-ADV-007
ARTIFACT_GENERATE	QTS-08-001 through QTS-08-007, QTS-10-001 through QTS-10-005	QTS-FC-007, QTS-AI-001 through QTS-AI-007
AUDIT_WRITE	QTS-09-001 through QTS-09-006	QTS-FC-005, QTS-AL-001 through QTS-AL-006
DECISION_EMIT	All terminal decision confirmation tests	All QTS-FC-* cases route to DENY at terminal state

SECTION 14

Control-to-Test Coverage Mapping Matrix

This matrix establishes that every control point is covered by at least one negative test, at least one determinism test, and at least one critical classification test. A control point with zero mapped tests renders this specification incomplete for that control. All twelve QCC control points are covered.

> **Coverage Completeness Requirement:** If any row in Table 14.1 contains a zero count in the Negative Cases, Determinism Tests, or Critical Tests columns, the specification is incomplete for that control point and conformance assessment is not permitted for that control.

TABLE 14.1: CONTROL-TO-TEST COVERAGE MAPPING MATRIX

CONTROL POINT	STATE TRANSITION	TEST IDS	TOTAL	NEG.	DETERMINISM TESTS	CONC.	CRIT.
QCC-01 · Token Validity	STRUCTURAL_VALIDATE	QTS-01-001 through QTS-01-007	7	6	QTS-01-001 (baseline)	N/A	5
QCC-02 · Signature Verify	SIGNATURE_VERIFY	QTS-02-001 through QTS-02-006	6	5	QTS-02-001 (baseline)	N/A	5
QCC-03 · Temporal Validity	TEMPORAL_VALIDATE	QTS-03-001 through QTS-03-005	5	4	QTS-03-002 (boundary)	N/A	3
QCC-04 · Revocation	REVOCATION_CHECK	QTS-04-001 through QTS-04-005 + QTS-RC-001 through QTS-RC-006	11	10	QTS-RC-004 (TTL boundary)	QTS-RC-005	9
QCC-05 · Policy Evaluation	POLICY_EVALUATE	QTS-05-001 through QTS-05-008	8	7	QTS-05-006 (replay)	QTS-05-007	7
QCC-06 · Context Hash	POLICY_EVALUATE	QTS-06-001 through QTS-06-006	6	4	QTS-06-003, -004, -006	N/A	4
QCC-07 · Policy Version	POLICY_EVALUATE	QTS-07-001 through QTS-07-005	5	3	QTS-07-002 (drift)	QTS-07-005	3
QCC-08 · Artifact Generation	ARTIFACT_GENERATE	QTS-08-001 through QTS-08-007	7	4	QTS-08-006 (replay)	N/A	4
QCC-09 · Audit Write	AUDIT_WRITE	QTS-09-001 through QTS-09-006	6	4	QTS-09-004 (chain)	QTS-09-005	4

CONTROL POINT	STATE TRANSITION	TEST IDS	TOTAL	NEG.	DETERMINISM TESTS	CONC.	CRIT.
QCC-10 · Data Minimisation	ARTIFACT_GENERATE	QTS-10-001 through QTS-10-005	5	3	QTS-10-005 (replay verify)	N/A	3
QCC-11 · Replay Detection	REVOCATION_CHECK / POLICY_EVALUATE	QTS-11-001 through QTS-11-006	6	5	QTS-11-002 (nonce replay)	QTS-11-006	5
QCC-12 · Concurrency	All states	QTS-12-001 through QTS-12-005	5	3	QTS-12-001 (isolation)	QTS-12-003, -005	4

14.1 Fail-Closed Invariant Mapping

TABLE 14.2: FAIL-CLOSED INVARIANT TO TEST MAPPING

FAIL-CLOSED INVARIANT	APPLICABLE STATE(S)	TEST IDS VERIFYING INVARIANT
DENY on missing token	INTAKE	QTS-01-002, QTS-FC-001
DENY on signature failure	SIGNATURE_VERIFY	QTS-02-002 through QTS-02-006, QTS-ADV-001
DENY on expired token	TEMPORAL_VALIDATE	QTS-03-002, QTS-03-003, QTS-03-004
DENY on revoked token	REVOCATION_CHECK	QTS-04-002, QTS-RC-001 through QTS-RC-006
DENY on registry unavailability	REVOCATION_CHECK	QTS-04-003, QTS-04-004, QTS-FC-002, QTS-RC-006
DENY on policy failure	POLICY_EVALUATE	QTS-05-003, QTS-05-004, QTS-05-008, QTS-FC-003, QTS-FC-008
DENY on artifact failure	ARTIFACT_GENERATE	QTS-08-002, QTS-08-003, QTS-FC-007
DENY on audit write failure	AUDIT_WRITE	QTS-09-002, QTS-09-006, QTS-FC-005

SECTION 15

Canonical Serialization Requirements

Determinism in hash computation and cross-architecture equivalence depends on canonical serialization. Any serialization deviation that produces a different byte sequence for logically identical data constitutes a determinism failure. The following requirements are normative and non-negotiable.

Normative Canonical Serialization Specification · QTS-CS-01

CHARACTER ENCODING	UTF-8 encoding ONLY. No UTF-16, UTF-32, or BOM-prefixed encodings are permitted at any serialization boundary.
OBJECT KEY ORDERING	All JSON object keys MUST be sorted in lexicographic byte order (ascending). Implementations MUST NOT rely on insertion order or language-native dictionary ordering.
WHITESPACE	No insignificant whitespace. No newlines, spaces, or tabs outside of string values. Canonical form is compact JSON with no padding.
INTEGER REPRESENTATION	Integers serialized as decimal without leading zeros. No scientific notation. No floating-point representation for integer-valued quantities. Range: signed 64-bit integer.
FLOATING-POINT	Floating-point values MUST NOT appear in any field that participates in hash computation. If unavoidable, serialize using IEEE 754 double-precision canonical decimal representation with no trailing zeros.
UNICODE NORMALIZATION	All string values MUST be NFC-normalized (Unicode Normalization Form C) before serialization. Strings that are logically equivalent but NFC-distinct MUST produce identical serialized form after normalization.
NULL AND BOOLEAN	Serialize as lowercase literals: `null`, `true`, `false`. No alternatives permitted.

ARRAY ORDERING	Array element order is semantically significant and MUST be preserved exactly. Arrays MUST NOT be re-sorted during serialization.
BYTE SEQUENCES	Raw byte sequences MUST be encoded as lowercase hexadecimal strings. No Base64 or Base64url encoding in hash-participating fields.

15.1 Serialization Conformance Tests

TABLE 15.1: SERIALIZATION CONFORMANCE TEST CASES

TEST ID	VALIDATION TARGET	PROCEDURE	PASS CONDITION	FAILURE CONDITION
QTS-SER-001	Key ordering: lexicographic	Serialize a known object with unordered keys; compare to reference canonical form	Output byte-identical to reference	Key order differs from lexicographic byte sort
QTS-SER-002	Unicode NFC normalization	Submit context with NFC and NFD forms of identical string; verify both produce identical canonical form	Both produce identical serialized bytes	NFC and NFD variants produce different serialized bytes
QTS-SER-003	Whitespace elimination	Submit context with and without insignificant whitespace; verify identical hash	Identical hash regardless of whitespace	Hash differs due to whitespace variation
QTS-SER-004	Integer representation	Serialize integer value 1000; verify representation is `1000` not `1e3`, `1.0e3`, or `01000`	Exact match to decimal integer form	Scientific notation or floating-point form used
QTS-SER-005	Cross-architecture serialization equivalence	Serialize identical context on x86-64 and ARM64; compare byte arrays	Byte-identical output on both architectures	Serialized output diverges across architectures
QTS-SER-006	UTF-8 encoding enforcement	Submit context containing non-ASCII characters; verify UTF-8 encoding in serialized form	Output is valid UTF-8; no BOM; no multi-byte encoding errors	Non-UTF-8 encoding present or BOM prepended

Serialization Divergence = Determinism Failure. Any deviation from the canonical serialization specification that causes two logically identical contexts to produce different byte sequences constitutes a determinism failure under QTS-P-01 and requires root cause isolation before conformance assessment continues.

SECTION 16

Evidence Packaging Requirements

All conformance evidence MUST be packaged in a structured evidence bundle that enables independent reproduction of test results by an external auditor. Evidence bundles not satisfying the following requirements are incomplete and SHALL NOT support a conformance finding.

TABLE 16.1: EVIDENCE BUNDLE COMPONENTS

COMPONENT	DESCRIPTION	FORMAT	OBLIGATION
Decision Logs	Structured log of all DENY/PERMIT decisions with timestamps, context IDs, decision basis, and control point references	Newline-delimited JSON	MUST
Frozen Context Records	Canonical serialized context record for each replay test with `frozen_context_id`	JSON, SHA-256 signed	MUST
Replay Iteration Logs	Full output log for each of N=10 replay iterations per frozen context	Newline-delimited JSON	MUST
Cross-Architecture Results	Decision, hash, and serialization output pairs for each tested architecture pair	JSON per architecture	MUST
Artifact Samples	Representative execution artifacts from positive path tests. Minimum 10 samples.	JSON	MUST
Audit Ledger Extract	Full ledger extract from test window with chain validation report	JSON + validation report	MUST
Environment Description	Hardware, OS, runtime versions, network topology, clock synchronization method	Markdown or structured text	MUST

COMPONENT	DESCRIPTION	FORMAT	OBLIGATION
Configuration Disclosure	Full SUT configuration at test time, excluding production secrets	YAML or JSON	MUST
Registry Fixture Definitions	Deterministic seed definitions for all registry states exercised	JSON	MUST
Serialization Test Outputs	Byte-level serialization outputs for QTS-SER-001 through QTS-SER-006	Hex-encoded byte arrays	MUST
Performance Measurement Exports	HDR histogram raw data, latency percentiles, throughput time series	CSV + summary JSON	MUST
Adversarial Input Corpus	Full set of adversarial payloads submitted during Section 10	JSON array	MUST

16.1 Evidence Integrity Requirements

The complete evidence bundle MUST be hash-sealed at submission time. The bundle manifest MUST include a SHA-256 hash of each component file. Any modification to evidence files after sealing is detectable. The bundle manifest MUST be signed by the test operator using a key verifiable by the receiving certification body.

SECTION 17

Conformance Result Classification

Classification is deterministic. There is no partial credit for proximity to a threshold. The following four classifications are exhaustive.

FULLY CONFORMANT

FC

All mandatory test cases passed. No critical non-conformance findings. Complete evidence bundle sealed and verifiable. Performance thresholds satisfied.

CONDITIONALLY CONFORMANT

CC

All critical test cases passed. One to three non-critical non-conformances. Evidence complete. Remediation plan required within 30 calendar days.

NON-CONFORMANT

NC

One or more mandatory test cases failed without critical designation, or four or more non-critical failures. Complete re-testing of affected control points required.

CRITICAL DETERMINISM FAILURE

CDF

Any fail-closed violation. Any replay divergence. Any cross-architecture divergence. Any critical artifact or ledger integrity failure. Full re-architecture and complete re-test required.

TABLE 17.1: CLASSIFICATION CRITERIA MATRIX

CONDITION	FC	CC	NC	CDF
All Section 6 fail-closed tests passed	Required	Required	Required	N/A
Any Section 6 fail-closed test failed	N/A	N/A	N/A	Automatic CDF
Any QTS-P-01 replay divergence	N/A	N/A	N/A	Automatic CDF
Any QTS-CA cross-architecture divergence	N/A	N/A	N/A	Automatic CDF
Any critical artifact integrity failure	N/A	N/A	N/A	Automatic CDF
Any critical audit ledger failure	N/A	N/A	N/A	Automatic CDF
Any serialization divergence (QTS-SER)	N/A	N/A	N/A	Automatic CDF
All mandatory control point tests passed	Required	Required	N/A	N/A

CONDITION	FC	CC	NC	CDF
1–3 non-critical test failures	N/A	Eligible	N/A	N/A
4+ non-critical test failures	N/A	N/A	Automatic NC	N/A
Evidence bundle complete and sealed	Required	Required	Required	Required
Cross-architecture evidence present	Required	Required	Required	Required
Performance thresholds met (all PASS)	Required	Optional	N/A	N/A

SECTION 18

Institutional Closing Statement

This specification defines the complete and authoritative conformance testing methodology for QODIQA-compliant enforcement systems. Version 1.0 defines twelve control points, with cross-architecture determinism validation, a deterministic gateway state machine model, full control-to-test coverage mapping, and deterministic benchmark methodology. The requirements herein are exhaustive with respect to the defined scope and non-negotiable in their enforcement.

FOUNDATIONAL CONFORMANCE AXIOMS

Conformance is measured through deterministic invariants. There is no probabilistic tolerance, no partial credit, and no acceptable ambiguity in enforcement outcome. A system that produces the correct decision in 99.9% of cases is not conformant.

ENFORCEMENT INVARIANT

Fail-closed behavior is non-negotiable. Any system that permits execution under ambiguous, degraded, or incomplete conditions has failed the most fundamental requirement of consent-bound enforcement. This cannot be remediated by subsequent logging, monitoring, or corrective action.

ARCHITECTURE INVARIANT

Cross-architecture determinism is a first-class conformance requirement. A system that produces identical results on x86-64 but divergent results on ARM64 is non-conformant regardless of deployment context. The enforcement invariant applies uniformly across all architectures and runtime environments.

VERIFICATION REQUIREMENT

Replay reproducibility is mandatory. An enforcement decision that cannot be independently reproduced from its evidence artifacts is not a verified decision. Certification requires not only correct behavior but demonstrably correct, reproducible, architecture-independent behavior.

DOCUMENT STATUS

Document Status and Classification

This document defines the formally complete conformance testing methodology for the QODIQA Gateway Enforcement Layer, the runtime component responsible for evaluating consent tokens, enforcing policy decisions, generating execution artifacts, and writing to the audit ledger.

This specification is not presented as a theoretical framework, but as the authoritative, implementable testing standard against which QODIQA-compliant systems are assessed. Version 1.0 achieves full test card granularity across all twelve control points.

The material contained herein is intended for:

- Systems architects and infrastructure engineers
- Enterprise AI platform designers and certification bodies
- Security and governance officers
- Regulatory and policy stakeholders
- Organizations building or operating autonomous AI systems

All test procedures defined in this document are designed to be production-oriented, independently reproducible, and architected for use within large-scale AI certification workflows. Conformance assessed under this specification is verifiable by an independent auditing body operating under ISO/IEC 17025 principles.

This document should be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0
- QODIQA: Threat Model and Abuse Case Specification, Version 1.0
- QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Duțescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-CTSS-2026-001
TITLE	Conformance Test Suite Specification
SUBTITLE	Formally Complete Verification Protocol for QODIQA Gateway Enforcement Layer
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Technical Test Specification
DOCUMENT STATUS	Normative: Technical Test Specification
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 08 PART II · NORMATIVE TECHNICAL SPECIFICATIONS

Conformance Verification Specification

Defines how conformance is verified, measured, and proven through deterministic verification mechanisms, observable evidence, and reproducible audit procedures.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-CVS-2026-001	1.0	2026	Verification Specification	Normative	NORMATIVE

Status note. No certification body or program exists yet. This is a conformance framework for self-assessment; any reference to certification or certified status describes a planned process, not an existing one.

SECTION 0

Abstract

ABSTRACT

This document defines how conformance to the standard is verified, measured, and proven. It establishes the verification model, verification layers, evidence requirements, and conformance assertion conditions applicable to all systems implementing the QODIQA deterministic runtime consent enforcement model. This document governs verification of conformance only - it does not define conformant behavior, which is established by the QODIQA Core Standard and the 68-Point Enforcement Framework. The governing principle of this document is that if compliance cannot be deterministically

verified, the system is NON-CONFORMANT. No probabilistic validation is permitted. No implicit trust is permitted. No verification process that cannot produce identical results under identical conditions is sufficient for conformance purposes.

> **VERIFICATION AXIOM**
>
> Verification defines conformance. A system that cannot produce deterministic, reproducible, and externally verifiable proof of compliance is NON-CONFORMANT.

SECTION 1

Scope

This document applies to all systems claiming conformance with the standard, all systems undergoing conformance assessment, and all audit processes conducted against QODIQA-certified deployments. The scope of verification coverage is as follows:

- The enforcement layer and all runtime consent enforcement decision pathways
- The policy evaluation engine and all policy resolution and application mechanisms
- Consent resolution processes, including registry queries, state validation, and resolution outputs
- Audit logging subsystems, including record production, integrity protection, and retention
- Cryptographic validation procedures, including signing, verification, and key management operations

This document governs verification of conformance only. It does not define what constitutes conformant behavior. Conformant behavior is defined in the QODIQA Core Standard, the 68-Point Enforcement Framework, and the related normative corpus documents. This document defines how conformance is established, demonstrated, and validated externally.

This document is normative. All verification requirements defined herein SHALL be treated as binding constraints on conformance assessment processes and on systems claiming conformance with this standard.

> CORE PRINCIPLE
>
> If compliance cannot be deterministically verified, the system is NON-CONFORMANT. The absence of demonstrated non-compliance does not constitute demonstrated compliance.

SECTION 2

Normative Foundations

All verification requirements in this document are derived from invariants and obligations established by the following QODIQA corpus documents. These documents are incorporated by reference and are not redefined herein. Verification procedures in this document constitute proof that the invariants of these documents are satisfied and that no failure modes defined therein are present.

- **QODIQA Core Standard for Deterministic Runtime Consent Enforcement** - establishes the mandatory behavioral invariants against which verification is performed.
- **QODIQA 68-Point Enforcement Framework** - enumerates specific enforcement obligations whose satisfaction must be verified.
- **QODIQA Non-Compliance Conditions and Failure Modes (QODIQA-FCM-2026-001)** - defines the failure modes and non-compliance conditions that verification must confirm are absent.
- **QODIQA Certification Framework for Deterministic Runtime Consent Enforcement** - defines certification levels, evidence requirements, and assessment procedures to which this document supplies verification inputs.

- **QODIQA Conformance Test Suite** - specifies the observable test procedures that operationalize verification requirements defined in this document.

Verification, in the context of this document, is the act of producing proof that enforcement invariants are satisfied and that no failure modes defined in QODIQA-FCM-2026-001 are present. Verification is distinct from implementation. A system that correctly implements QODIQA invariants but cannot produce externally verifiable proof of that implementation is NOT conformant.

SECTION 3

Definition of Conformance

A system is conformant with the standard if and only if all of the following conditions are satisfied simultaneously:

- All enforcement invariants defined in the Core Standard and 68-Point Enforcement Framework are satisfied.
- No failure modes defined in QODIQA-FCM-2026-001 are present in any operating condition, including degraded and adversarial conditions.
- All enforcement decisions are deterministic - identical inputs under identical policy state produce identical outputs, reproducibly and without exception.
- All enforcement actions are auditable - every decision is accompanied by a tamper-evident audit record that permits external reconstruction and verification of the decision.
- All results are externally verifiable - an independent auditor can confirm enforcement correctness using only the artifacts produced by the system and the verification procedures defined in this document.

Conformance is binary. A system satisfying all conditions is conformant. A system failing any single condition is non-conformant. Partial conformance is not recognized. A system that satisfies 67 of 68 enforcement points is non-

conformant.

> CONFORMANCE BOUNDARY
>
> A system is conformant if and only if all invariants are satisfied, no failure modes are present, all decisions are deterministic, all actions are auditable, and all results are externally verifiable. Any deviation from any condition results in non-conformance.

SECTION 4

Verification Model

4.1 Deterministic Verification Principle

All verification procedures defined in this document are deterministic. A verification process is deterministic if it produces identical results when applied to identical inputs under identical system state. The following constraints apply to all verification processes without exception:

- Verification MUST produce a binary outcome: PASS or FAIL. No indeterminate outcome is accepted.
- Verification MUST be reproducible. Applying the same verification procedure to the same artifacts MUST yield the same result on every application.
- Probabilistic validation is NOT permitted. No verification process that produces a confidence score, probability estimate, or heuristic assessment constitutes acceptable verification of a QODIQA invariant.
- Verification results MUST NOT depend on timing, execution order, or environmental conditions that are not explicitly specified as input parameters to the verification procedure.

A verification process that fails to satisfy these constraints is not a valid verification process for conformance purposes, regardless of the results it produces.

4.2 Verification Layers

Verification is organized into six layers. Each layer addresses a distinct enforcement surface. All layers MUST be verified for full conformance. Verification of a subset of layers does not constitute verification of conformance.

LAYER 1 - INPUT VALIDATION

What is verified: That all requests submitted to the enforcement layer are validated for structural and semantic correctness before policy evaluation commences. **How it is verified:** Submission of malformed, incomplete, and boundary-condition inputs, with observation that all such inputs are rejected prior to policy evaluation and that rejection is logged. **Observable output:** Rejection artifacts with input fingerprint, timestamp, and rejection reason code, present in the audit log for every rejected input.

LAYER 2 - POLICY EVALUATION VERIFICATION

What is verified: That the policy evaluation engine produces deterministic, correct decisions for all inputs under all applicable policy states. **How it is verified:** Replay of recorded request sets against recorded policy versions, with comparison of output decisions against expected decision records. **Observable output:** Decision artifacts, policy version identifiers, input fingerprints, and evaluation timestamps, all present in the audit log and cryptographically signed.

LAYER 3 - CONSENT RESOLUTION VERIFICATION

What is verified: That consent state resolved during policy evaluation matches current registry state and that resolution is based on registry-confirmed records only. **How it is verified:** Comparison of consent records cited in decision artifacts against registry state at the time of decision, verified against registry audit trail. **Observable output:** Consent record identifiers, registry timestamps, and resolution method present in each decision artifact.

LAYER 4 - ENFORCEMENT DECISION VERIFICATION

What is verified: That enforcement decisions issued by the policy engine are applied without modification, bypass, or delay. **How it is verified:** Comparison of enforcement gateway output records against execution records across all downstream components. No execution record may precede or exist without a corresponding enforcement gateway allow decision. **Observable output:** Enforcement decision artifacts, downstream execution records, and causal linkage tokens present in both gateway and execution logs.

LAYER 5 - AUDIT VERIFICATION

What is verified: That all audit records are complete, append-only, tamper-evident, and independently verifiable. **How it is verified:** Cryptographic chain verification of audit log entries; confirmation that no entries are missing, reordered, or modified; external log integrity verification using published cryptographic commitments. **Observable output:** Cryptographic hash chain, log entry signatures, and completeness proof artifacts.

LAYER 6 - CRYPTOGRAPHIC VERIFICATION

What is verified: That all cryptographic operations conform to the requirements of the QODIQA Security and Cryptographic Profile; that all signatures are valid; that all keys are within authorized operational bounds; and that no prohibited cryptographic operations have been performed. **How it is verified:** Signature verification of all decision artifacts; key inventory audit against authorized key registry; algorithm compliance check against approved algorithm list. **Observable output:** Signature validity records, key usage logs, and algorithm compliance attestations.

4.3 Verification Boundaries

The following conditions govern the scope and trust boundaries of all verification procedures:

- **MUST be verifiable:** Every enforcement decision, consent resolution, audit log entry, and cryptographic operation occurring within the system boundary. Systems MUST produce artifacts sufficient for independent external verification of each of these elements without requiring access to internal system state.
- **MUST NOT be trusted implicitly:** Assertions made by the system about its own conformance. Self-attestation without externally verifiable artifact support does not constitute evidence of conformance. Vendor documentation, configuration files, and code reviews do not substitute for operational verification against live or recorded system artifacts.
- **No hidden states allowed:** The system MUST NOT operate in states that are not reflected in the audit record. Any system state that influences enforcement decisions MUST be externally observable through the audit trail. Systems with hidden operational modes, undocumented configuration paths, or non-logged execution branches are non-conformant with respect to this requirement.

SECTION 5

Conformance Verification Matrix

The following matrix maps enforcement invariants to their corresponding failure modes, verification methods, observable signals, and Conformance Test Suite identifiers. Every invariant in the QODIQA corpus appears in this matrix. Verification of conformance requires that all entries in this matrix produce a PASS result under the specified verification method.

TABLE CVS-1 - CONFORMANCE VERIFICATION MATRIX

INVARIANT ID	FAILURE MODE	VERIFICATION METHOD	OBSERVABLE SIGNAL	TEST MAPPING
INV-ENF-01	FM-EF-1 (Enforcement Bypass)	Inject request without consent check; confirm system denies execution and logs denial artifact.	Denial record with enforcement trace ID in audit log. No execution record present.	CTS-EF-01, CTS-EF-01A
INV-ENF-02	FM-EF-2 (Deny Decision Not Enforced)	Issue deny decision from policy engine; confirm downstream system halts execution and produces halt record.	Policy deny artifact plus downstream halt record with causal token matching policy decision ID.	CTS-EF-02
INV-ENF-03	FM-EF-3 (Enforcement Decision Ignored)	Issue enforcement decision; verify all downstream components consume and act on decision artifact before proceeding.	Decision artifact consumption record in downstream execution log, timestamped after decision issuance.	CTS-EF-03
INV-DEC-01	FM-DI-1 (Policy Evaluation Inconsistency)	Replay identical request against same policy version across multiple instances; compare decision outputs.	Decision artifacts from all replay instances are byte-identical in decision field and policy version field.	CTS-DI-01
INV-DEC-02	FM-DI-2 (Non-Deterministic Output)	Submit well-formed inputs; confirm every output is either ALLOW or DENY with no intermediate or null values.	Decision field in all output artifacts contains only recognized decision tokens. No null, error, or indeterminate values.	CTS-DI-02

INVARIANT ID	FAILURE MODE	VERIFICATION METHOD	OBSERVABLE SIGNAL	TEST MAPPING
INV-DEC-03	FM-DI-3 (State Desynchronization)	Verify consent record timestamp cited in decision artifact against registry audit trail for same consent ID.	Decision artifact consent timestamp matches or postdates most recent registry write for that consent ID.	CTS-DI-03
INV-CON-01	FM-DR-1 (Registry Unavailability Bypass)	Simulate registry unavailability; confirm system applies deny-all and logs unavailability event.	Deny-all enforcement record with registry-unavailable status code in audit log during simulated outage window.	CTS-DR-01
INV-CON-02	FM-DR-2 (Stale Consent Cache)	Update registry consent record; confirm subsequent decisions reflect updated state within defined maximum staleness window.	Decision artifacts post-update cite registry records with timestamps falling within the staleness tolerance window.	CTS-DR-02
INV-AUD-01	FM-AF-1 (Audit Log Absent)	Execute enforcement decision sequence; confirm audit log contains entry for every decision event within defined latency bound.	One-to-one correspondence between enforcement event records and audit log entries, verified by event ID cross-reference.	CTS-AF-01
INV-AUD-02	FM-AF-2 (Log Integrity Failure)	Verify cryptographic hash chain of audit log; attempt controlled modification; confirm modification detection.	Hash chain verification produces PASS on unmodified log; modification attempt produces verifiable chain break detected by integrity check.	CTS-AF-02
INV-AUD-03	FM-AF-3 (Incomplete Audit Record)	Examine audit record for each decision event; confirm all mandatory fields are present and non-null.	All records contain: decision ID, timestamp, policy version, consent record reference, decision outcome, enforcement action, and signing key identifier.	CTS-AF-03
INV-CRYPT-01	FM-CF-1 (Invalid Signature)	Verify digital signature on each decision artifact using published verification key.	Signature verification produces VALID for all decision artifacts produced under normal operation.	CTS-CF-01
INV-CRYPT-02	FM-CF-2 (Prohibited Algorithm)	Audit all cryptographic operations against approved algorithm list in Security and Cryptographic Profile.	Cryptographic operation log contains only algorithm identifiers present in the approved list. No non-listed algorithms present.	CTS-CF-02

INVARIANT ID	FAILURE MODE	VERIFICATION METHOD	OBSERVABLE SIGNAL	TEST MAPPING
INV-BOUND-01	FM-BF-1 (Boundary Bypass)	Attempt execution through all known and probed boundary pathways without enforcement gateway traversal.	No execution record exists without a corresponding prior enforcement gateway allow decision record for the same request ID.	CTS-BF-01
INV-BOUND-02	FM-BF-2 (Control Plane Injection)	Attempt policy modification through non-authorized control paths; confirm modification is rejected and rejection is logged.	Policy version in decision artifacts remains unchanged following unauthorized modification attempt. Rejection event present in audit log.	CTS-BF-02

All invariant IDs correspond to the normative invariant registry defined in the QODIQA Core Standard. All failure mode IDs correspond to the classification in QODIQA-FCM-2026-001. All CTS identifiers correspond to the QODIQA Conformance Test Suite. Any invariant appearing in the Core Standard or 68-Point Enforcement Framework but not reflected in this matrix constitutes an error in this document and must be treated as a verification gap requiring immediate remediation.

SECTION 6

Deterministic Verification Conditions

Each verification rule below defines a complete verification condition. For a system to satisfy a verification rule, all four fields - Condition, Verification Method, Expected Result, and Failure Outcome - must be satisfied exactly as specified. Use of normative language (MUST, SHALL, MUST NOT) in these rules is binding.

VC-01 - ENFORCEMENT GATE COVERAGE

CONDITION	Every execution pathway within the system boundary traverses the enforcement gateway before proceeding.
VERIFICATION METHOD	Execution pathway enumeration against enforcement gateway traversal logs. Every execution event MUST correlate to a prior gateway allow record via request ID.
EXPECTED RESULT	One-to-one correspondence between execution records and gateway allow records. No orphan execution records present.
FAILURE OUTCOME	Any orphan execution record constitutes enforcement bypass. System is NON-CONFORMANT. INV-ENF-01 violated. FM-EF-1 present.

VC-02 - POLICY DECISION DETERMINISM

CONDITION	The policy evaluation engine SHALL produce identical decisions for structurally identical inputs against identical policy state.
VERIFICATION METHOD	Replay of recorded request sets against pinned policy versions. Decision fields in replay output MUST be byte-identical to original decision records.
EXPECTED RESULT	All replay decision artifacts are byte-identical in decision token and policy version to original recorded artifacts.
FAILURE OUTCOME	Any divergence constitutes non-deterministic evaluation. System is NON-CONFORMANT. INV-DEC-01 violated. FM-DI-1 present.

VC-03 - BINARY DECISION CONSTRAINT

CONDITION	The policy engine MUST NOT produce any output that is not a definitive ALLOW or DENY decision for any well-formed input.
VERIFICATION METHOD	Inspection of all decision artifacts across full test coverage. Decision field MUST contain only recognized ALLOW or DENY tokens.
EXPECTED RESULT	All decision artifacts contain exactly one recognized decision token. No null, error, pending, or undefined values present in any decision field.
FAILURE OUTCOME	Any non-binary decision value constitutes an indeterminate enforcement state. System is NON-CONFORMANT. INV-DEC-02 violated. FM-DI-2 present.

VC-04 - CONSENT REGISTRY CURRENCY

CONDITION	All consent records cited in enforcement decisions SHALL reflect registry state current within the maximum staleness tolerance defined in the Core Standard.
VERIFICATION METHOD	Cross-reference of consent record timestamps in decision artifacts against registry write timestamps for each cited consent identifier.
EXPECTED RESULT	All cited consent records are timestamped within the defined staleness tolerance relative to the registry write timestamp for that consent ID.
FAILURE OUTCOME	Any decision citing a consent record outside the staleness tolerance constitutes stale-state enforcement. System is NON-CONFORMANT. INV-DEC-03 violated. FM-DI-3 present.

VC-05 - FAIL-CLOSED UNDER DEGRADED CONDITIONS

CONDITION	The system SHALL apply deny-all enforcement when the consent registry is unavailable or when policy state cannot be confirmed.
VERIFICATION METHOD	Registry unavailability simulation followed by submission of standard request set. All requests MUST be denied. Audit log MUST contain unavailability event and deny-all activation record.
EXPECTED RESULT	Zero execution records during simulated unavailability window. Deny-all audit records present with unavailability status code.
FAILURE OUTCOME	Any execution during unavailability window constitutes registry-bypass enforcement. System is NON-CONFORMANT. INV-CON-01 violated. FM-DR-1 present.

VC-06 - AUDIT LOG COMPLETENESS

CONDITION	The audit log MUST contain a complete record for every enforcement decision event, with no gaps, silent events, or missing mandatory fields.
VERIFICATION METHOD	Event-to-log reconciliation across full test execution. Each enforcement event MUST have a corresponding audit log entry with all mandatory fields populated.
EXPECTED RESULT	Audit log entry count equals enforcement event count. All entries contain all mandatory fields. No silent events present.
FAILURE OUTCOME	Any missing entry, silent event, or incomplete record constitutes audit failure. System is NON-CONFORMANT. INV-AUD-01 or INV-AUD-03 violated.

VC-07 - AUDIT LOG INTEGRITY

CONDITION	The audit log SHALL be append-only and tamper-evident. Any modification to a committed log entry MUST be detectable.
VERIFICATION METHOD	Cryptographic hash chain verification of the full audit log. Controlled modification of one entry followed by re-verification; detection of chain break is required.
EXPECTED RESULT	Hash chain verification PASS on unmodified log. Chain break detected on modified log at the position of modification.
FAILURE OUTCOME	Undetected modification constitutes log integrity failure. System is NON-CONFORMANT. INV-AUD-02 violated. FM-AF-2 present.

VC-08 - CRYPTOGRAPHIC SIGNATURE VALIDITY

CONDITION	All decision artifacts MUST carry a valid digital signature produced using an authorized key and an approved algorithm.
VERIFICATION METHOD	Signature verification of all decision artifacts using the published verification key set. Algorithm field verification against approved algorithm list.
EXPECTED RESULT	All signatures valid. All algorithms present in approved list. No decision artifact with missing, invalid, or non-approved-algorithm signature.
FAILURE OUTCOME	Any invalid signature or non-approved algorithm constitutes cryptographic failure. System is NON-CONFORMANT. INV-CRYPT-01 or INV-CRYPT-02 violated.

SECTION 7

Evidence Requirements

Conformance assessment requires the following categories of evidence. Each category defines the artifact type, the integrity requirements for that artifact, and the conditions under which the artifact constitutes valid evidence. Evidence that does not satisfy all stated requirements for its category is not accepted for conformance purposes.

7.1 Enforcement Decision Artifacts

Every enforcement decision MUST produce a signed decision artifact. The artifact MUST contain: decision ID, request fingerprint, policy version identifier, consent record reference, resolution timestamp, decision token (ALLOW or DENY),

enforcement action taken, and signing key identifier. The artifact MUST be digitally signed using a key and algorithm compliant with the Security and Cryptographic Profile. Unsigned or partially signed artifacts are not valid evidence.

7.2 Audit Log Records

The audit log constitutes the primary evidence corpus for conformance verification. Audit log records MUST be append-only, cryptographically chained, and independently verifiable without access to internal system state. Logs MUST be retained for the duration specified in the Core Standard. Truncated, modified, or non-chained logs are not valid evidence. Logs that require access to system internals for verification are not valid evidence.

7.3 Decision Traces

For each enforcement decision, a decision trace MUST be producible on demand by the system. The trace MUST reconstruct the full evaluation path, including: input parameters, policy rules applied, consent state consulted, intermediate evaluation states, and final decision. Decision traces MUST be reproducible. A system that cannot produce a decision trace for any prior enforcement decision is non-conformant with respect to auditability requirements.

7.4 Cryptographic Validation Records

The system MUST maintain a cryptographic validation record for each signed artifact. Validation records MUST include: the artifact identifier, the verification key identifier used, the algorithm applied, the verification timestamp, and the verification result. Validation records MUST themselves be signed and included in the audit log chain.

7.5 Evidence Integrity Requirements

All evidence submitted for conformance verification MUST satisfy the following integrity requirements:

- **Tamper-evident:** Any modification to a submitted evidence artifact must be detectable by the verification process. Evidence that is not tamper-evident is not accepted.
- **Reproducible:** The evidence must have been produced by a deterministic process. Evidence derived from non-deterministic processes is not accepted.
- **Verifiable:** The correctness and completeness of the evidence must be confirmable by an independent party using only the artifacts themselves and the verification procedures in this document. Evidence requiring access to system internals for verification is not accepted.

SECTION 8

Verification Failure Conditions

The following conditions constitute verification failures. Each verification failure results in a finding of NON-CONFORMANCE with respect to the corresponding invariant. All findings are linked to specific failure modes defined in QODIQA-FCM-2026-001.

VF-01 - VERIFICATION CANNOT BE COMPLETED

CONDITION	Verification of any mandatory verification layer or verification condition cannot be completed due to missing artifacts, inaccessible logs, or system non-cooperation.
RESULT	System is NON-CONFORMANT. Incomplete verification does not default to a PASS finding. The burden of enabling verification rests with the system operator.
FCM REFERENCE	QODIQA-FCM-2026-001 Section 3.2 (Soft Non-Compliance - inability to demonstrate compliance)

VF-02 - VERIFICATION RESULT NOT REPRODUCIBLE

CONDITION	Applying the same verification procedure to the same artifacts produces different results on different applications.
RESULT	System is NON-CONFORMANT. Non-reproducible verification indicates non-deterministic enforcement or evidence tampering. Both conditions result in non-conformance.
FCM REFERENCE	QODIQA-FCM-2026-001 Section 4.2 (Decision Integrity Failure - FM-DI-1)

VF-03 - EVIDENCE DOES NOT SATISFY INTEGRITY REQUIREMENTS

CONDITION	Evidence submitted for conformance assessment fails the tamper-evident, reproducible, or verifiable requirements defined in Section 7.5.
RESULT	Evidence is rejected. System is NON-CONFORMANT with respect to auditability requirements. No conformance claim based on non-qualifying evidence is accepted.
FCM REFERENCE	QODIQA-FCM-2026-001 Section 4.5 (Audit Failure - FM-AF-2, FM-AF-3)

VF-04 - VERIFICATION PRODUCES NON-BINARY RESULT

CONDITION	A verification procedure produces an indeterminate, conditional, or probabilistic result rather than a binary PASS or FAIL.
RESULT	Indeterminate verification result is treated as FAIL. System is NON-CONFORMANT for the corresponding invariant until a PASS result is obtained from a valid deterministic verification procedure.
FCM REFERENCE	QODIQA-FCM-2026-001 Section 4.2 (Decision Integrity Failure - FM-DI-2)

VF-05 - AUDIT LOG VERIFICATION FAILS

CONDITION	Cryptographic hash chain verification of the audit log produces a failure, or the log contains gaps corresponding to enforcement events.
RESULT	System is NON-CONFORMANT. Audit log failure invalidates all evidence claims dependent on that log. The scope of non-conformance extends to all enforcement events within the affected log window.
FCM REFERENCE	QODIQA-FCM-2026-001 Section 4.5 (Audit Failure - FM-AF-1, FM-AF-2)

VERIFICATION FAILURE RESOLUTION

No verification failure may be resolved by resubmitting the same artifacts or by providing additional explanation. Verification failures are resolved only by providing new evidence from a system that has corrected the underlying non-conformance condition. Re-verification is required in full for the affected verification layer.

SECTION 9

Verification Observability

All verification processes defined in this document are based on externally observable system outputs. The following requirements govern observability of verification:

REQUIREMENT VO-1 - OBSERVABLE OUTPUTS REQUIRED

Every verification condition MUST produce an externally observable output. Verification processes that rely on internal system assertions, vendor claims, or code-level inspection do not satisfy this requirement. Observable output means an artifact produced by the system during operation that can be retrieved, inspected, and verified by an independent party without special system access.

REQUIREMENT VO-2 - NO SILENT VALIDATION

Systems MUST NOT perform validation processes that do not produce a corresponding audit record. Any validation step that influences enforcement behavior MUST leave an audit trail. Silent validation - validation that occurs but produces no externally observable record - is prohibited. A system that performs silent validation is non-conformant regardless of whether the validation itself would have produced a correct result.

REQUIREMENT VO-3 - EXTERNAL AUDIT MUST BE POSSIBLE

A qualified external auditor MUST be able to verify enforcement correctness using only the artifacts produced by the system during operation. The system MUST NOT require the auditor to have access to source code, internal databases, runtime state, or system internals to perform verification. All artifacts necessary for complete verification MUST be producible on demand by the system without privileged access.

REQUIREMENT VO-4 - VERIFICATION OUTPUT RETENTION

All artifacts produced during verification processes MUST be retained for the period specified in the Core Standard. Expired or deleted verification artifacts that were required for conformance demonstration constitute an evidence gap. Evidence gaps are treated as non-conformance conditions for the period in which the artifacts are unavailable.

SECTION 10

Verification Execution Model

Verification is executed at three defined points in the system lifecycle. All three verification phases MUST be supported. A system that supports verification at fewer than three phases is non-conformant with respect to lifecycle coverage requirements.

10.1 Pre-Execution Verification

Pre-execution verification is performed before the system enters operational service or before a significant configuration change takes effect. Pre-execution verification MUST confirm:

- Policy engine determinism under a comprehensive input test set
- Enforcement gateway coverage of all defined execution pathways
- Audit subsystem integrity and write capability
- Cryptographic subsystem key validity and algorithm compliance
- Consent registry connectivity and state currency mechanisms

Pre-execution verification MUST produce a signed pre-execution verification report. Entry into operational service is prohibited until pre-execution verification produces an all-PASS result.

10.2 Runtime Verification

Runtime verification is performed continuously during operational service. Runtime verification MUST include:

- Continuous audit log integrity monitoring
- Enforcement decision completeness monitoring - no enforcement event may occur without a corresponding audit record
- Consent registry state currency monitoring
- Cryptographic signature validation for all decision artifacts at issuance

Runtime verification MUST produce alerting artifacts for any detected anomaly. Anomaly detection is not sufficient; the anomaly MUST trigger a corresponding audit record and, where defined by the Core Standard, a fail-closed response.

10.3 Post-Execution Verification

Post-execution verification is performed following a defined operational period as part of ongoing compliance assurance, and following any suspected security incident or anomalous event. Post-execution verification MUST include:

- Full audit log hash chain verification for the covered period
- Decision trace reconstruction for a sampled set of enforcement decisions
- Policy evaluation replay against recorded request sets
- Cryptographic key usage audit

Post-execution verification MUST produce a signed post-execution verification report. Findings from post-execution verification MUST be resolved before the system is re-certified or before any certification claim is extended.

SECTION 11

Certification Integration

Verification results produced in accordance with this document are the primary input to the QODIQA Certification Framework assessment process. The relationship between verification and certification is governed by the following requirements.

11.1 Verification as Prerequisite to Certification

No certification claim at any level defined in the Certification Framework may be made without successful completion of all verification conditions defined in this document. Certification assessment begins only after all verification procedures have produced all-PASS results for the certification scope.

11.2 Mapping to Certification Levels

All certification levels defined in the Certification Framework require successful verification of all mandatory verification conditions in this document. There are no certification levels that are exempt from any subset of these verification conditions. The following mapping applies:

- **All certification levels:** VC-01 through VC-08 (full verification condition set), all six verification layers (Section 4.2), and all evidence categories (Section 7) are mandatory.
- **Higher certification levels** may require additional verification procedures as defined in the Certification Framework, but shall not require fewer procedures than those defined in this document.

11.3 Mandatory Verification Requirements for Certification

The following are mandatory for any certification claim:

- Signed pre-execution verification report with all-PASS result
- Runtime verification monitoring capability demonstrated and active
- Post-execution verification report for a representative operational period
- Complete evidence corpus meeting all Section 7 integrity requirements
- No open verification failure conditions at time of certification assessment

11.4 Verification Failure Impact on Certification

Any verification failure condition, if discovered after certification has been issued, MUST be reported to the certification authority as defined in the Certification Framework. Discovery of a verification failure condition during a certified operational period triggers the certification suspension procedures defined in the Certification Framework. Certification is not restored until the verification failure is resolved and post-execution verification produces an all-PASS result for the affected verification layers.

SECTION 12

Prohibited Verification Behaviors

The following behaviors are prohibited in any verification process presented in support of a conformance claim. Use of any prohibited verification behavior invalidates all findings produced by that verification process.

PVB-01 - PARTIAL VERIFICATION

PROHIBITED BEHAVIOR	Submitting verification results covering a subset of required verification conditions or layers as evidence of full conformance.
EFFECT	All findings from the partial verification are invalid. System MUST NOT claim conformance based on partial verification results.

PVB-02 - CACHED VALIDATION WITHOUT PROOF

PROHIBITED BEHAVIOR	Asserting that a prior verification result remains valid without re-executing verification or providing cryptographic proof of unchanged system state since the prior verification.
EFFECT	Cached results without cryptographic continuity proof are not accepted as evidence. Re-verification is required for all affected conditions.

PVB-03 - IMPLICIT TRUST

PROHIBITED BEHAVIOR	Asserting conformance based on trust in vendor claims, design documentation, code reviews, or architectural descriptions without operational artifact-based verification.
EFFECT	No trust-based assertion constitutes evidence of conformance. All conformance claims MUST be supported by operational artifacts produced by the system in execution.

PVB-04 - NON-DETERMINISTIC CHECKS

PROHIBITED BEHAVIOR	Using probabilistic, heuristic, or sampling-based verification methods as the basis for a binary conformance finding.
EFFECT	Non-deterministic checks do not produce valid PASS findings. Probabilistic analysis may be used as supplementary diagnostic information but MUST NOT be used as the basis for a conformance assertion.

PVB-05 - SILENT PASS

PROHIBITED BEHAVIOR	A verification procedure that produces a PASS result without generating a corresponding verification artifact - a signed record containing the procedure ID, input parameters, execution timestamp, and result.
EFFECT	Silent PASS results are not accepted as evidence of conformance. Every PASS result MUST be supported by a signed verification artifact that is included in the audit record and available for independent inspection.

SECTION 13

Conformance Assertion

A system MAY assert conformance with the standard if and only if all of the following conditions are simultaneously satisfied at the time of assertion:

ASSERTION CONDITION CA-1 - ALL VERIFICATION CONDITIONS SATISFIED

All verification conditions VC-01 through VC-08 have been executed in accordance with this document and have produced PASS results. Each PASS result is supported by a signed verification artifact present in the audit record. No open verification failure conditions exist.

ASSERTION CONDITION CA-2 - ALL RESULTS ARE REPRODUCIBLE

All verification results are reproducible. The system operator MUST be able to demonstrate, on demand, that re-execution of any verification procedure against the same artifacts produces the same result. Non-reproducible results invalidate the assertion regardless of the original result value.

ASSERTION CONDITION CA-3 - ALL EVIDENCE IS AVAILABLE

The complete evidence corpus required under Section 7 is available, intact, and satisfies all integrity requirements of Section 7.5. Evidence that has expired, been deleted, or fails integrity verification is not available for assertion purposes. The assertion period is bounded by the temporal coverage of available evidence.

ASSERTION CONDITION CA-4 - NO PROHIBITED BEHAVIORS PRESENT

None of the prohibited verification behaviors defined in Section 12 have been employed in the production of any verification result or evidence artifact presented in support of the assertion. Verification results produced using prohibited methods do not satisfy CA-1 regardless of the result value.

ASSERTION CONDITION CA-5 - NO OPEN FAILURE MODES

No failure modes defined in QODIQA-FCM-2026-001 are present in the system. All failure mode verification conditions in the Conformance Verification Matrix (Section 5) have produced PASS results confirming the absence of each defined failure mode.

CONFORMANCE ASSERTION CONSTRAINT

A conformance assertion is valid only for the scope and period covered by the evidence corpus. Changes to system configuration, policy versions, enforcement logic, or infrastructure components that post-date the evidence corpus require re-verification before the assertion may be extended to cover the changed system. The burden of demonstrating conformance rests with the system operator at all times.

SECTION 14

Residual Verification Limitations

This section identifies conditions under which complete verification cannot be achieved. These limitations are derived from the residual risk surfaces documented in the QODIQA Risk and Assumption Disclosure Annex and represent the boundaries of what deterministic verification can confirm. Systems subject to these limitations MUST apply the fail-closed response defined for each condition.

14.1 What Cannot Be Verified

The following conditions cannot be fully addressed by the verification procedures in this document:

- **Undisclosed execution pathways:** Verification can only confirm the absence of enforcement gaps for execution pathways that are known and enumerated. Previously unknown pathways, including pathways introduced by undisclosed system components or supply chain compromise, cannot be verified against.

- **Hardware-level consent bypass:** Verification procedures operate above the hardware abstraction layer. Enforcement bypass achieved at the hardware level, including through firmware modification or physical access, may not produce artifacts detectable by software-layer verification.
- **Cryptographic assumption failure:** Verification of cryptographic integrity depends on the continued strength of the algorithms and key lengths specified in the Security and Cryptographic Profile. Advances that undermine these assumptions may retroactively invalidate verification results without detection by the verification procedures themselves.
- **Registry state at exact decision time:** Verification confirms consent record currency within the defined staleness tolerance. It cannot confirm the exact registry state at the precise moment of a decision to sub-tolerance precision.

14.2 Required System Behavior Under Residual Limitations

Where verification cannot be completed due to a residual limitation, the system MUST apply fail-closed behavior. Specifically:

- Any execution pathway that cannot be confirmed as covered by the enforcement gateway MUST be disabled.
- Any cryptographic operation that cannot be verified against the approved algorithm list MUST be rejected.
- Any enforcement decision for which a complete audit record cannot be produced MUST be treated as if no decision was made, and deny-all MUST be applied.
- The existence of a residual limitation in a given area does not excuse reduced verification effort in that area. Systems MUST apply maximum available verification coverage even where residual limitations prevent complete verification.

RESIDUAL LIMITATION ACKNOWLEDGMENT

The identification of residual verification limitations in this section does not constitute acceptance that systems operating under those limitations are conformant. A system operating within a residual limitation surface is conformant only if it applies fail-closed behavior as required by Section 14.2. A system that does not apply fail-closed behavior in a residual limitation area is NON-CONFORMANT, regardless of whether the limitation is acknowledged in this document.

DOCUMENT STATUS

Document Status and Corpus Alignment

This document is a normative component of the QODIQA specification corpus. It defines the verification model, verification procedures, evidence requirements, and conformance assertion conditions applicable to all systems implementing the QODIQA deterministic runtime consent enforcement model.

This document governs verification of conformance only. It does not define conformant behavior, which is established by the Core Standard and the 68-Point Enforcement Framework. It must be read in conjunction with the normative corpus documents from which its verification requirements are derived and to which its verification outputs feed.

This document should be read together with the following related specifications:

- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Non-Compliance Conditions and Failure Modes
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement
- QODIQA: Residual Risk and Assumption Disclosure Annex

— QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Duțescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-CVS-2026-001
TITLE	Conformance Verification Specification - Deterministic Runtime Consent Enforcement
SUBTITLE	Verification Model, Evidence Requirements, and Conformance Assertion Conditions for Conformant Systems
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Conformance Verification Specification
DOCUMENT STATUS	Normative
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 09 PART II · NORMATIVE TECHNICAL SPECIFICATIONS

Threat Model and Abuse Case Specification

Formal adversarial threat model for deterministic runtime consent enforcement architecture describing threat actors, attack surfaces, bypass scenarios, and residual risks.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-SEC-TM-2026-001	1.0	2026	Security Analysis	Normative	NORMATIVE

ABSTRACT

Deterministic authorization systems require explicit adversarial analysis. This document provides a formal threat model for the QODIQA runtime enforcement architecture, analyzing the complete adversarial surface of a system whose security guarantees depend on the integrity of cryptographic identity, registry availability, policy evaluation determinism, and governance certification fidelity. The analysis enumerates threat actors, identifies attack surfaces, documents bypass scenarios, evaluates cryptographic compromise risks, and characterizes residual risks that persist under correct implementation. The document does not prescribe deployment controls; it defines the conditions under which architectural guarantees hold and the conditions under which they fail. QODIQA does not eliminate adversarial risk. It defines deterministic enforcement boundaries.

SECTION 0

Introduction

This document defines a formal adversarial threat model for the QODIQA runtime enforcement architecture. The purpose of adversarial modeling is to identify conditions under which the enforcement guarantees described in the QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, the QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, and the QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement may be weakened, bypassed, or invalidated by intentional adversarial action.

The importance of documenting enforcement failure modes is structural. A deterministic consent enforcement system derives its value from the predictability of its decisions. When adversarial actors successfully manipulate inputs, compromise infrastructure, or exploit deployment configurations, the resulting behavior may appear valid to the system while producing outcomes that violate the consent constraints it is designed to enforce. Understanding these scenarios is a prerequisite for meaningful conformance evaluation and operational security planning.

The scope of this analysis encompasses the complete QODIQA enforcement boundary as defined in the QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, including the consent registry, enforcement gateway, cryptographic verification layer, policy evaluation engine, identity infrastructure, and audit logging infrastructure. External AI inference systems that interface with the enforcement boundary are considered within the analysis to the extent that they represent attack vectors targeting enforcement integrity.

> **SCOPE LIMITATION**
>
> This document does not model threats to AI model behavior, training data integrity, or application-layer security posture beyond the enforcement boundary. Those threat domains are addressed in separate corpus documents. The analysis is bounded to the enforcement architecture as specified.

The limitations of this threat model follow from three constraints. First, the model assumes correct specification of the QODIQA architecture; defects in the underlying specification may introduce additional threat classes not enumerated here. Second, the model does not account for regulatory, legal, or contractual coercion mechanisms that operate outside the technical boundary. Third, adversarial capabilities are characterized at the class level; specific implementation vulnerabilities in particular deployments may alter the residual risk profile materially.

Readers should interpret this document as a formal analytical baseline, not as a comprehensive operational risk assessment for any specific deployment. Organizations implementing QODIQA-conformant systems are expected to perform deployment-specific threat assessments that extend from this baseline.

SECTION 1

Threat Modeling Methodology

This document employs a hybrid threat modeling methodology that combines the STRIDE threat taxonomy, trust boundary analysis, misuse case modeling, and adversarial objective decomposition. A hybrid methodology is required because the QODIQA enforcement architecture combines four distinct security domains: cryptographic identity, runtime enforcement, governance infrastructure, and certification frameworks. Each of these domains presents distinct threat classes and requires different analytical approaches to characterize adequately.

1.1 STRIDE Integration

The STRIDE taxonomy (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege) is applied systematically across identified attack surfaces. Each surface is evaluated against all six STRIDE categories to ensure coverage of both active and passive threat classes. The taxonomy is applied at the interface level rather than the component level, consistent with the enforcement boundary defined in the QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement.

1.2 Trust Boundary Analysis

Trust boundary analysis identifies the points at which data or control flow crosses a trust domain boundary within the QODIQA enforcement architecture. The enforcement gateway represents the primary trust boundary; all requests crossing this boundary are subject to adversarial scrutiny. Secondary boundaries exist between the gateway and the consent registry, between the registry and the identity infrastructure, and between the policy evaluation engine and the audit logging infrastructure. Each boundary transition is a candidate for tampering, injection, or interception attacks.

1.3 Misuse Case Modeling and Adversarial Objective Decomposition

Misuse case modeling extends the system use case model to represent adversarial interaction patterns. Each enforcement use case is paired with a corresponding misuse case that inverts the intended preconditions or outcomes. Adversarial objective decomposition structures the analysis around adversarial goals rather than system components, ensuring that multi-step attacks that traverse multiple components are captured as coherent threat scenarios rather than fragmented component-level events.

The combination of these methods provides layered coverage: STRIDE ensures no standard threat category is omitted; trust boundary analysis ensures boundary transitions are analyzed; misuse cases ensure adversarial intent is represented; and objective decomposition ensures that complex attack chains are captured at the scenario level.

SECTION 2

System Boundary Definition

The QODIQA enforcement boundary defines the set of components whose integrity is necessary and sufficient for the enforcement guarantees of the architecture to hold. The boundary is specified at the architectural level; deployment-specific instantiations may vary in topology and implementation, but the logical boundary remains constant.

2.1 Components Inside the Enforcement Boundary

The following components are considered inside the enforcement boundary for the purposes of this threat model:

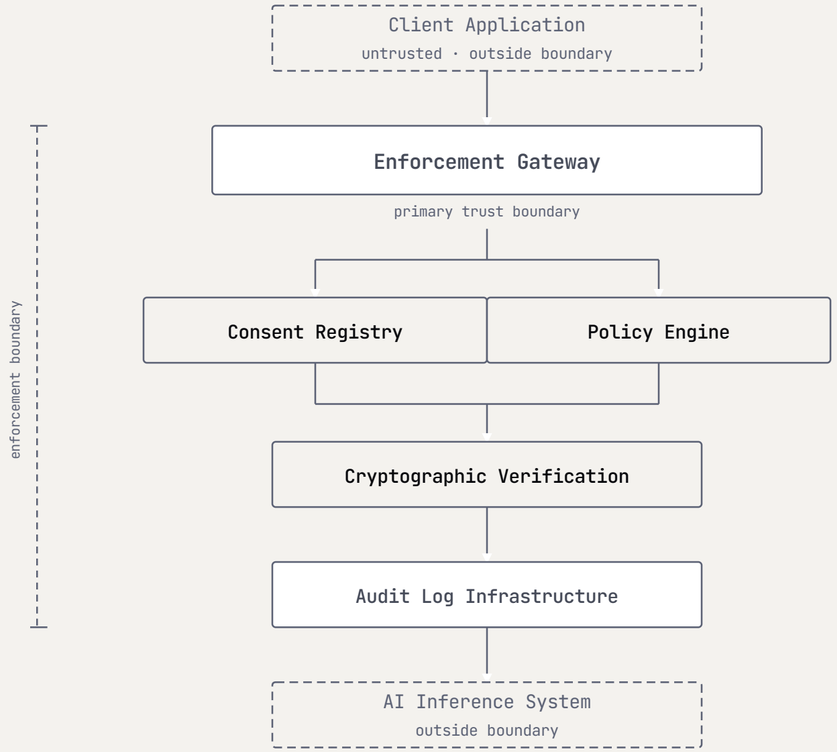
- **Enforcement Gateway.** The primary interception point for all AI inference requests. The gateway is responsible for intercepting requests, invoking consent verification, evaluating policy decisions, and either permitting or blocking execution. The integrity of the gateway is the single most critical component in the enforcement architecture.
- **Consent Registry.** The authoritative store of consent state, scoped by subject, purpose, data category, and temporal validity. The registry must maintain integrity and availability to support enforcement decisions. Consent state is read-only from the perspective of the enforcement gateway.
- **Identity Infrastructure.** The cryptographic identity layer that authenticates requestors, data subjects, and purpose declarations. This includes public key infrastructure, identity providers, and certificate management systems to the extent they are used in consent artifact generation and verification.
- **Cryptographic Verification Layer.** The component responsible for verifying the cryptographic authenticity and integrity of consent artifacts, decision tokens, and audit records. This layer validates signatures, checks certificate chains, and enforces expiration constraints on signed artifacts.
- **Policy Evaluation Engine.** The component that evaluates enforcement rules against declared intent, resolved consent state, and active policy versions. Policy evaluation must be deterministic, stateless with respect to external mutable state, and isolated from execution context.
- **Audit Log Infrastructure.** The tamper-evident logging system that records all enforcement decisions, inputs, and outcomes. Audit log integrity is required for post hoc verification of enforcement claims and conformance assessments.

2.2 Components Outside the Enforcement Boundary

The following are explicitly outside the enforcement boundary and are treated as untrusted inputs or external dependencies:

- **AI Inference Systems.** AI models, inference services, and model APIs are outside the boundary. The enforcement gateway controls whether these systems are invoked, but the behavior of the models themselves, including prompt handling, output generation, and sampling, is outside the enforcement scope. Compromise of an AI inference system does not by itself compromise enforcement decisions, provided the gateway remains intact.
- **Application Layer.** Client applications, orchestration frameworks, and integration middleware that construct and submit requests to the enforcement gateway are outside the boundary. These components are considered adversarial by default in the threat model.
- **Consent Collection Interfaces.** User interfaces, consent forms, and mechanisms by which data subjects provide or revoke consent operate outside the boundary. The accuracy and completeness of consent state in the registry depends on the integrity of these interfaces, but they are not components of the enforcement architecture.
- **Governance and Certification Bodies.** External organizations responsible for issuing QODIQA conformance certifications operate outside the enforcement boundary. Their integrity affects the trust value of certification claims but does not alter enforcement behavior.

Diagram: Enforcement Boundary Overview



THE ENFORCEMENT BOUNDARY ENCOMPASSES ALL COMPONENTS FROM THE ENFORCEMENT GATEWAY THROUGH THE AUDIT LOG INFRASTRUCTURE. THE CLIENT APPLICATION AND AI INFERENCE SYSTEM ARE OUTSIDE THE BOUNDARY AND ARE TREATED AS UNTRUSTED. ALL CONSENT RESOLUTION, POLICY EVALUATION, CRYPTOGRAPHIC VERIFICATION, AND AUDIT LOGGING OCCUR BEFORE INFERENCE EXECUTION IS PERMITTED.

SECTION 3

Modeling Assumptions

The threat model presented in this document is valid subject to the following baseline assumptions. Violation of any assumption does not invalidate the architecture, but it may render specific threat mitigations ineffective and may alter the residual risk profile materially.

> **FOUNDATIONAL CONSTRAINT**
>
> Incorrect operational practices invalidate the threat model. The architectural guarantees described in this document presuppose that the implementation, configuration, and operational management of QODIQA components conform to the specifications referenced in the corpus alignment section. Deviations from these specifications constitute conditions outside the scope of this analysis.

The following assumptions are stated explicitly for analytical scope:

- **Correct cryptographic implementation.** Cryptographic primitives, signature schemes, hash functions, key derivation functions, and random number generation, are implemented correctly and conform to the algorithm profiles specified in the QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement. Cryptographic implementation defects are out of scope.
- **Integrity of the consent registry at rest.** The consent registry is assumed to be correctly initialized with authentic consent state. Threats to the accuracy of initial consent state captured by collection interfaces are out of scope.
- **Correct deployment configuration.** The enforcement gateway is deployed in a network topology that ensures all AI inference requests traverse the gateway. Architectural bypass through misconfiguration, such as direct routing to inference endpoints that circumvents the gateway, is a deployment failure, not an architectural vulnerability.

- **Secure key management.** Private keys used in consent artifact signing and enforcement token issuance are managed in conformance with the key management provisions of the QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement. Key material is not accessible to components outside the defined trust boundary.
- **Audit log write integrity.** The audit logging infrastructure is write-only from the perspective of the enforcement components that generate records. Audit records cannot be modified or deleted by components within the enforcement boundary.
- **Policy version immutability.** Policy versions, once published, are immutable. Policy evaluation always references a specific, frozen version. Retroactive modification of published policy versions is not possible within the architecture.

Security Assumptions

In addition to the modeling assumptions above, the threat analysis presented in this document rests on a set of foundational security conditions. These conditions are assumed to hold at the time of enforcement system deployment and operation. The cryptographic primitives used for consent artifact integrity, including signature schemes and hash functions, are assumed to be computationally secure and not subject to known breaks or feasible cryptanalytic attacks. Private keys used for artifact signing and enforcement token issuance are assumed to be stored and managed using secure key-management mechanisms, such as hardware security modules or operationally equivalent controls that prevent unauthorized extraction. Communication between all enforcement boundary components is assumed to occur over authenticated and encrypted transport channels, such that in-transit interception or modification is not feasible for the threat actor classes defined in Section 5. Enforcement Gateway components are assumed to execute within trusted runtime environments whose process integrity is protected against unauthorized modification during operation. The Consent Registry is assumed to maintain append-only integrity guarantees, such that previously recorded consent state cannot be silently overwritten or deleted by components operating within the enforcement boundary.

Violation of any of these security assumptions may invalidate specific portions of the threat analysis presented in this document. Where an assumption is violated, the mitigations described for associated threats should not be relied upon

without independent evaluation of the resulting risk exposure.

SECTION 4

Severity Classification

Threat severity is classified on a four-level scale. Classification is determined by the intersection of four criteria: enforcement bypass feasibility, scale of impact, detectability, and reversibility. Threats that achieve a high score across multiple criteria receive the highest severity designation.

TABLE 1: SEVERITY CLASSIFICATION MODEL

LEVEL	DESIGNATION	BYPASS FEASIBILITY	SCALE OF IMPACT	DETECTABILITY	REVERSIBILITY
1	Critical	Direct or near-direct	System-wide; affects all enforcement decisions	Low; may be undetectable during active exploitation	Irreversible without full re-keying or re-deployment
2	High	Feasible with moderate capability	Targeted; affects specific decision classes or data domains	Low to moderate; detectable only via forensic review	Reversible with intervention; may require policy revision
3	Medium	Requires specific preconditions	Limited; affects single request flows or isolated subjects	Moderate; detectable through audit log analysis	Reversible; correctable through operational controls
4	Low	Theoretically possible; requires high effort or rare conditions	Minimal; affects fringe cases or degrades secondary properties	High; typically generates observable artifacts	Reversible; self-healing or administratively correctable

Enforcement bypass feasibility is the primary criterion. A threat that achieves any degree of direct enforcement bypass is classified as Critical or High regardless of detection properties or reversibility, because the foundational security

property of the architecture, that no inference executes without verified consent, is the invariant most requiring protection.

Threat Likelihood Model

Threats within this model are evaluated not only by severity of impact but also by the likelihood of occurrence in realistic adversarial conditions. Likelihood is assessed against three qualitative categories. **Low Likelihood** threats are those requiring exceptional technical capabilities, privileged access to protected infrastructure, or the coordinated compromise of multiple independent system components; such threats are theoretically feasible but demand resources and coordination that substantially constrain the population of credible adversaries. **Medium Likelihood** threats are those achievable by well-resourced adversaries, insiders with partial access, or sophisticated external attackers who have acquired partial knowledge of system configuration or internal trust domain structure. **High Likelihood** threats are those arising from configuration errors, operational mistakes, weak integration practices, or predictable patterns of misuse that do not require specialized adversarial capability to exploit.

Risk evaluation in the QODIQA threat model considers both impact severity and likelihood of occurrence. Likelihood is described qualitatively rather than through numerical probability scoring, as the adversarial conditions for enforcement bypass are highly deployment-specific and do not admit meaningful quantification at the architectural level. Readers should map likelihood categories to their specific deployment context when conducting deployment-specific risk assessments.

SECTION 5

Threat Actor Taxonomy

This document recognizes five adversarial actor classes. The classification is based on the actor's position relative to the enforcement boundary, their technical capability, and their primary adversarial objective.

5.1 External Attacker

An external attacker operates outside the deployment boundary and has no legitimate access to enforcement infrastructure. Capabilities include network-level interception, public API enumeration, and exploitation of exposed interfaces. The primary goal of an external attacker is to inject forged requests, replay valid consent artifacts, or enumerate information about policy configuration that facilitates more targeted attacks. External attackers have limited visibility into internal trust domain configurations and must operate through externally observable interfaces.

5.2 Malicious Integrator

A malicious integrator is an organization or entity that has been granted legitimate access to the enforcement gateway as an authorized integration partner, but that intentionally or negligently constructs requests in ways designed to obtain enforcement decisions beyond the scope of legitimate authorization. Capabilities include direct access to the request submission interface, knowledge of request schema, and the ability to craft syntactically valid but semantically manipulative intent declarations. The primary goal is to obtain permit decisions for actions that should be denied, through purpose misrepresentation or scope inflation in declared intent.

5.3 Model Operator Attempting Bypass

A model operator is a legitimate operator of an AI inference system that is subject to QODIQA enforcement constraints. The bypass goal is to route inference requests such that they avoid traversal of the enforcement gateway, obtaining the benefits of AI inference without the costs of enforcement compliance. Capabilities include control over the infrastructure topology in which the inference system is deployed, the ability to modify routing configurations, and knowledge of the expected enforcement behavior for specific request types. The primary risk vector is architectural: unauthorized direct routing to inference endpoints, circumventing gateway traversal requirements.

5.4 Compromised Infrastructure Provider

A compromised infrastructure provider is a component or service within the QODIQA enforcement boundary that has been subjected to supply chain compromise, insider attack, or external breach. This actor class differs from the others in that the threat originates from inside the trust boundary. Capabilities may include read or write access to the consent registry, access to signing key material, modification of policy evaluation logic, or suppression of audit records. The primary goal is to enable silent enforcement bypass or to introduce deniability into enforcement decisions. This actor class presents the highest severity risk profile because detection through audit logs may be compromised simultaneously with the enforcement integrity breach.

5.5 Governance Abuse Actor

A governance abuse actor targets the certification and conformance infrastructure that underlies trust in QODIQA-conformant systems. This actor may be a fraudulent certification applicant, a compromised auditor, a certification body subject to regulatory or economic capture, or an actor manipulating the supply chain of conformance test tooling. The primary goal is to obtain or confer certifications that falsely represent enforcement capability, creating systemic trust failures in the broader deployment ecosystem. Governance abuse does not directly bypass enforcement in any specific deployment but undermines the governance trust layer that enables relying parties to make deployment decisions based on certification claims.

SECTION 6

Attack Surface Enumeration

The following attack surfaces are identified across the enforcement boundary. Each surface is mapped to relevant STRIDE threat categories and to the actor classes most likely to exploit them.

6.1 Inference Request Path

The inference request path spans from the initial request submission by an integrator to the enforcement gateway, through gateway processing, and to the final permit or deny decision. Attack surfaces include the request submission interface, the declared intent structure embedded in the request, and the protocol used for transport. Relevant STRIDE categories: Spoofing (impersonation of authorized requestors), Tampering (modification of declared intent), Repudiation (ambiguous requestor identity), and Elevation of Privilege (scope inflation in declared purpose). This surface is exploitable by external attackers and malicious integrators.

6.2 Consent Verification Interface

The consent verification interface is the internal boundary crossing between the enforcement gateway and the consent registry. Attacks targeting this surface may inject false consent state, manipulate consent resolution results, or cause the verification to return incorrect validity signals for specific subjects or purposes. Relevant STRIDE categories: Tampering (manipulation of registry responses), Spoofing (impersonation of registry), Denial of Service (registry unavailability forcing fail behavior). This surface is exploitable by compromised infrastructure providers and by external attackers who achieve lateral movement into the internal trust domain.

6.3 Cryptographic Key Infrastructure

The cryptographic key infrastructure encompasses the key generation, storage, distribution, and rotation mechanisms used by the QODIQA enforcement components. Compromise of signing key material is the highest-severity attack surface in the architecture, as it enables the creation of forged consent artifacts and decision tokens that are indistinguishable from legitimate ones. Relevant STRIDE categories: Spoofing (use of compromised keys to impersonate legitimate signers), Elevation of Privilege (forged decision tokens). Exploitable primarily by compromised infrastructure providers and advanced external attackers.

6.4 Policy Evaluation System

The policy evaluation system processes declared intent and resolved consent state against active policy versions to produce enforcement decisions. Attack surfaces include the policy loading mechanism, the policy version reference integrity, the evaluation context construction, and the decision output path. Relevant STRIDE categories: Tampering (policy content modification), Repudiation (unclear policy version at time of evaluation), Elevation of Privilege (policy configuration enabling decisions that should be denied). Exploitable by malicious integrators with knowledge of policy logic and by compromised infrastructure providers.

6.5 Audit Logging Infrastructure

The audit logging infrastructure records enforcement decisions and serves as the primary mechanism for post hoc verification of enforcement claims. Attacks targeting the audit log aim to suppress records, retroactively modify records, or introduce false records that create misleading audit trails. Relevant STRIDE categories: Tampering (record modification), Repudiation (record suppression enabling deniability), Information Disclosure (audit log content exposure). Exploitable by compromised infrastructure providers; suppression attacks are of particular concern in scenarios where enforcement bypass is occurring simultaneously.

6.6 Governance and Certification Processes

The governance and certification processes that enable relying parties to trust QODIQA-conformant deployments constitute an attack surface at the ecosystem level. Attacks include fraudulent certification applications, manipulation of conformance test tooling to produce false pass results, and capture of certification bodies through economic or regulatory pressure. Relevant STRIDE categories: Spoofing (fraudulent certification claims), Tampering (manipulation of conformance test tooling). Exploitable by governance abuse actors.

SECTION 7

Adversarial Objective Model

Adversarial objectives are decomposed into goal hierarchies that map top-level outcomes to the enabling sub-goals required for their achievement. This structure enables the identification of intermediate attack steps that may be independently detectable or preventable.

OBJECTIVE CLASS A: ENFORCEMENT BYPASS

- Bypass consent verification: obtain a permit decision for a request that would not receive one under correct enforcement.
- Replay authorization artifacts: reuse a consent artifact valid for one context, purpose, or time window in a different context where consent has not been granted.

OBJECTIVE CLASS B: AUTHORIZATION FORGERY

- Forge authorization signals: create consent artifacts or decision tokens whose cryptographic structure is valid but whose semantic content is unauthorized.
- Inject fabricated decision tokens: introduce forged permit tokens into the processing pipeline without a corresponding enforcement decision having occurred.

OBJECTIVE CLASS C: ENFORCEMENT INTEGRITY DISRUPTION

- Corrupt enforcement logs: prevent accurate post hoc reconstruction of enforcement decisions or introduce false records that attribute compliance where violations occurred.
- Disable enforcement gateway: cause the enforcement gateway to become unavailable or to fail in a non-closed manner, permitting inference execution without enforcement.
- Manipulate policy state: modify active policy versions or load unauthorized policy configurations that expand the set of permitted decisions.

> OBJECTIVE CLASS D: IDENTITY MANIPULATION
>
> - Compromise identity binding: sever the binding between declared requestor identity and the cryptographic identity verified during enforcement, enabling impersonation.

Each primary objective requires one or more enabling sub-goals. For example, bypassing consent verification may require compromising the consent registry query path, forging a consent artifact that passes verification, or preventing the gateway from performing verification entirely. The sub-goal structure informs the defense-in-depth controls that are most effective at disrupting attack chains before a primary objective is achieved.

Adversarial success requires satisfying enabling sub-goals in a sequence that does not produce detectable artifacts prior to the primary objective being achieved. The architecture is designed such that enabling sub-goals for high-severity objectives are individually detectable through audit log analysis. The residual risk in this model is that audit log integrity may be compromised simultaneously with enforcement integrity.

Economic Attack Considerations

Attacks against deterministic consent enforcement are not exclusively motivated by technical objectives; they are frequently motivated by economic incentives that make non-compliant access commercially rational. Specific economic motivations include unauthorized inference access that avoids licensing, service, or compliance costs; circumvention of consent restrictions that would otherwise prevent access to commercially valuable data; and extraction of data for secondary commercial use in contexts where consent has not been granted for that purpose. These motivations are structurally relevant because they affect the resources adversaries are willing to expend on enforcement bypass and the persistence with which bypass attempts are likely to be repeated following detection and remediation.

The QODIQA enforcement architecture is designed to increase the operational cost of bypassing enforcement controls, creating economic asymmetry between compliant and non-compliant behavior. By requiring cryptographically grounded consent verification for each enforcement decision, the architecture raises the marginal cost of unauthorized access

above the level achievable through configuration manipulation alone. While the threat model focuses on technical mechanisms, economic motivations are considered part of the adversarial landscape and should inform deployment-specific risk prioritization, particularly in commercial environments where data access carries direct monetary value.

SECTION 8

Runtime Bypass Scenarios

The following misuse cases describe concrete runtime bypass scenarios against the enforcement architecture. Each case is analyzed along five dimensions: attack description, preconditions required, exploit mechanism, detection feasibility, and enforcement response under correct operation.

8.1 Direct Inference Endpoint Bypass

> ATTACK DESCRIPTION
>
> An adversary routes inference requests directly to the AI inference endpoint, circumventing the enforcement gateway entirely. No consent verification or policy evaluation occurs.

Preconditions. The adversary must have network-level or application-level access to the inference endpoint address that is not mediated by the enforcement gateway. This requires either a deployment misconfiguration that exposes the inference endpoint directly, or lateral movement within the deployment environment that grants access to internal routing.

Exploit Mechanism. The adversary constructs valid inference requests and submits them directly to the inference endpoint using the native API of the inference system, bypassing all QODIQA enforcement components.

Detection Feasibility. Low under this bypass scenario. Because the enforcement gateway is not traversed, no enforcement records are generated. Detection requires cross-correlation of inference system access logs with

enforcement gateway logs to identify inference activity not accompanied by a corresponding enforcement decision record.

Enforcement Response. Under correct deployment, the inference endpoint is not accessible without traversal of the enforcement gateway. Correct deployment is a deployment-layer control, not an architectural enforcement guarantee. The QODIQA architecture does not and cannot prevent direct endpoint access if the deployment topology exposes it.

8.2 Forged Authorization Token Injection

> ATTACK DESCRIPTION
>
> An adversary injects a forged enforcement decision token into the request processing pipeline, causing the downstream inference system to treat the request as having received a valid permit decision without actual enforcement having occurred.

Preconditions. The adversary must have access to a valid signing key or must be able to exploit a vulnerability in the decision token verification path that allows a token with an invalid or missing signature to be accepted as valid.

Exploit Mechanism. The adversary constructs a syntactically valid decision token containing a permit decision and injects it into the request pipeline at a point downstream of the enforcement gateway. The token's cryptographic validity is not re-verified, or the verification is bypassed.

Detection Feasibility. Moderate. If audit logs are correctly maintained, the absence of an enforcement record for a specific request, paired with evidence of inference execution, indicates a potential token injection event. However, if the injected token includes fabricated log entries, detection requires cryptographic verification of audit record integrity.

Enforcement Response. The enforcement architecture requires that decision tokens be cryptographically signed by the enforcement gateway with a key that is not accessible to components outside the trust boundary. Downstream systems

must verify this signature before treating a token as valid. Correct implementation of this verification provides strong resistance to token injection attacks that do not involve key compromise.

8.3 Enforcement Gateway Removal

> ATTACK DESCRIPTION
>
> An adversary with infrastructure access removes or disables the enforcement gateway component from the deployment, either transiently or permanently, causing inference requests to be processed without enforcement.

Preconditions. The adversary must have administrative access to the deployment infrastructure, or must be able to exploit a vulnerability in the gateway process that causes it to crash or become unavailable.

Exploit Mechanism. The gateway is halted, misconfigured to pass all requests, or replaced with a non-enforcing proxy that forwards requests without consent verification or policy evaluation.

Detection Feasibility. Moderate to high. Gateway availability monitoring provides detection for outage events. Replacement with a non-enforcing proxy may be detectable through audit log analysis if record formats or signing keys differ from canonical gateway behavior.

Enforcement Response. The enforcement architecture does not specify availability controls for the gateway itself; those are deployment concerns. The architectural response to gateway unavailability is fail-closed behavior: inference endpoints must refuse requests in the absence of a valid, current enforcement decision token. Correct implementation of fail-closed semantics limits the impact of gateway removal to service denial, not enforcement bypass.

8.4 Replay Attack on Consent Artifacts

> **ATTACK DESCRIPTION**
>
> An adversary replays a consent artifact that was valid for a prior request, with a different purpose, time window, or data scope, to obtain a permit decision in a context where consent has not been granted or has been revoked.

Preconditions. The adversary must have access to a previously captured consent artifact and must be able to submit it to the enforcement gateway in a context where the artifact's cryptographic validity would be confirmed but its contextual validity has not been verified.

Exploit Mechanism. The adversary submits the captured consent artifact with a new request whose declared intent differs from the purpose for which the artifact was originally granted. If the enforcement gateway verifies the artifact's cryptographic signature but does not verify purpose binding, temporal validity, or request-level contextual hash, the replay succeeds.

Detection Feasibility. Moderate. Replay attacks generate enforcement records, but the record may appear valid to a reviewer examining only the artifact signature. Detection requires cross-referencing the purpose binding in the artifact against the declared intent in the request, which requires semantic analysis of enforcement records beyond signature verification.

Enforcement Response. The QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement requires that consent artifacts bind to the specific context, purpose, and time window of the granted consent, and that enforcement decisions verify this binding explicitly. Correct implementation of context-bound artifact verification provides direct mitigation for replay attacks.

SECTION 9

Cryptographic Compromise Analysis

Cryptographic compromise represents the highest-severity threat class in the QODIQA threat model, because the enforcement architecture's authenticity and integrity guarantees are grounded in cryptographic properties. Failure of these properties may be silent, may affect all enforcement decisions system-wide, and may be irreversible without full re-keying.

9.1 Signing Key Compromise

Compromise of the private signing key used by the enforcement gateway to sign decision tokens enables an adversary to generate forged tokens that are cryptographically indistinguishable from legitimate ones. The impact is system-wide: all relying components that verify decision token signatures using the corresponding public key will accept forged decisions as valid. Mitigation requires key rotation, revocation of the compromised key, and forensic reconstruction of enforcement decisions made during the compromise window.

9.2 Consent Artifact Signature Forgery

Forgery of consent artifact signatures requires compromise of the signing key material used by the identity infrastructure to issue consent artifacts. Successful forgery enables the creation of consent artifacts for any subject, purpose, and data scope, effectively nullifying all consent constraints. The severity is Critical. The enforcement boundary assumes the integrity of the identity infrastructure; forgery that compromises this assumption requires evaluation outside the scope of this threat model as a dependent trust anchor failure.

9.3 Compromised Identity Providers

If the identity provider responsible for authenticating requestors and binding identities to consent artifacts is compromised, an adversary can obtain authentic credentials for arbitrary identities and use them to request consent artifacts that appear legitimately granted. This threat class is particularly significant for multi-tenant deployments where a shared identity provider serves multiple data subjects. Compromise does not require forging cryptographic signatures; it exploits the legitimate issuance path.

9.4 Replay of Signed Consent Artifacts

As described in Section 8.4, replay of previously valid signed artifacts represents a cryptographic-layer attack vector. The cryptographic signature on a replayed artifact may be valid, but the semantic binding to the current request context is not. This distinction is critical: replay attacks succeed in implementations that verify cryptographic authenticity without separately verifying contextual binding. The QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement requirement for contextual hashing is the primary architectural control for this scenario.

Across all cryptographic compromise scenarios, the impact on enforcement guarantees follows a common pattern: the architectural properties of determinism and verifiability are undermined because the inputs to enforcement decisions, consent artifacts and identity assertions, can no longer be trusted to represent their stated semantics. Recovery requires establishing a new trusted baseline, which may involve re-issuing all consent artifacts issued by the compromised infrastructure.

SECTION 10

Governance and Certification Abuse

Governance abuse threats operate at the ecosystem level and represent a category of attacks that do not directly bypass enforcement in any specific deployment, but that undermine the governance trust layer on which relying parties depend when making deployment and integration decisions.

10.1 Fraudulent Certification Claims

An organization may claim QODIQA conformance without having undergone valid certification processes. Fraudulent certification claims mislead relying parties into trusting enforcement guarantees that are not present. Detection requires that relying parties verify certification claims against records maintained by the governing authority. The QODIQA: Governance Charter for the QODIQA Standard Corpus and the QODIQA: Conformance Test Suite Specification define the certification process; fraudulent claims exploit the absence of a universally accessible certification verification mechanism.

10.2 Manipulated Conformance Tests

Conformance test tooling used to evaluate implementations against the QODIQA: Conformance Test Suite Specification may itself be subject to supply chain manipulation. If test tooling is compromised to produce false pass results for implementations that do not conform to specification, certifications issued on the basis of those results are invalid. Mitigation requires cryptographic signing of conformance test tooling by the governing authority, verification of tooling integrity before use, and independent audit of certification decisions.

10.3 Certification Body Capture

The certification body responsible for issuing conformance certifications may be subject to regulatory capture, economic pressure, or direct compromise. Captured certification bodies may issue certifications to non-conformant implementations, reduce the rigor of audit requirements, or extend certifications without revalidation. Detection requires independent oversight of certification body processes and periodic meta-audit of certification decisions against implementation evidence. This threat class is addressed in the QODIQA: Governance Charter for the QODIQA Standard Corpus through provisions for governing authority oversight and audit record maintenance.

10.4 Supply Chain Manipulation

Supply chain manipulation affects the integrity of enforcement components themselves: enforcement gateways, consent registry implementations, cryptographic libraries, or policy evaluation engines obtained from third-party sources may contain backdoors, weakened cryptography, or logic that enables adversarial outcomes. Mitigation requires verification of software supply chain provenance, cryptographic integrity verification of component artifacts, and conformance testing of third-party implementations against the QODIQA: Conformance Test Suite Specification before deployment.

SECTION 11

Integration and Dependency Failure

Integration failures are distinct from adversarial attacks in that they arise from operational or environmental conditions rather than intentional action. However, their impact on enforcement guarantees may be equivalent to that of targeted attacks, and they must be addressed in the threat model to characterize the full residual risk profile.

11.1 Consent Registry Outage

Unavailability of the consent registry prevents the enforcement gateway from resolving consent state for incoming requests. The expected architectural response is fail-closed behavior: requests for which consent state cannot be resolved must be denied. Fail-closed semantics ensure that registry outages degrade service availability without compromising enforcement integrity. Residual risk exists if fail-closed behavior is not correctly implemented; in that case, registry outages create a denial-of-service-equivalent enforcement bypass window.

11.2 Misconfigured Enforcement Gateways

Misconfigured gateways represent one of the most common operational failure modes. Configurations that disable cryptographic verification, suppress audit logging, or alter fail-closed behavior from the default may introduce enforcement gaps that are indistinguishable from deliberate bypass. Operational controls must include configuration auditing against a canonical baseline, automated detection of configuration drift, and immutable configuration management practices that prevent unauthorized modification.

11.3 Network Partitioning

Network partitioning that isolates the enforcement gateway from the consent registry or from the policy evaluation system creates a dependency failure equivalent to component outage. The fail-closed requirement applies equally to network-induced unavailability. In multi-region or multi-cloud deployments, partitioning scenarios must be explicitly tested and enforced during conformance evaluation.

11.4 Dependency Compromise

Dependencies of the enforcement gateway, TLS libraries, operating system components, container runtimes, or secret management systems, may be compromised through supply chain attacks or zero-day exploitation. Compromise of a dependency that handles cryptographic operations or controls gateway process execution may provide an adversary

with capabilities equivalent to those of a compromised infrastructure provider. Defense requires dependency inventory management, vulnerability monitoring, and rapid patching processes that operate with awareness of enforcement impact.

SECTION 12

Invariant Violation Analysis

The QODIQA enforcement architecture defines a set of core invariants: conditions that must hold for all enforcement decisions for the architectural guarantees to be valid. Invariant violations represent the most fundamental failure mode of the system and require explicit analysis of both causes and consequences.

12.1 Core Invariants

INVARIANT I-1: NO EXECUTION WITHOUT VERIFIED CONSENT

An AI inference request must not proceed to execution unless the enforcement gateway has issued a valid permit decision based on verified consent state for the specific request context. *Violation:* any inference execution that occurs without a corresponding permit decision from the enforcement gateway constitutes a direct bypass of the enforcement architecture, enabling unauthorized data processing or action execution.

INVARIANT I-2: CRYPTOGRAPHIC VERIFIABILITY OF CONSENT ARTIFACTS

All consent artifacts evaluated by the enforcement gateway must carry a cryptographic signature whose chain of trust is traceable to a recognized identity anchor, and that signature must be verified before any enforcement decision is issued. *Violation:* acceptance of unsigned or unverified consent artifacts allows forged or replayed artifacts to produce valid permit decisions, nullifying the authenticity guarantees of the enforcement model.

INVARIANT I-3: PRE-EXECUTION ENFORCEMENT DECISION

The enforcement gateway must produce a permit or deny decision before the inference request is forwarded to the AI system. No mechanism exists within the architecture for retroactive enforcement of decisions after execution. *Violation:* post hoc enforcement converts the architecture from a pre-execution control to an audit mechanism, eliminating its deterministic enforcement property and allowing unauthorized execution to occur before any governance response is possible.

> **INVARIANT I-4: POLICY VERSION RECOVERABILITY**
>
> The audit record for every enforcement decision must contain sufficient information to identify the policy version applied, enabling the decision to be reconstructed independently. *Violation:* absence of policy version in audit records prevents independent reproduction of enforcement decisions, rendering audit claims non-verifiable and eliminating the architecture's proof-of-compliance capability.

> **INVARIANT I-5: AUDIT RECORD COMMITMENT BEFORE PERMIT ISSUANCE**
>
> The audit record for an enforcement decision must be committed before the corresponding permit is issued. No enforcement decision may be issued speculatively or without a corresponding audit record. *Violation:* speculative permit issuance without committed audit records enables enforcement bypass that leaves no audit trail, eliminating the ability to detect, investigate, or prove enforcement failures after the fact.

12.2 Consequences of Invariant Violation

Violation of Invariant I-1 directly enables unauthorized inference execution. Violation of I-2 enables the acceptance of forged consent artifacts. Violation of I-3 converts QODIQA from a pre-execution enforcement layer to a post hoc audit mechanism, eliminating its deterministic enforcement property. Violation of I-4 prevents independent reproduction of enforcement decisions and renders audit claims non-verifiable. Violation of I-5 enables enforcement bypass that leaves no audit trail, eliminating the ability to detect or investigate enforcement failures.

Invariant violations may be caused by implementation defects, adversarial manipulation, or operational failures. The severity classification for any vulnerability or failure mode that causes an invariant violation is at minimum High, and Critical where the violation is undetectable or systemic.

Mitigation Mapping

The QODIQA architecture addresses identified threat categories through specific enforcement mechanisms and architectural components rather than through abstract policy declarations. The purpose of this mapping is to

demonstrate how the architecture operationalizes threat containment, connecting each threat class to the concrete system element responsible for its mitigation. Where multiple components participate in containment, the primary accountable element is identified. This mapping is intended to support conformance evaluators and deployment architects in verifying that each identified threat class has a corresponding architectural control with a defined locus of enforcement responsibility.

TABLE: MITIGATION MAPPING BY THREAT CATEGORY

THREAT CATEGORY	MITIGATION MECHANISM	QODIQA COMPONENT
Consent Artifact Tampering	Cryptographic signatures and artifact integrity verification	Artifact Generator
Runtime Policy Bypass	Deterministic enforcement at the inference boundary	Enforcement Gateway
Consent Registry Manipulation	Append-only registry integrity guarantees	Consent Registry
Unauthorized Inference Execution	Mandatory consent verification before inference execution	Enforcement Gateway
Certification or Governance Abuse	Independent conformance testing and certification requirements	QODIQA Certification Framework
Supply Chain or Integration Risk	Explicit trust boundaries and verifiable artifact generation	Reference Architecture

SECTION 13

Residual Risk Summary

Residual risks are threat conditions that persist under correct implementation of the QODIQA architecture. These risks are not mitigated by architectural controls alone and require operational, governance, or external controls to address.

13.1 Insider Abuse

Administrators with legitimate access to enforcement infrastructure retain the capability to modify configurations, access key material, or suppress audit records through authorized administrative channels. Insider threat is a residual risk in any architecture that requires human administration. Detection depends on administrative access audit logs maintained by a system that is itself independent of the enforcement infrastructure. Mitigation requires separation of duties in administrative roles, multi-party authorization for sensitive operations, and independent audit oversight.

13.2 Compromised Governance Infrastructure

Systematic compromise of the governance layer, including the certification authority, the governing charter administration, and the conformance test infrastructure, undermines the ecosystem-level trust guarantees that QODIQA provides to relying parties. This risk persists regardless of the technical security properties of any individual deployment. The absence of an independently verifiable, globally accessible registry of valid certifications is a structural residual risk in the current governance model.

13.3 Economic Coercion

Operators subject to QODIQA enforcement constraints may be subject to economic pressure from clients, regulators, or market conditions that create incentives for bypass. Economic coercion does not alter the technical enforcement properties of the architecture, but it may motivate deliberate deployment misconfiguration, selective enforcement gateway disablement, or fraudulent certification claims. This risk is addressed at the governance level through audit requirements and certification revocation provisions, but it is not eliminable through technical controls.

13.4 Large-Scale Key Compromise

Cryptanalytic advances, implementation vulnerabilities in widely deployed cryptographic libraries, or supply chain attacks on hardware security modules could compromise key material at scale across multiple deployments

simultaneously. This scenario is outside the scope of deployment-level mitigations and requires protocol-level responses including algorithm agility provisions and pre-planned migration paths to post-compromise cryptographic configurations. The QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement addresses algorithm agility requirements, but the residual risk of coordinated large-scale key compromise cannot be eliminated.

13.5 Consent State Accuracy Limitations

The enforcement architecture enforces consent as represented in the registry. It does not verify that the consent state in the registry accurately reflects the intentions of the data subjects from whose decisions it was derived. Consent collection failures, misrepresentation at the collection interface, or registry synchronization lag create conditions where enforcement is technically correct but semantically misaligned with actual consent. This residual risk is outside the enforcement boundary as defined in this document and is further documented in the QODIQA: Residual Risk and Assumption Disclosure Annex.

SECTION 14

Threat Coverage Matrix

The following matrix maps threat categories to specific attack scenarios, responsible threat actors, primary mitigation mechanisms, and residual risk under correct implementation.

TABLE 2: THREAT COVERAGE MATRIX

THREAT CATEGORY	ATTACK SCENARIO	THREAT ACTOR	SEVERITY	MITIGATION MECHANISM	RESIDUAL RISK
Enforcement Bypass	Direct inference endpoint bypass via routing circumvention	Model Operator; External Attacker	**Critical**	Deployment topology enforcement; network segmentation	Persists under misconfigured deployments

THREAT CATEGORY	ATTACK SCENARIO	THREAT ACTOR	SEVERITY	MITIGATION MECHANISM	RESIDUAL RISK
Token Forgery	Forged decision token injection using compromised signing key	Compromised Infrastructure Provider	**Critical**	Key isolation in HSM; signature verification at inference endpoint	Persists under key compromise scenarios
Gateway Disablement	Enforcement gateway removal or replacement with non-enforcing proxy	Model Operator; Compromised Infrastructure Provider	**Critical**	Fail-closed semantics at inference endpoint; availability monitoring	Limited to availability degradation under correct fail-closed implementation
Replay Attack	Replay of valid consent artifact for different purpose or time window	External Attacker; Malicious Integrator	**High**	Context-bound artifact structure; contextual hash verification	Low under correct binding verification implementation
Identity Spoofing	Impersonation of authorized requestor through credential theft	External Attacker; Malicious Integrator	**High**	Cryptographic requestor authentication; short-lived credential validity	Persists under compromised identity provider scenarios
Policy Tampering	Unauthorized modification of active policy versions	Compromised Infrastructure Provider; Insider	**High**	Policy version immutability; policy content signing	Low under correct policy management controls
Audit Log Suppression	Suppression or modification of enforcement decision records	Compromised Infrastructure Provider; Insider	**High**	Tamper-evident log structure; write-only access enforcement	Persists under insider or infrastructure compromise
Consent State Injection	Manipulation of consent registry query responses	Compromised Infrastructure Provider	**High**	Authenticated registry interface; response integrity signing	Persists under registry infrastructure compromise
Cryptographic Compromise	Signing key compromise enabling forged consent artifacts	Compromised Infrastructure Provider; External Attacker	**Critical**	HSM-based key storage; key rotation and revocation protocols	Persists; requires key revocation and artifact re-issuance
Fraudulent Certification	Certification claims asserted for non-conformant implementations	Governance Abuse Actor	**High**	Governance Charter oversight; certification registry maintenance	Persists under governance body capture scenarios
Supply Chain Attack	Compromised enforcement component containing backdoor logic	Compromised Infrastructure Provider; Governance Abuse Actor	**Critical**	Component integrity verification; conformance testing of third-party implementations	Persists; requires independent supply chain security program

THREAT CATEGORY	ATTACK SCENARIO	THREAT ACTOR	SEVERITY	MITIGATION MECHANISM	RESIDUAL RISK
Insider Abuse	Administrator-level configuration modification or signing key access	Insider (privileged administrator)	**High**	Separation of duties; administrative audit logging; multi-party authorization	Structural residual; requires governance controls outside architecture
Registry Outage	Consent registry unavailability during active enforcement	External Attacker (DoS); Operational Failure	**Medium**	Fail-closed enforcement behavior; registry redundancy	Service availability impact only under correct fail-closed implementation
Purpose Misrepresentation	Malicious integrator inflating declared purpose scope in request	Malicious Integrator	**Medium**	Purpose binding validation against granular consent; audit log review	Persists where consent granularity is coarse

SECTION 15

Formal Security Posture

The security posture of the QODIQA enforcement model is defined by the combination of architectural guarantees it provides, the operational dependencies on which those guarantees rest, and the limits of what the architecture can ensure independently of deployment context and governance infrastructure.

15.1 Architectural Guarantees

The QODIQA enforcement architecture provides the following guarantees under the assumptions stated in Section 3: all AI inference execution is preceded by a deterministic enforcement decision; enforcement decisions are cryptographically authenticated and auditably recorded; enforcement decisions are reproducible given the same inputs and policy version; and failure of any required enforcement dependency causes the system to deny rather than to permit inference execution.

These guarantees constitute a deterministic enforcement boundary. Within this boundary, the system's behavior is specified precisely and is not subject to probabilistic reasoning, contextual inference, or best-effort interpretation. The boundaries are structural, not statistical.

15.2 Operational Dependencies

The enforcement guarantees are contingent on: correct deployment configuration that ensures gateway traversal for all inference requests; correct implementation of cryptographic controls as specified in the QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement; operational integrity of the consent registry, identity infrastructure, and audit logging system; and governance practices that maintain the integrity of policy management and certification processes.

Violations of these operational dependencies do not indicate architectural defects. They represent conditions under which the architectural guarantees are no longer applicable because the prerequisites for those guarantees are not met. The enforcement boundary assumes that these prerequisites have been met; it does not enforce them.

15.3 Limits of Architectural Guarantees

The QODIQA enforcement architecture does not guarantee: the accuracy or completeness of consent state captured by external collection interfaces; the correctness or safety of AI model behavior within authorized execution boundaries; the accuracy of audit logs in the presence of insider attacks targeting logging infrastructure simultaneously with enforcement bypass; or the trustworthiness of certification claims in the absence of independent verification of certification body processes.

Residual risk remains where operational dependencies are not met, governance infrastructure is compromised, or threat actors possess capabilities that exceed those characterized in Section 5. These residual risks are documented in Section 13 and are acknowledged as conditions that the architecture is designed to make difficult but cannot eliminate.

FORMAL SECURITY POSTURE STATEMENT

The QODIQA enforcement architecture provides deterministic, cryptographically grounded, fail-closed enforcement of consent constraints for AI inference execution, within the system boundary defined in Section 2, under the operational assumptions stated in Section 3, subject to the residual risks enumerated in Section 13. It does not provide security guarantees outside the enforcement boundary, does not eliminate insider threat, and does not constitute a substitute for operational security practices, supply chain security programs, or governance oversight of certification processes.

This document presents a formal adversarial threat model and does not constitute legal, regulatory, or operational security advice. Organizations implementing QODIQA-conformant systems are required to conduct deployment-specific threat assessments that extend from the analytical baseline established here, accounting for deployment topology, threat actor capabilities, and operational context specific to their environment.

DOCUMENT STATUS

Document Status and Corpus Alignment

This document is part of the QODIQA specification corpus and provides the adversarial threat model and abuse case analysis for deterministic runtime consent enforcement architectures.

The document analyzes potential adversarial behaviors, attack surfaces, and bypass strategies relevant to consent enforcement at runtime. It defines the threat landscape associated with deterministic enforcement systems and identifies residual risks that remain under the architectural and cryptographic assumptions defined in the QODIQA standard corpus.

This document should be read together with the following related specifications:

- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement
- QODIQA: Residual Risk and Assumption Disclosure Annex

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dițescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-SEC-TM-2026-001
TITLE	Threat Model and Abuse Case Specification
SUBTITLE	Deterministic Runtime Consent Enforcement for Artificial Intelligence Systems
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Security Analysis Specification
DOCUMENT STATUS	Normative
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 10 PART II · NORMATIVE TECHNICAL SPECIFICATIONS

Extended Adversarial Threat Model

Formal enumeration of adversarial actors, attack surfaces, threat classes, and enforcement failure modes for systems implementing the framework.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-TM-2026-001	1.0	2026	Normative Technical	Normative	NORMATIVE

ABSTRACT

This document constitutes the Extended Adversarial Threat Model for systems implementing the QODIQA Deterministic Runtime Consent Enforcement framework. It provides a formal, structured enumeration of adversarial actors, attack surfaces, threat classes, bypass vectors, and enforcement failure conditions applicable to QODIQA-conformant deployments. The document is organized according to established threat modeling methodologies and is intended to inform security architecture decisions, conformance testing criteria, and risk acceptance processes for operators integrating QODIQA into AI-native execution environments. All threat descriptions are presented at a level of specificity sufficient to support defensive control design. The model addresses inference-time manipulation, consent bypass strategies, prompt injection, data exfiltration, side-channel leakage, supply chain compromise, insider threats, and adversarial automation. Residual and non-eliminable risks are explicitly identified. This document is normative within the QODIQA standard corpus and is subject to revision as the threat landscape evolves.

SECTION 1

Scope and Threat Modeling Boundaries

This threat model applies to any system that implements, integrates, or operationally depends upon the QODIQA Deterministic Runtime Consent Enforcement framework. The scope encompasses the enforcement engine, the consent and policy store, all interfaces through which requests are submitted and decisions are returned, and the audit subsystems that record enforcement outcomes.

The threat model does not extend to upstream AI model behavior, downstream application logic that consumes enforcement decisions, or regulatory compliance processes external to system execution. These domains are governed by separate specifications within the QODIQA corpus or by applicable jurisdictional frameworks.

> **MODELING BOUNDARY: FORMAL DEFINITION**
>
> The threat modeling boundary is defined as the set of components, interfaces, and data flows that participate in the evaluation cycle: request ingestion → consent resolution → policy evaluation → decision emission → audit record creation. Any component outside this cycle is considered out of scope for the purposes of this document.

The following are explicitly excluded from this threat model:

- Training-time attacks on AI models integrated downstream of the enforcement engine.
- Physical layer attacks on hardware infrastructure hosting QODIQA components.
- Social engineering directed at personnel responsible for policy authorship, unless such attacks result in automated exploitation of policy interfaces.
- Legal or regulatory non-compliance arising from policy content rather than enforcement mechanism failure.
- Denial-of-service attacks against network infrastructure, except where they specifically induce enforcement failure conditions relevant to QODIQA's fail-closed semantics.

The threat model is structured to support STRIDE-compatible analysis while extending that framework to address AI-specific threat classes not adequately represented in general-purpose security models. Threat entries are presented at a level of abstraction suitable for control design without prescribing specific implementation countermeasures, which are addressed in the QODIQA Security and Cryptographic Profile.

SECTION 2

System Context and Trust Boundaries

A QODIQA-conformant system operates as an enforcement intermediary between AI-native clients and the actions or data those clients request access to. The system's security posture depends on the integrity of three distinct trust domains, each of which presents a distinct attack surface.

2.1 Trust Domain: Request Originators

Request originators are AI agents, orchestration layers, human-operated interfaces, or automated pipelines that submit consent evaluation requests to the enforcement engine. These actors are considered partially trusted by default. They are authenticated but not assumed to be non-adversarial. Request contents are treated as potentially manipulated inputs at all times.

2.2 Trust Domain: Policy and Consent Store

The policy and consent store is the authoritative source for consent records and policy definitions. It is treated as a high-trust component subject to strict integrity requirements. Unauthorized modification of store contents constitutes a critical compromise of the enforcement system. The store's trust level requires cryptographic integrity guarantees and access control mechanisms that are specified in the QODIQA Security and Cryptographic Profile.

2.3 Trust Domain: Enforcement Engine

The enforcement engine is the highest-trust component in the system. It is responsible for deterministic evaluation and must not be subject to runtime modification, configuration drift, or interface exposure beyond what is minimally required. The engine's trust boundary is enforced through isolation, immutability controls, and a minimal external interface surface.

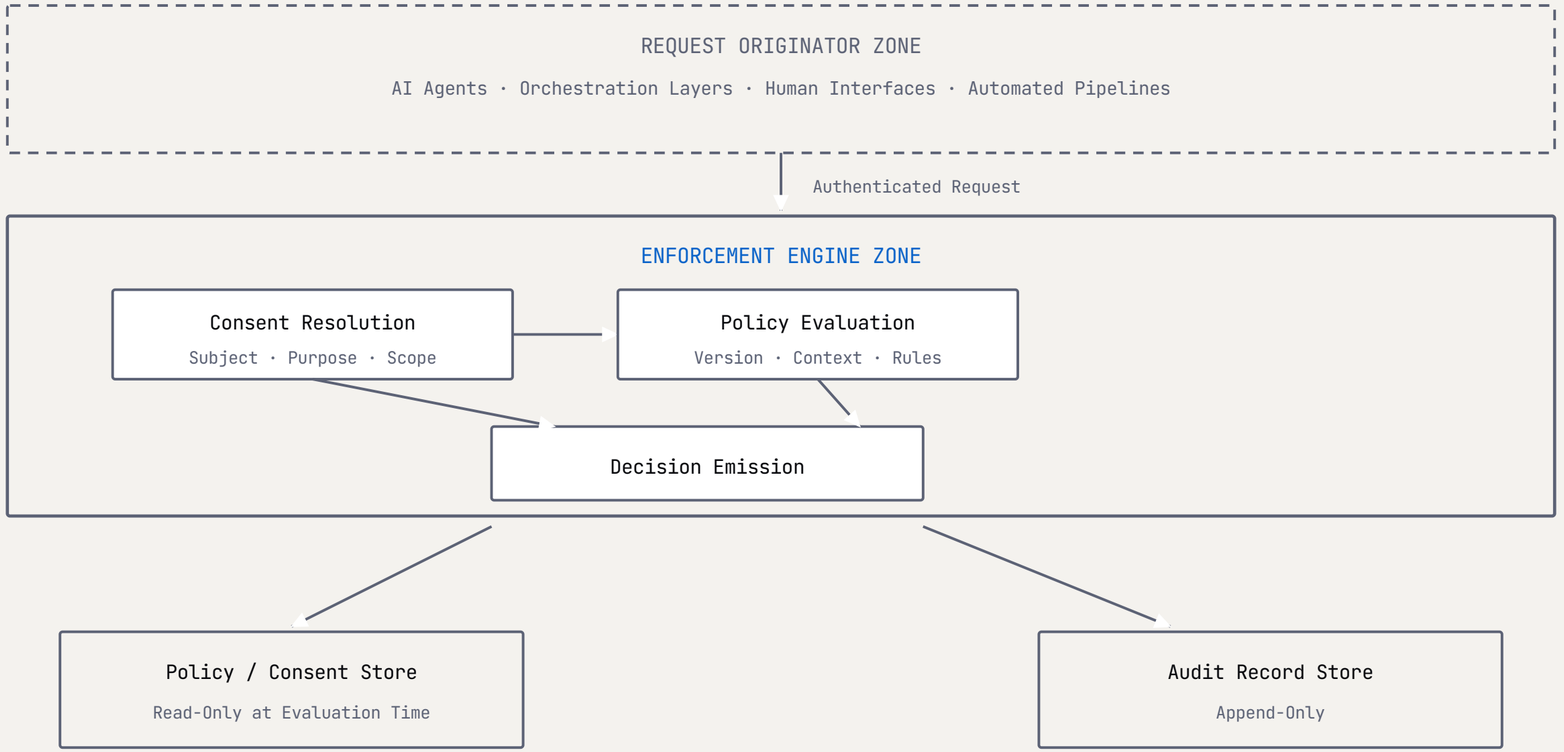


FIGURE 1. QODIQA SYSTEM TRUST DOMAINS AND PRIMARY DATA FLOWS. TRUST DECREASES FROM THE ENFORCEMENT ENGINE OUTWARD. ALL INTERFACES CROSSING TRUST BOUNDARIES CONSTITUTE PRIMARY ATTACK SURFACES.

2.4 Trust Domain: Audit and Logging Subsystem

The audit subsystem records enforcement decisions as immutable artifacts. Its integrity is essential for forensic and accountability functions. The subsystem is considered trusted for write operations originating from the enforcement engine, and untrusted for all other write operations. Read access is governed by separate access control policies outside the scope of this document.

SECTION 3

Adversary Classification Framework

This section defines the classification framework applied to adversaries considered within this threat model. The framework organizes adversaries along orthogonal axes of capability, access level, and intent, enabling systematic mapping of adversary classes to specific threat vectors.

3.1 Classification Axes

Adversaries are classified across three primary axes:

- **Capability:** The technical sophistication and resource endowment of the adversary, ranging from opportunistic actors with commodity tooling to nation-state or hyperscaler-level actors with custom hardware, insider access, and sustained operational capacity.
- **Access Level:** The degree of access the adversary holds relative to system components, ranging from external network-only access through authenticated API access, privileged operator access, and physical or hypervisor-level access.

- **Intent:** The adversary's operational objective, which may be consent bypass for unauthorized execution, exfiltration of protected data, disruption of enforcement availability, audit record manipulation, or long-term persistent access establishment.

3.2 Adversary Tiers

TABLE 3.2-A: ADVERSARY TIER DEFINITIONS

TIER	DESIGNATION	CAPABILITY LEVEL	TYPICAL ACCESS	PRIMARY OBJECTIVES
T1	Opportunistic External	Low: commodity tools, public techniques	Unauthenticated API, public interface	Consent bypass, unauthorized execution
T2	Authenticated Abuser	Medium: domain knowledge, scripted attacks	Authenticated API, limited operator role	Scope escalation, purpose misrepresentation
T3	Privileged Insider	High: system knowledge, elevated access	Operator console, policy store write access	Policy manipulation, audit suppression
T4	Sophisticated External	High: custom tooling, zero-day capability	External only, escalation via vulnerability	Engine compromise, cryptographic integrity attacks
T5	Nation-State / Supply Chain	Very high: persistent, multi-vector	Supply chain, third-party dependency, hardware	Long-term persistence, mass consent bypass

The adversary tiers defined above are not mutually exclusive in operational scenarios. A coordinated attack may involve T1 actors as a distraction vector while T3 or T5 actors pursue primary objectives. The threat model accounts for compound adversary scenarios in Sections 13 and 14.

SECTION 4

Attack Surface Enumeration

The attack surface of a QODIQA-conformant system is defined as the union of all interfaces, data paths, and components accessible to adversarial actors. This section enumerates primary attack surface elements and their associated exposure characteristics.

4.1 Input Interfaces

The enforcement engine exposes a request submission interface through which callers submit consent evaluation requests. This interface accepts structured request objects containing declared actor identity, action type, data scope identifiers, purpose declarations, and temporal parameters. Each field of the request object constitutes a potential manipulation target. Adversaries may supply false actor identities, misrepresent action types, artificially narrow or expand scope identifiers, fabricate purpose declarations, or manipulate temporal parameters to intersect with valid consent windows.

The interface itself represents a surface for protocol-level attacks including malformed request injection, schema boundary violations, and encoding-layer manipulation designed to induce parsing ambiguity in the evaluation engine.

4.2 Policy and Consent Store

The policy and consent store is the most sensitive data asset in the system. Attack vectors targeting this component include unauthorized write operations that modify active consent records, version rollback attacks that substitute current policy versions with previously valid but superseded versions, and store corruption that induces indeterminate evaluation states. Read-path attacks may target consent record confidentiality, exposing information about which actors hold consent for which purposes and data classes, constituting a structural information disclosure.

4.3 Enforcement Engine

The enforcement engine is the highest-value target in the system. Compromise of the engine invalidates all downstream security guarantees regardless of the integrity of other components. Engine attack vectors include exploitation of software vulnerabilities in evaluation logic, configuration interface abuse enabling runtime parameter modification, dependency compromise through libraries or runtime environments used by the engine, and resource exhaustion attacks designed to degrade evaluation latency below fail-closed thresholds.

4.4 Audit and Logging Subsystem

The audit subsystem is a target for adversaries seeking to eliminate forensic evidence of unauthorized activity. Attack vectors include log deletion or truncation, log injection designed to pollute or obscure genuine enforcement records, timestamp manipulation, and selective suppression of specific decision records. An adversary with write access to the audit subsystem can retroactively eliminate evidence of consent bypass events, undermining the accountability guarantees that the audit function is designed to provide.

SECTION 5

Threat Taxonomy

This section defines the primary threat categories applicable to QODIQA-conformant systems. The taxonomy extends the STRIDE framework with AI-specific threat classes. Each category is defined by a formal description, the assets at risk, and the enforcement properties violated.

TABLE 5-A: PRIMARY THREAT CATEGORY TAXONOMY

ID	CATEGORY	DEFINITION	ASSETS AT RISK	PROPERTIES VIOLATED
TC-01	Consent Bypass	Execution of an AI action in the absence of a valid, resolved consent state for the declared actor, purpose, and data scope.	Consent store, enforcement engine	Authorization, determinism
TC-02	Scope Escalation	Expansion of the effective data or action scope beyond what is authorized by the resolved consent record, achieved through request manipulation or policy ambiguity.	Consent records, policy definitions	Scope binding, purpose limitation
TC-03	Policy Corruption	Unauthorized modification of policy definitions or consent records resulting in altered evaluation outcomes for subsequent requests.	Policy store, consent store	Integrity, reproducibility
TC-04	Audit Suppression	Deletion, modification, or prevention of audit record creation for enforcement decisions, eliminating forensic accountability.	Audit subsystem	Auditability, non-repudiation
TC-05	Inference Manipulation	Exploitation of AI model behavior at inference time to produce outputs that circumvent downstream consent constraints.	AI model outputs, enforcement decisions	Determinism, output integrity
TC-06	Information Disclosure	Unauthorized extraction of consent state, policy structure, or enforcement decision contents through direct or side-channel means.	Consent records, policy definitions, decision artifacts	Confidentiality, privacy
TC-07	Availability Degradation	Reduction of enforcement engine availability below operational thresholds, potentially inducing fail-open behavior in non-conformant deployments.	Enforcement engine, availability guarantees	Fail-closed semantics, availability
TC-08	Supply Chain Compromise	Introduction of malicious logic through dependencies, build pipeline components, or third-party integrations that alter enforcement behavior at runtime.	Enforcement engine, all components	Integrity, trust chain

SECTION 6

Adversarial Objectives and Capabilities

This section maps adversary tiers to their primary operational objectives and characterizes the capabilities required to achieve each objective. Understanding the capability-objective relationship enables risk prioritization based on the realistic threat population rather than theoretical worst-case scenarios alone.

6.1 Objective: Unauthorized Action Execution

The primary objective for the majority of adversaries is the execution of actions that the QODIQA enforcement engine would otherwise deny. This may be motivated by competitive intelligence collection, unauthorized data access, regulatory evasion, or the triggering of downstream system effects. This objective is attainable by T1 through T4 adversaries through different vectors, from simple consent bypass via malformed requests to sophisticated engine compromise.

6.2 Objective: Persistent Access Establishment

T3 and T5 adversaries may seek to establish persistent, undetected access to the enforcement system rather than triggering individual bypass events. Persistence objectives involve modification of policy records, insertion of backdoor consent grants, or compromise of engine logic such that evaluation outcomes are predictably biased without triggering anomaly detection. Persistent access is the highest-severity objective category due to its potential to corrupt the integrity guarantees of the enforcement system across an extended operational window.

6.3 Objective: Accountability Elimination

Adversaries who have achieved unauthorized action execution may separately target the audit subsystem to eliminate forensic records. Accountability elimination is typically a secondary objective pursued after primary bypass, though it

may be a primary objective for T3 insiders who are willing to accept detection risk for the primary action provided the evidence can be expunged. This objective requires write access to the audit subsystem, which is typically accessible only to T3 and above.

6.4 Objective: Enforcement Disruption

Some adversaries seek to disable enforcement rather than bypass it, inducing a state in which the system cannot evaluate requests and fails in a manner that allows actions to proceed. This objective is viable only if the target deployment implements fail-open fallback behavior, which is prohibited by QODIQA conformance requirements. Against conformant deployments, enforcement disruption produces a denial-of-service condition that is harmful to availability but does not constitute a consent bypass.

TABLE 6-A: ADVERSARY CAPABILITY MATRIX

ADVERSARY TIER	UNAUTHORIZED EXECUTION	PERSISTENT ACCESS	AUDIT SUPPRESSION	ENGINE DISRUPTION	SUPPLY CHAIN
T1: Opportunistic	Low	None	None	Low	None
T2: Auth. Abuser	Medium	Low	None	Low	None
T3: Privileged Insider	High	Medium	High	Medium	Low
T4: Sophisticated External	Medium	Medium	Medium	High	Low
T5: Nation-State	High	High	High	High	High

SECTION 7

Inference-Time Threat Vectors

Inference-time threats target the period during which an AI model generates outputs in response to inputs that have passed enforcement evaluation. Because QODIQA operates as a pre-execution gate rather than an output inspector, actions authorized at enforcement time may produce outputs with properties that differ from those anticipated at the time consent was granted. This section addresses the structural vulnerabilities this creates.

7.1 Request Manipulation

An adversary may construct requests that satisfy enforcement evaluation criteria while encoding parameters that induce undesired model behavior at inference time. This class of attack exploits the semantic gap between the structured enforcement request and the semantic content processed by the AI model. A request authorized for purpose P against data scope D may, through carefully constructed prompt parameters, induce the model to operate on data outside D or produce outputs serving a purpose other than P.

> THREAT REFERENCE: TH-07-01
>
> Adversary submits enforcement request with declared purpose `summarize:public_records` and receives authorization. The AI model, operating on authorized input, is induced via prompt construction to include non-public record content in its output through inference path manipulation.

7.2 Context Flooding

Context flooding attacks supply the AI model's context window with adversarial content designed to override or dilute enforcement-relevant instructions embedded by the system. Where QODIQA enforcement decisions are communicated to the AI model through system prompts or structured context blocks, an adversary with input control may flood the

context to displace or contradict these constraints. This attack is most viable in deployments where the AI model is given enforcement context through the context window rather than through out-of-band mechanisms.

7.3 Timing Attacks

Enforcement evaluation occurs at a specific point in time against a specific consent state. Adversaries may exploit temporal gaps between enforcement evaluation and action execution. If the interval between enforcement decision and execution is sufficiently large, or if the system permits re-use of enforcement decisions across multiple requests, consent records that were valid at evaluation time may have expired or been revoked by the time execution occurs. QODIQA-conformant deployments must bind enforcement decisions to the specific execution event and prohibit decision reuse.

SECTION 8

Consent Bypass Strategies

Consent bypass is the highest-priority threat category in this model. It encompasses all strategies through which an adversary achieves execution of an action without a valid, current consent record authorizing that specific actor, purpose, data scope, and execution context.

8.1 Consent Record Fabrication

An adversary with write access to the consent store, or with the ability to inject forged consent records through an unprotected interface, may fabricate consent records that satisfy evaluation criteria without corresponding to any genuine consent event. Fabricated records that are syntactically valid and cryptographically coherent will pass evaluation unless the enforcement engine independently verifies consent record provenance against an authority external to the store itself.

8.2 Scope Misrepresentation

Scope misrepresentation involves declaring a narrower or differently framed action scope in the enforcement request than is actually intended for execution. A request authorized for action scope A is submitted, after which the execution layer performs action B under the assumption that the enforcement authorization extends to related actions not explicitly evaluated. This attack exploits deployments in which the mapping between enforcement scope identifiers and actual execution scope is loose, ambiguous, or controlled by the requester.

8.3 Purpose Laundering

Purpose laundering involves the sequential use of individually authorized purposes to achieve an aggregate outcome that would not have been authorized as a single unified purpose. An adversary executes a series of actions, each individually within its authorized purpose, but whose combined result constitutes a use of data that no consent record authorizes. This attack is a structural consequence of purpose-scoped consent systems that evaluate individual actions in isolation without tracking compositional purpose accumulation.

8.4 Temporal Window Exploitation

Consent records define temporal validity windows. Adversaries may exploit the boundaries of these windows by submitting requests marginally within validity windows, rapidly constructing chains of authorized actions before revocation propagates, or targeting the period between consent expiry detection and enforcement engine awareness of that expiry. The latter window is a function of consent store synchronization latency and is a specific target for T4 adversaries.

8.5 Delegation Chain Abuse

In deployments where consent may be delegated from a primary consent holder to secondary actors, adversaries may exploit weaknesses in the delegation verification logic. Attacks include the fabrication of delegation chains that appear to

originate from a legitimate consent holder, the extension of delegation scope beyond what the original consent authorized, and the re-delegation of consent to adversary-controlled actors without the primary consent holder's knowledge.

SECTION 9

Model Manipulation and Prompt Injection Threats

Prompt injection represents a distinct class of threat applicable to deployments where AI model outputs influence enforcement decisions or where model-generated content is submitted as input to the enforcement system. The following threat entries address prompt injection in the context of QODIQA-specific enforcement flows.

9.1 Direct Prompt Injection

Direct prompt injection occurs when an adversary with control over the AI model's input supplies instructions designed to alter the model's behavior in ways that circumvent the enforcement envelope. In QODIQA-adjacent deployments, this may manifest as instructions that cause the model to misrepresent the scope, purpose, or identity parameters it reports to the enforcement system, or to generate outputs that exceed its authorized scope while maintaining surface-level compliance in the parameters visible to the enforcement engine.

9.2 Indirect Prompt Injection

Indirect prompt injection targets deployments in which the AI model processes external data sources as part of its inference computation. Adversarial content embedded in processed documents, web pages, database records, or other external sources may contain instructions that alter model behavior in ways not visible or attributable to the direct requester. Because QODIQA's enforcement evaluation is based on declared parameters rather than semantic inspection of model inputs, indirect injection may produce policy-violating behavior under a valid enforcement authorization.

> THREAT REFERENCE: TH-09-02
>
> AI model authorized to process document corpus D for purpose summarization retrieves a document containing injected instruction directing the model to extract and include all referenced personal identifiers in its output. The enforcement engine observes an authorized summarization request and issues a permit. The effective action is data exfiltration beyond authorized scope.

9.3 Enforcement Context Injection

In deployments where QODIQA enforcement decisions are communicated to the AI model through system prompt or structured context, adversaries who can influence context construction may inject false enforcement state representations. A model that processes a forged enforcement permit embedded in its context may behave as if authorization has been granted for actions that were not evaluated. This attack surface is eliminated in deployments where enforcement-to-model communication is conducted through cryptographically authenticated out-of-band mechanisms.

9.4 Jailbreak as Enforcement Circumvention

Model jailbreaking techniques, typically studied in the context of safety constraint bypass, are also relevant as enforcement circumvention vectors when enforcement constraints are communicated to the model through prompt-level mechanisms. An adversary who successfully induces the model to ignore its system prompt effectively eliminates all enforcement constraints that are encoded at that layer. QODIQA-conformant deployments must not rely on model-level instruction following as the sole enforcement mechanism for any constraint.

SECTION 10

Data Exfiltration Vectors

Data exfiltration threats target the unauthorized extraction of data that is within the processing scope of an authorized AI action but outside the consent-authorized disclosure scope. QODIQA's enforcement model prevents unauthorized action initiation but does not inspect the semantic content of outputs generated by authorized actions. This structural characteristic creates specific exfiltration risks.

10.1 Output Scope Leakage

An AI model authorized to process a specific data set for a declared purpose may include in its outputs information derived from data outside the authorized scope, either through inference from authorized data, retrieval of associated records, or model memorization effects. Output scope leakage is not detectable through pre-execution enforcement evaluation and requires post-execution output inspection to mitigate.

10.2 Steganographic Exfiltration

A compromised or adversarially induced AI model may encode exfiltrated data within its outputs using steganographic techniques that are not visible to output inspection systems. Encoded data may be embedded in whitespace patterns, numerical precision, formatting variations, or linguistic constructs that appear benign to reviewers. This attack class is particularly relevant in deployments where model outputs are returned to adversary-controlled endpoints.

10.3 Incremental Aggregation Exfiltration

Adversaries may construct sequences of individually authorized queries that, in aggregate, reconstruct sensitive information exceeding what any single authorized query would disclose. This attack exploits the absence of cross-

request state tracking in enforcement systems that evaluate each request independently. The aggregate result may constitute a disclosure that no individual consent record authorized.

10.4 Model Weight and Architecture Exfiltration

In deployments where QODIQA governs access to proprietary AI models, adversaries may use authorized query access to reconstruct model weights or architecture through model extraction attacks. While this vector targets model intellectual property rather than data governed by consent records, it may constitute a policy violation if the consent record authorizes query access for declared purposes but prohibits model reconstruction.

SECTION 11

Side-Channel and Indirect Leakage Risks

Side-channel leakage occurs when information about enforcement decisions, consent state, or protected data is disclosed through observable system properties rather than through direct output. Side-channel attacks do not require access to enforcement engine internals or consent store contents; they operate on observable system behavior.

11.1 Timing Side-Channels

Enforcement evaluation latency may vary as a function of the complexity of the consent record resolved for a specific request. An adversary making repeated requests with controlled variation in declared parameters may observe latency differentials that reveal information about consent record structure, the number of applicable policy conditions, or the presence or absence of specific consent records. Timing oracles of this type can enable consent record enumeration without direct read access to the store.

11.2 Decision Outcome Oracle

The binary or ternary enforcement outcome (permit, deny, restricted permit) constitutes a side-channel that reveals information about consent state to any requester. An adversary making systematic requests with varied parameters and observing outcomes can infer the structure of consent records without requiring read access to the store. This enumeration vector is a direct consequence of providing informative enforcement responses rather than uniform refusals for all denied requests.

11.3 Error Message Information Disclosure

Enforcement denial responses that include informative error detail (specifying which parameter failed evaluation, what conditions were not met, or what consent record was consulted) provide adversaries with guidance for iterative refinement of bypass attempts. Enforcement responses must be designed to provide the minimum information necessary for legitimate operational purposes without disclosing internal evaluation logic or consent record contents.

11.4 Cache and State Inference

If the enforcement engine maintains internal caches of resolved consent records or partially evaluated policy results, cache timing differentials may reveal information about which consent records have been recently accessed or which policy paths have been recently traversed. This side-channel is relevant in high-throughput deployments where caching is used to reduce evaluation latency.

SECTION 12

Enforcement Failure Modes

Enforcement failure modes are conditions in which the enforcement engine produces incorrect decisions, fails to produce decisions, or produces decisions that do not accurately reflect the resolved consent and policy state. This section

catalogs primary failure modes and their root causes.

12.1 Evaluation Context Desynchronization

If the enforcement engine evaluates a request against a consent state that is stale relative to the current consent store state (due to caching, replication lag, or synchronization failure) the evaluation outcome may not reflect current consent. This failure mode may result in permits for actions against revoked consent, or denials for actions against consent that has been newly granted. The fail-closed principle requires that desynchronization conditions produce denials rather than permits.

12.2 Policy Version Ambiguity

In deployments where multiple policy versions are active concurrently, the enforcement engine must unambiguously resolve which policy version applies to each request. Failure to implement strict version binding may allow adversaries to induce evaluation against a policy version other than the one intended by the operator, potentially exploiting permissive conditions in earlier versions that have since been tightened.

12.3 Incomplete Scope Resolution

Policy definitions may specify scope conditions using identifiers that require resolution to concrete data or action sets at evaluation time. If scope resolution is incomplete (due to identifier ambiguity, missing resolution data, or resolution service unavailability) the enforcement engine faces an evaluation decision under incomplete information. Conformant deployments must treat incomplete scope resolution as a fail-closed condition, denying the request rather than proceeding with partial evaluation.

12.4 Identity Binding Failure

Enforcement decisions are only meaningful if the declared actor identity in the request can be reliably bound to a known and authenticated principal. Identity binding failure occurs when authentication credentials are accepted that do not provide sufficient assurance of the claimed identity, when identity tokens are reused, replayed, or forged, or when the identity space used by the enforcement system does not map injectively to the identity space of actual system actors.

12.5 Fail-Open Fallback

Fail-open behavior, in which the system defaults to permitting action execution when the enforcement engine is unavailable or returns an error, represents a critical architectural failure that is categorically prohibited by QODIQA conformance requirements. Any deployment that implements fail-open fallback does not conform to the QODIQA framework and must be treated as operating without enforcement guarantees. This failure mode is listed here because it is a common implementation error in systems that prioritize availability over security.

> **NORMATIVE REQUIREMENT**
>
> QODIQA-conformant deployments MUST implement fail-closed semantics at all enforcement points. In no condition shall the absence, unavailability, or error state of the enforcement engine result in a default permit for any action. The absence of an enforcement decision is equivalent to a denial.

SECTION 13

Multi-System and Supply Chain Threats

Multi-system and supply chain threats arise from the integration of QODIQA components within larger ecosystems of software dependencies, infrastructure services, and third-party integrations. These threats are distinguished by the fact that the attack vector is not the QODIQA system itself but rather the components it depends upon or integrates with.

13.1 Dependency Compromise

QODIQA enforcement engine implementations depend on software libraries for cryptographic operations, data serialization, storage access, and network communication. Compromise of any of these dependencies (through typosquatting, maintainer account takeover, or intentional backdoor insertion) may introduce malicious logic into the enforcement stack. Dependency compromise attacks may target cryptographic libraries to weaken signature verification, serialization libraries to induce parsing vulnerabilities, or storage libraries to enable unauthorized access to the consent store.

13.2 Build Pipeline Integrity

The integrity of the enforcement engine binary deployed in production depends on the integrity of the build pipeline that produced it. Compromise of build tools, CI/CD infrastructure, or artifact repositories may result in production deployment of an enforcement engine binary that differs from the reviewed source code. This class of attack produces an enforcement engine that appears legitimate but contains adversary-controlled modifications to evaluation logic.

13.3 Third-Party Consent Store Providers

Deployments that utilize third-party services for consent record storage introduce a trust dependency on the provider's operational security posture. A compromised or malicious consent store provider can supply false consent records, withhold legitimate ones, or manipulate record timestamps. Deployments relying on external consent store providers must implement independent cryptographic verification of record integrity that does not rely on the provider's attestations.

13.4 Orchestration Layer Injection

In multi-agent or orchestrated AI deployments, the orchestration layer that sequences agent actions and assembles enforcement requests may itself be compromised or adversarially influenced. A compromised orchestrator can construct

enforcement requests that misrepresent action types, scope, or purpose while remaining syntactically valid. This attack surface is specific to deployments where enforcement request construction is delegated to components with lower trust levels than the enforcement engine itself.

SECTION 14

Insider and Privileged Actor Threats

Insider threats are distinguished from external threats by the adversary's pre-existing legitimate access to system components. In the QODIQA context, privileged actors include policy administrators, consent store operators, enforcement engine operators, and personnel with access to audit subsystems. Insider threats represent some of the highest-severity risk scenarios due to the difficulty of detection and the depth of access available to internal actors.

14.1 Policy Administrator Abuse

Policy administrators with write access to the policy store can introduce policy definitions that appear syntactically valid while creating conditions under which actions that should be denied are permitted. Techniques include the introduction of overly broad scope definitions, the removal of required consent conditions for specific purpose categories, the insertion of time-limited permissive exceptions, and the modification of policy evaluation logic to exclude specific actor identities from enforcement.

14.2 Consent Store Operator Compromise

Operators with administrative access to the consent store can insert fabricated consent records, modify the temporal validity of existing records, suppress revocation events, or delete records that would have produced denials. This class of attack is distinguished from external consent record manipulation by the operator's possession of legitimate credentials, making access pattern analysis less effective as a detection mechanism.

14.3 Audit System Administrator Collusion

Personnel with administrative access to the audit subsystem can selectively delete or modify enforcement decision records, eliminating forensic evidence of unauthorized actions or suppressing records that would reveal the unauthorized activities of other principals. Effective audit integrity defense requires that audit records be replicated to infrastructure not accessible to operational administrators, and that cryptographic commitments to audit record sets be generated and published at intervals sufficient to detect retroactive modification.

14.4 Multi-Party Collusion

The most severe insider threat scenario involves collusion between multiple privileged actors across different administrative domains. A coordinated insider attack may combine policy modification by a policy administrator with audit suppression by an audit administrator, enabling persistent unauthorized access without leaving detectable audit artifacts. Defense against multi-party collusion requires administrative domain separation enforced by technical controls rather than organizational policies alone.

SECTION 15

Adversarial Automation and Scaling Risks

Adversarial automation threats are characterized by the use of programmatic tooling to conduct attacks at scale, volume, or speed that would be impractical for manual operation. These threats are particularly relevant for enforcement systems that process high volumes of requests, as automated attacks can exploit rate limits, extract information through enumeration, or probe for bypass conditions across the full parameter space.

15.1 Automated Consent Enumeration

Automated enumeration attacks submit large volumes of systematically varied requests to infer the structure of the consent store from enforcement decision outcomes. By observing which parameter combinations produce permits versus denials, an adversary can reconstruct a partial map of active consent records without read access to the store. Countermeasures require uniform response timing, consistent response format for all denial outcomes, and rate limiting sufficient to make exhaustive enumeration operationally impractical.

15.2 Fuzzing of Enforcement Interfaces

Automated fuzzing directs malformed, boundary, and edge-case inputs at the enforcement request interface to identify parsing vulnerabilities, evaluation logic errors, or unhandled exception conditions that may produce unintended permit outcomes. The enforcement engine's input validation layer must be designed to treat all non-conformant input as a deny condition rather than as a recoverable error that may default to permit.

15.3 Replay and Token Harvesting at Scale

Automated systems can capture and replay enforcement decision tokens, authentication credentials, or consent grant artifacts across sessions and time periods at a scale that enables systematic exploitation of any replay vulnerability. Token harvesting from compromised endpoints or network captures can be automated to produce a repository of usable credentials that are submitted programmatically at high volume.

15.4 Model Extraction at Scale

In deployments where QODIQA governs access to AI model inference, automated query generation can conduct model extraction at a scale and systematicity that produces high-fidelity surrogate models within operational cost bounds. Scale-enabled model extraction may violate consent records that permit individual inference but prohibit systematic extraction, and represents a purpose violation achievable through individually authorized actions performed in aggregate.

SECTION 16

Detection and Observability Constraints

Effective threat detection in QODIQA-conformant deployments is constrained by structural characteristics of the enforcement model. This section identifies the primary constraints on detection capability and their implications for threat response.

16.1 Pre-Execution Detection Boundary

QODIQA enforcement operates as a pre-execution gate. Threats that manifest through post-execution behavior (including output scope leakage, steganographic exfiltration, and incremental aggregation attacks) are not observable to the enforcement engine and are outside the detection scope of the enforcement subsystem. Comprehensive threat detection requires supplementary post-execution monitoring infrastructure operating at the output layer.

16.2 Semantic Opacity

The enforcement engine evaluates structured enforcement request parameters without semantic inspection of model inputs or outputs. Threats that operate through semantic manipulation (including indirect prompt injection, purpose laundering through individually authorized requests, and steganographic exfiltration) are not detectable through enforcement-layer observability. Detection of these threat classes requires semantic analysis capabilities that are outside the scope of the QODIQA enforcement layer.

16.3 Insider Threat Detection Limitations

Insider threats exercised through legitimate administrative credentials and within the operational parameters of the actor's authorization scope may not produce anomaly signals distinguishable from legitimate administrative activity.

Detection of insider threats requires behavioral baselining, peer comparison, and independent administrative review that is not a function of the enforcement system itself.

16.4 Supply Chain Attack Detectability

Supply chain attacks that modify enforcement engine behavior at the binary level are not detectable through operational monitoring of evaluation outcomes alone. Detection requires reproducible build verification, binary integrity attestation, and runtime behavioral consistency monitoring that can identify deviations from expected evaluation logic without requiring access to the source of those deviations.

SECTION 17

Residual Risk and Non-Eliminable Threats

Certain threat classes cannot be fully eliminated through the design and operation of a QODIQA-conformant enforcement system. These residual risks arise from structural characteristics of AI inference, from the nature of consent as a social and legal construct, or from threat actors with capabilities that exceed the security model's design assumptions.

17.1 Semantic Indeterminacy of AI Outputs

AI model outputs are not deterministically predictable from authorized inputs and consent parameters. Even with full enforcement correctness, the semantic content of model outputs may include information that, while derived from authorized processing, constitutes a consent violation in its disclosed form. This residual risk is non-eliminable through pre-execution enforcement and requires acceptance, mitigation through post-execution inspection, or constraint of authorized action scope.

17.2 Consent Validity Under Coercion

QODIQA's consent model assumes that consent records represent genuine expressions of informed and uncoerced consent by the consent holder. Consent records obtained under coercion, deception, or incapacity are not distinguishable at the enforcement layer from legitimately obtained records. This residual risk is a structural consequence of the enforcement model's position downstream of consent elicitation and is not addressable through enforcement mechanism design.

17.3 Nation-State Level Adversaries

Adversaries with nation-state capabilities and persistent access objectives may possess the resources to compromise hardware infrastructure, conduct extended covert supply chain operations, or leverage zero-day vulnerabilities against any software component in the QODIQA stack. Defense against this adversary class requires operational security postures and hardware trust anchors beyond the scope of any software-only enforcement framework. QODIQA provides meaningful structural defense against this adversary class but cannot provide categorical guarantees.

17.4 Unknown Attack Techniques

This threat model reflects the current state of knowledge regarding adversarial techniques applicable to AI enforcement systems. Novel attack techniques, including those enabled by advances in AI capability itself, may render current defensive assumptions inadequate. The model must be reviewed and updated on a defined cadence to incorporate newly identified threat classes.

SECTION 18

Assumptions and Limitations

This threat model is constructed on a set of explicit assumptions. Deployments in which these assumptions do not hold must extend the model to account for the resulting additional threat exposure.

18.1 Assumed Properties of the Deployment Environment

- The enforcement engine is deployed in an environment where physical access is restricted to authorized personnel.
- Network communication between system components occurs over authenticated and encrypted channels.
- Operating system and hypervisor integrity is maintained through established operational security practices.
- Authentication mechanisms for all system interfaces provide at minimum strong single-factor assurance, with multi-factor assurance for privileged administrative interfaces.
- Time synchronization across system components is accurate to within the granularity of consent record temporal validity windows.

18.2 Assumed Properties of Consent Records

- Consent records are created through processes that verify the identity, capacity, and informed understanding of the consent holder at the time of consent grant.
- Consent records are cryptographically signed at creation time and the signing key is not accessible to enforcement engine operators.
- Revocation events are propagated to the enforcement engine within a defined maximum latency that is shorter than the minimum practical window for temporal bypass attacks.

18.3 Modeling Limitations

This threat model does not model the full attack surface of AI models operating within QODIQA-governed deployments, including training-time poisoning, weight manipulation, or hardware-level inference attacks. It does not model regulatory or jurisdictional threats that may affect the legal validity of consent records. It does not model threats arising from consent holder behavior, including voluntary disclosure of consent credentials or consent grant to adversary-controlled actors.

> This document is a living specification. Identified limitations and assumption failures should be reported to the document maintainer for incorporation into future revisions.

SECTION 19

Appendix: Threat Mapping Tables

A.1 Threat-to-Adversary-Tier Mapping

TABLE A.1: THREAT CATEGORY APPLICABILITY BY ADVERSARY TIER

THREAT CATEGORY	T1	T2	T3	T4	T5
TC-01: Consent Bypass	Low	Medium	High	High	High
TC-02: Scope Escalation	Low	High	High	Medium	High
TC-03: Policy Corruption	None	Low	High	Medium	High
TC-04: Audit Suppression	None	None	High	Low	High

THREAT CATEGORY	T1	T2	T3	T4	T5
TC-05: Inference Manipulation	Medium	High	Medium	High	High
TC-06: Information Disclosure	Medium	High	High	High	High
TC-07: Availability Degradation	Medium	Medium	Medium	High	High
TC-08: Supply Chain	None	None	Low	Low	High

A.2 Threat-to-Attack-Surface Mapping

TABLE A.2: THREAT-TO-ATTACK-SURFACE RELATIONSHIP MATRIX

ATTACK SURFACE	APPLICABLE THREATS	PRIMARY ADVERSARY TIERS	DETECTION FEASIBILITY
Input Interfaces	TC-01, TC-02, TC-05, TC-06	T1, T2, T4	Medium: request-level anomaly detection feasible
Policy and Consent Store	TC-01, TC-02, TC-03, TC-06	T3, T5	Medium: integrity monitoring required
Enforcement Engine	TC-01, TC-03, TC-07, TC-08	T4, T5	Low: binary attestation required
Audit Subsystem	TC-04, TC-06	T3, T5	Low: external commitment verification required
AI Model (Downstream)	TC-02, TC-05, TC-06	T1, T2, T3, T4	Low: out-of-scope for enforcement layer
Build and Dependency Pipeline	TC-03, TC-07, TC-08	T5	Low: reproducible build verification required

A.3 Control Mapping Summary

TABLE A.3: THREAT COUNTERMEASURE REFERENCE SUMMARY

ID	PRIMARY COUNTERMEASURES	REFERENCE
TC-01	Fail-closed semantics; cryptographic consent record verification; identity binding enforcement	QODIQA-SEC-2026-001 §4, §7
TC-02	Strict scope binding; purpose declaration verification; scope resolution integrity	QODIQA-SEC-2026-001 §5
TC-03	Cryptographic integrity protection of policy store; administrative access controls; change audit logging	QODIQA-SEC-2026-001 §6
TC-04	Append-only audit store; cryptographic commitment chains; external replication; independent integrity verification	QODIQA-SEC-2026-001 §8
TC-05	Out-of-band enforcement communication; post-execution output inspection; model isolation	QODIQA-REF-2026-001 §9
TC-06	Uniform response timing; minimal error disclosure; encrypted response channels; differential privacy for enumeration prevention	QODIQA-SEC-2026-001 §9
TC-07	Fail-closed default; availability monitoring; capacity planning; circuit breakers	QODIQA-REF-2026-001 §6
TC-08	Reproducible builds; dependency pinning and verification; binary attestation; SBOM generation and monitoring	QODIQA-SEC-2026-001 §11

1 Threat likelihood designations (High / Medium / Low / None) reflect the structural accessibility of the threat to the identified adversary tier given assumed baseline security controls. They do not represent probabilistic risk estimates and should not be interpreted as such without supplementary quantitative risk analysis.

2 Reference document identifiers (e.g., QODIQA-SEC-2026-001) refer to specifications within the QODIQA corpus. Consult the QODIQA Governance Charter for the authoritative corpus index.

3 This threat model is subject to revision as the operational environment, AI capability landscape, and adversarial technique space evolve. The document version control process is defined in the QODIQA Governance Charter.

DOCUMENT STATUS

Document Status and Corpus Position

This document constitutes a normative specification within the QODIQA standard corpus. It is designated as the authoritative adversarial threat model for all systems implementing or claiming conformance with the QODIQA Deterministic Runtime Consent Enforcement framework. Implementers, auditors, and certification bodies referencing this document should verify they hold the current version as identified in the QODIQA Governance Charter.
This document should be read together with the following related specifications:

- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement
- QODIQA: Terminology and Normative Definitions
- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Governance Charter for the QODIQA Standard Corpus

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dițescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-TM-2026-001
TITLE	Extended Adversarial Threat Model
SUBTITLE	Comprehensive Adversarial Threat Modeling for Deterministic Runtime Consent Enforcement Systems
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Technical: Normative
CLASSIFICATION	Technical / Normative / Security Analysis
DOCUMENT STATUS	Normative
GOVERNING AUTHORITY	QODIQA Governance Charter
CONTACT	contact@qodiqa.com
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 11 PART II · NORMATIVE TECHNICAL SPECIFICATIONS

Global Enforcement Invariants

Globally binding enforcement invariants formalizing constraints implicit within the corpus.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-GEI-2026-001	1.0	2026	Normative Specification	Normative Extension	NORMATIVE

ABSTRACT

This document defines the Global Enforcement Invariants of the QODIQA specification corpus. It centralizes constraints that are globally binding across all QODIQA-conformant systems, formalizing enforcement obligations that are implicit within the Core Standard, Reference Architecture, Formal Enforcement Model, and Threat Model. Each invariant is absolute, exceptionless, binary, testable, and auditable. This document does not modify, override, or reinterpret any existing QODIQA specification. It defines globally enforced invariants that formalize constraints already implicit within the QODIQA corpus.

SECTION 1

Purpose and Scope

This document centralizes enforcement invariants applicable across all QODIQA-conformant systems. Invariants defined herein are globally binding and override any implicit interpretation of QODIQA corpus documents.

> NON-MODIFICATION CLAUSE
>
> This document does not modify, override, or reinterpret any existing QODIQA specification. It defines globally enforced invariants that formalize constraints already implicit within the QODIQA corpus.

No new behavior is introduced by this document. No new mechanisms, system components, or scope expansions are defined. All invariants contained herein are extracted, formalized, and centralized from constraints already present within the QODIQA specification corpus.

This document SHALL be treated as a normative extension layer. All QODIQA-conformant systems SHALL satisfy every invariant defined in Section 4 without exception.

SECTION 2

Relationship to QODIQA Corpus

This document consolidates invariant constraints across the following corpus instruments:

- **Core Standard** (QODIQA-CORE-2026-001), defines normative runtime consent enforcement requirements.
- **Reference Architecture**: defines the structural execution model and component boundaries.
- **Formal Enforcement Model** (QODIQA-SPEC-68P-V3-2026), defines the 68-point enforcement framework and decision logic.
- **Threat Model and Abuse Case Analysis**, defines adversarial conditions and failure categories.

Invariants defined herein are derived exclusively from constraints present within the above instruments. No invariant introduces behavior absent from the referenced corpus.

In all cases of apparent conflict between this document and a referenced corpus instrument, the more restrictive constraint SHALL govern.

SECTION 3

Invariant Model Overview

Invariants defined in this document satisfy the following properties:

- Invariants are globally binding across all execution contexts in which QODIQA enforcement applies.
- Invariants override any implicit interpretation of corpus documents.
- Invariants are mandatory. No invariant is advisory or conditional.
- Invariants are binary. A system either satisfies an invariant or it does not. No partial satisfaction is recognized.
- Invariants are testable. Each invariant admits a deterministic verification procedure.
- Invariants are auditable. Conformance evidence SHALL be producible for each invariant from system artifacts alone.

Each invariant is assigned a unique identifier of the form `GI-XX`. Identifiers are stable across corpus versions. Invariants SHALL NOT be renumbered.

SECTION 4

Global Enforcement Invariants

The following invariants are globally binding on all QODIQA-conformant systems. Violation of any single invariant constitutes a non-compliant system.

4.1 Consent Enforcement

GI-01

Runtime Consent Requirement

STATEMENT

The system SHALL NOT execute any inference operation without verified valid consent at runtime.

NON-COMPLIANCE CONDITION

Any execution path in which inference occurs without runtime consent validation having been performed and affirmatively resolved.

REFERENCES

- Core Standard 2.1
- Formal Enforcement Model 7
- Threat Model 4

GI-02

Consent Scope Binding

STATEMENT

The system SHALL NOT execute any operation whose scope exceeds the scope explicitly bound within the validated consent artifact.

NON-COMPLIANCE CONDITION

Any operation whose action type, data domain, purpose, or principal identifiers extend beyond those specified in the resolved consent artifact governing that execution.

REFERENCES

- Core Standard 3.2
- Formal Enforcement Model 12
- Reference Architecture 5.1

GI-03

Consent Validity Window

STATEMENT

The system SHALL NOT honor a consent artifact whose validity window has expired or whose revocation timestamp predates the execution timestamp.

NON-COMPLIANCE CONDITION

Any execution authorized by a consent artifact whose `valid_until` field is in the past relative to the authoritative execution clock, or for which a valid revocation record exists.

REFERENCES

- Core Standard 4.3
- Formal Enforcement Model 15
- Threat Model 6

4.2 Determinism

GI-04

Deterministic Evaluation Requirement

STATEMENT

The system SHALL produce an identical enforcement decision for any identical combination of intent declaration, consent artifact, and policy set, irrespective of execution time, environment, or instance.

NON-COMPLIANCE CONDITION

Any execution path in which the enforcement decision for a given input triple is not guaranteed to be identical across all invocations of the enforcement layer.

REFERENCES

- Core Standard 5.1
- Reference Architecture 3.4
- Formal Enforcement Model 2

GI-05

Replay Invariant

STATEMENT

The system SHALL produce an enforcement decision that, when replayed against the frozen input state recorded in the corresponding decision artifact, yields an identical outcome.

NON-COMPLIANCE CONDITION

Any enforcement decision for which replay against the recorded frozen input state produces a different outcome than the original decision.

REFERENCES

- Core Standard 7.3
- Formal Enforcement Model 19
- Reference Architecture 6.2

4.3 Execution Flow Integrity

GI-06

Mandatory Enforcement Layer Traversal

STATEMENT

The system SHALL NOT permit any inference operation to reach an execution endpoint without traversing the QODIQA enforcement layer in its entirety.

NON-COMPLIANCE CONDITION

Any execution path in which inference reaches an endpoint via a route that bypasses, short-circuits, or partially traverses the enforcement layer.

REFERENCES

- Core Standard 2.3
- Reference Architecture 4.1
- Threat Model 3.2

GI-07

Intent Declaration Precedence

STATEMENT

The system SHALL NOT initiate consent resolution or policy evaluation prior to receiving and validating a fully formed, structurally conformant intent declaration.

NON-COMPLIANCE CONDITION

Any consent resolution or policy evaluation initiated against an absent, malformed, or structurally non-conformant intent declaration.

REFERENCES

- Core Standard 3.1
- Reference Architecture 5.2
- Formal Enforcement Model 8

4.4 Audit Requirements

GI-08

Mandatory Decision Artifact Production

STATEMENT

The system SHALL produce a cryptographically signed, tamper-evident decision artifact for every enforcement decision, whether the outcome is PERMIT or DENY.

NON-COMPLIANCE CONDITION

Any enforcement decision for which no corresponding signed decision artifact is produced and durably recorded, including DENY decisions and error-path terminations.

REFERENCES

- Core Standard 6.4
- Formal Enforcement Model 22
- Reference Architecture 6.1

GI-09

Frozen Context Capture

STATEMENT

The system SHALL capture and bind the complete input state (intent declaration, resolved consent artifact, and applicable policy set) as an immutable frozen context within the decision artifact at the moment of enforcement evaluation.

NON-COMPLIANCE CONDITION

Any decision artifact that omits any element of the input triple, or that records input state that has been modified after enforcement evaluation was initiated.

REFERENCES

- Core Standard 6.5
- Reference Architecture 6.2
- Formal Enforcement Model 20

4.5 State Machine Integrity

GI-10

Enforcement State Sequencing

STATEMENT

The system SHALL execute enforcement states in the prescribed sequence: intent declaration validation, consent resolution, policy evaluation, decision production. No state SHALL be skipped, reordered, or repeated.

NON-COMPLIANCE CONDITION

Any execution in which enforcement states are executed out of prescribed sequence, any state is skipped, or any state is repeated without system restart.

REFERENCES

- Reference Architecture 4.3
- Formal Enforcement Model 5
- Core Standard 2.4

GI-11

Terminal State Finality

STATEMENT

The system SHALL NOT permit re-entry into any enforcement state after a terminal decision state, PERMIT or DENY, has been reached for a given execution request.

NON-COMPLIANCE CONDITION

Any execution path in which enforcement states are re-entered or enforcement logic is re-applied to the same execution request after a terminal decision has been issued.

REFERENCES

- Reference Architecture 4.4
- Formal Enforcement Model 6
- Threat Model 5.1

4.6 Failure Behavior

GI-12

Fail-Closed Default

STATEMENT

The system SHALL default to DENY and terminate execution upon any failure, error, timeout, or indeterminate state encountered within the enforcement layer, irrespective of the failure source.

NON-COMPLIANCE CONDITION

Any execution path in which an enforcement layer failure, error, or timeout results in any outcome other than DENY, or in which execution continues past the point of failure.

REFERENCES

- Core Standard 4.6
- Formal Enforcement Model 30
- Threat Model 7

GI-13

Prohibition of Fallback Execution

STATEMENT

The system SHALL NOT implement any fallback execution path that permits inference to proceed when the enforcement layer is unavailable, degraded, or has returned a non-affirmative result.

NON-COMPLIANCE CONDITION

The existence of any code path, configuration option, or operational mode in which inference execution proceeds under conditions of enforcement layer unavailability or degradation.

REFERENCES

- Core Standard 4.6
- Reference Architecture 7.1
- Threat Model 8

4.7 Data Integrity

GI-14

Consent Artifact Immutability

STATEMENT

The system SHALL NOT permit modification of a consent artifact after it has been issued and recorded. Any consent modification SHALL require issuance of a new consent artifact with a distinct identifier and timestamp.

NON-COMPLIANCE CONDITION

Any system state in which a previously issued consent artifact can be modified in-place, or in which modified consent data is presented under an identifier belonging to a previously issued artifact.

REFERENCES

- Core Standard 5.3
- Reference Architecture 5.4
- Formal Enforcement Model 11

GI-15

Decision Artifact Tamper Evidence

STATEMENT

The system SHALL produce decision artifacts whose cryptographic integrity is verifiable by any party in possession of the applicable public verification key, at any time after artifact issuance.

NON-COMPLIANCE CONDITION

Any decision artifact for which cryptographic integrity verification fails, or for which the verification mechanism depends on mutable system state rather than the artifact's own content.

REFERENCES

- Core Standard 6.6
- Security and Cryptographic Profile 4
- Reference Architecture 6.3

4.8 Policy Enforcement

GI-16

Policy Immutability at Evaluation Time

STATEMENT

The system SHALL evaluate all enforcement decisions against the policy set that was active and locked at the moment the intent declaration was received. Policy updates that occur after intent declaration receipt SHALL NOT affect the evaluation of that request.

NON-COMPLIANCE CONDITION

Any enforcement decision in which the policy set applied differs from the policy set that was active at intent declaration receipt time for that request.

REFERENCES

- Core Standard 5.4
- Formal Enforcement Model 17
- Reference Architecture 5.3

GI-17

Prohibition of Policy Override

STATEMENT

The system SHALL NOT permit any runtime signal, operator instruction, user request, or external interface to override, suspend, or bypass an applicable policy constraint during enforcement evaluation.

NON-COMPLIANCE CONDITION

The existence of any mechanism (including administrative interfaces, debug modes, or API parameters) through which a policy constraint may be suspended or bypassed during a live enforcement evaluation.

REFERENCES

- Core Standard 5.5
- Formal Enforcement Model 31
- Threat Model 9

4.9 Non-Ambiguity

GI-18

Binary Decision Requirement

STATEMENT

The system SHALL produce exclusively binary enforcement decisions: PERMIT or DENY. The system SHALL NOT produce probabilistic, conditional, deferred, or advisory outcomes.

NON-COMPLIANCE CONDITION

Any enforcement outcome that is expressed as a probability, a condition to be resolved post-execution, a recommendation, or any value other than an unambiguous PERMIT or DENY.

REFERENCES

- Core Standard 5.1
- Formal Enforcement Model 3
- Reference Architecture 4.5

4.10 Prohibition of Implicit Behavior

GI-19

Prohibition of Implicit Consent

STATEMENT

The system SHALL NOT treat the absence of an explicit DENY signal, the passage of time without response, or any inferred behavioral indicator as equivalent to explicit consent. Consent MUST be affirmatively and explicitly represented in a resolved consent artifact.

NON-COMPLIANCE CONDITION

Any execution path in which consent is inferred, assumed, or defaulted to PERMIT in the absence of a validated, explicitly structured consent artifact.

REFERENCES

- Core Standard 2.2
- Formal Enforcement Model 9
- Threat Model 4.1

GI-20

Prohibition of Interpretive Execution

STATEMENT

The system SHALL NOT apply semantic interpretation, probabilistic inference, or contextual inference to resolve ambiguity in consent artifacts, intent declarations, or policy constructs. All enforcement inputs MUST be machine-readable and unambiguously structured prior to enforcement evaluation.

NON-COMPLIANCE CONDITION

Any enforcement decision in which the system applies reasoning, inference, or interpretation to resolve ambiguity in any enforcement input rather than issuing an immediate DENY for non-conformant inputs.

REFERENCES

- Core Standard 5.6
- Formal Enforcement Model 4
- Reference Architecture 3.1

SECTION 5

Invariant Interdependency

Invariants defined in this document are mutually reinforcing. No invariant may be read in isolation. The following structural interdependencies apply:

- GI-01 through GI-03 establish the consent preconditions without which no enforcement evaluation may proceed. GI-06 and GI-07 establish the structural preconditions for consent resolution and policy evaluation respectively. These invariants are collectively prerequisite to all subsequent invariants.
- GI-04 and GI-05 govern the deterministic and replayable character of all enforcement decisions. They apply to every PERMIT and DENY outcome produced under GI-01 through GI-03.
- GI-08 and GI-09 govern artifact production obligations that arise upon resolution of any enforcement decision. They depend on the successful traversal of GI-06 and GI-07.
- GI-10 and GI-11 govern the state machine through which all preceding invariants are executed. Violation of GI-10 or GI-11 constitutes a violation of the enforcement preconditions for all other invariants.
- GI-12 and GI-13 govern failure conditions. They apply whenever any preceding invariant cannot be satisfied. Failure to apply GI-12 or GI-13 constitutes a systemic violation.
- GI-14 through GI-17 govern data and policy integrity across the full enforcement lifecycle. They apply to all inputs consumed and outputs produced by the enforcement layer.
- GI-18 through GI-20 govern the structural properties of all inputs and outputs. Violation of GI-18, GI-19, or GI-20 constitutes a structural precondition failure that renders all dependent invariants untestable.

No invariant may be bypassed. Violation of one invariant constitutes a violation of the system as a whole.

SECTION 6

Non-Compliance Conditions (Global)

A system is NON-COMPLIANT with this document if any single invariant defined in Section 4 is violated.

PARTIAL COMPLIANCE

No partial compliance with this document is recognized. A system that satisfies 19 of 20 invariants is a non-compliant system.

DEGRADED COMPLIANCE

No degraded compliance mode is recognized. A system operating in any mode (including maintenance, testing, debugging, or reduced-capability modes) SHALL satisfy all invariants or SHALL NOT execute inference operations.

CONDITIONAL COMPLIANCE

No conditional compliance is recognized. Invariant satisfaction MUST NOT depend on runtime configuration flags, deployment environment properties, or operator discretion.

Non-compliance determination is binary. A system either satisfies all invariants or it does not. No scoring, weighting, or tiered assessment of invariant satisfaction is defined or recognized by this document.

SECTION 7

Cross-Document Traceability

The following table maps each invariant to its primary corpus references.

TABLE 1: INVARIANT-TO-CORPUS TRACEABILITY MATRIX

INVARIANT	CORE STANDARD	FORMAL MODEL	THREAT MODEL	REFERENCE ARCHITECTURE
GI-01	2.1	7	4	N/A
GI-02	3.2	12	N/A	5.1
GI-03	4.3	15	6	N/A
GI-04	5.1	2	N/A	3.4
GI-05	7.3	19	N/A	6.2
GI-06	2.3	N/A	3.2	4.1
GI-07	3.1	8	N/A	5.2
GI-08	6.4	22	N/A	6.1
GI-09	6.5	20	N/A	6.2
GI-10	2.4	5	N/A	4.3
GI-11	N/A	6	5.1	4.4
GI-12	4.6	30	7	N/A
GI-13	4.6	N/A	8	7.1
GI-14	5.3	11	N/A	5.4
GI-15	6.6	N/A	N/A	6.3
GI-16	5.4	17	N/A	5.3
GI-17	5.5	31	9	N/A

INVARIANT	CORE STANDARD	FORMAL MODEL	THREAT MODEL	REFERENCE ARCHITECTURE
GI-18	5.1	3	N/A	4.5
GI-19	2.2	9	4.1	N/A
GI-20	5.6	4	N/A	3.1

SECTION 8

Enforcement Implications

Systems subject to this document SHALL implement enforcement behavior consistent with the following obligations:

- The system MUST fail-closed. Every failure, timeout, indeterminate state, or unresolvable input within the enforcement layer MUST produce a DENY outcome and terminate the affected execution request.
- An invariant violation MUST terminate execution of the affected request immediately. The system MUST NOT attempt to recover, retry, or continue execution past the point of invariant violation.
- No fallback execution path is permitted. The system MUST NOT implement any execution route that allows inference to proceed when the enforcement layer has not affirmatively produced a PERMIT decision.
- Enforcement layer unavailability MUST be treated as equivalent to a DENY decision. The system MUST NOT defer execution pending restoration of enforcement layer availability.
- All enforcement decisions MUST be recorded as signed decision artifacts prior to the release of any execution result to any downstream system or caller.
- The system MUST NOT expose any interface, parameter, mode, or configuration by which the enforcement layer may be disabled, suspended, or bypassed in production operation.

SECTION 9

Conclusion

This document defines 20 globally binding enforcement invariants for QODIQA-conformant systems. Each invariant is absolute, exceptionless, binary, testable, and auditable. No new concepts, mechanisms, or scope expansions are introduced. All invariants formalize constraints already implicit within the QODIQA specification corpus.

Conformance to this document requires full satisfaction of all invariants defined in Section 4 across all execution contexts, operational modes, and deployment configurations.

DOCUMENT STATUS

Document Status and Corpus Alignment

This document is a normative extension layer within the QODIQA specification corpus. It centralizes globally binding enforcement invariants that formalize constraints implicit across the corpus. It does not modify, override, or reinterpret any existing QODIQA specification.

This document SHALL be read together with the following related specifications:

- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement
- QODIQA: Terminology and Normative Definitions
- QODIQA: Threat Model and Abuse Case Specification
- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dițescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-GEI-2026-001
TITLE	Global Enforcement Invariants
SUBTITLE	Deterministic Runtime Consent Enforcement for Artificial Intelligence Systems
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Normative Specification
DOCUMENT STATUS	Normative Extension
SCOPE	Global Constraint Layer
CLASSIFICATION	Extension Layer
AUTHORITY	QODIQA Corpus
ROLE	Global Constraint Definition
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

PART III

III

Operational documents

9 documents

12 Implementation Playbook 445
13 Implementation Conformance Checklist 506
14 Reference Implementation 524
15 Failure Handling and Recovery 555
16 Non-Compliance and Failure Modes 578
17 System Boundary and Trust Model 613
18 Interoperability and Deployment 651
19 Terminology and Definitions 691
20 Worked Example 730

CHAPTER 12 PART III · OPERATIONAL DOCUMENTS

Implementation Playbook for Deterministic Runtime Consent Enforcement

Operationalizes the Core Standard into a concrete infrastructure specification for production deployments of deterministic runtime consent enforcement.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-IMPL-2026-001	1.0	2026	Technical Specification	Institutional	NORMATIVE

SCOPE AND PURPOSE

This document operationalizes the QODIQA Core Standard into a deployable infrastructure specification. It translates normative requirements (defined in the Core Standard and the 68-Point Enforcement Framework) into concrete component architectures, deployment topologies, integration patterns, security controls, and operational procedures.

The Playbook does not establish new normative requirements. It does not reinterpret enforcement philosophy. Every architectural decision traces to an existing control in the QODIQA corpus. Where the Core Standard permits implementation latitude, this Playbook defines the technical constraints within which that latitude may be exercised. **The governing principle is unchanged: enforcement is deterministic, or it is not enforcement.**

SECTION 1

Executive Technical Overview

The QODIQA Core Standard defines what must be enforced: consent tokens govern all inference-time data processing; enforcement is deterministic and fail-closed; revocation propagates within defined time bounds; and an immutable audit record is produced for every enforcement decision. This Playbook defines how those requirements are realized in infrastructure.

1.1 Purpose of This Playbook

This Playbook is addressed to principal architects, infrastructure engineers, security reviewers, and technical governance leads responsible for implementing deterministic runtime consent enforcement within production systems. Its audience is not policy authors or legal counsel; it is the engineering teams responsible for building and operating the enforcement layer.

> This document answers six operational questions about QODIQA infrastructure: **how to build it, how to deploy it, how to integrate it, how to scale it, how to validate it, and how to secure it**, without becoming a product manual or introducing scope beyond runtime enforcement.

1.2 Relationship to the Core Standard

The Core Standard is normative. This Playbook is subordinate. In the event of conflict between this document and the Core Standard, the 68-Point Enforcement Framework, or the QODIQA Whitepaper, those documents prevail. This Playbook may be revised independently to reflect updated deployment practices, provided all revisions maintain strict compliance with Core Standard normative requirements.

1.3 Scope Limitations

This Playbook is scoped strictly to runtime enforcement infrastructure. Pre-deployment consent collection workflows, legal agreement structures, data retention policies, user interface design, model training governance, upstream data provenance, and organizational policy authorship are outside scope unless they directly intersect with enforcement component behavior.

1.4 System Boundary Definition

The QODIQA enforcement boundary begins at the point at which a processing request enters the enforcement perimeter (defined as the first component capable of intercepting and blocking that request) and ends at the point at which an audit record for the corresponding enforcement decision is written to the Audit Ledger in an immutable state. This boundary is a hard system invariant: no processing decision exists within the QODIQA semantics until both conditions are satisfied simultaneously. A PERMIT without a committed AL record is undefined; the system must behave as if no decision occurred.

Any processing that occurs outside these boundaries is not governed by this Playbook and is not considered QODIQA-conformant, regardless of the controls applied to it. The boundary is not a logical construct; it is an operational demarcation that must be verifiable through AL record completeness audit at any point in time.

ARCHITECTURAL ASSUMPTIONS: INVARIANT CLASS

- All components are operated by the same institutional entity responsible for the inference system
- No component shares a process space with the inference system under enforcement
- Cross-institutional component sharing requires additional trust negotiation not specified here
- Synchronous enforcement path; asynchronous processing does not satisfy enforcement requirements
- All inter-component channels are authenticated, encrypted, and integrity-protected at the transport layer; unauthenticated responses are treated as component failure
- Time is monotonic within each component; no component may use wall-clock time as the sole basis for a deterministic enforcement decision

These assumptions hold as architectural invariants across all six deployment topologies defined in Section 3. Deviation from any invariant places the deployment outside the conformance boundary.

1.5 Global System Invariants

The following invariants must hold at all times across the entire deployed system, regardless of topology, load condition, or failure state. They are stated as universal propositions against which any operational state may be tested. Violation of any invariant is a conformance failure requiring immediate remediation and forensic audit.

GLOBAL INVARIANT SET: QODIQA-IMPL-INV — INVARIANTS

```
// INV-01: Completeness
∀ inference execution e:
  ∃ audit_record r ∈ AL : r.decision = PERMIT ∧ r.consent_id = e.consent_id
    ∧ r.context_hash = e.context_hash ∧ r.committed_before(e.execution_start)

// INV-02: Fail-Closed
∀ component c ∈ {EG, PE, CR, AL, AE}:
  failure(c) → ∀ requests: decision = DENY

// INV-03: Revocation Propagation Bound
```

```
∀ revocation event rev committed at time t:
  ∀ EG instances i: i.cache.state(rev.consent_id) = REVOKED
    within [t, t + TTL]

// INV-04: Artifact Binding
∀ inference execution e:
  e.artifact.context_hash = e.context_hash
    ∧ e.artifact.consent_id = e.consent_id
    ∧ e.artifact.policy_version = e.policy_version
    ∧ verify_hmac(e.artifact) = TRUE

// INV-05: Ledger Monotonicity
∀ records r_i, r_j ∈ AL: i < j → r_i.committed_at < r_j.committed_at
  ∧ r_j.prev_hash = sha256(r_i)

// INV-06: Policy Immutability
∀ policy version pv marked FROZEN:
  ¬∃ mutation m : m.target = pv ∧ m.committed_at > pv.frozen_at

// INV-07: Evaluation Determinism
∀ T, H, P: E(T,H,P) at time t_1 = E(T,H,P) at time t_2
  where policy_version(P) is FROZEN and T.state is identical at t_1 and t_2
```

SECTION 2

System Architecture Blueprint

A conformant QODIQA enforcement infrastructure requires exactly seven canonical components. Each component has a strictly defined responsibility domain. No component may assume responsibility for a domain assigned to another without

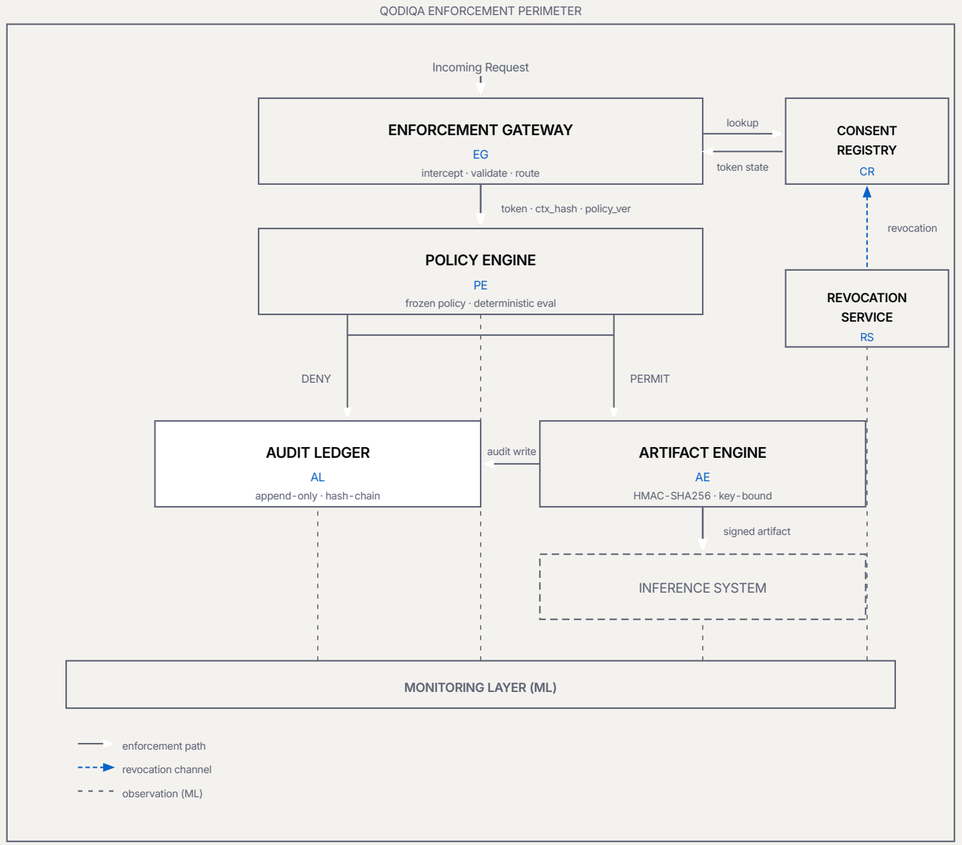
explicit interface delegation. The architecture is minimal by design: every component present is necessary; no component present is optional on the enforcement path.

2.1 Component Inventory

TABLE 2.1: CANONICAL COMPONENT DEFINITIONS

COMPONENT	DESIGNATION	PRIMARY RESPONSIBILITY	FAILURE MODE
Enforcement Gateway	EG	Request interception; synchronous block or pass based on Policy Engine decision	Fail-closed
Policy Engine	PE	Deterministic evaluation of consent tokens against versioned, frozen policy rules	Fail-closed
Consent Registry	CR	Authoritative storage and retrieval of consent token state	Read unavailable ↳ deny
Revocation Service	RS	Propagation of revocation events to all enforcement surfaces within TTL bounds	Propagation failure ↳ stale-deny
Audit Ledger	AL	Append-only, tamper-evident record of all enforcement decisions	Write unavailable ↳ block execution
Artifact Engine	AE	Generation and verification of cryptographic consent artifacts attached to inference outputs	Generation failure ↳ suppress output
Monitoring Layer	ML	Real-time metric collection, alert dispatch, and control health verification	Degraded; non-blocking

2.2 Canonical Architecture Diagram



SEVEN CANONICAL COMPONENTS ARRANGED IN A VERTICAL ENFORCEMENT FLOW. THE ENFORCEMENT GATEWAY INTERCEPTS ALL REQUESTS AND COORDINATES WITH THE CONSENT REGISTRY AND POLICY ENGINE. ON A PERMIT DECISION, THE ARTIFACT ENGINE GENERATES A CRYPTOGRAPHIC ARTIFACT; BOTH THE DECISION AND ARTIFACT ARE COMMITTED TO THE AUDIT LEDGER BEFORE THE INFERENCE SYSTEM IS RELEASED. THE MONITORING LAYER OBSERVES ALL COMPONENTS WITHOUT PARTICIPATING IN THE ENFORCEMENT PATH.

2.3 Data Flow Model

Every enforcement-bound request traverses a defined sequence of component interactions. The sequence is invariant: no step may be skipped, reordered, or substituted. The data flow model formalizes these interactions as a directed acyclic graph with a single terminal condition per request.

2.4 Failure Isolation

Each component failure mode is defined and bounded. No component failure may produce a PERMIT outcome. Component failures are classified into two categories: blocking failures, which halt the enforcement path and deny all requests; and degraded failures, which reduce observability without affecting enforcement decisions. The Monitoring Layer is the only component with a degraded failure mode; all enforcement-path components have blocking failure modes.

2.5 Failure Cascade Simulation

The following failure cascade simulations define the required system behavior under specific multi-component failure scenarios. Each cascade must be validated during conformance testing. Failure cascade FC-01 through FC-05 are defined below.

FAILURE CASCADE · FC-01: CONSENT REGISTRY UNAVAILABILITY CASCADE

```
// FC-01: CR becomes unavailable during active request processing

State[0]: REQUEST RECEIVED
  EG has validated request fields; consent_id present and well-formed
  EG issues CR query: GET /token/{consent_id}

State[1]: CR TIMEOUT
  CR does not respond within T_CR_ceiling (30ms default)
  EG circuit breaker records failure count +1

State[2]: CIRCUIT EVALUATION
  If failure_count < N (threshold): single-request fail-closed
    → decision = DENY; reason = REGISTRY_UNAVAILABLE
    → AL.append(DENY record)
    → return DENY to caller
  If failure_count ≥ N within window W:
    → circuit OPENS; all subsequent requests denied without CR query
    → ML alert dispatched: CRITICAL: CR circuit breaker OPEN

State[3]: CIRCUIT OPEN BEHAVIOR
  All requests: decision = DENY; reason = CIRCUIT_OPEN
  AL.append(DENY record) for each request
  No inference execution is permitted during circuit-open state
  No requests are queued or deferred; all are immediately denied

State[4]: RECOVERY
  Circuit enters half-open after recovery interval
  Single probe request issued to CR
  If probe succeeds: circuit CLOSES; normal processing resumes
  If probe fails: circuit remains OPEN; recovery interval resets
```

FAILURE CASCADE · FC-02: AUDIT LEDGER WRITE FAILURE — CASCADE

```
// FC-02: AL write fails after PERMIT decision

State[0]: PERMIT DECISION REACHED
  PE evaluated E(T,H,P) = PERMIT
  AE generated artifact; artifact is valid
  EG issues AL write: APPEND enforcement_record

State[1]: AL WRITE TIMEOUT OR REJECTION
  AL does not acknowledge write within T_AL_ceiling (20ms)
  OR AL returns error code

State[2]: MANDATORY RESPONSE
  EG MUST NOT release request to inference system
  Decision is BLOCKED; not PERMIT; not DENY
  EG returns HTTP 503 to caller
  AL.append(AUDIT_WRITE_FAILURE record) if AL recovers within 5ms
  ML alert dispatched: CRITICAL: AL write failure

State[3]: RETRY BOUNDARY
  One retry attempt permitted within T_AL_ceiling + 5ms
  If retry succeeds: AL write acknowledged; normal PERMIT path continues
  If retry fails: request is terminated; DENY returned to caller

// Invariant: No inference execution begins without committed AL record.
// A PERMIT that cannot be durably recorded is not a valid PERMIT.
```

FAILURE CASCADE · FC-03: REVOCATION PROPAGATION DELAY — CASCADE

```
// FC-03: Revocation event not acknowledged by all EG instances within TTL/2
```

```
State[0]: REVOCATION COMMITTED
  RS receives revocation event for consent_id C at time T_rev
  RS begins propagation to all registered EG instances

State[1]: PROPAGATION MONITORING
  At time T_rev + TTL/2:
    RS checks acknowledgment receipts from all EG instances
    If all acknowledged: propagation complete; no escalation
    If any EG instance unacknowledged: FC-03 condition

State[2]: CONSERVATIVE MODE ACTIVATION
  For unacknowledged EG instance(s):
    All requests referencing consent_id C: DENY
    Reason: REVOCATION_PENDING_CONSERVATIVE
    AL records: DENY with reason=CONSERVATIVE_REVOCATION_MODE
    Conservative mode cannot be lifted by automated system alone

// Recovery Invariant:
  Conservative mode exit requires: cross-region sync confirmed AND
  EG instance acknowledges revocation event AND
  human operator sign-off on propagation resolution log

// FC-03c: Clock Skew Edge Cases
  Token expiry is evaluated against the EG's local monotonic clock.
  If EG clock skew relative to CR-issued timestamps exceeds
  the operator-configured skew_tolerance (max: 30 seconds):
    EG must treat all timestamp comparisons as unreliable
    EG must deny any request where token.expires_at is within
    [now - skew_tolerance, now + skew_tolerance]
  Clock skew exceeding skew_tolerance triggers ML alert: TIME_SKEW_EXCEEDED
  No request may be permitted on the basis of a timestamp comparison
  when clock skew is unresolved.
```

FAILURE CASCADE · FC-04: CROSS-REGION CONSENSUS FAILURE — CASCADE

```
// FC-04: Multi-region CR consensus disrupted; CR state divergence between regions

State[0]: DIVERGENCE DETECTED
  Region A CR and Region B CR report different states for consent_id C
  Divergence detected by RS cross-region consistency check

State[1]: CONSERVATIVE MODE: ALL REGIONS
  All requests for affected consent_ids: DENY with REVOCATION_PENDING
  AL records: DENY with reason=CONSERVATIVE_REVOCATION_MODE
  Conservative mode cannot be lifted by automated system alone

// Recovery Invariant:
  Conservative mode exit requires: cross-region sync confirmed AND
  Region B CR hash matches Region A CR hash for affected range AND
  human operator sign-off on divergence resolution log

// FC-04c: Clock Skew Edge Cases

  Token expiry is evaluated against the EG's local monotonic clock.
  If EG clock skew relative to CR-issued timestamps exceeds
  the operator-configured skew_tolerance (max: 30 seconds):
    EG must treat all timestamp comparisons as unreliable
    EG must deny any request where token.expires_at is within
    [now - skew_tolerance, now + skew_tolerance]
  Clock skew exceeding skew_tolerance triggers ML alert: TIME_SKEW_EXCEEDED
  No request may be permitted on the basis of a timestamp comparison
  when clock skew is unresolved.
```

FAILURE CASCADE · FC-05: ARTIFACT MISMATCH PROPAGATION — CASCADE

```
// FC-05: Inference system receives artifact that fails verification

State[0]: ARTIFACT DELIVERED
  AE generated artifact A for request R; AL record committed
  EG released request to inference system with artifact attached

State[1]: INFERENCE SYSTEM ARTIFACT VERIFICATION
  Inference system verifies HMAC of artifact A
  Verification outcome: FAIL
    Possible causes:
      (a) Key rotation in progress; artifact signed with outgoing key;
          inference system only loaded incoming key
      (b) Artifact corruption in transit
      (c) Artifact replay from different request context (context_hash mismatch)
      (d) AE signing key revoked or expired

State[2]: INFERENCE SYSTEM RESPONSE
  Inference system must refuse execution regardless of PERMIT in artifact
  Inference system emits: ARTIFACT_VERIFY_FAIL{artifact_id=A.id, cause=detected}
  Request is terminated; no inference computation begins
  Inference system returns error to upstream caller

State[3]: EG DETECTION AND RESPONSE
  EG receives ARTIFACT_VERIFY_FAIL event from inference system
  EG emits: ARTIFACT_MISMATCH{request_id=R, artifact_id=A.id}
  ML alert dispatched: CRITICAL: artifact integrity failure
  If cause = (a): verify key rotation window; resume if within window
  If cause = (b): infrastructure integrity concern; halt and investigate
  If cause = (c): replay attack suspected; suspend consent_id; full audit
  If cause = (d): AE key management failure; halt AE; human review

// Non-Recovery Conditions:
  Cause (c) confirmed replay: consent_id suspended; full AL audit required;
```

```
    human authorization required to restore consent_id processing
  Cause (d) confirmed: AE must not restart until key hierarchy verified by
    out-of-band inspection; all artifacts from affected AE instance re-verified
```

SECTION 3

Deployment Topologies

Six deployment topologies are defined. Each is a complete, conformant configuration of the seven canonical components. Topologies differ in where components are hosted, which trust boundaries they cross, and what revocation propagation guarantees are achievable. Operators must declare their topology and document the trust assumptions and revocation TTL applicable to that topology.

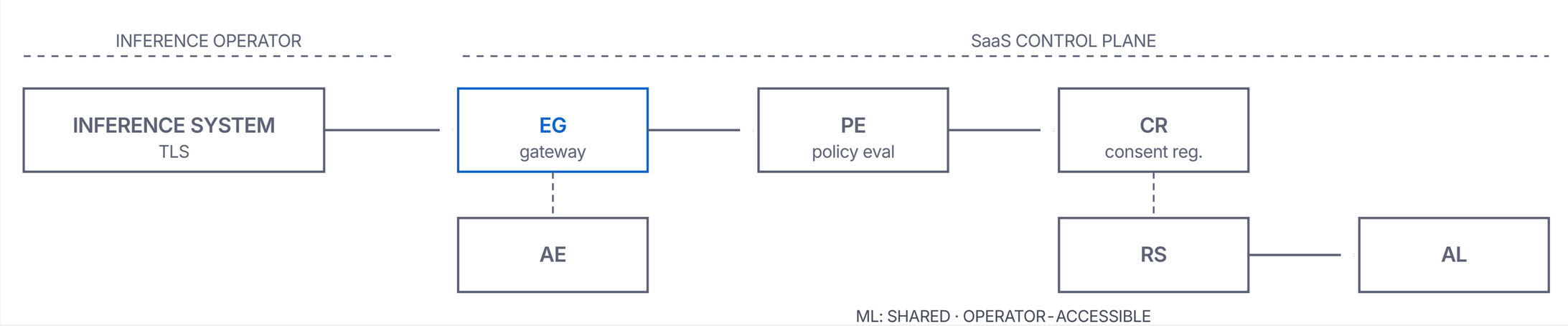
> The enforcement architecture is designed to operate without requiring modification of underlying AI models. Enforcement occurs at the inference boundary through externalized policy evaluation and consent verification, enabling deployment without altering model internals, training pipelines, or inference logic.

All seven canonical components (Enforcement Gateway, Policy Engine, Consent Registry, Revocation Service, Audit Ledger, Artifact Engine, and Monitoring Layer) operate as infrastructure surrounding the inference system. The inference system is a passive recipient of enforcement decisions: it receives a signed consent artifact authorizing execution or a denial signal terminating the request. No changes to model weights, serving code, or runtime configuration of the inference system are required or permitted as substitutes for proper enforcement component deployment. This separation is architecturally intentional and must be preserved across all six deployment topologies defined in this section.

3.1 SaaS Control Plane

All seven components are operated by the enforcement provider and exposed to the inference system operator over authenticated HTTPS APIs. The inference operator is the policy author; the SaaS operator manages component availability and enforces SLA commitments against TTL bounds.

TOPOLOGY 3.1 **SAAS CONTROL PLANE**



LATENCY OVERHEAD
5–40ms round-trip; dominated by geographic distance to SaaS EG

TRUST ASSUMPTION
Operator trusts SaaS to enforce faithfully; AL must be independently verifiable by operator

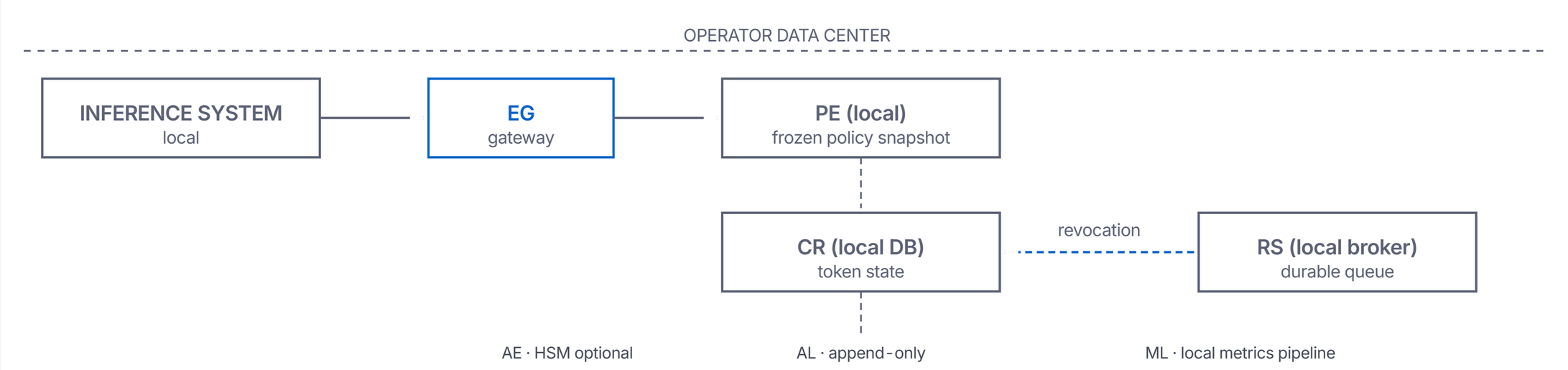
REVOCATION PROPAGATION
Internal to SaaS plane; operator revocation call receives propagation confirmation within TTL

FAILURE SCENARIO
SaaS unavailability triggers local circuit breaker at inference system; all requests denied

3.2 On-Premises Gateway

All seven components are deployed within the operator's own infrastructure. No enforcement traffic traverses an external network boundary. Required where data residency constraints prohibit external control plane communication.

TOPOLOGY 3.2 **ON-PREMISES GATEWAY**



LATENCY OVERHEAD
1–5ms; intra-host or intra-datacenter; no WAN overhead

TRUST ASSUMPTION
Operator controls full stack; no external dependency on enforcement path

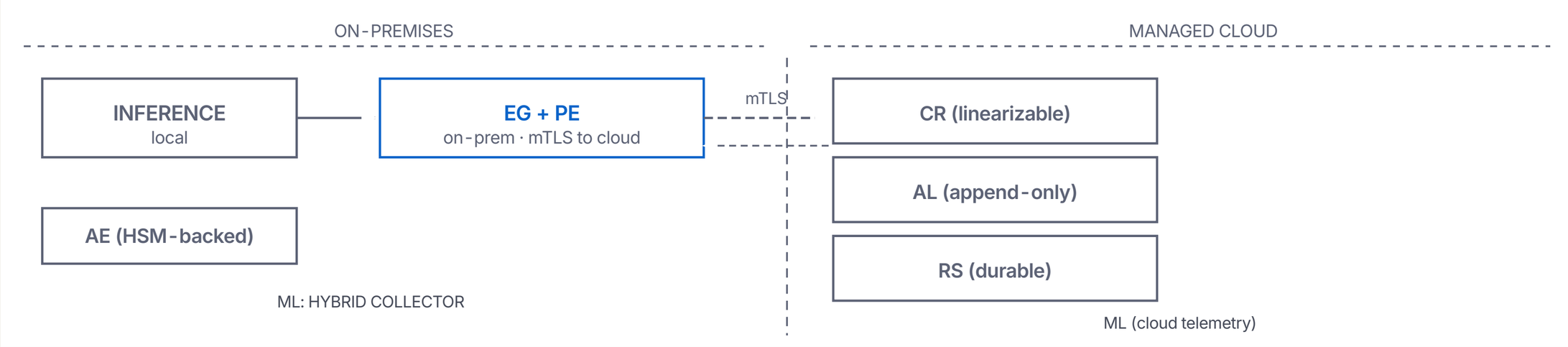
REVOCATION PROPAGATION
RS as internal durable queue; revocation lag ≤ defined TTL; replication lag must not exceed TTL

FAILURE SCENARIO
Single-datacenter failure; HA requires CR and AL replicated across ≥ 2 independent failure domains

3.3 Hybrid Deployment

EG and PE are deployed on-premises. CR, RS, and AL are operated in a managed cloud environment with strong consistency guarantees. AE may be co-located with EG or with the cloud boundary.

TOPOLOGY 3.3 **HYBRID DEPLOYMENT**



LATENCY OVERHEAD
WAN latency for CR reads is dominant; local read-cache with TTL ≤ revocation propagation bound permitted

TRUST ASSUMPTION
On-prem EG must verify every CR and AL response cryptographically; unverifiable responses treated as unavailable

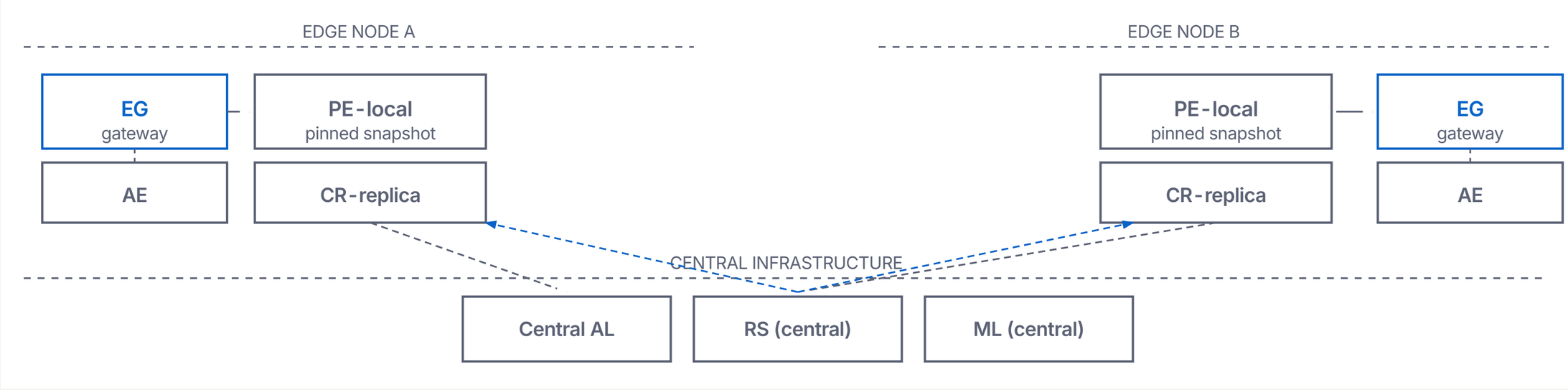
REVOCATION PROPAGATION
Cloud RS propagates to on-prem EG; cache must invalidate synchronously on revocation event receipt

FAILURE SCENARIO
WAN loss triggers fail-closed on CR queries; circuit breaker opens; all requests denied until connectivity restored

3.4 Edge Enforcement

EG instances are deployed at the network edge, physically proximate to inference compute. PE operates with a locally cached, version-pinned policy snapshot. CR is read-replicated to the edge with revocation invalidation. AL writes propagate to a central ledger store with bounded durability acknowledgment.

TOPOLOGY 3.4 **EDGE ENFORCEMENT**



REVOCATION PROPAGATION
RS pushes to all edge EG within TTL; unacknowledged at TTL/2 → conservative mode: affected consent_id denied

FAILURE SCENARIO
Edge isolation: default is deny all; operator may configure cached-state enforcement with documented TTL bounds

TRUST ASSUMPTION
Central AL is authoritative; edge AL writes are provisional until central confirmation

LATENCY OVERHEAD
Local PE and CR-replica: <5ms enforcement; AL write to central adds 20–60ms depending on WAN

3.5 Multi-Region Deployment

All seven components are replicated across geographic regions. CR is operated as a globally consistent store with linearizable read semantics for the revocation flag. AL is replicated with region-level durability. PE uses a globally synchronized, frozen policy snapshot identical across all regions.

Cross-region revocation propagation must complete within the defined TTL regardless of inter-region network conditions. RS must maintain a propagation record per region per revocation event. Unconfirmed propagation after TTL/2 triggers escalation. **Eventual consistency is not acceptable for revocation state under any topology.**

3.6 Air-Gapped Environment

All components operate within a network boundary with no external connectivity. Policy updates, external revocation events, and ledger exports are transmitted via authenticated offline media under defined transfer protocols. The RS operates as a local broadcast bus; externally-originated revocation events arrive as signed bundles on authenticated media.

> Air-gapped deployments must not process requests referencing consent tokens whose last-known state is older than the operator-documented revocation propagation bound, unless the token was issued within the current air-gap session with a locally resolvable issuer. **The effective revocation propagation bound equals the media transfer schedule interval and must be disclosed to all affected parties.**

SECTION 4

API Integration Model

Five integration patterns are defined for embedding QODIQA enforcement into existing systems. Each pattern addresses a different placement of the enforcement call relative to the inference system's architecture. All patterns are equivalent in enforcement semantics; they differ only in where the EG function is invoked and how enforcement metadata is transported.

4.1 Preflight Decision Call Pattern

The calling system issues a blocking pre-check before initiating inference computation. The caller presents a consent token reference and a frozen context hash; EG returns a deterministic permit or deny before any inference work begins.

POST /V1/ENFORCE/PREFLIGHT — JSON

```
// Request
{
  "consent_id":     "cns_a1b2c3d4e5f6",
  "context_hash":   "sha256:9f86d081884c7d659a2feaa0c55ad015",
  "policy_version": "pv_2026-03-01-frozen",
  "caller_id":      "svc_inference_primary",
  "request_nonce":  "nonce_0f1e2d3c4b5a"
}

// Response: PERMIT
{
  "decision":       "PERMIT",
  "artifact_id":    "art_7a8b9c0d1e2f",
  "audit_ref":      "aud_3c4d5e6f7a8b",
  "policy_version": "pv_2026-03-01-frozen",
```

```
  "expires_at":     "2026-03-01T12:05:00Z"
}

// Response: DENY
{
  "decision":       "DENY",
  "reason_code":    "TOKEN_REVOKED",
  "audit_ref":      "aud_9b8a7c6d5e4f",
  "policy_version": "pv_2026-03-01-frozen"
}
```

4.2 Inline Gateway Interception

EG is deployed as a transparent network proxy or sidecar. All traffic to the inference endpoint passes through EG without requiring application-layer modification. Enforcement metadata is carried in defined HTTP headers.

INLINE GATEWAY: REQUIRED HEADERS — HTTP

```
// Inbound: required on all inference-bound requests
X-QODIQA-Consent-ID:      cns_a1b2c3d4e5f6
X-QODIQA-Context-Hash:    sha256:9f86d081884c7d659a2feaa0c55ad015
X-QODIQA-Policy-Version:  pv_2026-03-01-frozen
X-QODIQA-Request-Nonce:   nonce_0f1e2d3c4b5a

// EG forwards on PERMIT with artifact headers
X-QODIQA-Artifact-ID:     art_7a8b9c0d1e2f
X-QODIQA-Audit-Ref:       aud_3c4d5e6f7a8b

// EG returns HTTP 403 on DENY
{
  "error": "QODIQA_DENY",
  "reason_code": "POLICY_NOT_SATISFIED",
```

```
  "audit_ref": "aud_9b8a7c6d5e4f"
}
```

4.3 SDK Wrapper Pattern

The SDK wraps the inference system's native call interface. Application code calls the SDK's enforcement-aware wrapper; the SDK handles preflight, artifact attachment, and audit propagation transparently. The inference system's native interface is never called directly by application code in a conformant deployment.

SDK WRAPPER: PSEUDOCODE — PSEUDO

```
function enforced_infer(consent_id, context, payload):
    ctx_hash = sha256_canonical(freeze(context))
    decision = qodiqa_client.preflight(
        consent_id, ctx_hash, POLICY_VERSION
    )
    if decision.status ≠ "PERMIT":
        raise QodiqaDenyError(decision.reason_code, decision.audit_ref)

    result = inference_backend.call(
        payload,
        artifact_id=decision.artifact_id
    )
    qodiqa_client.emit_audit(
        decision.audit_ref,
        outcome="EXECUTED",
        output_hash=sha256_canonical(result)
    )
    return result
```

4.4 Service Mesh Filter Pattern

An Envoy or compatible L7 filter intercepts inference-bound traffic at the sidecar proxy layer. The filter implements the EG function inline, calling PE and CR via gRPC before forwarding or rejecting the request.

SERVICE MESH FILTER: LOGICAL CONFIGURATION — YAML

```yaml
filter:
  name:  qodiqa.enforcement.v1
  type:  HTTP
  phase: REQUEST
  rules:
    - match:
        path_prefix: /v1/inference/
      action:
        call_external:
          endpoint:    qodiqa-pe.enforcement.svc:9090
          method:      POST /evaluate
          timeout_ms:  50
          on_timeout:  DENY
          on_error:    DENY
          forward_headers:
            - X-QODIQA-Consent-ID
            - X-QODIQA-Context-Hash
            - X-QODIQA-Policy-Version
```

4.5 Agent-Based Orchestration Hook

In multi-agent orchestration systems, enforcement is applied at the point where an individual agent receives a task context and prepares to invoke an inference operation. Each agent call requires an independent enforcement decision; parent-to-subagent delegation does not propagate PERMIT decisions.

> **Non-Transitivity Requirement:** In multi-agent systems, enforcement decisions are not transitive. A PERMIT issued for Agent A's call does not authorize Agent A to forward that context to Agent B. Agent B requires an independent PERMIT for its own call, using its own consent reference and its own frozen context hash.

4.6 Artifact Propagation and Error Taxonomy

CONSENT ARTIFACT: PROPAGATION HEADER FIELDS — HTTP

```
X-QODIQA-Artifact-ID:       art_7a8b9c0d1e2f
X-QODIQA-Artifact-Sig:      base64url(HMAC-SHA256(payload, signing_key))
X-QODIQA-Artifact-Issued:   2026-03-01T12:00:00Z
X-QODIQA-Artifact-Expires:  2026-03-01T12:05:00Z
X-QODIQA-Policy-Binding:    pv_2026-03-01-frozen
X-QODIQA-Consent-Ref:       cns_a1b2c3d4e5f6
```

TABLE 4.1: ERROR TAXONOMY

REASON CODE	HTTP	CONDITION	REQUIRED ACTION
`TOKEN_NOT_FOUND`	403	consent_id absent from CR	Deny; audit
`TOKEN_REVOKED`	403	Revocation flag set	Deny; audit
`TOKEN_EXPIRED`	403	Token TTL exceeded	Deny; audit
`POLICY_NOT_SATISFIED`	403	Token active; rule not met	Deny; audit
`CONTEXT_HASH_MISMATCH`	403	Hash does not match registered context	Deny; audit; alert

REASON CODE	HTTP	CONDITION	REQUIRED ACTION
`REGISTRY_UNAVAILABLE`	503	CR unreachable or timeout	Fail-closed deny; audit
`AUDIT_WRITE_FAILURE`	503	AL write did not complete	Block execution; alert
`ARTIFACT_GENERATION_FAILURE`	503	AE could not produce artifact	Suppress output; audit
`ENFORCEMENT_TIMEOUT`	503	Path exceeded timeout ceiling	Fail-closed deny; TIMEOUT_DENY audit
`POLICY_VERSION_UNKNOWN`	400	Requested version not available in PE	Deny; notify caller

SECTION 5

Enforcement Lifecycle: Operational Detail

The enforcement lifecycle comprises seven sequential steps. No step may be skipped. No step may be reordered. The lifecycle is synchronous: execution of the inference request is gated on completion of all seven steps in the defined order.

Step 01	**Intent Intake** EG receives request; validates presence and syntax of `consent_id`, `context_hash`, `policy_version`, `request_nonce`. Missing fields result in immediate deny without downstream calls.	Ceiling: 2ms
Step 02	**Token Verification** EG queries CR with `consent_id`. CR returns token record: state, issuance and expiry timestamps, revocation flag, issuer identifier. EG validates token has not expired and revocation flag is clear.	Ceiling: 30ms

Step 03	**Registry Resolution** EG confirms the token's issuer is recognized under the declared policy version. Unrecognized issuers deny without PE evaluation.	In-process
Step 04	**Policy Evaluation** EG passes `{token_record, context_hash, policy_version}` to PE. PE loads frozen policy snapshot, evaluates all applicable rules, returns deterministic `PERMIT` or `DENY`. No dynamic rule loading during evaluation.	Ceiling: 10ms
Step 05	**Artifact Generation** On `PERMIT`: AE receives `{consent_id, context_hash, policy_version, decision_timestamp}`. AE generates HMAC-SHA256 signed artifact. Timeout or signing failure → suppress request.	Ceiling: 5ms
Step 06	**Audit Write** AL receives audit record for every decision (both PERMIT and DENY). Write must be acknowledged before enforcement decision is returned to caller. Write failure → deny execution; emit `AUDIT_WRITE_FAILURE` alert.	Ceiling: 20ms
Step 07	**Execution Release** Only after confirmed AL write and valid AE artifact does EG release the request to the inference system. Inference system must verify artifact presence before beginning computation. Missing or malformed artifacts cause refusal.	Path ceiling: 75ms

5.1 Circuit Breaker Behavior

Each downstream component call from EG is wrapped in an independent circuit breaker. State transitions: Closed to Open after N consecutive failures within rolling window W (N ∈ [3,10]; W ∈ [10s,60s], operator-configurable within bounds). Open state: all requests that would have called the failed component are immediately denied. Open to Half-open: probe request after recovery interval; success closes circuit, failure resets interval.

> **Fail-Closed Requirement:** An open circuit breaker state must never be interpreted as a PERMIT. An open circuit on any enforcement-path component means enforcement cannot be completed. All requests must be denied for the duration of the open state, without exception.

SECTION 6

Performance and Scalability Model

The enforcement path is designed for synchronous, per-request evaluation. The throughput ceiling is determined by the slowest component under load. Optimization is permitted where it does not compromise determinism or fail-closed behavior. Caching is permitted under strict constraints; it does not relax enforcement. All performance specifications in this section are structural bounds derived from component interaction topology, not empirical benchmarks. Operators must not treat these as performance commitments; they must measure and bound their own deployments against these structural models.

6.1 Latency Envelope: Bounded Overhead Model

The enforcement path overhead is the sum of latencies of all synchronous component calls on the critical path. The critical path is the sequence of calls that cannot be parallelized without violating the enforcement ordering invariant. The bounded overhead model is expressed as:

LATENCY BOUND MODEL: CRITICAL PATH — FORMAL

```
// Critical path: sequential, non-parallelizable
L_enforcement = L_CR + L_PE + L_AE + L_AL + L_EG_overhead
```

```
// Where each component latency is bounded:
L_CR  ≤ T_CR_ceiling   (default: 30ms; network-dominated)
L_PE  ≤ T_PE_ceiling   (default: 10ms; CPU-dominated)
L_AE  ≤ T_AE_ceiling   (default:  5ms; crypto-dominated)
L_AL  ≤ T_AL_ceiling   (default: 20ms; durability-dominated)
L_EG_overhead ≤ 3ms    (routing, extraction, circuit check)

// Therefore:
L_enforcement ≤ T_CR + T_PE + T_AE + T_AL + 3ms
              ≤ 30 + 10 + 5 + 20 + 3 = 68ms (no-cache upper bound)

// With CR read cache (warm, valid TTL):
L_enforcement_cached ≤ T_cache_hit + T_PE + T_AE + T_AL + 3ms
                     ≤ 1 + 10 + 5 + 20 + 3 = 39ms

// Hash computation cost, context_hash generation (caller-side):
C_hash(n) = O(n)   where n = byte length of serialized context object
            n is bounded by operator-defined context_max_bytes
            All hash computations must complete in O(n) time;
            no hash function with super-linear behavior is conformant.

// Per-request overhead as a fraction of inference latency:
If L_inference = duration of inference computation
  QODIQA_overhead_fraction = L_enforcement / (L_enforcement + L_inference)
  For L_inference ≥ 200ms: overhead ≤ 25% (no-cache)
  For L_inference ≥ 500ms: overhead ≤ 12% (no-cache)
  Operators with L_inference < 100ms must use cached enforcement path.
```

6.2 Throughput Envelope

TABLE 6.1: COMPONENT THROUGHPUT REFERENCE

COMPONENT	THROUGHPUT (PER INSTANCE)	SCALING MODEL	BOTTLENECK
EG	5,000–15,000 req/s	Horizontal (stateless)	Downstream call latency
PE	10,000–50,000 eval/s	Horizontal (stateless; pinned policy)	Policy ruleset size
CR reads	20,000–100,000/s	Read replicas	Consistency model
CR writes	2,000–10,000/s	Single-primary writes	Linearizability req.
AL	5,000–20,000 append/s	Horizontal partitioned	Durability guarantee
AE (software)	20,000–100,000 sig/s	Horizontal	CPU (HMAC-SHA256)
AE (HSM)	1,000–5,000 sig/s per partition	HSM partition scale-out	HSM transaction limit

Throughput Saturation Model: System throughput ceiling is min(EG_throughput, CR_throughput, AL_throughput). Saturation at any single component does not cause graceful degradation; it causes enforcement latency to exceed ceiling thresholds, triggering fail-closed behavior across all downstream requests. Operators must provision each component such that peak request rate does not exceed 70% of the component's single-instance throughput ceiling, to preserve headroom for revocation bursts and audit write spikes.

HORIZONTAL SCALING INVARIANTS — FORMAL

```
// EG scaling: stateless, linear throughput growth
throughput(EG, n) = n × throughput(EG, 1)
  Invariant: Each EG instance is independent; no shared mutable state
```

```
  Load balancer must not route requests to EG instance in circuit-open state

// PE scaling: stateless, linear, policy-snapshot pinned
throughput(PE, n) = n × throughput(PE, 1)
  Invariant: All PE instances must hold the identical frozen policy snapshot
  Policy version deployment must be atomic across all PE instances;
  partial deployment (some PE at v_n, some at v_n+1) is non-conformant

// CR scaling: read-replicas permitted; write linearizability required
read_throughput(CR, n_replicas) ≤ n_replicas × read_throughput(CR, 1)
  Invariant: Revocation flag read must be linearizable; stale-read replicas
  are not acceptable for revocation state without TTL-bounded cache protocol
write_throughput(CR) is bounded by single-primary commit rate
  Adding replicas does not increase write throughput

// AL scaling: partitioned append permitted; addressability invariant holds
throughput(AL, n_partitions) ≤ n_partitions × throughput(AL, 1)
  Invariant: All audit records for a given consent_id must be retrievable
  from a single query interface regardless of partition count
  Hash chain integrity must be verifiable per partition independently

// Cache invalidation determinism:
  cache.invalidate(consent_id) must complete before cache.read(consent_id)
  returns the new state for any subsequent request.
  Invalidation is synchronous. There is no "eventually consistent"
  cache invalidation in a conformant deployment.
  Cost: O(1) per invalidation event per EG instance.
  Total invalidation cost per revocation event:
    O(n_EG_instances) distributed; must complete within TTL.
```

6.3 Per-Request Overhead

WITHOUT CACHING	WITH CR READ CACHE (WARM)
• CR read: 15–30ms (dominant)	→ CR cache hit: <1ms
• PE eval: <1ms (local)	→ PE eval: <1ms (local instance)
• AE sign: 1–5ms (software)	→ AE sign: 1–5ms (software)
• AL write: 10–20ms	→ AL write: 10–20ms
Total: 40–75ms per request	Total: 10–25ms per request

6.4 Caching Constraints

Caching of CR token state is permitted only under three conditions simultaneously: cache entries have a TTL no greater than the configured revocation propagation bound; the cache is invalidated synchronously upon receipt of a revocation event for the cached token; and cache entries for tokens in any state other than `ACTIVE` must not be cached. PE policy snapshots may be cached indefinitely once frozen; no cache invalidation is required for version-bound PE cache entries.

6.5 Revocation TTL Implications

A shorter revocation TTL reduces the window during which a revoked token might be honored; it increases pressure on CR read consistency and RS propagation infrastructure. Operators must document the deployed revocation TTL and its measurement basis. The TTL is a commitment to affected parties: it defines the maximum exposure window after a revocation event is committed.

6.6 Horizontal Scaling Model

EG and PE are stateless and scale horizontally without coordination. CR scale-out must not relax linearizability for the revocation flag. AL partitioning must ensure all audit records for a given `consent_id` are addressable from a single query interface. AE scales horizontally in software mode; HSM-backed AE must be over-provisioned relative to peak request rate by a factor of at least 2× to account for HSM transaction jitter.

SECTION 7

Security Hardening Guidelines

7.1 Key Management Model

All cryptographic operations in the enforcement path use keys managed through a three-tier hierarchy. The Root Key is offline, stored in hardware, used only to sign intermediate keys. The Intermediate Signing Key has a maximum 90-day lifetime and derives per-deployment signing keys. The Deployment Signing Key has a maximum 30-day lifetime and is used by AE for artifact HMAC generation.

Key rotation must not interrupt in-flight enforcement; dual-key windows are required during rotation, with both outgoing and incoming keys accepted for artifact verification for the duration of the rotation window. The rotation window must be bounded: its minimum duration is determined by the maximum in-flight request lifetime, defined as the enforcement ceiling (75ms) plus the inference system's maximum processing time. Its maximum duration must not exceed 24 hours. Key rotation events are recorded in AL under record type `KEY_ROTATION_EVENT` with the outgoing key fingerprint, incoming key fingerprint, and rotation window boundaries.

KEY ROTATION INVARIANTS — INVARIANTS

```
// KR-INV-01: Dual-key window coverage
∀ artifact A generated at time T during rotation window [T_start, T_end]:
  verify(A, key_outgoing) = TRUE ∨ verify(A, key_incoming) = TRUE

// KR-INV-02: Window boundaries
T_end - T_start ≥ max_in_flight_lifetime (≥ 75ms + max_inference_time)
T_end - T_start ≤ 86400 seconds (24 hours hard ceiling)

// KR-INV-03: Key retirement
After T_end: key_outgoing is no longer accepted for new artifact verification
Before T_start: key_incoming is not accepted for any artifact verification

// KR-INV-04: Audit binding
∀ key rotation event:
  AL.append({type=KEY_ROTATION_EVENT, outgoing_fp, incoming_fp,
             T_start, T_end, authorized_by, root_sig_of_incoming})

// KR-INV-05: No silent rotation
Key rotation without prior AL record is non-conformant.
AE must not accept a new key that has no corresponding AL record.
```

7.2 Revocation Registry Protection

Write access to the CR revocation flag is restricted to the RS component exclusively. No inference system, application layer, or human operator may set or clear the revocation flag directly. All revocation state changes are logged in AL as a separate record type with issuer attribution. CR must implement write-ahead logging for revocation flag updates; a revocation that is not durably written must not be considered committed.

The write path to the CR revocation flag must be isolated by an authenticated, non-delegable interface. RS must hold the sole cryptographic credential authorized to write to the revocation flag endpoint. This credential must not be derivable

from any other credential in the system and must be rotated independently of the AE signing key hierarchy. The revocation write credential must be stored in hardware if available.

7.3 Ledger Integrity and Tamper-Evidence

AL is an append-only store. Deletion and modification operations are disabled at the storage layer. Ledger integrity is verified through a hash chain: each record includes the SHA-256 hash of the previous record in the same partition. The hash chain construction is as follows: for record r_n, `r_n.chain_hash = SHA-256(r_(n-1).chain_hash ‖ r_n.content_hash)`, where `content_hash = SHA-256(serialize_canonical(r_n.payload))`. The separator ‖ denotes fixed-length concatenation with no ambiguity boundary.

LEDGER TAMPER-EVIDENCE PROOF LOGIC — INVARIANTS

```
// AL-INV-01: Hash chain continuity
∀ records r_n, r_(n+1) ∈ AL partition P:
  r_(n+1).prev_hash = SHA-256(serialize_canonical(r_n))

// AL-INV-02: Tamper detection procedure
function verify_chain(partition P, range [i, j]):
  computed_hash = hash(r_i)
  for k = i+1 to j:
    if r_k.prev_hash ≠ computed_hash: return CHAIN_BREAK at position k
    computed_hash = hash(r_k)
  return CHAIN_VALID

// AL-INV-03: Export integrity
∀ AL export E:
  E.signature = sign(AL_signing_key, SHA-256(E.content))
  Verifier: ∀ imported exports E: verify(AL_public_key, E.signature) = TRUE

// AL-INV-04: Deletion prohibition · enforced at storage layer
```

```
∀ storage operation op on AL:
  op.type ∈ {APPEND} is the only permitted operation
  op.type ∈ {DELETE, UPDATE, TRUNCATE} must be rejected with error code
    OPERATION_PROHIBITED and emitted as ML alert

// AL-INV-05: Integrity heartbeat
  AL verifier runs chain verification on every newly appended record
  at Level 3 (High-Assurance); at minimum weekly at Level 1
  Verification latency: O(records_since_last_verification)
  If verification fails: AL transitions to READ_ONLY mode; writes suspended;
  ML alert: CRITICAL: LEDGER_CHAIN_BREAK; human intervention required
```

7.4 Secure Enclave Boundary Isolation

Where PE and AE are deployed within hardware secure enclaves (Intel TDX, AMD SEV-SNP, or equivalent), the following boundary isolation requirements apply. The enclave boundary must encompass the entire evaluation function E(T,H,P) and the HMAC signing function; no intermediate evaluation state may be observable outside the enclave boundary. Remote attestation must be performed before any key material is provisioned into the enclave, and attestation must be re-verified after any enclave restart.

The attestation evidence (measurement, signing certificate chain) must be archived in AL at enclave initialization time under record type `ENCLAVE_ATTESTATION`. If remote attestation fails or is unavailable, the enclave must not accept key provisioning and the enforcement component must not enter operational state. Degraded attestation is not a conformant operational state.

7.5 Abuse Throttling Determinism

EG implements rate limiting at the caller and `consent_id` dimensions. Per-caller limiting prevents enforcement infrastructure saturation; per-`consent_id` limiting detects anomalous access patterns on individual tokens. Rate limit

exceedance produces a deny decision with reason code `RATE_LIMIT_EXCEEDED` and an audit record. Rate limiting is an abuse protection layer applied before enforcement evaluation begins; it does not substitute for enforcement.

RATE LIMIT DETERMINISM INVARIANTS — INVARIANTS

```
// RL-INV-01: Rate limit application is pre-enforcement, not post-decision
∀ request R:
  if rate_limit_exceeded(R.caller_id) ∨ rate_limit_exceeded(R.consent_id):
    decision = DENY; reason = RATE_LIMIT_EXCEEDED
    AL.append(deny_record); return DENY
  else: proceed to enforcement path

// RL-INV-02: Rate limit counters are per-EG-instance
  Rate limit state is not shared across EG instances.
  Distributed rate limiting introduces coordination overhead
  that may exceed enforcement latency ceiling.
  Per-instance limits must be set to (global_limit / n_EG_instances)
  where n_EG_instances is the current active instance count.

// RL-INV-03: Rate limit denial is audited
∀ RATE_LIMIT_EXCEEDED denial:
  AL.append({type=DENY, reason=RATE_LIMIT_EXCEEDED,
             consent_id, caller_id, timestamp, limit_dimension})
  No rate-limit denial is silent.

// RL-INV-04: Per-consent_id anomaly detection threshold
  If request_rate(consent_id, window=60s) > anomaly_threshold:
    ML alert: HIGH: CONSENT_ID_ACCESS_ANOMALY{consent_id}
    Enforcement continues; alert is advisory; human review required
    Operator may escalate to automated consent_id suspension
    Automated suspension requires AL record: CONSENT_ID_SUSPENDED
```

SECTION 8

Determinism Assurance Procedures

8.1 Frozen Context Serialization: Formal Constraints

The context hash presented to the enforcement system must be produced by a canonicalization procedure that yields an identical output for semantically identical inputs regardless of field ordering, whitespace, or encoding differences. The procedure must be deterministic, order-normalized (all map keys sorted lexicographically by UTF-8 byte value), whitespace-stripped, encoding-fixed (UTF-8 throughout, no BOM, no surrogate pairs), and version-tagged (canonicalization algorithm version included in hash prefix). The canonicalization procedure must be published as a versioned, frozen specification; changes to the procedure require a new version identifier and a policy version bump.

CANONICAL SERIALIZATION INVARIANTS — INVARIANTS

```
// CS-INV-01: Key ordering · lexicographic by UTF-8 byte sequence
∀ map M with keys k₁, k₂: k₁ < k₂ in UTF-8 byte order
  → serialize(M) = "{" k₁ ":" v₁ "," k₂ ":" v₂ "}"

// CS-INV-02: No whitespace in serialized output
∀ serialized S: ¬∃ character c ∈ S: c ∈ {SP, HT, CR, LF}

// CS-INV-03: Encoding fixation
∀ string value v: encode(v) = UTF-8 without BOM
  ¬∃ surrogate pair in encoded output
  All non-ASCII characters must be encoded as UTF-8, not escaped as \uXXXX
  unless the character has no UTF-8 representation

// CS-INV-04: Version prefix
∀ context hash H:
  H = "sha256:qodiqa-ctx-v{N}:" + sha256_hex(serialize(context))
```

```
  where N is the canonicalization algorithm version
  H must be treated as opaque by EG; PE does not re-derive H

// CS-INV-05: Semantic equivalence implies hash equality
∀ context objects C1, C2: semantically_equal(C1, C2)
  → canonical_hash(C1) = canonical_hash(C2)
  This invariant must be verified by the determinism test protocol
  using a corpus of semantically-equal but syntactically-distinct inputs.

// CS-INV-06: No floating-point values in context objects
  Floating-point representation is implementation-defined and
  may produce different serializations across platforms.
  ∀ numeric value in context: must be integer or fixed-decimal string.
  Floating-point values must be rejected with error: CONTEXT_TYPE_VIOLATION.
```

CONTEXT CANONICALIZATION: PSEUDOCODE — PSEUDO

```
function canonical_hash(context_object):
    normalized = deep_sort_keys(context_object)
    serialized = json_encode(normalized,
        separators=(',', ':'),
        ensure_ascii=False,
        sort_keys=True
    )
    prefix = "qodiqa-ctx-v1:"
    return "sha256:" + sha256_hex(prefix + serialized)
```

8.2 Policy Version Binding: Proof Model

Every enforcement decision is bound to an explicit, immutable policy version identifier. Policy versions are frozen at publication; a frozen version cannot be modified. The PE rejects evaluation requests referencing a version not present in

its local snapshot store. Policy version identifiers follow the structure `pv_{ISO-date}-frozen`; the `-frozen` suffix is a normative marker of immutability.

POLICY VERSION BINDING PROOF SKETCH — FORMAL

```
// Definitions
PV = policy version identifier (string, immutable reference)
S(PV) = rule_set bound to PV: a finite, ordered set of evaluation predicates
H(PV) = SHA-256(canonical_serialize(S(PV))): the policy snapshot hash

// Binding invariant
At freeze time T_freeze:
  S(PV) is finalized; H(PV) computed and published
  AL.append({type=POLICY_FROZEN, pv=PV, snapshot_hash=H(PV), frozen_at=T_freeze})
  PE loads S(PV) and verifies: SHA-256(loaded_rules) = H(PV)
  If verification fails: PE rejects the policy version; emits POLICY_LOAD_FAILURE

// Evaluation binding
∀ evaluation request (T, hash_H, PV):
  PE evaluates E(T, hash_H, PV) using S(PV) loaded and verified at startup
  The evaluation result is bound to PV by construction:
  changing PV changes S(PV) and therefore may change E(T, hash_H, PV)
  No two distinct PVs are permitted to have the same H(PV)
  (uniqueness enforced by date-based identifier and content hash in AL)

// Replay validation logic
∀ replayed request R (same consent_id, context_hash, policy_version, nonce):
  nonce must be unique per request; R.nonce ∉ used_nonces(consent_id)
  If R.nonce ∈ used_nonces: DENY with reason=NONCE_REPLAY_DETECTED
  used_nonces is maintained per EG instance with TTL = token.expires_at
  Replay detection does not prevent legitimate re-evaluation;
  it prevents exact request duplication with identical nonces.
```

```
// Frozen context immutability guarantee
∀ context hash H presented to enforcement at time T:
  H is the caller's commitment to the context at T
  The context cannot be modified post-hash without producing a different H
  EG does not re-derive or modify H; it is carried opaquely from caller to PE
  PE does not validate context content; it validates H binds to the decision
```

8.2a Reproducibility Test Algorithm

DETERMINISM VERIFICATION PROCEDURE: STEP-BY-STEP — PROCEDURE

```
procedure VERIFY_DETERMINISM(policy_version PV, corpus_size N ≥ 500):

  Step 1, Corpus Generation:
    Generate N test vectors (T_i, H_i, PV) with known expected decisions D_i
    Include: edge cases, boundary tokens, expired tokens, revoked tokens,
             unknown issuers, scope mismatches, default-deny cases
    Corpus must be archived with SHA-256 hash before test execution

  Step 2, First Evaluation Pass:
    For each vector (T_i, H_i, PV):
      result_1[i] = E(T_i, H_i, PV) via live enforcement path
      if result_1[i] ≠ D_i: FAIL: policy_mismatch at vector i
        record failure; terminate; block deployment

  Step 3, PE Restart:
    Restart all PE instances; verify policy reload: SHA-256(S(PV)) = H(PV)
    If reload verification fails: FAIL: policy snapshot corrupted

  Step 4, Second Evaluation Pass:
    For each vector (T_i, H_i, PV):
```

```
      result_2[i] = E(T_i, H_i, PV) via live enforcement path
      if result_2[i] ≠ result_1[i]: FAIL: non-deterministic evaluation
        record: vector i, result_1[i], result_2[i]; escalate immediately

  Step 5, Archive:
    If all checks pass:
      AL.append({type=DETERMINISM_TEST, pv=PV, corpus_hash=H(corpus),
                 N=N, pass_rate=1.0, tested_at=now})
      Policy version approved for deployment gate

  Step 6, Semantic Equivalence Test:
    For a subset of M ≥ 50 vectors: generate semantically-equal variants
    (same context, different field ordering/whitespace)
    Verify canonical_hash produces identical H for each variant pair
    Any variance fails the canonicalization invariant
```

8.3 Pseudo-Formal Evaluation Model

FORMAL EVALUATION MODEL: NOTATION — FORMAL

```
// Definitions
Let T = token_record = { id, state, issuer, scope, issued_at, expires_at }
Let H = context_hash ∈ SHA256_STRING
Let P = policy_version ∈ FROZEN_POLICY_ID
Let R = rule_set(P): immutable set of evaluation predicates

// Evaluation function
E(T, H, P) → { PERMIT, DENY } × reason_code

E(T, H, P) =
  if T.state ∈ { REVOKED, EXPIRED, NOT_FOUND }:
      return (DENY, token_state_as_reason(T.state))
```

```
  if ¬ issuer_recognized(T.issuer, P):
      return (DENY, POLICY_NOT_SATISFIED)
  let matched = { r ∈ R | r.applies(T, H) }
  if ∃ r ∈ matched: r.effect = DENY:
      return (DENY, POLICY_NOT_SATISFIED)
  if ∀ r ∈ R applicable to (T, H): r.effect = PERMIT:
      return (PERMIT, nil)
  return (DENY, POLICY_NOT_SATISFIED)  // default-deny

// Required properties
∀ T, H, P: E(T, H, P) is total and deterministic
∀ T, H, P: E(T, H, P) is pure (no side effects during evaluation)
∀ T₁=T₂, H₁=H₂, P₁=P₂: E(T₁,H₁,P₁) = E(T₂,H₂,P₂)
```

8.4 Determinism Test Protocol

Prior to production deployment of any policy version, the VERIFY_DETERMINISM procedure defined in §8.2a must be executed in full and results archived in AL under record type `DETERMINISM_TEST`. The procedure is a deployment gate: no policy version may be activated on any production PE instance without a passing DETERMINISM_TEST record in AL. The archived corpus hash must be verifiable by any conformance auditor with read access to AL. Any deviation from the step-by-step procedure invalidates the test result and requires re-execution.

SECTION 9

Observability and Monitoring

The Monitoring Layer (ML) is a non-blocking observer of the enforcement path. Its failure does not halt enforcement. Its purpose is to detect, classify, and escalate deviations from expected system state before those deviations become

enforcement failures. ML must have read access to all component health endpoints; it must not have write access to any enforcement-path component. All ML alerts are classified according to the failure severity taxonomy defined in §9.3.

9.1 Required Metrics

TABLE 9.1: MANDATORY METRIC DEFINITIONS

METRIC	TYPE	DESCRIPTION	LABELS
`qodiqa_decisions_total`	Counter	All decisions by outcome	outcome, reason_code
`qodiqa_enforcement_latency_ms`	Histogram	End-to-end enforcement path latency	topology
`qodiqa_component_latency_ms`	Histogram	Per-component latency	component
`qodiqa_circuit_breaker_state`	Gauge	State per circuit (0=closed, 1=half-open, 2=open)	component
`qodiqa_revocation_events_total`	Counter	Revocation events received and propagated	issuer, status
`qodiqa_revocation_propagation_ms`	Histogram	Revocation commit to EG acknowledgment latency	region
`qodiqa_audit_write_errors_total`	Counter	AL write failures	reason
`qodiqa_artifact_mismatch_total`	Counter	Outputs with missing or invalid artifact	detection_point
`qodiqa_ledger_chain_integrity`	Gauge	Last successful chain verification (epoch)	partition

9.2 Alert Thresholds

TABLE 9.2: ALERT DEFINITIONS

CONDITION	THRESHOLD	SEVERITY	RESPONSE
Circuit breaker open (any enforcement component)	State = OPEN	CRITICAL	Immediate escalation; all requests denied
Enforcement latency P99	> 100ms	HIGH	Capacity review; scale-out
AL write error rate	> 0.01%	CRITICAL	Block new enforcement; investigate AL
Revocation propagation latency	> TTL/2	HIGH	RS incident; manual verification
Artifact mismatch detected	> 0	CRITICAL	Immediate investigation; possible integrity breach
Ledger chain integrity failure	Any break	CRITICAL	Halt AL writes; forensic review
Deny rate anomaly	> 3σ from baseline	HIGH	Policy review; possible misconfiguration

9.3 Failure Severity Classification

All failure conditions detected by ML are classified into four severity classes. Classification determines the automated response, escalation path, and required audit record. Classification is deterministic: given an observed condition, the class is uniquely determined by the rules below. No condition may be reclassified downward without a committed AL record documenting the basis for reclassification and human authorization.

CLASS I

Hard Fail-Closed

AL write failure · Circuit breaker OPEN on any enforcement-path component · Ledger chain break · Unauthorized revocation flag write · Artifact mismatch (cause not key rotation) · Enclave attestation failure. Automated response: all enforcement suspended immediately; all requests denied; human intervention required before resumption. AL record: ENFORCEMENT_HALTED.

CLASS II

Control Degradation

Revocation propagation latency > TTL/2 · Enforcement P99 latency > 100ms · CR cache miss rate > 50% sustained · Key rotation window active beyond 12 hours · Cross-region consensus lag approaching TTL. Automated response: conservative mode activated for affected consent_ids; human review required within 4 hours; capacity action initiated. Enforcement continues under conservatism.

CLASS III

Observability Anomaly

ML component degraded · Metric collection gap > 60 seconds · Deny rate > 3σ from baseline without known policy change · consent_id access anomaly above threshold. Automated response: alert dispatched; enforcement path unaffected; monitoring blind spot must be resolved within recovery window. Human review required; no automated enforcement change.

CLASS 0

Integrity Breach

Artifact mismatch confirmed replay · AL chain break with evidence of external modification · CR revocation flag written by unauthorized principal · Policy version mutation post-freeze detected. Full forensic review required; no automated recovery pathway. All affected consent_ids suspended. External notification to governance authority required.

9.4 Deterministic Drift Classification Matrix

TABLE 9.3: DRIFT CLASSIFICATION MATRIX

OBSERVED DRIFT	DETECTION METRIC	THRESHOLD	CLASS	AUTOMATED ACTION
Enforcement latency drift	`qodiqa_enforcement_latency_ms P99`	> 75ms	II	Scale-out alert; cache review
Enforcement latency breach	`qodiqa_enforcement_latency_ms P99`	> 100ms	II+	Conservative mode; human escalation
Revocation lag approaching TTL	`qodiqa_revocation_propagation_ms P95`	> TTL/2	II	RS incident; conservative mode
Revocation lag at TTL	`qodiqa_revocation_propagation_ms P95`	= TTL	I	Affected consent_ids denied; halt

OBSERVED DRIFT	DETECTION METRIC	THRESHOLD	CLASS	AUTOMATED ACTION
AL write error rate	`qodiqa_audit_write_errors_total rate/min`	> 0	I	Enforcement suspended
Artifact divergence detected	`qodiqa_artifact_mismatch_total`	> 0	I / 0	Halt; forensic per FC-05
Ledger chain break	`qodiqa_ledger_chain_integrity`	Any break	0	AL READ_ONLY; full forensic
Clock skew exceeded	`qodiqa_clock_skew_seconds`	> skew_tolerance	II	Timestamp-based denies; NTP fix
Deny rate anomaly	`qodiqa_decisions_total outcome=DENY rate`	> 3σ baseline	III	Policy review alert; no enforcement change
Circuit breaker OPEN	`qodiqa_circuit_breaker_state`	= 2 (OPEN)	I	Enforcement suspended; all DENY

9.5 Artifact Divergence Response Tree

ARTIFACT DIVERGENCE RESPONSE: DECISION TREE — PROCEDURE

```
on ARTIFACT_VERIFY_FAIL{artifact_id=A, consent_id=C, cause=X}:

  1. Retrieve AL record for A: confirm PERMIT record exists for C
     If AL record missing → Class 0: AUDIT_INTEGRITY_BREACH; halt

  2. Determine cause X:

     X = KEY_NOT_RECOGNIZED:
       → Check active key rotation window in AL
       If within rotation window and key is outgoing key: RESUME
         (artifact is valid under dual-key window; log and continue)
```

```
      If outside rotation window: Class I; AE signing key compromised
        → AE halted; key rotation emergency procedure initiated

    X = CONTEXT_HASH_MISMATCH:
      → artifact.context_hash ≠ request.context_hash
      → Possible replay: check nonce in AL for prior use
      If nonce previously used: Class 0: REPLAY_ATTACK; suspend consent_id
      If nonce not previously used: Class I: artifact corruption or tampering

    X = CONSENT_ID_MISMATCH:
      → artifact.consent_id ≠ request.consent_id
      → Class 0: AE issued artifact with wrong binding; full AE audit required

    X = HMAC_INVALID:
      → Artifact bit corruption in transit or at rest
      → Class I: infrastructure integrity concern; isolate AE instance; investigate

3. All Class 0 and Class I outcomes:
   → Suspend affected consent_id(s) immediately
   → AL.append(ENFORCEMENT_SUSPENDED{consent_id=C, reason=X, timestamp})
   → Human authorization required to unsuspend

4. SLA breach escalation:
   If Class I condition persists > 15 minutes without human acknowledgment:
     → Escalate to organizational governance authority
     → All enforcement for affected deployment suspended automatically
```

9.6 Integrity Heartbeat Design

The ML integrity heartbeat is a periodic, independent verification of enforcement system consistency. It is not a health check; it is a bounded consistency audit. The heartbeat runs on a defined schedule (minimum interval: 60 seconds;

maximum interval: operator-configured per maturity level) and verifies the following invariants in sequence:

- AL chain integrity: verify the most recent N records form an unbroken chain; N is configured as (heartbeat_interval_seconds × peak_decisions_per_second × 1.2)
- Policy snapshot hash: verify all active PE instances report the same policy version and snapshot hash; divergence between instances is a Class I condition
- Revocation sync state: verify all active EG instances acknowledge the most recent revocation event within TTL; any unacknowledged instance triggers Class II escalation
- Circuit breaker states: verify all component circuit breakers report CLOSED; any non-CLOSED state that is not already alerting triggers immediate Class I escalation
- Key rotation window: verify no key rotation window has been open beyond the maximum window duration; breach triggers Class II escalation

The heartbeat result is recorded in ML with timestamp and a pass/fail status for each verification. Heartbeat failures that are not already active alerts are treated as new Class I conditions. A heartbeat that cannot complete within its scheduled interval due to ML load is itself a Class III condition.

SECTION 10

Conformance Readiness Checklist

The following checklist maps to normative controls QCC-01 through QCC-12 in the QODIQA Core Standard. All applicable items must be satisfied before a deployment may be declared conformant at any maturity level. Items marked with a QCC reference are directly normative; items without are implementation readiness criteria derived from this

Playbook. For each item, the validation method, evidence artifact, and fail-closed proof requirement are specified. This checklist functions as a certification pre-audit instrument: a deployment that cannot produce the specified evidence artifact for each item is not conformant, regardless of operational behavior.

10.1 Architecture and Component Completeness

- ☐ QCC-01.1 All seven canonical components (EG, PE, CR, RS, AL, AE, ML) are deployed and operationally independent. *Validation: architecture diagram cross-referenced against network topology; each component has a distinct network address and process boundary. Evidence: network topology map signed by infrastructure lead. Fail-closed proof: simultaneous shutdown of inference system component; verify AL record exists for every request processed prior to shutdown.*
- ☐ QCC-01.2 No component shares a process space with the inference system under enforcement. *Validation: process isolation audit; no shared PID namespace; container or VM boundary verified. Evidence: runtime process tree snapshot with process-to-component mapping. Fail-closed proof: kill inference process; verify EG, PE, CR, AL remain operational.*
- ☐ QCC-01.3 Inter-component interfaces are mutually authenticated (mTLS or equivalent). *Validation: TLS certificate inspection; certificate pinning verification; test with expired certificate. Evidence: certificate chain for each inter-component channel, archived with expiry dates. Fail-closed proof: present expired certificate at EG-CR interface; verify EG denies and emits alert.*
- ☐ QCC-02.1 EG is positioned as the sole enforcement gateway; no bypass path exists to inference system. *Validation: network path analysis; verify inference system has no listener reachable without traversing EG. Evidence: network access control list and routing table for inference system host. Fail-closed proof: attempt direct network connection to inference system port from external host; verify blocked.*

10.2 Fail-Closed Validation

- ☐ QCC-03.1 EG verified to deny all requests on CR unavailability. *Validation: controlled CR shutdown test; verify 100% of requests denied within circuit open window. Evidence: test execution log with timestamp, request count, DENY count; must show 0 PERMITs. Failure simulation: terminate CR process; send 50 requests; verify all 50 in AL as DENY with CR_UNAVAILABLE.*

☐	QCC-03.2	EG verified to deny all requests on PE unavailability. *Validation: controlled PE shutdown test. Evidence: test log with 0 PERMITs during PE outage. Failure simulation: kill PE process; send 20 requests; verify all DENY with PE_UNAVAILABLE.*
☐	QCC-03.3	EG verified to deny all requests on AL write failure. *Validation: AL write fault injection (network partition EG→AL). Evidence: test log confirming no request proceeds when AL write times out. Failure simulation: block AL port; send 10 requests; verify all blocked with AUDIT_COMMIT_FAILURE; verify 0 inference executions.*
☐	QCC-03.4	EG verified to deny all requests on AE generation failure. *Validation: AE signing failure injection (corrupt AE signing key). Evidence: test log confirming PERMIT path is blocked when AE fails. Failure simulation: revoke AE signing key; send 10 PERMIT-eligible requests; verify all blocked.*
☐	QCC-03.5	Timeout ceilings for each component configured within specified bounds. *Validation: configuration inspection; verify timeout values against Table 5.1. Evidence: configuration snapshot with timestamp; values must match declared ceilings. Determinism validation: verify timeout-triggered denial produces AL record within 5ms of timeout expiry.*
☐	QCC-03.6	Circuit breaker parameters (N, W) configured within specified bounds. *Validation: configuration inspection; $N \in [3,10]$, $W \in [10s,60s]$. Evidence: configuration snapshot. Fail-closed proof: trigger N consecutive failures within W; verify circuit opens; verify all subsequent requests denied without component query.*

10.3 Consent Registry and Revocation

☐	QCC-04.1	CR implements linearizable reads for the revocation flag. *Validation: consistency model documentation from CR storage vendor; linearizability guarantee must be explicit, not implied. Evidence: storage system configuration with consistency level set to linearizable/strong. Fail-closed proof: concurrent write (revocation) and read test under load; verify no stale reads observed in 1,000 concurrent read/write trials.*
☐	QCC-04.2	CR cache TTL is no greater than the configured revocation propagation bound. *Validation: cache configuration inspection; $TTL_{cache} \leq TTL_{revocation}$. Evidence: cache configuration with TTL values and comparison to revocation TTL. Determinism validation: verify cache expiry triggers CR re-read, not default-permit.*

- ☐ QCC-04.3 CR cache invalidates synchronously on revocation event receipt. *Validation: revocation event injection; measure time from event to cache invalidation. Evidence: measurement log with timestamps; invalidation latency must be ≤ 100ms. Fail-closed proof: submit request for recently-revoked consent_id within 200ms of revocation; verify DENY.*
- ☐ QCC-05.1 RS propagates revocation events to all EG instances within documented TTL. *Validation: controlled revocation test across all EG instances; measure propagation latency per instance. Evidence: propagation measurement log with per-EG timestamps; all must be ≤ TTL. Failure simulation: network partition one EG from RS; verify conservative mode activates at TTL/2.*
- ☐ QCC-05.2 Unacknowledged propagation events trigger Class II alert at TTL/2. *Validation: propagation delay test; withhold RS→EG acknowledgment; verify ML alert at TTL/2. Evidence: ML alert log with timestamp; alert must fire before TTL/2 + 5 seconds. Fail-closed proof: verify conservative mode activates for affected consent_ids simultaneously with alert.*

10.4 Audit Ledger Integrity

- ☐ QCC-06.1 AL is append-only; delete and modify operations disabled at storage layer. *Validation: attempt DELETE and UPDATE operations against AL storage endpoint; verify rejection. Evidence: rejection log with error codes OPERATION_PROHIBITED. Fail-closed proof: verify rejection generates ML alert; verify AL integrity metric unchanged after rejection attempt.*
- ☐ QCC-06.2 AL hash chain integrity verified on defined schedule. *Validation: verify chain verification tooling operational; verify last run timestamp in ML. Evidence: chain verification run log; must show CHAIN_VALID for all partitions at last execution. Determinism validation method: re-run chain verification independently; result must be identical to archived result.*
- ☐ QCC-06.3 AL exports are cryptographically signed by AL component key. *Validation: export a sample AL partition; verify signature against AL public key. Evidence: export file with signature; verification command and result. Fail-closed proof: tamper one byte of export; verify signature verification fails.*
- ☐ QCC-06.4 AL records contain all required fields including token_record_snapshot. *Validation: schema validation against required fields for both PERMIT and DENY record types. Evidence: sample records for each type with field-by-field validation. Determinism validation: verify record schema is identical across all AL partitions.*

10.5 Determinism and Artifact Integrity

☐	QCC-07.1	AE uses HMAC-SHA256 with key from defined three-tier key hierarchy. *Validation: AE key provenance audit; verify signing key traces to intermediate key, which traces to root key signature. Evidence: key certificate chain with root key fingerprint. Fail-closed proof: generate artifact; verify HMAC with published AE public key; verify inference system refuses artifact with tampered HMAC.*
☐	QCC-07.2	Artifact fields include all required fields per §4.6 specification. *Validation: artifact schema inspection for a sample of ≥ 100 live artifacts. Evidence: field-by-field validation report. Determinism validation: generate artifact for same inputs twice after AE restart; verify bit-identical output for HMAC (same key, same nonce).*
☐	QCC-07.3	Inference system performs artifact verification before beginning computation. *Validation: send request with malformed artifact; verify inference system refuses before computation begins. Evidence: inference system refusal log with timestamp; must precede any computation metric. Fail-closed proof: remove artifact entirely; verify inference system refuses with MISSING_ARTIFACT.*
☐	QCC-08.1	Context canonicalization procedure is documented and version-tagged. *Validation: canonicalization specification exists as versioned document; version tag in context hash matches. Evidence: specification document hash; sample context hash with version prefix decoded. Determinism validation: generate context hash for semantically-equivalent inputs with different field ordering; verify identical output.*
☐	QCC-08.2	Policy versions are frozen at publication; no post-publication modification possible. *Validation: attempt to modify rule in frozen policy version; verify rejection at PE and AL. Evidence: rejection log. Fail-closed proof: verify AL record exists for policy freeze event; verify policy snapshot hash matches AL record hash.*
☐	QCC-08.3	Determinism test protocol (§8.2a, VERIFY_DETERMINISM) executed and archived for all deployed policy versions. *Validation: AL query for DETERMINISM_TEST records; verify one exists per deployed policy version. Evidence: AL records with corpus hash, test timestamp, pass_rate=1.0. Failure simulation requirement: execute Steps 2 and 4 with one intentionally incorrect expected decision; verify procedure fails and deployment is blocked.*

10.6 Security, Observability and Topology

☐	QCC-09.1	Key hierarchy documented; root key is offline. *Validation: key management documentation; verify root key has no network-accessible interface. Evidence: hardware storage inventory for root key; last loaded timestamp (must be at key establishment, not recent). Fail-closed proof: verify root key cannot be queried from any network-connected system.*
☐	QCC-09.2	Signing key rotation schedule operational within defined bounds. *Validation: rotation log in AL; verify KEY_ROTATION_EVENT records within 30-day ceiling. Evidence: AL export of KEY_ROTATION_EVENT records for past 90 days. Fail-closed proof: verify dual-key window is operational during active rotation; submit request during rotation; verify artifact accepted by both keys.*
☐	QCC-10.1	All required metrics from Table 9.1 collected and available to ML. *Validation: metric endpoint inspection; verify each metric from Table 9.1 is present and updating. Evidence: metric snapshot at two timestamps; verify counter increments for decision metrics. Determinism validation: verify metric labels are consistent and non-ambiguous.*
☐	QCC-10.2	All defined alert thresholds from Table 9.2 active and routed to escalation paths. *Validation: alert configuration inspection; trigger test alerts; verify delivery. Evidence: alert configuration snapshot; test alert delivery receipt. Fail-closed proof: trigger each Class I condition; verify enforcement suspension within 30 seconds of threshold breach.*
☐	QCC-11.1	Agent delegation independence verified; no transitive permit inheritance. *Validation: submit request with delegated agent token; verify enforcement re-evaluates independently for each delegation level. Evidence: test log with delegation chain and per-level decision records in AL. Fail-closed proof: revoke intermediate delegation; verify downstream agent requests denied without re-issuing root token.*
☐	QCC-12.1	Deployed topology declared; trust assumptions and revocation TTL documented and disclosed to all affected parties. *Validation: topology declaration document exists; TTL value disclosed in system documentation. Evidence: signed topology declaration with TTL value and disclosure acknowledgment. Determinism validation: verify declared TTL is the measured TTL; ML revocation propagation histogram P99 must be ≤ declared TTL.*

SECTION 11

Implementation Maturity Levels

Three maturity levels are defined for QODIQA enforcement deployments. Each level represents a complete, conformant deployment. Higher levels extend the baseline with additional hardening, availability, and assurance properties. A deployment must fully satisfy its declared level before claiming that level designation. Levels are not cumulative designations; each is a complete, independently auditable configuration. A deployment that satisfies Level 2 in some dimensions but not others is a Level 1 deployment until all Level 2 requirements are met without exception.

Bounded Liveness Statement: Across all three maturity levels, the enforcement path provides bounded liveness under the following preconditions: all enforcement-path components are operational; all component response times are within their configured ceilings; the revocation propagation round-trip completes within TTL; and the AL write acknowledgment is received within its ceiling. If any precondition fails, liveness is forfeited and fail-closed behavior is mandatory. The system does not guarantee liveness under partial failure; it guarantees only the safety property: no inference execution proceeds without a valid, committed enforcement decision. This is a deliberate architectural choice, not a limitation to be optimized away.

Failure Containment Boundary: Each maturity level defines its failure containment boundary: the scope within which a component failure is contained without requiring cross-boundary intervention. At Level 1, the containment boundary is the single deployment; a datacenter failure requires operator action. At Level 2, the boundary is the failure domain; a single failure domain failure is contained by replicated components. At Level 3, the boundary is the region; a regional failure is contained by cross-region consensus with conservative mode activation. In all cases, the safety property (fail-closed) holds within any boundary.

LEVEL 1 **Minimal Deployable**	LEVEL 2 **Enterprise Hardened**	LEVEL 3 **High-Assurance**
All seven components deployed, single-region, single-datacenter	All Level 1 requirements satisfied	All Level 2 requirements satisfied
Software-based AE; no HSM requirement	GR and AL replicated across ≥ 2 independent failure domains	Multi-region deployment; linearizable cross-region revocation
GR with single-primary, synchronous revocation write	EG and PE deployed with ≥ 2 active instances behind load balancer	HSM-backed AE mandatory
AL as append-only file or single-instance database	RS as durable message broker with delivery guarantees	PE and AE optionally in hardware secure enclave with attestation
Hash chain integrity verified weekly	Revocation TTL ≤ 60 seconds	Revocation TTL ≤ 15 seconds across all regions
Revocation TTL ≤ 300 seconds	Hash chain integrity verified daily	Hash chain integrity verified continuously (per-write)
All alert thresholds active	Out-of-band AL export to immutable storage (daily minimum)	Continuous AL export to operator-controlled immutable storage
Determinism test protocol executed for all policy versions	Key rotation operational (30-day deployment key ceiling)	Formal determinism audit by independent third party pre-deployment
No HA requirement; planned maintenance windows permitted	SLO defined for enforcement path availability (≥ 99.9%)	Annual penetration test of enforcement infrastructure
		Full audit trail verifiable by operator without SaaS intermediation
		SLO for enforcement path availability (≥ 99.99%)

11.1 Deterministic State Transition Model

The enforcement system operates as a deterministic finite state machine at the request level. The following state transition model defines all valid state sequences for a single enforcement request. Transitions not present in this model are prohibited; any system state that cannot be reached by a valid sequence of transitions from the RECEIVED state is an error state requiring investigation.

PER-REQUEST DETERMINISTIC STATE TRANSITIONS

FSM

```
// States: RECEIVED → {RATE_LIMITED, CR_QUERIED} → ...
// All terminal states produce exactly one AL record.

RECEIVED
    → RATE_LIMITED         [if rate_limit_exceeded]              → DENY-TERMINAL
    → VALIDATION_FAILED    [if request missing required fields]  → DENY-TERMINAL
    → CR_QUERIED           [otherwise]

CR_QUERIED
    → CR_UNAVAILABLE       [timeout or circuit open]             → DENY-TERMINAL
    → CR_UNRECOGNIZED      [invalid signing key on response]     → DENY-TERMINAL
    → TOKEN_REVOKED        [T.state ∈ {REVOKED, EXPIRED}]        → DENY-TERMINAL
    → TOKEN_NOT_FOUND      [T.state = NOT_FOUND]                 → DENY-TERMINAL
    → PE_QUERIED           [T.state = ACTIVE, issuer recognized]

PE_QUERIED
    → PE_UNAVAILABLE       [timeout or circuit open]             → DENY-TERMINAL
    → PE_DENY              [E(T,H,P) = DENY]                     → DENY-TERMINAL
    → AE_REQUESTED         [E(T,H,P) = PERMIT]

AE_REQUESTED
    → AE_UNAVAILABLE       [timeout or signing failure]          → DENY-TERMINAL
    → AL_WRITE_PENDING     [artifact generated]

AL_WRITE_PENDING
    → AL_WRITE_FAILED      [timeout or error; retry exhausted]   → DENY-TERMINAL
    → EXECUTION_RELEASED   [AL write acknowledged]

EXECUTION_RELEASED
    → ARTIFACT_VERIFY_OK   [inference system verifies artifact]  → PERMIT-TERMINAL
    → ARTIFACT_VERIFY_FAIL [inference system rejects artifact]   → FC-05 procedure
```

```
DENY-TERMINAL   → AL.append(DENY record); return DENY to caller
PERMIT-TERMINAL → AL.append(PERMIT record already committed); inference proceeds

// Invariant: Each request terminates in exactly one terminal state.
// No request may remain in a non-terminal state beyond the enforcement ceiling.
// A request that has not reached a terminal state after ceiling + 5ms
// is treated as a timeout; EG transitions to DENY-TERMINAL.
```

SECTION 12

Institutional Closing Statement

This Playbook completes the QODIQA corpus. It does not introduce new obligations. It does not revise the normative framework established by the Core Standard, the 68-Point Enforcement Framework, or the Whitepaper. It translates those obligations into the concrete architectural, operational, and procedural terms required to build infrastructure that satisfies them.

> Enforcement is deterministic, or it is not enforcement. A consent decision that can be bypassed by a component failure, a cache inconsistency, an artifact omission, or a propagation delay is not a decision; it is merely a recommendation.
> **Enforcement infrastructure cannot operate on recommendations.**

Every design choice in this document reflects fail-closed discipline. Where a specification might have permitted optimistic execution (proceeding on the assumption that enforcement would likely succeed), this Playbook requires

verification before execution. The overhead of verification is the cost of determinism. That cost is borne by the infrastructure, not by the persons whose consent is being enforced.

The Audit Ledger is not a logging mechanism. It is the institutional record of every commitment made and kept, or denied, in the enforcement system's operational history. Its integrity is not a performance characteristic; it is a precondition of accountability.

> Operators who implement this Playbook in conformance with the declared maturity level are not merely deploying software. They are instantiating an infrastructure posture: that **consent, once granted, governs processing; that consent, once revoked, terminates it;** that no inference occurs without a recorded, verifiable, cryptographically bound decision authorizing it.

12.1 Corpus Alignment Statement

This Implementation Playbook (QODIQA-IMPL-2026-001 v1.0) is normatively subordinate to and must be read in conjunction with the following corpus documents. In the event of conflict between this Playbook and any of the foregoing, the foregoing documents prevail. This Playbook introduces Annex-grade failure modeling (FC-01 through FC-05), formalized latency envelope expressions, canonical serialization invariants, policy version binding proof sketch, observability drift classification matrix, artifact divergence response tree, integrity heartbeat design, expanded key rotation invariants, ledger tamper-evidence proof logic, and certification pre-audit conformance checklist with validation criteria and evidence requirements. No normative enforcement semantics have been modified.

DOCUMENT STATUS

Document Status and Classification

This document is the Implementation Playbook of the QODIQA specification corpus. It operationalizes the Core Standard into a deployable infrastructure specification, translating normative requirements into concrete component architectures, deployment topologies, integration patterns, security controls, and operational procedures for production environments.

This is not a product specification, a commercial offering, or a legal instrument. It defines infrastructure requirements for implementing deterministic runtime consent enforcement in conformance with the Core Standard.

This document is intended for:

- Principal architects responsible for enforcement infrastructure design
- Zero-trust and security engineers implementing component hardening
- Technical governance leads validating conformance readiness
- Compliance and audit functions requiring infrastructure accountability evidence
- CTO and board-level reviewers assessing deployment posture

This document should be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0

- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0
- QODIQA: Threat Model and Abuse Case Specification, Version 1.0
- QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dușescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-IMPL-2026-001
TITLE	Implementation Playbook for Deterministic Runtime Consent Enforcement
SUBTITLE	Deployment, Integration, and Operationalization of QODIQA Enforcement Infrastructure
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Technical Specification
DOCUMENT STATUS	Normative: Technical Specification
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 13 PART III · OPERATIONAL DOCUMENTS

Implementation Conformance Checklist

The operational bridge between the 68-Point Enforcement Framework and a running system: each control point reduced to a single binary condition, organized across three conformance profiles.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-CL-2026-001	1.0	2026	Operational Instrument	Normative	NORMATIVE

Status note. No certification body or program exists yet. This is a conformance framework for self-assessment; any reference to certification or certified status describes a planned process, not an existing one.

PURPOSE & SCOPE

This document is a normative operational instrument for verifying conformance to the QODIQA Core Standard for Deterministic Runtime Consent Enforcement. It reduces each control point of the 68-Point Enforcement Framework to a single, binary-verifiable condition and organizes those conditions across three conformance profiles: CORE-MIN, CORE-VER, and CORE-ENT. This checklist does not replace the QODIQA Conformance Test Suite; it makes the Test Suite operationally accessible by providing implementors with a structured self-assessment instrument prior to formal certification.

SECTION 1

How to Use This Checklist

Each row in the checklist tables below represents one control point (CP) drawn from the QODIQA 68-Point Enforcement Framework. For each control point, a **Verifiable Condition** is stated as a binary proposition: it is either satisfied or not satisfied. No partial credit applies at the control-point level.

The three rightmost columns indicate applicability per profile. A filled indicator (●) means the control point is **mandatory** for that profile. An open indicator (○) means the control point is **recommended but not required**. A dash (N/A) means the control point does **not apply** to that profile.

The rightmost column provides a **tick box** for manual verification during self-assessment. During formal certification, this column is replaced by the Test Suite result record.

● MANDATORY FOR PROFILE ○ RECOMMENDED, NOT REQUIRED N/A: NOT APPLICABLE ☐ SELF-ASSESSMENT TICK

CONFORMANCE RULE

A system achieves conformance to a profile if and only if every mandatory control point for that profile is satisfied. Failure of any single mandatory control point constitutes non-conformance, regardless of the total number of satisfied conditions.

SECTION 2

Profile Overview

QODIQA defines three conformance profiles corresponding to distinct operational contexts and risk postures. Each profile is cumulative: CORE-VER includes all mandatory conditions of CORE-MIN, and CORE-ENT includes all mandatory conditions of CORE-VER.

TABLE 1: CONFORMANCE PROFILE SUMMARY

PROFILE	INTENDED DEPLOYMENT CONTEXT	MANDATORY CPS	CUMULATIVE
CORE-MIN	Single-tenant systems, internal tooling, early-stage production deployments with bounded scope and manual oversight.	24	*Baseline*
CORE-VER	Multi-service architectures, regulated environments, systems subject to external audit or contractual accountability requirements.	44	*MIN + 20*
CORE-ENT	Enterprise-scale deployments, multi-tenant platforms, critical infrastructure, and systems requiring formal certification under QODIQA governance.	68	*VER + 24*

CORE-MIN · 24 CP | CORE-VER · 44 CP | CORE-ENT · 68 CP

SECTION 3

CORE-MIN Conformance Checklist

CORE-MIN *Minimum Enforcement Profile* — 24 MANDATORY CONTROL POINTS

Applicable to: single-tenant systems, internal tooling, and early-stage production deployments. All 24 control points below are mandatory for CORE-MIN conformance.

DOMAIN 1: INTENT DECLARATION

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	✓
CL-01	CP-01	Every AI action request is accompanied by a declared intent field that explicitly identifies the action type before execution begins.	●	●	●	☐
CL-02	CP-02	Every request contains a declared purpose field specifying the operational reason for the requested action.	●	●	●	☐
CL-03	CP-03	Every request identifies the data subjects and data categories involved at declaration time, not inferred post hoc.	●	●	●	☐
CL-04	CP-04	The system rejects any request in which the intent field is absent, empty, or malformed, without attempting inference or substitution.	●	●	●	☐
CL-05	CP-05	Request contracts are closed at submission and cannot be modified during evaluation.	●	●	●	☐

DOMAIN 2: CONSENT STATE RESOLUTION

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	✓
CL-06	CP-09	Consent state is resolved explicitly as a discrete value (granted / denied / expired / revoked) before any policy logic is applied.	●	●	●	☐
CL-07	CP-10	Each consent record is scope-bound: it specifies a defined purpose, data category, and acting entity for which it is valid.	●	●	●	☐

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	
CL-08	CP-11	Each consent record carries an explicit validity period; the system treats records without a validity period as invalid.	●	●	●	☐
CL-09	CP-12	Revocation of consent takes effect before the next evaluation cycle; there is no execution window during which revoked consent is treated as valid.	●	●	●	☐
CL-10	CP-13	The system does not infer, assume, or inherit consent across purposes, data categories, or acting entities.	●	●	●	☐

DOMAIN 3: POLICY EVALUATION

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	✓
CL-11	CP-17	Policies declare required conditions; they do not perform consent resolution and do not access mutable runtime state during evaluation.	●	●	●	☐
CL-12	CP-18	Each policy version is immutable after publication; changes are introduced only through explicit versioning with a new policy identifier.	●	●	●	☐
CL-13	CP-19	The evaluation context is frozen at the moment of evaluation; no policy condition can read state that changes during evaluation.	●	●	●	☐
CL-14	CP-20	Every evaluation produces exactly one of three outcomes: ALLOW, DENY, or ALLOW-WITH-RESTRICTIONS. No outcome outside this set is produced.	●	●	●	☐

DOMAIN 4: DECISION ARTIFACTS

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	✓
CL-15	CP-24	Every enforcement decision produces a persistent decision artifact that records the outcome, the policy version applied, the consent state resolved, and the timestamp of evaluation.	●	●	●	☐
CL-16	CP-25	Decision artifacts are immutable after creation; no post-evaluation process can alter a stored artifact.	●	●	●	☐
CL-17	CP-26	Decision artifacts are retained for a minimum period sufficient to satisfy all applicable audit, regulatory, and contractual obligations in the deployment context.	●	●	●	☐

DOMAIN 5: FAIL-CLOSED BEHAVIOR

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	✓
CL-18	CP-30	In any failure condition (missing input, unavailable consent store, policy resolution error), the system produces DENY rather than a default-allow or best-effort outcome.	●	●	●	☐
CL-19	CP-31	Fail-closed behavior is unconditional and cannot be overridden by configuration, environment flag, or caller-supplied parameter.	●	●	●	☐
CL-20	CP-32	Every DENY outcome produced under a failure condition is recorded with a distinct failure reason code differentiating it from a policy-determined DENY.	●	●	●	☐
CL-21	CP-33	The enforcement layer does not expose internal error details to the calling system; failure responses carry a structured status code only.	●	●	●	☐

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	
CL-22	CP-34	Enforcement latency under failure conditions does not exceed the threshold defined for normal-path evaluation in the deployment specification.	●	●	●	☐
CL-23	CP-35	Downstream systems cannot distinguish a failure-induced DENY from a policy-determined DENY based on the enforcement response alone.	●	●	●	☐
CL-24	CP-36	The system is demonstrably isolated from the execution path it governs: a failure within the enforcement layer does not cause uncontrolled execution of the rejected action.	●	●	●	☐

CORE-MIN total: 24 mandatory control points. All 24 must be satisfied for CORE-MIN conformance. CORE-VER and CORE-ENT inherit all 24 mandatory conditions above and add the conditions defined in Sections 4 and 5 respectively.

SECTION 4

CORE-VER Conformance Checklist

CORE-VER *Verifiable Enforcement Profile* **+20 MANDATORY CONTROL POINTS (44 TOTAL)**

CORE-VER inherits all 24 CORE-MIN conditions. The 20 control points below are additional mandatory requirements for CORE-VER conformance. Applicable to: multi-service architectures, regulated environments, and systems subject to external audit.

DOMAIN 6: DETERMINISM GUARANTEES

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	✓
CL-25	CP-37	Given identical inputs (request contract, policy version, consent state, evaluation timestamp), the enforcement engine always produces the identical decision outcome.	N/A	○	○	☐
CL-26	CP-38	The evaluation engine contains no non-deterministic components: no probabilistic sampling, no randomized tie-breaking, and no time-dependent branching outside the declared evaluation timestamp.	N/A	○	○	☐
CL-27	CP-39	The determinism property is verifiable through a documented test procedure that can be executed by a third-party auditor without access to internal system components.	N/A	○	○	☐
CL-28	CP-40	The system maintains a determinism boundary log that records any condition under which the determinism guarantee could not be upheld, including the reason and the resulting DENY outcome.	N/A	○	○	☐

DOMAIN 7: REPLAY INVARIANCE & AUDIT

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	✓
CL-29	CP-41	Any past decision can be replayed against the original frozen evaluation context and yields the identical outcome as the original evaluation.	N/A	○	○	☐
CL-30	CP-42	The system provides an independent replay interface that does not require the production enforcement path to be active during audit operations.	N/A	○	○	☐
CL-31	CP-43	Audit query results are reproducible: the same audit query against the same artifact set always returns the same result.	N/A	○	○	☐

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	
CL-32	CP-44	All decision artifacts contributing to a specific audit finding are traceable to their originating request without requiring reconstruction or inference.	N/A	●	●	☐
CL-33	CP-45	Audit access is governed by access controls that are independent of the enforcement path and cannot be modified by entities whose actions are subject to audit.	N/A	●	●	☐

DOMAIN 8: CRYPTOGRAPHIC INTEGRITY

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	✓
CL-34	CP-48	Each decision artifact carries a cryptographic digest (SHA-256 minimum) computed over the full artifact content at the time of creation.	○	●	●	☐
CL-35	CP-49	Policy documents are stored with a cryptographic digest; any discrepancy between stored and computed digest causes immediate rejection of the policy.	○	●	●	☐
CL-36	CP-50	The cryptographic integrity of all stored artifacts can be verified independently using only publicly documented algorithms and artifact contents.	N/A	●	●	☐
CL-37	CP-51	Consent records are cryptographically bound to the identity of the issuing authority; the binding is verifiable without access to the enforcement layer.	N/A	●	●	☐

DOMAIN 9: LATENCY & PERFORMANCE CONTROLS

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	✓
CL-38	CP-54	Enforcement latency (measured from request receipt to decision emission) is bounded by a documented Service Level Objective (SLO) enforced at the p99 percentile.	N/A	●	●	☐
CL-39	CP-55	SLO breaches are logged with sufficient detail (timestamp, duration, failure reason) to support root-cause analysis without relying on memory or inference.	N/A	●	●	☐
CL-40	CP-56	The enforcement layer imposes a hard timeout on consent resolution; if resolution does not complete within the declared timeout, a fail-closed DENY is issued.	N/A	●	●	☐
CL-41	CP-57	Consent resolution and policy evaluation execute on independent processing paths and do not share mutable runtime resources that could introduce timing dependencies.	N/A	●	●	☐
CL-42	CP-58	Enforcement throughput can be demonstrated to meet production load requirements without degrading below the declared latency SLO, as evidenced by documented load test results.	N/A	●	●	☐
CL-43	CP-59	Latency measurements are produced by a monitoring component that is independent of the enforcement path and cannot be influenced by enforcement layer failures.	N/A	●	●	☐
CL-44	CP-60	The system provides a documented capacity ceiling: a maximum request rate above which enforcement behavior is undefined, and a mechanism to reject requests that exceed this ceiling before enforcement is attempted.	N/A	●	●	☐

CORE-VER total: 44 mandatory control points (24 inherited from CORE-MIN + 20 additional). All 44 must be satisfied for CORE-VER conformance.

SECTION 5

CORE-ENT Conformance Checklist

CORE-ENT *Enterprise Enforcement Profile* **+24 MANDATORY CONTROL POINTS (68 TOTAL)**

CORE-ENT inherits all 44 CORE-VER conditions. The 24 control points below are additional mandatory requirements for CORE-ENT conformance. Applicable to: enterprise-scale, multi-tenant, and critical infrastructure deployments.

DOMAIN 10: GOVERNANCE INTEGRATION

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	✓
CL-45	CP-61	The enforcement layer is integrated with the organization's formal governance structure: policy changes require an authorized approval workflow before taking effect in production.	N/A	N/A	●	☐
CL-46	CP-62	The governance approval workflow produces a signed approval record that is stored as a prerequisite artifact linked to the policy version it authorized.	N/A	N/A	●	☐
CL-47	CP-63	A designated governance authority is named and accountable for all policy decisions within the QODIQA deployment; this authority is recorded in the governance charter artifact.	N/A	N/A	●	☐
CL-48	CP-64	Governance audit reports are produced on a defined periodic schedule and are delivered to the designated governance authority without requiring manual data assembly.	N/A	N/A	●	☐
CL-49	CP-65	The system exposes a governance API that allows the designated governance authority to query aggregate enforcement statistics, policy utilization, and exception rates without accessing individual decision artifacts.	N/A	N/A	●	☐

DOMAIN 11: MULTI-TENANT ISOLATION

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	✓
CL-50	CP-66	Consent records, policy sets, and decision artifacts are partitioned by tenant identifier; no evaluation can access data belonging to a different tenant.	N/A	N/A	●	☐
CL-51	CP-67	Tenant isolation is enforced at the storage layer, not solely at the application layer; a direct storage query cannot return cross-tenant data regardless of the query origin.	N/A	N/A	●	☐
CL-52	CP-68	An isolation boundary test demonstrating the absence of cross-tenant data access is executed and recorded as part of each release validation cycle.	N/A	N/A	●	☐
CL-53	CP-69	Audit logs are partitioned by tenant; a tenant-scoped auditor role cannot access log entries belonging to another tenant.	N/A	N/A	●	☐
CL-54	CP-70	Resource consumption (processing time, storage quota, throughput) is bounded per tenant; one tenant's load cannot degrade enforcement latency for another tenant beyond the declared SLO.	N/A	N/A	●	☐

DOMAIN 12: DELEGATION & ESCALATION

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	✓
CL-55	CP-71	Consent delegation is represented as an explicit, signed delegation record identifying the delegating authority, the delegated entity, the scope of delegation, and its validity period.	N/A	N/A	●	☐
CL-56	CP-72	Delegated consent cannot exceed the scope of the original grant; any delegation that attempts to expand scope is rejected at evaluation time.	N/A	N/A	●	☐

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	
CL-57	CP-73	The full delegation chain is resolved and recorded in the decision artifact for every evaluation that relies on delegated consent.	N/A	N/A	●	☐
CL-58	CP-74	The system provides a defined escalation path for enforcement decisions that require human review; escalated decisions are not executed until the review is complete and the outcome is recorded.	N/A	N/A	●	☐
CL-59	CP-75	Escalation outcomes are stored as first-class decision artifacts and are subject to the same immutability, integrity, and retention requirements as automated enforcement decisions.	N/A	N/A	●	☐

DOMAIN 13: OPERATIONAL RESILIENCE & CONTINUITY

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	✓
CL-60	CP-76	The enforcement layer is deployed with redundancy such that the failure of a single component does not cause unavailability of the enforcement service; redundancy topology is documented.	N/A	N/A	●	☐
CL-61	CP-77	Failover between redundant components does not produce a window during which enforcement decisions are bypassed; fail-closed semantics apply throughout all failover transitions.	N/A	N/A	●	☐
CL-62	CP-78	A documented disaster recovery procedure exists, has been tested within the past 12 months, and demonstrates restoration of full enforcement capability within a declared Recovery Time Objective (RTO).	N/A	N/A	●	☐
CL-63	CP-79	Decision artifact storage is replicated to a geographically separate location; the recovery of artifacts from the replica is demonstrably sufficient to reconstruct the complete enforcement history.	N/A	N/A	●	☐
CL-64	CP-80	The system's operational health is continuously monitored by a component independent of the enforcement layer; health state changes trigger alerts that do not require manual detection.	N/A	N/A	●	☐

ID	CP REF	VERIFIABLE CONDITION	MIN	VER	ENT	
CL-65	CP-81	Change management for the enforcement layer follows a documented procedure that includes impact assessment, staged rollout, rollback capability, and mandatory post-deployment verification of enforcement correctness.	N/A	N/A	●	☐
CL-66	CP-82	An annual third-party security assessment of the enforcement layer is conducted; findings are tracked to resolution and evidence of resolution is stored as governance artifacts.	N/A	N/A	●	☐
CL-67	CP-83	The enforcement layer exposes a machine-readable status endpoint that allows downstream systems to detect enforcement unavailability and activate fail-closed behavior independently.	N/A	N/A	●	☐
CL-68	CP-84	Capacity planning for the enforcement layer is reviewed at least quarterly and produces a documented forecast of headroom relative to projected request volumes for the next operational period.	N/A	N/A	●	☐

CORE-ENT total: 68 mandatory control points (44 inherited from CORE-VER + 24 additional). All 68 must be satisfied for CORE-ENT conformance.

SECTION 6

Scoring and Conformance Determination

Conformance is binary at the profile level. A system either satisfies all mandatory control points for a given profile or it does not. There is no partial conformance designation, no weighted scoring, and no grace period for mandatory conditions. This design is intentional: partial conformance creates ambiguity about which properties can be relied upon by downstream systems, consumers, and auditors.

Conformance Thresholds by Profile

All mandatory CPs must be satisfied. Recommended (open circle) conditions do not affect conformance status but are recorded in the self-assessment record.

CORE-MIN	CORE-VER	CORE-ENT
24 / 24	**44 / 44**	**68 / 68**
All mandatory CPs satisfied	All MIN + VER CPs satisfied	All VER + ENT CPs satisfied

Self-Assessment Procedure

Before initiating formal certification under the QODIQA Certification Framework, implementors should complete this checklist as a self-assessment instrument. For each control point relevant to the target profile, tick the verification box only when the condition is demonstrably satisfied: not when it is planned, partially implemented, or approximated.

Any unsatisfied mandatory condition should be recorded in an implementation gap register that identifies the control point, the nature of the gap, the responsible team, and a target remediation date. The gap register is a prerequisite artifact for formal certification submission.

> IMPORTANT: SELF-ASSESSMENT LIMITATION
>
> This checklist is a preparatory instrument. Self-assessment results do not constitute conformance certification. Formal QODIQA conformance is established only through the QODIQA Conformance Test Suite administered by an accredited evaluation body under the QODIQA Certification Framework.

Gap Classification

When a mandatory control point is not satisfied, classify the gap by severity to prioritize remediation effort:

TABLE 2: GAP SEVERITY CLASSIFICATION

SEVERITY	DEFINITION	CERTIFICATION IMPACT
BLOCKING	The condition is entirely unimplemented or the implementation explicitly violates the stated requirement.	Non-conformance. Cannot certify.
MARGINAL	The condition is partially implemented; a specific, bounded capability is missing or the implementation deviates in a limited way.	Non-conformance. Remediation required.
EVIDENCE	The condition is implemented but supporting evidence or documentation is insufficient to demonstrate compliance to an external auditor.	Non-conformance. Documentation required.

DOCUMENT STATUS

Document Status and Corpus Alignment

This document is a normative operational instrument within the QODIQA specification corpus. It derives its control points from the QODIQA 68-Point Enforcement Framework and is intended to be used in conjunction with the QODIQA Conformance Test Suite. It does not supersede or modify any normative requirement in the Core Standard or the 68-Point Framework.

This checklist is designed as an implementation-facing complement to the following related specifications:

- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement
- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper

Version 1.0 represents the initial normative release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Duțescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-CL-2026-001
TITLE	Implementation Conformance Checklist
SUBTITLE	Normative Binary Verification Instrument for CORE-MIN, CORE-VER, and CORE-ENT Profiles
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Normative Operational Instrument
DOCUMENT STATUS	Normative
GOVERNING AUTHORITY	QODIQA Governance Charter
DERIVES FROM	QODIQA 68-Point Enforcement Framework · Core Standard v1.0
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 14 PART III · OPERATIONAL DOCUMENTS

Reference Implementation Specification

A minimal, deterministic runtime enforcement stack demonstrating consent validation at the inference boundary, fail-closed decisioning, and verifiable audit logging.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-RIS-2026-001	1.0	2026	Reference Implementation	Normative	NORMATIVE

ABSTRACT

This document specifies a minimal, deterministic reference implementation of the QODIQA runtime consent enforcement stack. It defines the components, interfaces, data structures, and execution flow required to demonstrate consent validation at the AI inference boundary, deterministic allow/deny decisioning, fail-closed behavior, and verifiable audit logging. The implementation is explicitly minimal in scope: it is not a production system, not a scalable architecture, and not a complete security solution. Its purpose is to instantiate the core behavioral guarantees of the QODIQA Core Standard and the 68-Point Enforcement Framework in a concrete, executable form that can be inspected, validated, and used as a reference for production implementations. The stack is fully deterministic: given identical inputs and policy state, the system will always produce identical outputs. All undefined or missing inputs result in denial. No silent fallback, inference, or probabilistic behavior is permitted at any stage of the enforcement pipeline.

SECTION 1

Executive Overview

The QODIQA Reference Implementation Specification defines a minimal, self-contained enforcement stack that demonstrates the behavioral properties required by the QODIQA Core Standard. It is not a production deployment blueprint. It is a structured reference: a concrete instantiation of the enforcement model in its simplest correct form, against which production implementations can be calibrated.

The reference stack addresses a specific validation gap. Specifications describe what a system must do; reference implementations demonstrate how those requirements are satisfied in practice. Without a concrete reference, specification conformance is assessed against documentation alone, which is insufficient for systems whose correctness properties must be mechanically verifiable.

This document covers the complete enforcement pipeline from request ingestion to decision artifact generation. Each component is defined with explicit inputs, outputs, responsibilities, and failure behaviors. The enforcement flow is defined step by step with no ambiguous transitions. All failure modes resolve to denial. No component is permitted to infer, assume, or fall back silently.

The scope of this specification is deliberately bounded. It covers runtime consent validation, deterministic policy evaluation, fail-closed enforcement at the inference boundary, and append-only audit logging. It does not cover distributed deployment, high-availability configurations, advanced identity systems, cryptographic proof generation, or full production security controls. These are addressed in companion specifications within the QODIQA corpus.

The reference implementation serves three functions. First, it validates that the QODIQA enforcement model is implementable in a minimal configuration. Second, it provides a concrete behavioral reference for teams building

production systems. Third, it enables conformance testing by defining validation scenarios against which any implementation can be assessed.

SECTION 2

System Scope and Constraints

The boundary of this reference implementation is explicitly defined. Conformant behavior is only required within the stated scope. Behaviors outside this scope are not evaluated by this specification and must be addressed by production-specific architecture decisions and companion QODIQA documents.

2.1 Included

The following capabilities are within scope and must be implemented by any system claiming conformance with this reference specification:

- Runtime consent validation prior to AI inference execution
- Deterministic allow/deny decisioning based on explicit consent state and policy
- Fail-closed behavior: all undefined or error states resolve to DENY
- Append-only audit logging of all enforcement decisions
- Minimal evidence artifact generation for each decision
- Request validation including identity extraction and context parsing
- Consent registry with mock or in-memory state adequate for single-node operation
- Policy engine operating on immutable, versioned policy snapshots

2.2 Excluded

The following are explicitly outside scope and must not be assumed by any conformant implementation of this reference specification:

- Distributed consensus, replication, or multi-node coordination
- High-availability or fault-tolerant deployment configurations
- Advanced identity and authentication systems (PKI, hardware tokens, federated identity)
- Cryptographic proof generation or zero-knowledge constructs
- Full production security hardening, threat modeling, or penetration-tested configurations
- Scalability or throughput optimization
- AI model governance beyond the enforcement boundary
- User-facing consent capture, consent management interfaces, or consent workflow orchestration
- Legal or regulatory compliance verification

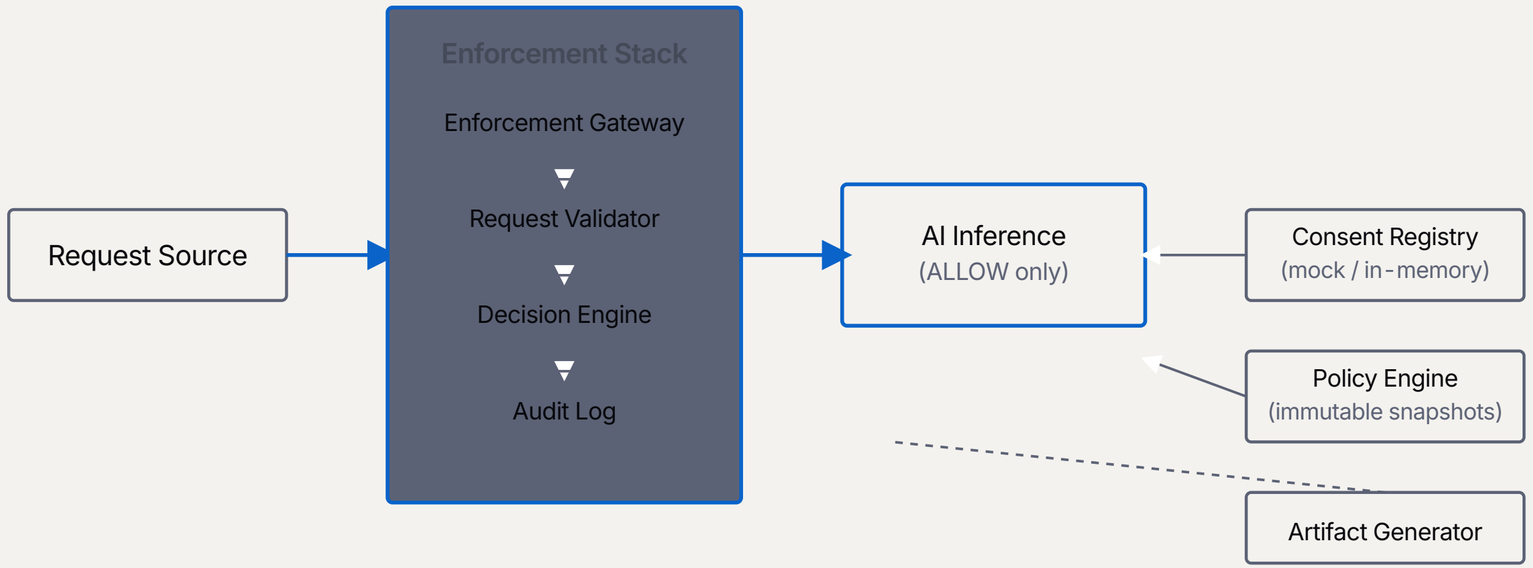
This scope boundary is a design constraint, not a limitation of the QODIQA framework. The minimal reference implementation is intentionally restricted to isolate and demonstrate core enforcement behavior without conflation with operational concerns. Production systems must extend this reference according to their deployment requirements and applicable QODIQA companion specifications.

SECTION 3

Architectural Overview

3.1 System Topology

The reference implementation operates as a single-node enforcement layer interposed between the request source and the AI inference execution environment. All inbound requests to the inference boundary must pass through the enforcement stack. No request may reach the inference layer without a positive ALLOW decision issued by the Decision Engine.



All paths through enforcement stack. No bypass permitted.

FIGURE 1. REFERENCE IMPLEMENTATION TOPOLOGY. ALL REQUESTS TRAVERSE THE ENFORCEMENT STACK. AI INFERENCE EXECUTES ONLY UPON RECEIPT OF A POSITIVE ALLOW DECISION. CONSENT REGISTRY AND POLICY ENGINE ARE READ-ONLY INPUTS TO THE DECISION ENGINE.

3.2 Trust Boundaries

The reference implementation defines two trust domains. The **external domain** encompasses all request sources, including clients, agents, orchestration layers, and upstream services. No entity in the external domain is trusted by default. All claims made by external entities, including identity assertions, scope declarations, and purpose statements, must be validated within the enforcement stack before any decision is issued.

The **internal domain** encompasses the enforcement stack components themselves: the Enforcement Gateway, Request Validator, Policy Engine, Consent Registry, Decision Engine, Audit Log, and Artifact Generator. These components are assumed to be co-located in a trusted runtime environment in the minimal reference configuration. In production deployments, inter-component trust must be explicitly secured; this is outside the scope of this specification.

The AI inference environment is treated as a downstream consumer of enforcement decisions. It is not part of the enforcement stack and must not be trusted to self-authorize execution. It receives only a signed decision artifact and must refuse to execute if no valid ALLOW artifact is present.

3.3 Enforcement Boundary

The enforcement boundary is the precise point at which the Decision Engine's output is applied. In this reference implementation, the boundary is defined as the interface between the Enforcement Gateway and the AI inference execution environment. No inference computation, model invocation, data retrieval, or output generation may occur prior to the receipt and validation of a positive decision artifact at this boundary.

> **CRITICAL INVARIANT**
>
> The enforcement boundary is absolute. Any architecture in which AI inference can be initiated without a valid ALLOW decision from this enforcement stack is non-conformant, regardless of any other controls present in the system.

SECTION 4

Core Components

Each component in the enforcement stack is defined below with its precise responsibility, input contract, output contract, and failure behavior. Components must not exceed their defined responsibilities. Cross-component calls must follow the defined flow sequence specified in Section 5.

4.1 Enforcement Gateway

Responsibility: The Enforcement Gateway is the single ingress point for all requests targeting the AI inference boundary. It receives raw inbound requests, performs initial structural validation, and routes conformant requests into the enforcement pipeline. It is responsible for rejecting malformed requests before they enter the pipeline and for enforcing the fail-closed contract at the system perimeter.

Inputs: Raw inbound request payload; caller network context.

Outputs: Structured request object passed to Request Validator; or a DENY response to caller with reason code `MALFORMED_REQUEST`.

> FAILURE BEHAVIOR
>
> If the inbound request cannot be parsed, is missing required structural fields, or exceeds defined payload constraints, the Gateway must immediately return DENY with reason code `MALFORMED_REQUEST`. No partial parsing or recovery is permitted. The failure event must be logged to the Audit Log before the response is returned.

4.2 Policy Engine (Deterministic)

Responsibility: The Policy Engine evaluates a resolved request context against an immutable policy snapshot to produce a binary policy result: PERMITTED or NOT_PERMITTED. It does not resolve consent state. It does not perform probabilistic or heuristic evaluation. Given identical inputs, it must produce identical outputs without exception.

Inputs: Validated request context object; immutable policy snapshot identified by policy version identifier.

Outputs: Policy result: `PERMITTED` or `NOT_PERMITTED`; policy version identifier used in evaluation; evaluation timestamp.

> FAILURE BEHAVIOR
>
> If the policy snapshot cannot be located, loaded, or parsed, the engine must return `NOT_PERMITTED` with reason code `POLICY_UNAVAILABLE`. If any policy rule evaluation produces an undefined result due to missing fields, the rule must be treated as NOT_PERMITTED. No default-allow, probabilistic, or inferred results are permitted.

4.3 Consent Registry (Mock / Minimal)

Responsibility: The Consent Registry provides a queryable store of consent records. In the minimal reference implementation, this may be an in-memory structure, a static file, or a simple key-value store. It does not capture consent; it reads consent state. Consent records must be pre-populated prior to system operation. The registry is read-only during enforcement execution.

Inputs: Consent lookup key comprising subject identifier, scope identifier, and purpose identifier.

Outputs: Consent object (see Section 8.1) if a valid, non-expired record exists; or a structured absence response: `CONSENT_NOT_FOUND` or `CONSENT_EXPIRED`.

> FAILURE BEHAVIOR
>
> If the registry is unavailable, if the lookup produces an error, or if the returned record fails structural validation, the registry must return `CONSENT_UNAVAILABLE`. The Decision Engine must treat all non-positive consent responses as denial conditions. There is no fallback consent state.

4.4 Request Validator

Responsibility: The Request Validator performs semantic validation of the structured request object produced by the Enforcement Gateway. It extracts and validates identity context, declared purpose, requested scope, and any timing or contextual constraints. It produces a validated request context object for consumption by the Decision Engine.

Inputs: Structured request object from the Enforcement Gateway.

Outputs: Validated request context object; or a structured validation failure with reason code.

> FAILURE BEHAVIOR
>
> If any required field is absent, malformed, or semantically invalid, the validator must return a validation failure with the appropriate reason code. Partial validation results are not permitted. The validation failure is forwarded to the Decision Engine, which must resolve it as DENY.

4.5 Decision Engine (Allow / Deny Only)

Responsibility: The Decision Engine is the central enforcement component. It receives a validated request context, queries the Consent Registry and Policy Engine, and produces a final binary enforcement decision: ALLOW or DENY. It does not perform probabilistic assessment, risk scoring, or conditional outcomes. It applies consent state and policy result to produce a deterministic decision artifact.

Inputs: Validated request context; Consent Registry response; Policy Engine result.

Outputs: Decision record (see Section 8.4): `ALLOW` or `DENY`; decision reason code; evaluation trace.

> FAILURE BEHAVIOR
>
> - Any consent response other than a valid, non-expired, scope-matched consent object → DENY
> - Any policy result other than `PERMITTED` → DENY
> - Any validation failure from Request Validator → DENY
> - Any internal error, timeout, or exception during evaluation → DENY with reason code `EVALUATION_ERROR`
> - Absence of any required input → DENY with reason code `MISSING_INPUT`

4.6 Audit Log (Append-Only)

Responsibility: The Audit Log records all enforcement decisions and system events in an append-only, tamper-evident log. Every decision produced by the Decision Engine must be written to the Audit Log before the decision is returned to the Enforcement Gateway. The log is a system-of-record for enforcement behavior.

Inputs: Decision record from Decision Engine; system event records from all components.

Outputs: None (write-only during enforcement). Log entries are available for read-only audit access through a separate interface not part of the enforcement pipeline.

> FAILURE BEHAVIOR
>
> If the Audit Log cannot accept a write, the Enforcement Gateway must return DENY to the caller and must not proceed with execution. A system that cannot audit its decisions must not make them. Log write failure is a hard enforcement failure condition, not a degraded mode.

4.7 Artifact Generator

Responsibility: The Artifact Generator produces a structured evidence artifact for each ALLOW decision. The artifact encodes the decision record, the consent record identifier, the policy version identifier, and the evaluation timestamp. It is attached to the execution context and forwarded to the AI inference environment as the sole authorization credential for the requested inference operation.

Inputs: Decision record (ALLOW decisions only); consent record identifier; policy version identifier.

Outputs: Decision artifact object; artifact identifier.

> FAILURE BEHAVIOR
>
> If artifact generation fails for any reason, the Enforcement Gateway must return DENY. No inference execution is permitted without a successfully generated and validated artifact. DENY decisions do not generate artifacts; they generate denial records in the Audit Log only.

SECTION 5

Deterministic Enforcement Flow

The enforcement flow is a linear, sequential pipeline. No step may be skipped, reordered, or parallelized in a manner that allows execution to proceed without a completed prior step. Each step is defined with its preconditions and the failure behavior that applies if the step cannot be completed successfully.

ENFORCEMENT PIPELINE: STEP SEQUENCE

STEP	ACTION	COMPONENT	FAILURE OUTCOME
1	Request ingestion and structural validation	Enforcement Gateway	DENY / MALFORMED_REQUEST
2	Identity and context extraction	Request Validator	DENY / INVALID_CONTEXT

STEP	ACTION	COMPONENT	FAILURE OUTCOME
3	Consent lookup by subject, scope, purpose	Consent Registry	DENY / CONSENT_NOT_FOUND or CONSENT_EXPIRED
4	Policy evaluation against versioned snapshot	Policy Engine	DENY / NOT_PERMITTED or POLICY_UNAVAILABLE
5	Decision synthesis (ALLOW / DENY)	Decision Engine	DENY / EVALUATION_ERROR
6	Enforcement at inference boundary	Enforcement Gateway	DENY / ARTIFACT_MISSING
7	Audit log write and artifact generation	Audit Log + Artifact Generator	DENY / LOG_WRITE_FAILURE

Step 1: Request Ingestion

The Enforcement Gateway receives the raw inbound request. It validates the structural envelope: required fields present, payload within size bounds, content type declared. Requests that fail structural validation are rejected immediately with DENY and reason code `MALFORMED_REQUEST`. Conformant requests are passed as structured objects to the Request Validator.

Step 2: Identity and Context Extraction

The Request Validator extracts and validates the following fields from the structured request: subject identifier (who is acting), declared purpose (why the action is performed), requested scope (what data or capability is targeted), and request timestamp. All fields are mandatory. Any absent or malformed field causes the validator to return a validation failure with the corresponding reason code.

Step 3: Consent Lookup

The Decision Engine queries the Consent Registry using the composite key: subject identifier, scope identifier, purpose identifier. The registry returns the consent object if a valid, non-expired, scope-matched record exists. Any other

response, including absence, expiry, scope mismatch, or registry error, is treated as a denial condition. The Decision Engine does not proceed to policy evaluation if consent is not positively confirmed.

Step 4: Policy Evaluation

The Policy Engine is invoked with the validated request context and the current policy version identifier. It evaluates all applicable policy rules from the immutable policy snapshot. The result is `PERMITTED` only if all applicable rules are satisfied. If any rule is not satisfied, if any rule produces an undefined result, or if the policy snapshot cannot be loaded, the result is `NOT_PERMITTED`.

Step 5: Decision Synthesis

The Decision Engine synthesizes the final enforcement decision. ALLOW requires: a valid consent object returned from the Consent Registry, and a `PERMITTED` result from the Policy Engine, and no validation failures from the Request Validator. If any of these conditions is not met, the decision is DENY. The decision record is produced with a complete evaluation trace including all inputs, their resolution, and the final outcome.

Step 6: Enforcement at Inference Boundary

The Enforcement Gateway applies the decision at the inference boundary. If the decision is ALLOW, the decision artifact is attached to the execution context and inference execution is permitted. If the decision is DENY, inference execution is blocked and the denial reason is returned to the caller. The inference environment must verify the artifact before accepting any execution request.

Step 7: Logging and Artifact Generation

All enforcement decisions, including denials, are written to the Audit Log before the response is returned to the caller. ALLOW decisions additionally trigger artifact generation. The artifact contains the decision record, consent record

identifier, policy version, and evaluation timestamp. If the Audit Log write fails, the Enforcement Gateway returns DENY regardless of the decision outcome.

SECTION 6

Fail-Closed Guarantees

Fail-closed behavior is a core invariant of the QODIQA enforcement model. It is not a configuration option or a default setting. It is a structural property that must hold unconditionally across all operating conditions.

> **FAIL-CLOSED DEFINITION**
>
> Any condition that cannot be positively resolved to an ALLOW decision must result in DENY. This includes errors, timeouts, missing inputs, undefined states, partial results, and system failures. There is no degraded mode. There is no default-allow fallback. There is no ambiguity state that permits execution.

Explicit Denial Conditions

The following conditions must each individually produce DENY with the corresponding reason code:

DENIAL CONDITION REGISTRY

CONDITION	REASON CODE
Request structurally malformed	`MALFORMED_REQUEST`
Subject identifier absent or invalid	`INVALID_SUBJECT`
Purpose not declared	`PURPOSE_MISSING`

CONDITION	REASON CODE
Scope not declared or invalid	SCOPE_INVALID
No consent record found	CONSENT_NOT_FOUND
Consent record expired	CONSENT_EXPIRED
Consent scope does not match request scope	CONSENT_SCOPE_MISMATCH
Consent purpose does not match declared purpose	CONSENT_PURPOSE_MISMATCH
Policy not found or unavailable	POLICY_UNAVAILABLE
Policy evaluation result NOT_PERMITTED	POLICY_DENIED
Decision Engine internal error	EVALUATION_ERROR
Audit Log write failure	LOG_WRITE_FAILURE
Artifact generation failure	ARTIFACT_GENERATION_FAILURE
Any unclassified system error	SYSTEM_ERROR

System Failure Handling

System failures, including process crashes, memory exhaustion, network timeouts between components, and unhandled exceptions, must all resolve to DENY. No component is permitted to swallow exceptions and proceed. No component is permitted to produce an ambiguous or partial result that allows downstream components to interpret the outcome as ALLOW. Every error boundary must terminate the request with DENY.

No Silent Fallback

Silent fallback, defined as any behavior that permits execution to proceed without a positive enforcement decision, is categorically prohibited. This includes: treating an absent consent record as implied consent; applying a default policy when a named policy is unavailable; proceeding with inference when the Audit Log is unavailable; or returning ALLOW when evaluation is incomplete. Any such behavior constitutes a conformance failure.

SECTION 7

Audit and Evidence Model

What Is Logged

Every enforcement event must be logged. The following event types must each produce a log entry:

- Request received by Enforcement Gateway (structural outcome: accepted / rejected)
- Validation result from Request Validator (outcome and reason code)
- Consent lookup result (outcome, consent record identifier if found, reason code if not)
- Policy evaluation result (outcome, policy version, applicable rule identifiers)
- Final decision from Decision Engine (ALLOW or DENY, complete reason chain)
- Audit Log write confirmation or failure
- Artifact generation outcome (for ALLOW decisions)

Log Entry Format

Each log entry must include at minimum: event type identifier; timestamp (ISO 8601, UTC); request identifier; subject identifier; decision outcome; reason code; policy version identifier; consent record identifier (if applicable). Log entries

must be structured and machine-parseable. Free-form text fields are prohibited in mandatory fields.

LOG ENTRY SCHEMA (MINIMAL)

```
{
  "event_type":        string,      // GATEWAY_RECEIVED | VALIDATION_RESULT | CONSENT_LOOKUP
                                    // POLICY_EVAL | DECISION | ARTIFACT_GENERATED
  "timestamp":         ISO8601/UTC,
  "request_id":        uuid,
  "subject_id":        string,
  "decision":          "ALLOW" | "DENY",
  "reason_code":       string,      // from Denial Condition Registry
  "policy_version":    string,
  "consent_record_id": string | null,
  "artifact_id":       string | null  // present on ALLOW decisions only
}
```

Immutability Assumptions

In the minimal reference implementation, append-only immutability is implemented by prohibiting any delete, update, or overwrite operation on the log store. Log entries are written sequentially with a monotonically increasing sequence number. In production systems, cryptographic immutability guarantees must be applied; this is specified in the QODIQA Security and Cryptographic Profile.

Evidence Generation Triggers

Decision artifacts are generated only for ALLOW decisions. Artifacts are not generated for DENY decisions; the Audit Log entry for a DENY decision constitutes the full evidence record. Artifact generation is triggered immediately after the Decision Engine issues an ALLOW decision and before the decision is returned to the Enforcement Gateway.

SECTION 8

Minimal Data Model

8.1 Consent Object

CONSENT OBJECT SCHEMA

```
{
  "consent_id":       uuid,          // unique record identifier
  "subject_id":       string,        // data subject or acting entity identifier
  "scope":            string[],      // list of authorized scope identifiers
  "purpose":          string[],      // list of authorized purpose identifiers
  "granted_at":       ISO8601/UTC,   // timestamp of consent grant
  "expires_at":       ISO8601/UTC,   // hard expiry; null not permitted
  "revoked":          boolean,       // true if consent has been revoked
  "revoked_at":       ISO8601/UTC | null,
  "grantor_id":       string         // identity of consent-granting entity
}
```

A consent object is valid if and only if: `revoked` is `false`, the current timestamp is before `expires_at`, the requested scope is contained within the `scope` array, and the declared purpose is contained within the `purpose` array. Any condition not met renders the consent object invalid for the given request.

8.2 Request Context

REQUEST CONTEXT SCHEMA

```
{
  "request_id":     uuid,          // unique request identifier, caller-supplied
  "subject_id":     string,        // identity of the acting entity
```

```
  "declared_purpose": string,      // single purpose identifier for this request
  "requested_scope":  string,      // single scope identifier for this request
  "timestamp":      ISO8601/UTC,   // request creation timestamp
  "caller_context": {
    "source_ip":    string | null,
    "agent_id":     string | null   // optional upstream agent identifier
  }
}
```

8.3 Policy Structure

POLICY SNAPSHOT SCHEMA

```
{
  "policy_id":      string,        // unique policy identifier
  "version":        string,        // semver or monotonic version string
  "effective_at":   ISO8601/UTC,
  "rules": [
    {
      "rule_id":    string,
      "scope":      string,        // scope this rule applies to
      "purpose":    string,        // purpose this rule applies to
      "condition":  object,        // structured condition expression (no free text)
      "effect":     "PERMIT" | "DENY"
    }
  ]
}
```

Policy snapshots are immutable once published. The Policy Engine must load the snapshot identified by the current policy version and evaluate all applicable rules. Modifications to published snapshots are prohibited. New versions must be published as distinct snapshots with new version identifiers.

8.4 Decision Record

DECISION RECORD SCHEMA

```
{
  "decision_id":        uuid,
  "request_id":         uuid,
  "subject_id":         string,
  "decision":           "ALLOW" | "DENY",
  "reason_code":        string,
  "consent_record_id":  uuid | null,
  "policy_version":     string,
  "evaluation_trace": {
    "validation_result":  "VALID" | "INVALID",
    "consent_result":     "FOUND" | string,    // reason code if not FOUND
    "policy_result":      "PERMITTED" | string // reason code if not PERMITTED
  },
  "decided_at":         ISO8601/UTC,
  "artifact_id":        uuid | null            // populated for ALLOW decisions only
}
```

SECTION 9

Deployment Model

The reference implementation is designed for single-node deployment. All components run within a single process or within a single trusted host environment. No distributed coordination, consensus protocol, or network-separated component topology is required or assumed.

SINGLE-NODE ACCEPTABILITY

A single-node deployment is fully conformant with this specification. The reference implementation does not require replication, load balancing, or distributed storage. These are production concerns addressed in companion specifications.

The Consent Registry may be implemented as an in-memory key-value store seeded from a static file at startup. The Audit Log may be implemented as a local append-only file with sequential write semantics. The Policy Engine loads policy snapshots from a local filesystem path at initialization. No external service dependencies are required for baseline operation.

The runtime boundary is defined as: the enforcement stack process, the consent registry data source, the policy snapshot store, and the audit log destination. All must be present and operational before the Enforcement Gateway accepts any inbound requests. A startup check must verify all component availability and must refuse to accept requests if any component is unavailable.

Network exposure of the enforcement stack is outside the scope of this specification. In production deployments, the enforcement stack must be protected from unauthorized access; this is addressed in the QODIQA Security and Cryptographic Profile.

SECTION 10

Non-Goals

The following are explicitly outside the scope of this specification. An implementation that addresses these areas is not prohibited, but conformance with this specification does not require them and does not assess them.

SCALABILITY

This specification makes no performance or throughput requirements. Horizontal scaling, load distribution, caching, and latency optimization are not addressed and must not be assumed from this reference.

FULL PRODUCTION SECURITY

This specification does not define a production security posture. Authentication of callers, transport security, component isolation, secret management, and hardening against adversarial inputs are outside scope and are addressed in the QODIQA Security and Cryptographic Profile.

DISTRIBUTED SYSTEMS

No distributed consensus, eventual consistency, partition tolerance, or multi-node coordination is defined or required. These concerns are addressed in the QODIQA Reference Architecture for Deterministic Runtime Consent Enforcement.

ADVANCED IDENTITY SYSTEMS

This specification uses opaque string identifiers for subjects and grantors. PKI, federated identity, hardware attestation, and verifiable credentials are not addressed and must not be assumed from the minimal data model defined herein.

CONSENT CAPTURE AND LIFECYCLE MANAGEMENT

This specification addresses consent enforcement only. Consent capture, revocation workflows, consent versioning, and user-facing consent management interfaces are outside scope.

SECTION 11

Conformance Alignment

This reference implementation is designed to demonstrate compliance with the QODIQA Core Standard and the 68-Point Enforcement Framework. The following table maps key implementation behaviors to their governing framework principles.

CONFORMANCE MAPPING: IMPLEMENTATION TO QODIQA FRAMEWORK

IMPLEMENTATION BEHAVIOR	QODIQA CORE INVARIANT	68-POINT FRAMEWORK
All requests pass through Enforcement Gateway before inference	Enforcement boundary is absolute	Points 1–4: Boundary enforcement
Consent explicitly resolved before policy evaluation	Consent separation from policy	Points 12–18: Consent resolution
Deterministic decision: ALLOW or DENY only	Binary decisioning, no ambiguity	Points 19–26: Decision determinism
All undefined states resolve to DENY	Fail-closed semantics	Points 27–34: Fail-closed guarantees
Every decision written to append-only log before response	Auditability is unconditional	Points 40–48: Audit requirements
Policy evaluated against immutable versioned snapshot	Policy immutability	Points 49–55: Policy integrity
ALLOW decisions produce signed decision artifacts	Verifiable authorization	Points 56–62: Evidence model

SECTION 12

Validation Scenarios

The following scenarios constitute the minimal conformance test suite for this reference implementation. Any conformant implementation must produce the specified outcome for each scenario. These scenarios may be executed as automated tests against a running implementation.

Scenario 1 · Valid Consent, Permitted Policy: ALLOW

> PRECONDITIONS
>
> - Consent record exists for subject, scope, and purpose
> - Consent record is not expired and not revoked
> - Policy snapshot is loaded and applicable rule effect is PERMIT

Expected outcome: Decision Engine issues ALLOW. Decision artifact is generated. Audit Log entry written with decision ALLOW. Inference execution is permitted.

Scenario 2 · Missing Consent: DENY

> PRECONDITIONS
>
> - No consent record exists for the given subject, scope, and purpose combination
> - Policy snapshot is otherwise valid

Expected outcome: Consent Registry returns `CONSENT_NOT_FOUND`. Decision Engine issues DENY with reason code `CONSENT_NOT_FOUND`. No artifact is generated. Audit Log entry written with decision DENY. Inference execution is blocked.

Scenario 3 · Invalid Scope: DENY

> PRECONDITIONS
>
> - Consent record exists but does not include the requested scope identifier
> - All other inputs are valid

Expected outcome: Decision Engine issues DENY with reason code `CONSENT_SCOPE_MISMATCH`. No artifact is generated. Audit Log entry written with decision DENY. Inference execution is blocked.

Scenario 4 · System Failure: DENY

> PRECONDITIONS
>
> - Consent Registry is unavailable (simulated by shutting down registry component)
> - Request is otherwise well-formed and valid

Expected outcome: Decision Engine issues DENY with reason code `CONSENT_UNAVAILABLE`. No artifact is generated. System does not fall back to implicit consent or default-allow. Audit Log entry written if log is available; if Audit Log is also unavailable, DENY is still returned and system records the failure in its internal error state.

Scenario 5 · Expired Consent: DENY

> PRECONDITIONS
>
> - Consent record exists but `expires_at` is in the past
> - All other inputs are valid

Expected outcome: Consent Registry returns `CONSENT_EXPIRED`. Decision Engine issues DENY with reason code `CONSENT_EXPIRED`. Inference execution is blocked.

SECTION 13

Conclusion

This specification defines a minimal, self-contained reference implementation of the QODIQA enforcement model. It is not a production system. It is the smallest correct instantiation of the enforcement properties required by the QODIQA Core Standard, expressed in concrete component definitions, data structures, flow sequences, and validation scenarios.

The reference implementation demonstrates that deterministic runtime consent enforcement is achievable in a minimal configuration. Seven components are sufficient to implement the complete enforcement pipeline: a gateway, a validator, a consent registry, a policy engine, a decision engine, an audit log, and an artifact generator. Each has a precisely bounded responsibility. The interaction between them is fully specified. All failure modes resolve to denial. No ambiguity is permitted at any stage.

This specification serves as the bridge between the QODIQA Core Standard's normative requirements and the operational realities of production deployment. Teams building production enforcement systems should use this reference as their behavioral baseline, validating each component against the scenarios defined in Section 12 before extending the stack with production-grade capabilities: distributed deployment, cryptographic proof generation, high-availability configurations, and advanced identity integration.

The enforcement model defined here is intentionally minimal, but it is not simplified. Every component and every constraint is present because removing it would compromise the correctness of the enforcement guarantee. Fail-closed behavior, audit unconditionally, consent-before-policy, and binary decisioning are not optional features; they are the properties that make the enforcement model trustworthy. Production implementations may add capabilities above this baseline, but they must not remove any of its behavioral guarantees.

Runtime consent enforcement represents a structural shift in how AI systems are governed: from documentation and retrospective accountability toward execution-layer control and verifiable authorization. This reference implementation is a concrete demonstration that the shift is achievable and that its core properties can be implemented, tested, and validated with precision.

DOCUMENT STATUS

Document Status and Corpus Alignment

This document is the Reference Implementation Specification of the QODIQA standard corpus. It defines a minimal, deterministic enforcement stack demonstrating runtime consent validation, fail-closed decisioning, and verifiable audit logging at the AI inference boundary.

This specification is normative within its defined scope. It provides the concrete behavioral reference against which production implementations of the QODIQA enforcement model are validated. It should be read in conjunction with the following documents:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement
- QODIQA: Terminology and Normative Definitions
- QODIQA: Threat Model and Abuse Case Specification
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement

— QODIQA: Governance Charter for the QODIQA Standard Corpus

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dițescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-RIS-2026-001
TITLE	Reference Implementation Specification (Minimal Deterministic Enforcement Stack)
SUBTITLE	Minimal Deterministic Enforcement Stack for Runtime Consent Validation
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Reference Implementation Specification
DOCUMENT STATUS	Normative
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 15 PART III · OPERATIONAL DOCUMENTS

Formal Enforcement Model Deterministic Execution Semantics

Formal definition of execution semantics, state machine constraints, decision invariants, and non-compliance conditions for the runtime enforcement layer.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-FEM-2026-001	1.0	2026	Formal Specification	Normative Extension	NORMATIVE

ABSTRACT

This document formalizes a deterministic execution model for runtime consent enforcement, defining a closed state machine, a total decision function, and invariant-based compliance constraints.

SECTION 1

Purpose and Scope

This document defines formal execution semantics for the runtime consent enforcement layer. Its purpose is to eliminate ambiguity in the execution model by specifying all permitted states, transitions, decision logic, and enforcement invariants in a binary, testable, and auditable form.

This document does not modify, override, or reinterpret any existing QODIQA specification. It provides formalization, constraint definition, and deterministic execution semantics only.

The scope of this document is strictly limited to the execution semantics layer. This document SHALL NOT be construed to alter the behavioral requirements, architectural definitions, governance structure, or certification criteria established in other QODIQA corpus documents. All normative content herein is additive and subordinate to the QODIQA Core Standard.

> **SCOPE BOUNDARY**
>
> This document defines: execution states, state transitions, decision function constraints, enforcement invariants, and non-compliance classification. This document does not define: consent capture mechanisms, policy authoring, cryptographic protocols, certification criteria, or governance procedures.

> **NON-GOALS**
>
> This document does not define consent acquisition mechanisms, user interfaces, policy authoring systems, or cryptographic primitives. Its scope is strictly limited to execution semantics.

SECTION 2

Relationship to QODIQA Corpus

This document is a normative extension to the QODIQA corpus. It formalizes the execution semantics of enforcement logic that is defined in substantive terms across the following corpus documents. All definitions and constraints herein are derived from and consistent with those documents.

TABLE 1: CORPUS DOCUMENT RELATIONSHIPS

DOCUMENT	IDENTIFIER	RELATIONSHIP TO THIS DOCUMENT
Core Standard	QODIQA-CORE-2026-001	Primary normative authority. This document formalizes its enforcement logic.
Reference Architecture	QODIQA-RA-2026-001	Defines the architectural components whose execution semantics are specified herein.

DOCUMENT	IDENTIFIER	RELATIONSHIP TO THIS DOCUMENT
Implementation Playbook	QODIQA-IMPL-2026-001	Provides implementation guidance consistent with the execution model defined herein.
68-Point Enforcement Framework	QODIQA-SPEC-68P-V3-2026	Enumerates enforcement points that SHALL conform to the execution semantics herein.
Conformance Test Suite	QODIQA-CTSS-2026-001	Test criteria SHALL be derived from the invariants and constraints defined herein.
Security and Cryptographic Profile	QODIQA-SCP-2026-001	Defines integrity requirements for state and audit artifacts referenced herein.

In the event of apparent conflict between this document and any other QODIQA corpus document, the QODIQA Core Standard SHALL take precedence. All conflicts SHALL be reported to the QODIQA Governance Authority for resolution.

SECTION 3

Deterministic Execution Model

The enforcement layer SHALL operate as a deterministic finite-state machine. Every execution path through the enforcement layer SHALL be fully defined, fully traceable, and free of implicit or undefined behavior.

> **DETERMINISM REQUIREMENT**
>
> For any given input triple (Request, ConsentState, PolicySet), the enforcement layer SHALL produce exactly one output. No execution path SHALL exist that produces more than one output, no output, or a probabilistic output for identical inputs.

The enforcement layer SHALL NOT contain ambiguous execution paths. An ambiguous execution path is defined as any path whose outcome depends on factors not present in the explicit input triple (Request, ConsentState, PolicySet) at the moment of evaluation.

Implicit behavior is defined as any system behavior that is not explicitly specified by a defined state, a defined transition, or a defined constraint. Implicit behavior is prohibited. Every behavior of the enforcement layer SHALL be the direct result of an explicitly defined execution step.

The enforcement layer SHALL NOT defer evaluation to external components during the active decision path. All inputs required for evaluation SHALL be fully resolved prior to the initiation of the `PolicyEvaluation` state. Any failure to resolve required inputs SHALL result in an immediate transition to the `Deny` decision state.

SECTION 4

Canonical State Machine

The enforcement layer SHALL implement precisely the state machine defined in this section. No implementation SHALL add states, remove states, merge states, or reorder states relative to the canonical definition below.

STATE MACHINE: FORMAL DEFINITION

```
StateMachine = (S, T, s0, F)

where:
  S  := {S1, S2, S3, S4a, S4b, S5}
  T  := set of allowed transitions defined in Table 3
  s0 := S1 (initial state)
  F  := {S5} (terminal state)
```

4.1 Defined States

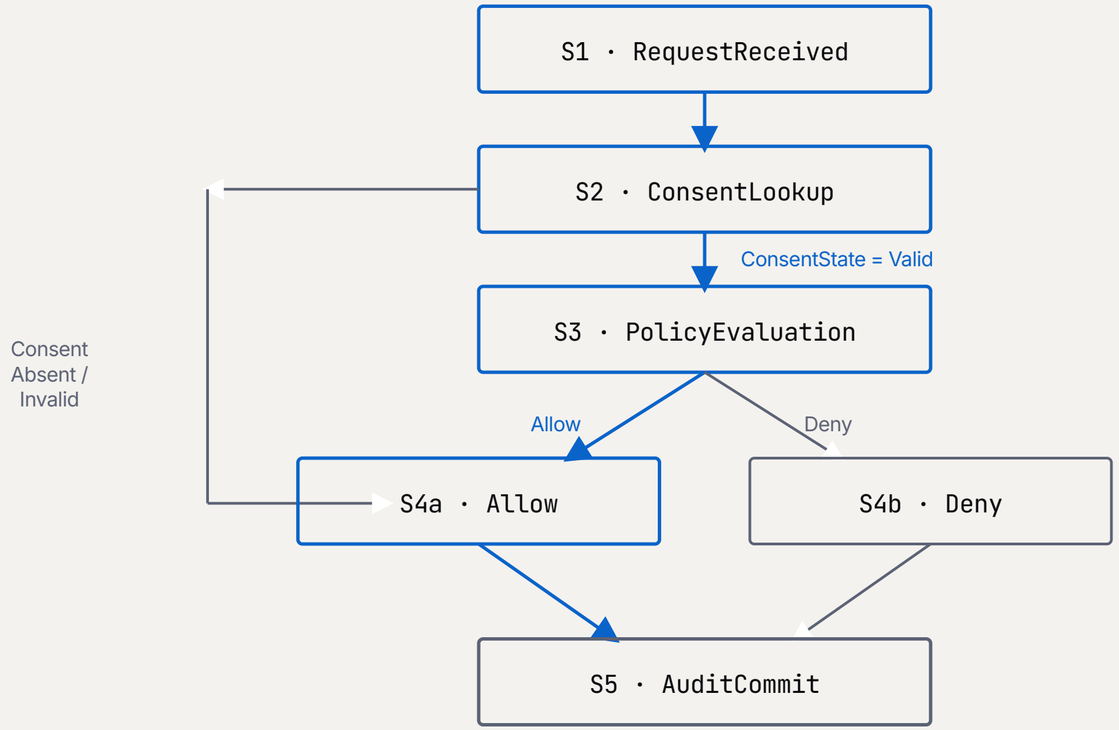
The canonical state machine consists of exactly five (5) states. All five states are mandatory. No state is optional.

TABLE 2: CANONICAL EXECUTION STATES

STATE ID	STATE NAME	CLASSIFICATION	DESCRIPTION
S1	RequestReceived	Entry State	The enforcement layer has received a complete, well-formed execution request. No evaluation has commenced.
S2	ConsentLookup	Intermediate State	The enforcement layer is resolving the current consent state for the declared subject, purpose, and scope. No policy evaluation has commenced.
S3	PolicyEvaluation	Intermediate State	The enforcement layer is evaluating the applicable policy set against the resolved consent state and the declared request parameters.
S4a	Decision: Allow	Terminal Decision State	The enforcement layer has determined that the request satisfies all consent and policy requirements. Execution MAY proceed.
S4b	Decision: Deny	Terminal Decision State	The enforcement layer has determined that the request does not satisfy one or more consent or policy requirements. Execution MUST NOT proceed.
S5	AuditCommit	Terminal Audit State	The enforcement layer has committed an immutable, tamper-evident audit record of the decision, its inputs, and its evaluation context.

4.2 Allowed Transitions

The following transitions are the complete and exclusive set of permitted state transitions. No transition not listed in this section is permitted.



All paths terminate at S5. No terminal state is exempt from AuditCommit.

FIGURE 1. CANONICAL QODIQA ENFORCEMENT STATE MACHINE. ALL PERMITTED TRANSITIONS ARE SHOWN. ANY TRANSITION NOT DEPICTED IS PROHIBITED.

TABLE 3: PERMITTED STATE TRANSITIONS

FROM STATE	TO STATE	CONDITION
`S1 · RequestReceived`	S2 · ConsentLookup	Request is well-formed and complete.
`S2 · ConsentLookup`	S3 · PolicyEvaluation	ConsentState resolves to Valid.
`S2 · ConsentLookup`	S4b · Deny	ConsentState resolves to Absent, Expired, Revoked, or Invalid.
`S3 · PolicyEvaluation`	S4a · Allow	All applicable policy rules evaluate to satisfied.
`S3 · PolicyEvaluation`	S4b · Deny	One or more applicable policy rules evaluate to unsatisfied.
`S4a · Allow`	S5 · AuditCommit	Unconditional. No exception is permitted.
`S4b · Deny`	S5 · AuditCommit	Unconditional. No exception is permitted.

4.3 Prohibited Transitions

The following transition categories are explicitly and unconditionally prohibited. Any system that implements one or more of these transitions is non-compliant with this specification.

- Any transition that bypasses `ConsentLookup` on the path from `RequestReceived` to `PolicyEvaluation`.
- Any transition from `RequestReceived` directly to `PolicyEvaluation`, `Allow`, `Deny`, or `AuditCommit`.
- Any transition from `ConsentLookup` directly to `Allow` or `AuditCommit`.
- Any transition from `PolicyEvaluation` directly to `AuditCommit` without passing through a terminal `Decision` state.
- Any transition from `Allow` or `Deny` to any state other than `AuditCommit`.
- Any cyclic transition that returns a state to a prior position in the sequence.
- Any parallel transition that places the machine in two states simultaneously.

- Any transition to an undefined state not listed in Section 4.1.

> **TERMINATION GUARANTEE**
>
> All execution paths SHALL terminate in a finite number of steps at the AuditCommit state. Infinite execution paths or non-terminating evaluations are strictly prohibited.

SECTION 5

Deterministic Decision Function

5.1 Formal Definition

The enforcement decision function is formally defined as follows:

DECISION FUNCTION: FORMAL DEFINITION

```
Decision = f(Request, ConsentState, PolicySet)

where:
  Request      := { Subject, Purpose, Scope, DataCategory, Timestamp }
  ConsentState := { Valid | Absent | Expired | Revoked | Invalid }
  PolicySet    := { P1, P2, ..., Pn } (immutable snapshot at evaluation time)
  Decision     := { Allow | Deny }
```

The function `f` is a total function. For every valid combination of (Request, ConsentState, PolicySet), `f` SHALL produce exactly one element of the set {Allow, Deny}. The function SHALL NOT be partial, probabilistic, or undefined for any input within its specified domain.

5.2 Input Constraints

The following constraints apply to all inputs to the decision function. A request that violates any input constraint SHALL result in an immediate `Deny` decision without entering `PolicyEvaluation`.

- The `Request` SHALL be complete. A request is complete if and only if all five fields (Subject, Purpose, Scope, DataCategory, Timestamp) are present and non-null.
- The `ConsentState` SHALL be resolved prior to the initiation of `PolicyEvaluation`. An unresolved `ConsentState` SHALL be treated as `Invalid`.
- The `PolicySet` SHALL be an immutable snapshot captured at the moment `PolicyEvaluation` commences. The `PolicySet` SHALL NOT change during evaluation.
- The `PolicySet` SHALL NOT be empty. A request evaluated against an empty `PolicySet` SHALL result in a `Deny` decision.
- All inputs SHALL be evaluated from explicit, machine-readable state. No input SHALL be inferred, assumed, or inherited from prior execution contexts.

5.3 Output Constraints

The decision function output SHALL satisfy the following constraints unconditionally.

- The output SHALL be exactly one of: `Allow` or `Deny`. No other output value is permitted.
- For identical inputs (Request, ConsentState, PolicySet), the function SHALL produce identical output on every invocation. Non-deterministic output for identical inputs constitutes a non-compliance condition.
- The output SHALL NOT depend on execution time, invocation count, system load, external network state, or any factor not present in the explicit input triple.
- The output SHALL NOT be null, undefined, pending, or deferred.

- The output SHALL be produced in strict sequence following completion of `PolicyEvaluation`. No asynchronous decision emission is permitted.

> **DETERMINISM GUARANTEE**
>
> The decision function is strictly deterministic. No entropy source, probabilistic mechanism, or external state dependency SHALL influence the output for identical inputs under any condition.

SECTION 6

Execution Flow Constraints

The following constraints govern the execution flow of the enforcement layer. Each constraint is absolute and admits no exception.

> **STRICT SEQUENTIAL EVALUATION**
>
> States SHALL be entered and exited in the sequence defined in Section 4. No state SHALL be entered before its predecessor state has been fully completed and exited.

> **NO STAGE SKIPPING**
>
> No implementation SHALL skip, abbreviate, or combine any defined state. The `ConsentLookup` state SHALL NOT be skipped under any condition, including prior consent validation, cached consent state, or operator override.

> **NO PARTIAL EVALUATION**
>
> Each state SHALL be fully completed before the transition to the subsequent state is initiated. Partial completion of a state is classified as an invalid execution state.

NO ASYNCHRONOUS BYPASS

No asynchronous mechanism SHALL be used to initiate a transition before the current state has completed its defined processing. Asynchronous callbacks, deferred evaluations, and event-driven state bypasses are prohibited within the enforcement execution path.

NO FALLBACK LOGIC

No fallback, default-allow, best-effort, or degraded evaluation mode is permitted. When a required input is unavailable, the enforcement layer SHALL immediately produce a `Deny` decision and proceed to `AuditCommit`.

NO EXTERNAL DEPENDENCIES DURING EVALUATION

The active decision path from `ConsentLookup` through `AuditCommit` SHALL NOT initiate synchronous calls to external systems. All required state SHALL be resolved prior to entering `ConsentLookup`.

NO SILENT CONTINUATION

Any execution error, state resolution failure, or evaluation exception that occurs during the enforcement path SHALL terminate evaluation immediately and produce a `Deny` decision. Errors SHALL NOT be silently suppressed or deferred.

FAILURE SEMANTICS

Any failure in input resolution, state transition, or evaluation SHALL result in immediate termination of execution and emission of a Deny decision. No retry, fallback, or degraded execution mode is permitted.

This section defines execution constraints in a strictly formal manner. Implementation strategies may vary, provided that all constraints, invariants, and state transition rules remain fully preserved without deviation.

SECTION 7

Formal Enforcement Invariants

The following invariants are absolute. Each invariant SHALL hold under all conditions, for all requests, across all deployment configurations. No invariant admits an exception, a grace period, or a configuration-dependent override.

7.1 Consent Invariants

INV-C-01

IF ConsentState ≠ Valid THEN Decision = Deny

INV-C-02

NO execution path SHALL exist where Decision = Allow AND ConsentState = Absent

INV-C-03

NO execution path SHALL exist where Decision = Allow AND ConsentState = Expired

INV-C-04

NO execution path SHALL exist where Decision = Allow AND ConsentState = Revoked

INV-C-05

NO execution path SHALL exist where Decision = Allow AND ConsentState = Invalid

INV-C-06

ConsentState SHALL be resolved as an explicit, machine-readable value. ConsentState SHALL NOT be inferred, assumed, or inherited.

7.2 Audit Invariants

INV-A-01

AuditCommit SHALL occur for every Decision, regardless of Decision value.

INV-A-02

NO execution path SHALL terminate at Decision (Allow or Deny) without subsequent AuditCommit.

INV-A-03

The AuditCommit record SHALL include: Decision value, ConsentState value, PolicySet snapshot identifier, Request parameters, and evaluation Timestamp.

INV-A-04

The AuditCommit record SHALL be immutable after commit. Post-commit modification of any audit field is prohibited.

INV-A-05

AuditCommit SHALL occur synchronously within the same execution context as the Decision. Deferred or asynchronous audit commit is prohibited.

INV-A-06

Decision SHALL precede AuditCommit in strict execution order.

INV-A-07

Decision and AuditCommit SHALL occur within the same execution context and MUST be atomic.

AUDIT TRACEABILITY

Every audit record MUST be uniquely attributable to a single execution instance and MUST allow full reconstruction of the decision context without reliance on external or implicit state.

7.3 Sequencing Invariants

INV-S-01

PolicyEvaluation SHALL NOT execute without prior completed ConsentLookup.

INV-S-02

ConsentLookup SHALL NOT execute without prior completed RequestReceived.

INV-S-03

Decision SHALL NOT be emitted without prior completed PolicyEvaluation, except where ConsentState ≠ Valid, in which case Decision = Deny SHALL be emitted immediately upon ConsentLookup completion.

INV-S-04

AuditCommit SHALL NOT execute before Decision has been produced.

INV-S-05

No state in the sequence SHALL execute more than once per request.

7.4 Decision Invariants

INV-D-01

f(Request, ConsentState, PolicySet) = f(Request, ConsentState, PolicySet) for all invocations with identical inputs.

INV-D-02

Decision SHALL be exactly one of {Allow, Deny}. No other value is a valid decision output.

INV-D-03

Decision SHALL NOT be null, deferred, or pending.

INV-D-04

PolicySet SHALL be immutable during PolicyEvaluation. Any modification to PolicySet after the commencement of PolicyEvaluation SHALL constitute a non-compliance condition.

INV-D-05

An empty PolicySet SHALL produce Decision = Deny.

INV-D-06

A malformed or incomplete Request SHALL produce Decision = Deny without entering ConsentLookup.

SECTION 8

Invalid Execution States

The following execution conditions are classified as non-compliant. Any system that enters or permits any of the following states is non-compliant with this specification. Non-compliant systems SHALL NOT be certified under the QODIQA Certification Framework.

TABLE 4: INVALID EXECUTION STATE CLASSIFICATION

ID	INVALID STATE	CLASSIFICATION
IES-01	Undefined state transition: any transition not listed in Table 3.	NON-COMPLIANT
IES-02	Missing evaluation step: any execution that reaches Decision without having completed all prior mandatory states.	NON-COMPLIANT
IES-03	Partial execution path: any execution that enters a state but does not fully complete it before transitioning.	NON-COMPLIANT
IES-04	Inconsistent state combination: simultaneous occupancy of two or more states.	NON-COMPLIANT
IES-05	Decision without AuditCommit: any execution path that terminates at a Decision state without subsequent AuditCommit.	NON-COMPLIANT
IES-06	ConsentLookup skipped: any execution that transitions from RequestReceived to PolicyEvaluation without completing ConsentLookup.	NON-COMPLIANT
IES-07	PolicyEvaluation against mutable PolicySet: any evaluation where the PolicySet changes during PolicyEvaluation.	NON-COMPLIANT
IES-08	Silent fallback: any execution that silently substitutes a default or degraded decision in place of a Deny upon input resolution failure.	NON-COMPLIANT
IES-09	Asynchronous Decision emission: any Decision produced via asynchronous callback, deferred evaluation, or event-driven bypass of the sequential execution path.	NON-COMPLIANT
IES-10	Re-entry into a completed state: any execution that re-enters a state that has already been completed within the same request lifecycle.	NON-COMPLIANT

TABLE 5: FAILURE SEVERITY CLASSIFICATION

SEVERITY LEVEL	DESCRIPTION
CRITICAL	Violations that compromise determinism, audit integrity, or decision correctness. Immediate non-compliance.
HIGH	Violations that affect execution ordering, state transitions, or enforcement guarantees.
NON-COMPLIANT	Any violation of defined invariants or execution constraints.

SECTION 9

Non-Compliance Conditions

The following conditions constitute explicit non-compliance with this specification. The enumeration below is exhaustive within the scope defined in Section 1. Each condition is independent; satisfaction of all conditions except one does not constitute partial compliance.

> **NC-01: ABSENCE OF CONSENT VALIDATION**
>
> A system is non-compliant if it initiates PolicyEvaluation without having completed a ConsentLookup that resolves ConsentState to an explicit value.

> **NC-02: NON-DETERMINISTIC DECISION LOGIC**
>
> A system is non-compliant if its decision function produces differing outputs for identical inputs (Request, ConsentState, PolicySet) across any two invocations.

> **NC-03: SKIPPED AUDIT COMMIT**
>
> A system is non-compliant if any Decision, Allow or Deny, is produced without a subsequent, complete, and immutable AuditCommit within the same execution context.

NC-04: PARALLEL INCONSISTENT EXECUTION

A system is non-compliant if it executes two or more concurrent enforcement paths for the same request such that conflicting decisions could be produced or state consistency cannot be guaranteed.

NC-05: STATE MACHINE VIOLATION

A system is non-compliant if it implements any state transition not listed in Table 3, or if it fails to implement any state listed in Table 2.

NC-06: ALLOW DECISION WITH INVALID CONSENTSTATE

A system is non-compliant if it produces Decision = Allow when ConsentState is Absent, Expired, Revoked, or Invalid, under any condition or configuration.

NC-07: MUTABLE POLICYSET DURING EVALUATION

A system is non-compliant if the PolicySet is modified, updated, or replaced during the execution of PolicyEvaluation for any active request.

NC-08: IMPLICIT OR INFERRED CONSENTSTATE

A system is non-compliant if ConsentState is not resolved from an explicit, machine-readable consent record but is instead inferred from role membership, prior authorization, or assumed context.

NC-09: DEGRADED EVALUATION MODE

A system is non-compliant if it operates in any mode that reduces, approximates, or bypasses the full execution sequence defined in Section 4, regardless of the operational justification offered for such a mode.

NC-10: MISSING OR INCOMPLETE AUDITCOMMIT RECORD

A system is non-compliant if the AuditCommit record for any Decision does not include all fields enumerated in INV-A-03.

SECTION 10

Cross-Document Traceability

Each major rule and invariant defined in this document is traceable to one or more normative sources within the QODIQA corpus. The following table establishes the primary traceability mappings. Implementors and auditors SHALL use these mappings to verify consistency between this document and the broader corpus.

TABLE 6: CROSS-DOCUMENT TRACEABILITY MATRIX

THIS DOCUMENT	CORE STANDARD	REFERENCE ARCHITECTURE	68-POINT FRAMEWORK	CONFORMANCE TEST SUITE
State Machine (4)	3: Enforcement Model	4: Enforcement Layer	Points 1–12	TC-SM-001 – TC-SM-012
Decision Function (5)	4: Decision Logic	5: Decision Engine	Points 13–24	TC-DF-001 – TC-DF-012
Execution Flow Constraints (6)	5: Execution Requirements	6: Execution Path	Points 25–36	TC-EF-001 – TC-EF-012
Consent Invariants (7.1)	2: Consent Requirements	3: Consent Resolution	Points 1–6, 37–42	TC-CI-001 – TC-CI-006
Audit Invariants (7.2)	6: Audit Requirements	7: Audit Subsystem	Points 43–52	TC-AI-001 – TC-AI-005
Non-Compliance Conditions (9)	7: Compliance Criteria	8: Failure Semantics	Points 53–68	TC-NC-001 – TC-NC-010

Where a corpus document has been amended subsequent to the publication date of this specification, the traceability mappings in Table 6 SHALL be reviewed against the amended version. In the event of traceability gaps introduced by amendment, the QODIQA Governance Authority SHALL be notified in accordance with the Change Log and Version Control Protocol (QODIQA-CL-2026-001).

SECTION 11

Enforcement Implications

The following enforcement implications are derived directly from the constraints and invariants defined in this document. Each implication is stated as a binding requirement.

FAIL-CLOSED BEHAVIOR

The enforcement layer SHALL fail closed. A fail-closed system is defined as a system that produces Decision = Deny upon any failure to resolve required inputs, complete any mandatory state, or satisfy any defined invariant. Fail-open behavior, producing Decision = Allow upon failure, is prohibited.

IMMEDIATE TERMINATION ON INVALID STATE

Upon detection of any invalid execution state enumerated in Section 8, the enforcement layer SHALL immediately terminate evaluation and produce Decision = Deny. Continued evaluation after detection of an invalid state is prohibited.

NO DEGRADED EXECUTION

No degraded execution mode is permitted. A degraded execution mode is defined as any mode in which one or more mandatory states are abbreviated, skipped, or combined, or in which one or more defined invariants are suspended. Systems SHALL operate in full compliance or SHALL NOT operate.

NO SILENT FALLBACK

Silent fallback is defined as any mechanism that substitutes a default decision for a properly evaluated decision without producing an explicit error, a non-compliance record, and an audit entry. Silent fallback is prohibited in all configurations and deployment modes.

MANDATORY AUDITCOMMIT ON ALL PATHS

AuditCommit is mandatory on all execution paths, including paths that terminate at Decision = Deny due to input failure, state resolution failure, or invariant violation. No execution outcome is exempt from the audit requirement.

NO OPERATOR OVERRIDE OF INVARIANTS

No configuration, operator instruction, runtime parameter, or deployment-time override SHALL suspend, modify, or disable any invariant defined in Section 7. Invariants are not configurable. They are structural properties of the enforcement layer.

This section does not introduce new architectural components. Its purpose is to state the operational consequences of the invariants and constraints defined in preceding sections. These implications are derived, not additive.

SECTION 12

Conclusion

This document has specified the canonical state machine, deterministic decision function, execution flow constraints, enforcement invariants, invalid execution states, non-compliance conditions, cross-document traceability, and enforcement implications of the runtime consent enforcement layer.

This document does not modify, override, or reinterpret any existing QODIQA specification. It provides formalization, constraint definition, and deterministic execution semantics only.

All implementations claiming conformance with the QODIQA Core Standard SHALL satisfy the constraints and invariants defined herein. Conformance verification SHALL be performed in accordance with the QODIQA Conformance Test Suite (QODIQA-CTSS-2026-001).

DOCUMENT STATUS

Document Status and Identity

Document Status

Document Type	Formal Specification
Version	1.0
Status	Normative Extension
Scope	Deterministic Execution Semantics Layer
Publication Date	2026
Supersedes	None. Initial release.

This document is a normative extension to the QODIQA corpus. It does not supersede, replace, or modify any existing corpus document. The corpus documents to which this specification provides formal execution semantics are listed in Section 2.

This specification is authored and maintained by the QODIQA principal architect. Inquiries regarding this document SHALL be directed to the QODIQA Governance Authority.

Bogdan Duțescu

Principal Architect, QODIQA

contact@qodiqa.com

www.qodiqa.com

This document represents a formal deterministic specification intended for high-assurance environments, including enterprise deployment, regulatory alignment, and independent audit evaluation.

TITLE	QODIQA Formal Enforcement Model: Deterministic Execution Semantics
DOCUMENT IDENTIFIER	QODIQA-FEM-2026-001
VERSION	1.0
CLASSIFICATION	Extension Layer: Normative
AUTHORITY	QODIQA Corpus
RELATIONSHIP	Formalization of Runtime Enforcement Logic
PRIMARY REFERENCE	QODIQA Core Standard: QODIQA-CORE-2026-001
GOVERNANCE AUTHORITY	QODIQA Governance Charter: QODIQA-GC-2026-002
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 16 PART III · OPERATIONAL DOCUMENTS

Non-Compliance Conditions and Failure Modes

Formal classification of invalid system states, enforcement failure modes, and certification invalidation conditions within the runtime consent enforcement model.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-FCM-2026-001	1.0	2026	Failure Classification	Normative	NORMATIVE

ABSTRACT

This document defines the invalid states, failure modes, and non-compliance conditions of systems implementing the QODIQA deterministic runtime consent enforcement model. It classifies all conditions under which a system fails to enforce consent correctly, fails to produce verifiable enforcement decisions, or otherwise violates the invariants established by the QODIQA normative corpus. This document operates exclusively as a fail-closed enforcement model: any condition not positively confirmed as compliant is treated as non-compliant. It does not provide guidance, recommendations, or implementation advice. It defines states that are impermissible, conditions that invalidate conformance, and triggers that require certification revocation.

SECTION 1

Scope

This document applies to all systems claiming conformance with the QODIQA standard, including systems under evaluation, systems holding active certification, and systems in production deployment. The scope encompasses the following enforcement surfaces:

- The runtime enforcement layer and all enforcement decision pathways
- The policy evaluation engine and all policy resolution mechanisms
- All interactions with the consent registry, including read, write, and validation operations
- The enforcement gateway and all request interception and disposition mechanisms
- The audit and logging subsystem and all record production pathways
- The cryptographic subsystem and all signing, verification, and key management operations

This document governs failure and non-compliance states only. It does not define conformant behavior. Conformant behavior is defined in the QODIQA Core Standard and the 68-Point Enforcement Framework. This document defines the conditions under which conformance is absent, invalidated, or suspended.

This document is normative. All conditions defined herein SHALL be treated as binding constraints on systems claiming QODIQA conformance.

SECTION 2

Normative Foundations

All failure conditions defined in this document derive from violations of the invariants established by the following QODIQA corpus documents. These documents are incorporated by reference and are not redefined herein:

- **QODIQA Core Standard for Deterministic Runtime Consent Enforcement** - establishes the mandatory behavioral invariants of a conformant enforcement system.
- **QODIQA 68-Point Enforcement Framework** - enumerates the specific enforcement obligations against which non-compliance is measured.
- **QODIQA Certification Framework for Deterministic Runtime Consent Enforcement** - defines certification levels, assessment criteria, and revocation procedures.
- **QODIQA Conformance Test Suite** - specifies the observable test conditions that verify or refute conformance claims.
- **QODIQA Threat Model and Abuse Case Analysis** - defines the adversarial conditions and attack surfaces against which enforcement invariants must hold.
- **QODIQA Security and Cryptographic Profile** - establishes the cryptographic requirements whose violation constitutes a failure condition.
- **QODIQA Risk and Assumption Disclosure Annex** - identifies residual risk surfaces referenced in Section 12 of this document.

A failure condition is present whenever a system state, transition, or output violates an invariant established by any of the above documents. This document classifies such violations. It does not establish new invariants.

SECTION 3

Definition of Non-Compliance

Non-compliance is the state of a system in which one or more required invariants are not satisfied. Non-compliance is not a gradient. A system is either compliant or non-compliant with respect to each invariant individually. Partial compliance with respect to an invariant is classified as non-compliance.

Non-compliance occurs in the following conditions:

- A required invariant is not enforced by the system.
- Enforcement occurs but is subsequently bypassed, overridden, or rendered ineffective.
- The outcome of enforcement is indeterminate - the system cannot produce a definitive allow or deny decision.
- The enforcement decision is not externally verifiable - no auditable record is produced, or existing records are insufficient for verification.

Non-compliance is categorized as follows.

3.1 Hard Non-Compliance

Hard non-compliance is a deterministic violation of a required invariant. The violation is observable, reproducible, and verifiable without inference. Hard non-compliance includes: execution proceeding without a completed consent check; a consent check returning a deny decision that is not enforced; enforcement being bypassed by an alternate execution path; and logs that are absent, truncated, or non-append-only.

Hard non-compliance results in immediate loss of conformance status. No cure period applies.

3.2 Soft Non-Compliance

Soft non-compliance is a condition in which the system cannot demonstrate that a required invariant is satisfied. The violation may not be directly observable, but the absence of verifiable proof of compliance constitutes non-compliance. Soft non-compliance includes: enforcement decisions that are not signed or cryptographically attributable; audit logs that exist but cannot be verified for integrity; and consent states that are asserted but not backed by registry-confirmed records.

Soft non-compliance results in loss of conformance status. A system cannot claim conformance on the basis of undemonstratable compliance.

3.3 Conditional Non-Compliance

Conditional non-compliance is a violation that is contingent on system state, operational context, or input conditions. The system may satisfy an invariant under nominal conditions but fail under specific states. Conditional non-compliance includes: enforcement decisions that diverge when the consent registry is unavailable; policy evaluation that produces different results for structurally identical inputs across execution instances; and audit logging that fails under high throughput or degraded infrastructure conditions.

Conditional non-compliance results in loss of conformance status for all states in which the invariant is violated. A conformant system MUST satisfy all required invariants under all defined operating conditions, including degraded conditions.

SECTION 3A

Failure State Transition Model

This section defines the four discrete states in which a QODIQA-conformant system may exist with respect to compliance status, and the deterministic conditions that govern transitions between those states. State transitions are not discretionary. Each transition is triggered by a defined condition and results in a required system response.

Compliant State

A system is in the Compliant State when all of the following conditions are simultaneously satisfied: all required invariants are enforced; all enforcement decisions are deterministic, signed, and auditable; the consent registry is reachable and synchronized within the permitted staleness threshold; the audit log is append-only, integrity-protected, and externally verifiable; and no failure mode defined in Section 4 is active. Certification is valid and operative in this state.

Degraded State

A system enters the Degraded State when one or more operational conditions are impaired but no enforcement invariant has yet been violated. Conditions that may produce a Degraded State include: elevated registry synchronization latency approaching the staleness threshold; key management system response times approaching the defined timeout; or audit log write latency approaching the maximum permitted delay. A system in the Degraded State MUST alert governance and MUST apply heightened monitoring. If any impaired condition crosses its defined threshold, the system MUST immediately transition to the Non-Compliant State. The system MUST NOT remain in the Degraded State indefinitely. The Degraded State does not permit execution on a permissive basis; fail-closed requirements remain fully operative.

Non-Compliant State

A system enters the Non-Compliant State when one or more invariants are violated, as defined in Section 3 and Section 4. The Non-Compliant State is triggered by any active failure condition classified under the taxonomy in Section 4, or by any deterministic trigger condition defined in Section 5. Upon entering the Non-Compliant State, the system MUST deny all execution requests, MUST produce an observable signal identifying the failure condition, and MUST NOT resume enforcement without completing the re-validation process defined in Section 11. Certification is suspended upon entry to the Non-Compliant State. The severity of the non-compliance condition, as classified in Section 7, determines whether the transition is from suspension to revocation.

Certification Revoked State

A system enters the Certification Revoked State when a Level 1 non-compliance condition is confirmed, or when a Level 2 condition is not remediated within the period specified by the Certification Framework, resulting in escalation. In this state, the system has no valid certification. The system MUST NOT represent itself as QODIQA-conformant or reference its prior certification. No auto-recovery, operator action, or remediation restores certification status in this state. A new certification process, initiated under the QODIQA Certification Framework, is the sole path to re-entering the Compliant State from the Certification Revoked State.

Deterministic Transition Rules

The following transitions are defined and are non-discretionary:

- **Compliant to Degraded:** One or more operational conditions approach but have not crossed a defined invariant threshold. Triggered by monitoring signals. Does not require external notification.

- **Compliant or Degraded to Non-Compliant:** An invariant threshold is crossed or a failure condition defined in Section 4 becomes active. Triggered deterministically by the conditions in Table 5-A. Requires immediate fail-closed enforcement and certification suspension notification.
- **Non-Compliant to Compliant:** All failure conditions are resolved and the system has completed the re-validation process under Section 11. Requires positive confirmation by the certification body. Not achievable through self-attestation.
- **Non-Compliant to Certification Revoked:** A Level 1 condition is confirmed, or a Level 2 condition is not remediated within the required period. Triggered by the certification body under the Certification Framework. Requires initiation of a new certification process to exit.
- **Certification Revoked to Compliant:** Only achievable through successful completion of a full re-certification process under the QODIQA Certification Framework. All prior enforcement records from the revocation period are considered invalid.

SECTION 4

Failure Mode Taxonomy

The following taxonomy classifies all failure modes recognized within the QODIQA enforcement model. Each failure mode class identifies distinct categories of invariant violation. A system may simultaneously exhibit failures across multiple classes.

4.1 Enforcement Failure

Enforcement failure occurs when the enforcement layer does not produce or apply a correct enforcement decision for an incoming execution request.

EF-1 - CONSENT CHECK ABSENT

CONDITION	An execution request proceeds to the target system without a consent check having been completed by the enforcement gateway.
DETECTION	Audit log contains an execution record with no corresponding consent evaluation record. Enforcement gateway emits no decision artifact for the request.
RESULTING STATE	Unauthorized execution. Hard non-compliance. Immediate loss of conformance status.
REFERENCED INVARIANT	INV-ENF-01 (Core Standard Section 4.1 - Mandatory Consent Evaluation)
CONFORMANCE TEST	CTS-EF-01

EF-2 - CONSENT CHECKED BUT NOT ENFORCED

CONDITION	A consent check is completed and produces a deny decision, but the request is permitted to proceed.
DETECTION	Audit log contains a deny decision record followed by an execution record for the same request identifier within the same session context.
RESULTING STATE	Enforcement decision ignored. Hard non-compliance. Immediate loss of conformance status.
REFERENCED INVARIANT	INV-ENF-02 (Core Standard Section 4.2 - Enforcement Decision Binding)
CONFORMANCE TEST	CTS-EF-02

EF-3 - ENFORCEMENT DECISION IGNORED

CONDITION	The enforcement gateway produces a decision artifact, but downstream components do not act on that decision.
DETECTION	Enforcement gateway log shows decision issued; execution log shows request completed without reference to enforcement decision artifact.
RESULTING STATE	Enforcement bypassed at downstream boundary. Hard non-compliance.
REFERENCED INVARIANT	INV-ENF-03 (68-Point Framework Section 1.3 - Downstream Enforcement Obligation)
CONFORMANCE TEST	CTS-EF-03

4.2 Decision Integrity Failure

Decision integrity failure occurs when the policy evaluation engine produces incorrect, inconsistent, or non-deterministic outputs for a given input set.

DI-1 - POLICY EVALUATION INCONSISTENCY

CONDITION	Structurally identical inputs to the policy engine produce different decisions across two or more evaluation instances under identical policy state.
DETECTION	Replay of a recorded request against the same policy version yields a different decision than the original evaluation log records.
RESULTING STATE	Non-deterministic enforcement. Hard non-compliance.
REFERENCED INVARIANT	INV-DEC-01 (Core Standard Section 5.1 - Deterministic Policy Evaluation)
CONFORMANCE TEST	CTS-DI-01

DI-2 - NON-DETERMINISTIC OUTPUT

CONDITION	The policy engine returns an output that is neither a definitive allow nor a definitive deny for a valid, well-formed input.
DETECTION	Decision artifact is absent, malformed, null, or contains an unrecognized decision value. Conformance test suite DI-series tests fail.
RESULTING STATE	Indeterminate enforcement state. System MUST apply deny. Soft non-compliance if deny is applied; hard non-compliance if execution proceeds.
REFERENCED INVARIANT	INV-DEC-02 (Core Standard Section 5.2 - Binary Decision Requirement)
CONFORMANCE TEST	CTS-DI-02

DI-3 - STATE DESYNCHRONIZATION

CONDITION	The policy engine evaluates against a consent state that does not match the current consent registry state due to cache staleness, replication lag, or partial update failure.
DETECTION	Consent record timestamp in enforcement decision artifact predates the most recent registry update for the subject consent identifier.
RESULTING STATE	Enforcement based on stale state. Conditional non-compliance. Decisions issued against desynchronized state are invalid.
REFERENCED INVARIANT	INV-DEC-03 (Core Standard Section 5.3 - Registry State Currency)
CONFORMANCE TEST	CTS-DI-03

4.3 Data Integrity Failure

Data integrity failure occurs when the consent state, registry data, or enforcement inputs have been corrupted, altered, or cannot be verified as authentic.

DAI-1 - CORRUPTED CONSENT STATE

CONDITION	The consent record for a subject cannot be read, is structurally invalid, or fails integrity verification at the time of policy evaluation.
DETECTION	Consent record fails schema validation or cryptographic integrity check. Policy engine cannot complete evaluation.
RESULTING STATE	Consent resolution failure. System MUST apply deny. Hard non-compliance if execution proceeds.
REFERENCED INVARIANT	INV-DAT-01 (Core Standard Section 6.1 - Consent Record Integrity)
CONFORMANCE TEST	CTS-DAI-01

DAI-2 - STALE REGISTRY DATA

CONDITION	The enforcement layer operates against registry data that has exceeded the maximum permitted staleness threshold defined in the QODIQA Core Standard.
DETECTION	Registry synchronization timestamp in enforcement context exceeds permitted staleness window. Registry replication health check fails.
RESULTING STATE	Conditional non-compliance. All decisions issued during the staleness window are invalid and must be treated as unenforced.
REFERENCED INVARIANT	INV-DAT-02 (Core Standard Section 6.2 - Maximum Staleness Threshold)
CONFORMANCE TEST	CTS-DAI-02

DAI-3 - TAMPERED INPUTS

CONDITION	Enforcement inputs - including request context, subject identity claims, or consent token - have been modified after initial capture and before policy evaluation.
DETECTION	Input bundle signature verification fails. Hash of enforcement input context does not match the hash recorded at capture time.
RESULTING STATE	Enforcement decision issued on invalid inputs. Hard non-compliance. Decision artifact is void.
REFERENCED INVARIANT	INV-DAT-03 (68-Point Framework Section 2.4 - Input Bundle Integrity)
CONFORMANCE TEST	CTS-DAI-03

4.4 Control Plane Failure

Control plane failure occurs when a critical system component required for enforcement is unavailable, degraded, or not responding within the permitted time constraints.

CP-1 - REGISTRY UNAVAILABLE

CONDITION	The consent registry is unreachable or returns no valid response within the maximum resolution timeout defined in the Core Standard.
DETECTION	Registry health check returns failure. Enforcement gateway cannot complete consent resolution. Execution queued or attempted without resolved consent state.
RESULTING STATE	Consent unresolvable. System MUST apply deny to all requests during the unavailability window. Any execution permitted during this window constitutes hard non-compliance.
REFERENCED INVARIANT	INV-CTL-01 (Core Standard Section 7.1 - Registry Availability Requirement)
CONFORMANCE TEST	CTS-CP-01

CP-2 - POLICY ENGINE UNAVAILABLE

CONDITION	The policy evaluation engine is unreachable, non-responsive, or produces no output for submitted evaluation requests.
DETECTION	Policy engine health endpoint returns failure or timeout. No decision artifact is produced for incoming enforcement requests.
RESULTING STATE	Enforcement decision impossible. System MUST apply deny. Any execution permitted constitutes hard non-compliance.
REFERENCED INVARIANT	INV-CTL-02 (Core Standard Section 7.2 - Policy Engine Availability Requirement)
CONFORMANCE TEST	CTS-CP-02

CP-3 - KEY MANAGEMENT FAILURE

CONDITION	The key management system is unavailable, returns invalid material, or cannot issue keys required for signing or verification of enforcement artifacts.
DETECTION	Signing operations fail. Verification operations return invalid results. Key material retrieval timeout or error.
RESULTING STATE	Enforcement artifacts cannot be cryptographically attributed. All decisions issued without valid signatures are void. Soft non-compliance transitions to hard non-compliance if execution proceeds.
REFERENCED INVARIANT	INV-CTL-03 (Security and Cryptographic Profile Section 3.1 - Key Material Availability)
CONFORMANCE TEST	CTS-CP-03

4.5 Audit Failure

Audit failure occurs when the audit and logging subsystem fails to produce complete, verifiable, and tamper-evident records of all enforcement operations.

AF-1 - MISSING LOGS

CONDITION	An enforcement operation completes but no corresponding audit record is produced, or the produced record is absent from the audit store.
DETECTION	Reconciliation of enforcement gateway event count against audit store record count reveals a deficit. Request identifiers present in execution logs are absent from audit logs.
RESULTING STATE	Unverifiable enforcement. Soft non-compliance for each unlogged operation. Operations without audit records cannot be treated as conformant.
REFERENCED INVARIANT	INV-AUD-01 (Core Standard Section 8.1 - Complete Audit Coverage)
CONFORMANCE TEST	CTS-AF-01

AF-2 - NON-APPEND-ONLY LOGS

CONDITION	The audit store permits modification, deletion, or overwrite of existing records.
DETECTION	Audit store configuration audit reveals write permissions other than append. Hash-chain verification of log sequence fails at one or more positions.
RESULTING STATE	Audit integrity structurally compromised. Hard non-compliance. The entire audit record for the affected period is considered invalid.
REFERENCED INVARIANT	INV-AUD-02 (Core Standard Section 8.2 - Append-Only Log Requirement)
CONFORMANCE TEST	CTS-AF-02

AF-3 - TAMPERED LOGS

CONDITION	One or more audit records have been modified after initial write.
DETECTION	Hash-chain verification failure. Record signature verification failure. Audit record timestamp inconsistency.
RESULTING STATE	Audit integrity breach. Hard non-compliance. All records in the affected sequence are invalid.
REFERENCED INVARIANT	INV-AUD-03 (Core Standard Section 8.3 - Audit Record Immutability)
CONFORMANCE TEST	CTS-AF-03

AF-4 - NON-VERIFIABLE LOGS

CONDITION	Audit records exist but cannot be externally verified due to absent or invalid signatures, unavailable verification keys, or missing hash-chain anchors.
DETECTION	External audit verification process returns failure or cannot complete. Verification keys are not available to the verifying party.
RESULTING STATE	Soft non-compliance. Records that cannot be externally verified do not satisfy the auditability invariant.
REFERENCED INVARIANT	INV-AUD-04 (68-Point Framework Section 3.4 - External Verifiability)
CONFORMANCE TEST	CTS-AF-04

4.6 Cryptographic Failure

Cryptographic failure occurs when the cryptographic mechanisms required for enforcement artifact integrity, audit log integrity, or consent state authentication do not operate correctly.

CF-1 - INVALID SIGNATURES

CONDITION	An enforcement decision artifact, consent token, or audit record carries a digital signature that fails verification.
DETECTION	Signature verification returns invalid. Verification key does not correspond to the signing key referenced in the artifact header.
RESULTING STATE	Artifact authenticity cannot be established. The artifact is void. If the artifact is a decision, that decision is invalid. Hard non-compliance if voided decision was acted upon.
REFERENCED INVARIANT	INV-CRY-01 (Security and Cryptographic Profile Section 4.1 - Artifact Signature Validity)
CONFORMANCE TEST	CTS-CF-01

CF-2 - BROKEN TRUST CHAIN

CONDITION	The chain of cryptographic trust from the enforcement artifact to the trust anchor defined in the Security and Cryptographic Profile is broken at one or more links.
DETECTION	Trust chain traversal fails. An intermediate certificate or key attestation in the chain is absent, expired, or revoked.
RESULTING STATE	Unattributable enforcement. Hard non-compliance. All artifacts dependent on the broken chain are void.
REFERENCED INVARIANT	INV-CRY-02 (Security and Cryptographic Profile Section 4.2 - Trust Chain Continuity)
CONFORMANCE TEST	CTS-CF-02

CF-3 - KEY MISUSE

CONDITION	A cryptographic key is used outside its defined purpose, scope, or validity period as specified in the Security and Cryptographic Profile.
DETECTION	Key usage field in certificate or key attestation does not permit the operation performed. Key is expired or its defined scope does not include the signing context.
RESULTING STATE	Cryptographic invariant violated. All artifacts produced using the misused key are void. Hard non-compliance.
REFERENCED INVARIANT	INV-CRY-03 (Security and Cryptographic Profile Section 4.3 - Key Usage Constraints)
CONFORMANCE TEST	CTS-CF-03

4.7 Boundary Failure

Boundary failure occurs when enforcement is bypassed by an execution path that does not pass through the enforcement gateway, or when execution occurs in a context that is not subject to enforcement evaluation.

BF-1 - ENFORCEMENT BYPASS VIA ALTERNATE PATH

CONDITION	An execution request reaches the target system via a path that does not include the enforcement gateway.
DETECTION	Execution records exist for requests with no corresponding enforcement gateway entry. Network topology analysis reveals direct access routes to protected systems that bypass the enforcement plane.
RESULTING STATE	Complete enforcement bypass. Hard non-compliance. All executions occurring via the bypassed path are unauthorized.
REFERENCED INVARIANT	INV-BND-01 (Core Standard Section 9.1 - Enforcement Boundary Completeness)
CONFORMANCE TEST	CTS-BF-01

BF-2 - SHADOW EXECUTION PATH

CONDITION	An execution context exists within the system boundary that does not emit observable signals to the enforcement or audit subsystem and is not subject to consent evaluation.
DETECTION	System boundary analysis identifies execution contexts without enforcement plane registration. Audit log coverage analysis reveals gaps corresponding to specific execution contexts.
RESULTING STATE	Unmonitored execution surface. Hard non-compliance. The existence of a shadow path constitutes a violation independent of whether unauthorized execution has occurred.
REFERENCED INVARIANT	INV-BND-02 (68-Point Framework Section 4.2 - Execution Context Registration)
CONFORMANCE TEST	CTS-BF-02

BF-3 - SIDE-CHANNEL EXECUTION

CONDITION	An action with material effect on the consent-protected resource is achieved through a mechanism other than the protected execution interface, and that mechanism is not subject to enforcement evaluation.
DETECTION	Threat model coverage analysis identifies effect-equivalent access paths not covered by enforcement evaluation. Conformance test suite BF-series tests fail.
RESULTING STATE	Enforcement model incomplete. Hard non-compliance. The protected resource is not fully under enforcement control.
REFERENCED INVARIANT	INV-BND-03 (Threat Model Section 5.3 - Side-Channel Attack Surface)
CONFORMANCE TEST	CTS-BF-03

SECTION 5

Deterministic Failure Conditions

The following conditions are defined as deterministic triggers for fail-closed behavior. Each condition is expressed as a verifiable system state. Upon detection of any of these conditions, the system MUST apply the specified resulting state immediately and without exception.

TABLE 5-A - DETERMINISTIC FAILURE TRIGGER CONDITIONS

ID	TRIGGER CONDITION	DETECTION MECHANISM	REQUIRED SYSTEM RESPONSE	REFERENCED INVARIANT	TEST MAPPING
FC-01	Consent record for subject cannot be resolved within maximum timeout	Registry resolution timeout event	MUST deny execution. MUST log unresolvable consent condition.	INV-DAT-02, INV-CTL-01	CTS-CP-01, CTS-DAI-02
FC-02	Policy evaluation returns indeterminate or null output	Decision artifact absent or contains invalid decision value	MUST deny execution. MUST NOT treat null as allow.	INV-DEC-02	CTS-DI-02
FC-03	Enforcement decision artifact signature fails verification	Signature verification returns invalid	MUST void the artifact. MUST deny execution. MUST log signature failure.	INV-CRY-01	CTS-CF-01
FC-04	Consent registry is unreachable at time of evaluation	Registry health check failure; no response within timeout	MUST deny all execution requests until registry is reachable and verified.	INV-CTL-01	CTS-CP-01
FC-05	Policy engine is unavailable at time of evaluation request	Policy engine health failure; no decision artifact produced	MUST deny all execution requests until policy engine is available and verified.	INV-CTL-02	CTS-CP-02
FC-06	Audit log write fails for an enforcement operation	Audit subsystem returns write error or no confirmation	MUST deny the associated execution. MUST NOT proceed without a confirmed audit record.	INV-AUD-01	CTS-AF-01

ID	TRIGGER CONDITION	DETECTION MECHANISM	REQUIRED SYSTEM RESPONSE	REFERENCED INVARIANT	TEST MAPPING
FC-07	Consent record fails integrity check at evaluation time	Record hash or signature verification failure	MUST deny execution. MUST treat record as invalid. MUST log integrity failure.	INV-DAT-01	CTS-DAI-01
FC-08	Enforcement input bundle has been modified after capture	Input bundle hash mismatch at evaluation time	MUST void the evaluation. MUST deny execution. MUST log tampering condition.	INV-DAT-03	CTS-DAI-03
FC-09	Audit log sequence hash-chain verification fails	Hash-chain traversal returns inconsistency at one or more positions	MUST flag entire affected sequence as compromised. MUST NOT treat affected records as valid evidence of compliance.	INV-AUD-03	CTS-AF-03
FC-10	Execution request arrives via path not registered with enforcement gateway	Source path not present in enforcement boundary registry	MUST deny execution. MUST log boundary violation. MUST alert governance.	INV-BND-01	CTS-BF-01

No failure condition defined in this table permits a fallback to permissive behavior. The absence of a positive, verified enforcement decision is equivalent to a deny decision. Systems MUST be designed such that the default state under uncertainty is execution denial.

SECTION 6

Fail-Closed Requirements

The QODIQA model is a fail-closed enforcement model. The requirements in this section are absolute. No operational condition, system constraint, availability target, or performance requirement overrides these requirements.

REQUIREMENT FC-R1 - NO FALLBACK TO PERMISSIVE STATE

A system MUST NOT transition from an enforcement state to a permissive state as a result of any failure condition. A system experiencing a failure condition MUST remain in a deny state until the failure condition is resolved and the system has returned to a verified compliant state.

REQUIREMENT FC-R2 - NO DEGRADED PERMISSIVE OPERATION

A system operating in a degraded mode MUST NOT permit execution of consent-protected requests during that degraded mode. There is no defined degraded operational mode that allows execution. A system that cannot fully enforce consent MUST deny all execution.

REQUIREMENT FC-R3 - NO SILENT BYPASS

A system MUST NOT permit execution to proceed without a completed enforcement decision, regardless of whether that bypass is caused by a configuration error, infrastructure failure, component unavailability, or operator action. Silent bypass is a hard non-compliance condition independent of its cause.

REQUIREMENT FC-R4 - DEFAULT DENY UNDER UNCERTAINTY

A system MUST default to denying execution under any condition of uncertainty. Uncertainty includes: inability to resolve consent, inability to evaluate policy, inability to verify enforcement artifact integrity, and inability to write an audit record. No uncertainty condition permits execution.

These requirements apply without exception to all system states, including startup, shutdown, maintenance, degraded operation, failover, and recovery. A system that satisfies fail-closed requirements under nominal operation but not under exceptional conditions is non-conformant.

SECTION 7

Non-Compliance Severity Levels

Non-compliance conditions are classified into three severity levels. Severity level determines the immediate required outcome and the certification impact. Severity levels are assigned per the conditions specified below and are not subject to operator interpretation or reclassification.

7.1 Level 1 - Critical Non-Compliance

CRITICAL A Level 1 condition is a direct violation of a mandatory enforcement invariant that results in unauthorized execution, undetected bypass, or irrecoverable audit failure. Level 1 conditions require no remediation period. They result in immediate loss of conformance status and immediate initiation of certification revocation proceedings.

Conditions classified as Level 1:

- Execution proceeding without a completed consent check (EF-1)
- A deny decision not enforced and execution permitted (EF-2)
- Execution bypassing the enforcement gateway via an alternate path (BF-1)
- Execution via a shadow path not registered with the enforcement plane (BF-2)
- Audit log records found to be modified after initial write (AF-3)
- Audit store configuration permitting non-append-only writes (AF-2)
- Enforcement proceeding on a voided or signature-invalid decision artifact, where execution was permitted (CF-1, CF-2)

Required outcome: Immediate suspension of conformance status. Initiation of certification revocation under the QODIQA Certification Framework. No continued operation as a QODIQA-conformant system. Re-certification required before any

compliance claim may be renewed. Each condition listed above SHALL be traceable to its originating invariant and validated through the corresponding Conformance Test Suite mapping.

7.2 Level 2 - Major Non-Compliance

MAJOR A Level 2 condition is a violation of a required invariant that does not result in proven unauthorized execution but renders the system's enforcement decisions unverifiable, unreliable, or structurally unsound. Level 2 conditions result in loss of conformance status and require remediation and re-assessment before conformance status may be restored.

Conditions classified as Level 2:

- Policy evaluation returning inconsistent results for structurally identical inputs (DI-1)
- State desynchronization between enforcement layer and consent registry (DI-3)
- Consent registry unavailable, without confirmed fail-closed response (CP-1)
- Policy engine unavailable, without confirmed fail-closed response (CP-2)
- Key management failure preventing signing or verification of artifacts (CP-3)
- Audit records missing for one or more enforcement operations (AF-1)
- Non-verifiable audit records due to absent or invalid signatures (AF-4)
- Trust chain broken at one or more links (CF-2)
- Side-channel execution path identified and not covered by enforcement evaluation (BF-3)

Required outcome: Immediate suspension of conformance status. System is considered non-conformant until the condition is remediated and independently re-assessed. No conformance claim may be maintained while a Level 2 condition is present. Each condition listed above SHALL be traceable to its originating invariant and validated through the corresponding Conformance Test Suite mapping.

7.3 Level 3 - Minor Non-Compliance

MINOR A Level 3 condition is a violation of a structural or procedural requirement that does not directly break enforcement or produce unverifiable decisions, but represents a deviation from normative specification requirements. Level 3 conditions do not by themselves result in loss of conformance status, but MUST be remediated within the timeframe specified in the Certification Framework. A Level 3 condition that is not remediated within the specified period is reclassified as Level 2.

Conditions classified as Level 3:

- Enforcement decision artifacts that satisfy functional requirements but do not conform to the normative artifact schema
- Audit records that satisfy content requirements but do not conform to the normative record format
- Registry interaction that satisfies functional requirements but uses a non-normative protocol binding
- Documentation or configuration artifacts that do not satisfy corpus reference requirements

Required outcome: Documented finding against the conformance record. Remediation plan required within the period specified by the Certification Framework. No immediate suspension of conformance status, subject to escalation conditions above. Each condition listed above SHALL be traceable to its originating invariant and validated through the corresponding Conformance Test Suite mapping.

SECTION 8

Certification Impact

Failure conditions map to certification outcomes as defined in the QODIQA Certification Framework. The following table specifies the required certification action for each severity level and condition class.

TABLE 8-A - FAILURE SEVERITY TO CERTIFICATION IMPACT MAPPING

SEVERITY LEVEL	CONDITION CLASS	CERTIFICATION ACTION	RE-CERTIFICATION REQUIRED
Level 1 - Critical	EF-1, EF-2, BF-1, BF-2, AF-2, AF-3, CF-1 (with execution), CF-2 (with execution)	Immediate revocation	Yes - full re-certification at applicable level
Level 2 - Major	DI-1, DI-3, CP-1, CP-2, CP-3, AF-1, AF-4, CF-2 (no execution), BF-3	Immediate suspension	Yes - re-assessment against all affected framework points
Level 3 - Minor	Structural and procedural deviations	Documented finding; remediation required within specified period	No - unless reclassified to Level 2 due to non-remediation
Level 2 (escalated from Level 3)	Any Level 3 condition not remediated within specified period	Immediate suspension	Yes - re-assessment against affected framework points

Certification suspension is distinct from certification revocation. Suspension is a temporary state from which reinstatement is possible upon successful re-assessment. Revocation is a permanent invalidation of the certification record. A revoked certification cannot be reinstated; a new certification process must be initiated.

The certification body operating under the QODIQA Certification Framework is the sole authority for determining certification status. Operator self-assessment does not constitute certification status restoration. A system remains non-conformant until the certification body issues a positive conformance determination.

Any misrepresentation of certification status following revocation or suspension, including continued use of conformance claims or QODIQA compliance attestations, constitutes a violation of the Governance Charter and the terms of certification.

SECTION 9

Detection and Verifiability Constraints

A failure condition that cannot be detected is not exempt from non-compliance classification. The inability to detect a failure condition is itself a non-compliance condition. The following requirements govern detection and verifiability of all failure states.

REQUIREMENT DV-R1 - EXTERNAL TESTABILITY

All failure conditions defined in this document MUST be externally testable. The Conformance Test Suite defines the test procedures for each condition. A system that cannot be subjected to these test procedures is non-conformant with respect to verifiability requirements, independent of whether any failure condition is present.

REQUIREMENT DV-R2 - OBSERVABLE SIGNALS

All failure conditions MUST produce observable signals. These signals include: enforcement decision artifacts, audit log records, health status outputs, and error responses. A failure condition that occurs without producing any observable signal constitutes a silent failure and is classified as hard non-compliance (see Section 10).

REQUIREMENT DV-R3 - NO HIDDEN FAILURE STATES

A system MUST NOT contain operational states in which a failure condition exists but produces no externally observable indication. All system states MUST be enumerable and testable. The existence of internal states not observable through defined interfaces is a boundary failure condition (BF-2).

REQUIREMENT DV-R4 - FAILURE SIGNAL INTEGRITY

Failure signals themselves MUST be integrity-protected. A system that can suppress, forge, or delay failure signals is considered to be in a state of hard non-compliance with respect to the failure condition that the signal would have indicated.

SECTION 10

Prohibited System Behaviors

The following behaviors are prohibited in all systems claiming QODIQA conformance. These prohibitions are absolute. No operational justification, performance requirement, or system constraint constitutes a basis for exception.

PROHIBITED BEHAVIOR PB-1 - SILENT FAILURE

A system MUST NOT fail to enforce consent without producing an observable signal indicating that enforcement did not occur. Silent failure includes: timeout expiry with no recorded outcome; component crash with no logged state transition; and audit write failure with no system-level alarm or fallback.

PROHIBITED BEHAVIOR PB-2 - IMPLICIT ALLOW

A system MUST NOT default to allowing execution in the absence of an explicit, verified allow decision. The absence of a deny decision does not constitute an allow decision. An allow decision MUST be explicitly issued, signed, and recorded before execution is permitted.

PROHIBITED BEHAVIOR PB-3 - PARTIAL ENFORCEMENT

A system MUST NOT enforce consent for a subset of requests while permitting uninspected requests to proceed. Partial enforcement is not a degraded mode of operation. It is a non-conformant configuration in which the enforcement boundary is incomplete.

PROHIBITED BEHAVIOR PB-4 - LOGGING WITHOUT INTEGRITY

A system MUST NOT produce audit log records that are not integrity-protected. Unprotected log records satisfy the volume requirement of logging but not the integrity requirement. Production of unprotected records does not constitute audit compliance.

PROHIBITED BEHAVIOR PB-5 - RUNTIME OVERRIDES

A system MUST NOT implement or expose any mechanism by which an enforcement decision can be overridden, bypassed, or suppressed at runtime by an operator, administrator, or automated process. The enforcement decision issued by the policy engine is final for its evaluated context. Runtime override capability constitutes a structural non-compliance condition independent of whether it is exercised.

PROHIBITED BEHAVIOR PB-6 - COMPLIANCE CLAIMS UNDER SUSPENSION OR REVOCATION

A system MUST NOT assert QODIQA conformance, reference QODIQA certification, or represent itself as meeting QODIQA requirements while its certification is suspended or revoked.

SECTION 11

Recovery Constraints

Recovery from a failure condition does not restore conformance status. Recovery and compliance are distinct states. A system that has experienced a failure condition and subsequently returned to operational function has not demonstrated conformance unless it has re-entered a verified compliant state through the processes defined in this section.

REQUIREMENT RC-R1 - RECOVERY IS NOT COMPLIANCE

A system that resolves a failure condition through automated restart, failover, or operator intervention has not restored conformance. Recovery denotes a return to operational function. Compliance denotes a verified state in which all required invariants are satisfied. These are distinct and recovery does not imply compliance.

REQUIREMENT RC-R2 - RE-ENTRY INTO VERIFIED COMPLIANT STATE

A system MUST re-enter a verified compliant state following any Level 1 or Level 2 failure condition. Re-entry requires: a complete integrity check of all enforcement subsystems; verification that all failure conditions have been resolved; confirmation that no data from the failure period is being treated as valid enforcement evidence; and, for Level 1 conditions, initiation of the re-certification process with the certification body.

REQUIREMENT RC-R3 - NO AUTO-RECOVERY WITHOUT RE-VALIDATION

A system MUST NOT automatically resume enforcement operations following a Level 1 or Level 2 failure condition without an explicit re-validation step. Auto-recovery mechanisms MUST halt at a deny-all state and require explicit operator confirmation of system integrity before resuming enforcement. An auto-recovery sequence that resumes enforcement without re-validation constitutes an implicit allow condition and is classified as hard non-compliance under PB-2.

REQUIREMENT RC-R4 - FAILURE PERIOD RECORDS

All enforcement operations attempted during a failure period MUST be treated as unenforced, regardless of whether they completed successfully from an application perspective. Audit records produced during a failure period that cannot be independently verified are not valid evidence of enforcement.

SECTION 12

Residual Failure Surfaces

The QODIQA Risk and Assumption Disclosure Annex identifies residual risk surfaces that are acknowledged as present within the QODIQA model and that cannot be fully eliminated through system design. This section defines how such residual surfaces interact with the non-compliance and failure classification framework.

The existence of a residual failure surface does not constitute permission for that surface to produce non-fail-closed behavior. The following requirements apply to all residual surfaces identified in the Disclosure Annex.

REQUIREMENT RS-R1 - FAIL-CLOSED BEHAVIOR REQUIRED DESPITE RESIDUAL SURFACE

A system MUST apply fail-closed behavior on all residual failure surfaces. The acknowledgment of a residual surface in the Disclosure Annex documents its existence; it does not authorize execution in the presence of that surface. If a residual surface manifests as an active failure condition, the system MUST deny execution.

REQUIREMENT RS-R2 - RESIDUAL SURFACES MUST BE OBSERVABLE

All residual failure surfaces MUST produce observable signals when active. A residual surface that activates without producing an observable signal constitutes a silent failure under PB-1 and is classified as hard non-compliance.

REQUIREMENT RS-R3 - RESIDUAL SURFACE MANIFESTATION TRIGGERS NON-COMPLIANCE CLASSIFICATION

When a residual failure surface manifests as an active failure condition, the condition is classified under the taxonomy in Section 4 and receives a severity level under Section 7 as if it were any other failure condition. The residual nature of the surface does not modify the severity classification or the required certification impact.

The distinction between a residual surface and an unmitigated vulnerability is determined by the Disclosure Annex. A failure condition not identified in the Disclosure Annex as a residual surface is classified as an unmitigated vulnerability and is treated as a Level 1 non-compliance condition unless assessment indicates otherwise.

SECTION 13

Conformance Requirements

A system is conformant with this document if and only if all of the following conditions are simultaneously satisfied. Partial satisfaction of the conditions below does not constitute conformance.

CONFORMANCE CONDITION CC-1 - ABSENCE OF DEFINED FAILURE MODES

None of the failure modes defined in Section 4 are present in the system. A conformant system has no active enforcement failures, no active decision integrity failures, no active data integrity failures, no active control plane failures, no active audit failures, no active cryptographic failures, and no active boundary failures.

CONFORMANCE CONDITION CC-2 - FAIL-CLOSED UNDER ALL FAILURE TRIGGERS

All deterministic failure conditions listed in Section 5 result in fail-closed behavior. The system MUST deny execution in response to each trigger condition. Verification of this condition is performed through the Conformance Test Suite failure-condition test series.

CONFORMANCE CONDITION CC-3 - OBSERVABLE AND AUDITABLE FAILURES

All failure conditions that occur produce externally observable signals and are recorded in the audit log in a manner that satisfies the integrity requirements of Section 4.5. No failure condition occurs silently. No failure produces an audit record that cannot be independently verified.

CONFORMANCE CONDITION CC-4 - ABSENCE OF PROHIBITED BEHAVIORS

None of the behaviors enumerated in Section 10 are implemented or achievable in the system. The system contains no silent failure paths, no implicit allow mechanisms, no partial enforcement configurations, no logging without integrity, no runtime override mechanisms, and no capability to misrepresent certification status.

CONFORMANCE CONDITION CC-5 - RECOVERY COMPLIANCE

The system satisfies all recovery constraints in Section 11. No automated recovery mechanism exists that could restore permissive execution without an explicit re-validation step.

Conformance with this document is a necessary component of QODIQA certification at any level defined in the Certification Framework. Conformance with this document alone does not constitute full corpus conformance. Systems asserting QODIQA certification MUST satisfy this document in conjunction with all other normative corpus documents.

FINAL CONFORMANCE STATEMENT

A system is QODIQA-conformant with respect to this document if and only if conditions CC-1 through CC-5 are all satisfied simultaneously, as verified by the QODIQA Conformance Test Suite under the assessment procedures of the QODIQA Certification Framework. The burden of demonstrating conformance rests with the system operator. The absence of demonstrated non-compliance does not constitute demonstrated compliance.

DOCUMENT STATUS

Document Status and Corpus Alignment

This document is a normative component of the QODIQA specification corpus. It defines the invalid states, failure modes, non-compliance conditions, and certification invalidation triggers applicable to all systems implementing the QODIQA deterministic runtime consent enforcement model.

This document does not define conformant behavior. It defines exclusively the conditions under which conformance is absent, invalidated, or suspended. It must be read in conjunction with the normative corpus documents from which its failure conditions are derived.

This document should be read together with the following related specifications:

- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement
- QODIQA: Threat Model and Abuse Case Specification
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement
- QODIQA: Residual Risk and Assumption Disclosure Annex

— QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Duțescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-FCM-2026-001
TITLE	Non-Compliance Conditions and Failure Modes - Deterministic Runtime Consent Enforcement
SUBTITLE	Formal Classification of Invalid System States, Enforcement Failures, and Certification Invalidation Conditions
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Failure Classification Specification
DOCUMENT STATUS	Normative
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 17 PART III · OPERATIONAL DOCUMENTS

System Boundary and Trust Model Specification

Formal definition of system boundaries, trust zones, enforcement location, and trust invariants for deterministic runtime consent enforcement architecture.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-SBTM-2026-001	1.0	2026	Foundational Specification	Final	NORMATIVE

ABSTRACT

Deterministic runtime consent enforcement requires an unambiguous, closed definition of where the system begins, where it ends, which components carry trust, and how that trust is governed. Without explicit boundary definitions, enforcement guarantees cannot be stated, tested, or audited. This document defines the complete system boundary and trust model for the QODIQA architecture. Nine canonical trust zones are established, each with a precisely bounded trust level, enumerated capabilities, prohibited assumptions, and a characterized attack surface. The Enforcement Gateway is defined as the single, non-relocatable, non-delegable enforcement boundary. Rules governing cross-boundary data flows, trust downgrade requirements, containment of untrusted components, and mandatory fail-closed responses are specified without qualification. System invariants are enumerated and referenced throughout the specification. Every statement in this document is normative. Every statement is enforceable, testable, and auditable. Any ambiguity in any boundary or trust definition constitutes a non-compliance condition and invalidates all enforcement guarantees for the affected deployment.

SECTION 1

Executive Definition

Three foundational definitions govern this specification. All subsequent boundary and trust requirements are derived from and constrained by these definitions. Deviation from any definition in an implementation constitutes an immediate non-compliance condition.

DEFINITION 1: SYSTEM BOUNDARY

The system boundary is the complete, explicit demarcation separating components subject to QODIQA enforcement governance from components that are not. The boundary is defined structurally. A component is inside the boundary if and only if it satisfies every inclusion criterion in Section 2.2. All other components are outside the boundary. No component occupies an intermediate or ambiguous position. Intermediate positions are INVALID.

DEFINITION 2: TRUST MODEL

The trust model is the complete set of explicit trust assignments, trust contracts, trust constraints, and trust revocation rules governing all interactions among all components within and across the system boundary. Trust is NEVER inferred, assumed, or inherited. Every trust relationship MUST be explicitly defined, minimally scoped, cryptographically verifiable where required, and revocable on demand. The default trust assignment for any component not explicitly granted trust is zero.

DEFINITION 3: ENFORCEMENT BOUNDARY

The enforcement boundary is the single, uniquely defined structural location at which all consent enforcement decisions are made. The enforcement boundary is coincident with the Enforcement Gateway (TZ-4). No enforcement decision may be made outside this location. No inference request may proceed to model execution without traversing this boundary. The enforcement boundary MUST NOT be relocated, duplicated, or bypassed by any conformant implementation.

INVARIANT I-00: DEFINITION IMMUTABILITY

Definitions 1, 2, and 3 are immutable at runtime. No operational condition, integration requirement, or performance optimization modifies the scope of any definition. An implementation that narrows, broadens, or reinterprets any definition is NON-COMPLIANT.

SECTION 2

System Boundary Definition

The system boundary is fully specified by the four subsections below. Each subsection is normative. Implementations MUST satisfy all four without exception or qualification.

2.1 Formal Boundary Definition

INSIDE THE BOUNDARY: ENUMERATION

- The Enforcement Gateway (TZ-4) in its entirety, including all enforcement decision logic, request validation routines, and blocking mechanisms.
- The Consent Registry (TZ-5), including all consent state storage, resolution interfaces, and validity verification mechanisms.
- The Policy Engine (TZ-6), including all policy definition storage, version management, and evaluation logic.
- The Audit and Logging Layer (TZ-9), including all decision record generation, tamper-evidence mechanisms, and log integrity verification.
- All cryptographic key management infrastructure used to sign or verify enforcement artifacts, consent tokens, and decision records.
- All inter-component communication channels between the above components, including transport security layers dedicated exclusively to enforcement operations.

OUTSIDE THE BOUNDARY: ENUMERATION

- All AI model inference engines, regardless of deployment location or ownership.
- All application layers that submit inference requests to the enforcement gateway.
- All user-controlled interfaces, including consent declaration surfaces, revocation interfaces, and subject-facing portals.
- All external APIs, third-party services, and downstream data consumers that receive outputs of permitted inference operations.
- All infrastructure not enumerated in the inside-boundary list above, including general-purpose compute, storage, and network layers not dedicated exclusively to enforcement operations.

VIOLATION CONSEQUENCE

Any component that participates in enforcement-related operations but is not assigned to the inside-boundary enumeration above is operating outside the enforcement boundary. Actions taken by such a component MUST NOT be treated as enforcement decisions. Enforcement claims based on such actions are INVALID.

2.2 Inclusion Criteria

A component is included within the system boundary if and only if ALL of the following conditions are satisfied simultaneously. Satisfaction of a subset is insufficient.

INCLUSION CRITERIA: FULL SET

- The component participates directly in the enforcement decision path, the consent resolution path, the policy evaluation path, or the audit record generation path.
- The component is subject to the access controls, integrity requirements, and cryptographic verification requirements specified in the QODIQA Security and Cryptographic Profile.
- The component's behavior is fully deterministic with respect to its inputs. Probabilistic or adaptive behavior disqualifies a component from inclusion.
- The component can be independently audited, and its outputs can be verified against declared inputs.
- The component is operated under a governance configuration satisfying the QODIQA Certification Framework requirements for its functional class.

CONSTRAINT: PARTIAL CRITERIA SATISFACTION

A component that satisfies some but not all inclusion criteria MUST NOT be assigned to the inside-boundary component set. Partial satisfaction is equivalent to non-satisfaction for boundary assignment purposes.

2.3 Exclusion Criteria

A component is excluded from the system boundary if any one of the following conditions is true. The exclusion criteria are disjunctive; a single positive condition mandates exclusion.

EXCLUSION CRITERIA: ANY ONE SUFFICIENT

- The component produces probabilistic, stochastic, or adaptive outputs, including but not limited to neural language models, retrieval-augmented generation systems, and reinforcement learning agents.
- The component is controlled, operated, or configurable by an entity that has not obtained QODIQA certification for that component class.
- The component's behavior cannot be fully reproduced from a defined set of inputs without external state dependencies outside the enforcement boundary.
- The component is a user-facing interface through which consent declarations or revocations are submitted.
- The component serves functions unrelated to enforcement, consent resolution, policy evaluation, or audit, even if physically co-located with inside-boundary components.

VIOLATION CONSEQUENCE

Any component meeting one or more exclusion criteria that has been placed inside the enforcement boundary introduces an unverified component into the trusted execution environment. All enforcement decisions produced while this condition exists are INVALID. The deployment MUST be suspended pending boundary remediation.

2.4 Boundary Immutability Constraint

CONSTRAINT: RUNTIME IMMUTABILITY

- No component may be added to the inside-boundary set without a formal revision of this specification under the QODIQA Governance Charter.
- No runtime condition, operational shortcut, or integration requirement may cause the boundary to expand beyond its defined extent.
- No trust granted to an inside-boundary component may be interpreted as extending to outside-boundary components, regardless of operational proximity or shared infrastructure.
- Implementers MUST document the complete inside-boundary component inventory and maintain it as a controlled artifact under the QODIQA Change Log and Version Control Protocol.

INVARIANT I-08: BOUNDARY IMMUTABILITY AT RUNTIME

The composition of the inside-boundary component set MUST NOT change at runtime. Any change requires a governed revision of this specification and re-certification of the affected deployment. An undocumented runtime boundary change is a critical security incident.

VIOLATION CONSEQUENCE

Any implementation in which the effective enforcement boundary differs from the boundary defined in this document, through expansion, contraction, or ambiguity, is NON-COMPLIANT. Non-compliance invalidates all enforcement guarantees for the affected deployment with immediate effect.

SECTION 3

Canonical Trust Zones

The QODIQA architecture defines exactly nine canonical trust zones. No trust zone other than those defined in this section may be recognized by a conformant implementation. Any component not assignable to one of the nine canonical zones MUST be treated as External Services (TZ-8) with zero trust. Each zone definition is structured as: trust level, capabilities, prohibited assumptions, and attack surface. This structure is uniform across all zones.

CONSTRAINT: ZONE ASSIGNMENT COMPLETENESS

Every component interacting with the QODIQA enforcement infrastructure MUST be assigned to exactly one canonical trust zone before any interaction is processed. Unassigned components are PROHIBITED from interacting with any inside-boundary component. A component with ambiguous zone assignment MUST be treated as TZ-8.

TZ-1 User-Controlled Zone — UNTRUSTED

TRUST LEVEL	Zero. All inputs from this zone are treated as adversarial until validated at the enforcement boundary. No elevated trust is granted regardless of claimed user identity or prior session history.
CAPABILITIES	Submission of consent declarations and revocations to the Consent Registry (TZ-5) through authenticated channels exclusively. No direct access to TZ-4, TZ-6, TZ-9, or cryptographic key infrastructure.
PROHIBITED	Direct access to the enforcement gateway, policy engine, audit log, or signing key material. Assumption that submitted consent is active without registry confirmation. Inference of prior consent validity.
ATTACK SURFACE	Consent injection through forged declarations. Replay of previously valid consent after revocation. Scope inflation in consent declaration parameters. Social engineering of consent submission workflows to manufacture broader authorization.

TZ-2 Input Zone — UNTRUSTED

TRUST LEVEL	Zero. All data and control signals entering from this zone are subjected to full structural and cryptographic validation before any enforcement processing. No exceptions are permitted.
CAPABILITIES	Submission of inference requests to the enforcement gateway through the published request interface only. Provision of intent declarations, subject identifiers, and purpose specifications as part of structured request payloads.
PROHIBITED	Assertion of enforcement outcomes. Pre-declaration of consent validity. Provision of policy directives or evaluation instructions. Any attempt to influence enforcement logic through input channel manipulation.
ATTACK SURFACE	Malformed intent declarations designed to trigger evaluation edge cases or evaluation-logic errors. Purpose field manipulation to obtain permits for non-consented actions. Request flooding to degrade enforcement availability. Injection of structured data mimicking internal enforcement signals to cause misclassification.

TZ-3 Application Layer — UNTRUSTED

TRUST LEVEL	Zero. The application layer is outside the enforcement boundary. Its correct operation is not assumed. Its claimed identity MUST be verified at the enforcement gateway on every request without exception.
CAPABILITIES	Construction and submission of enforcement requests. Receipt of enforcement decisions. Routing of permitted inference requests to model providers upon receipt of a valid, gateway-issued permit token for that specific request.
PROHIBITED	Routing of inference requests to model providers without a valid gateway-issued permit token for that specific request. Caching of permit tokens across requests. Modification of declared intent between gateway submission and permit receipt. Self-assertion of enforcement authority.
ATTACK SURFACE	Permit token replay across distinct requests. Gateway bypass through direct model provider routing. Permit token fabrication or parameter modification after issuance. Scope extension of permitted actions beyond what the permit specifies. Misrepresentation of requestor identity to obtain permits.

TZ-4 Enforcement Gateway — TRUSTED CORE

TRUST LEVEL	Trusted. Trust is contingent on continuous satisfaction of the certification requirements in the QODIQA Certification Framework. Certification lapse immediately reverts TZ-4 to untrusted status.
CAPABILITIES	Receipt and full structural validation of inference requests from TZ-2 and TZ-3. Invocation of consent resolution against TZ-5. Invocation of policy evaluation against TZ-6. Issuance of cryptographically signed permit or deny decisions. Generation of decision records to TZ-9. Blocking of all requests for which a valid permit cannot be issued.

PROHIBITED	Delegation of enforcement decisions to any other component. Issuance of conditional or provisional permits. Granting of trust to TZ-7 or TZ-8 components. Processing of requests outside defined input channels. Modification of audit records after generation. Permit issuance when any pre-condition in Section 5 is unconfirmed.
ATTACK SURFACE	Process integrity compromise enabling silent decision manipulation. Injection of forged consent responses from a compromised TZ-5 component. Policy recommendation tampering from a compromised TZ-6 component. Denial-of-service degradation targeting fail-open behavior. Timing side-channel analysis of decision latency to infer enforcement logic.

INVARIANT I-01: NO INFERENCE WITHOUT ENFORCEMENT

No inference request may produce a model output unless a valid, gateway-issued permit token for that specific request exists and has been verified by the forwarding component. This invariant holds without exception under all operational conditions.

TZ-5 Consent Registry

CONDITIONALLY TRUSTED

TRUST LEVEL	Conditional. Trusted only for consent state resolution responses that are cryptographically signed and correspond to a registered, non-revoked consent record within its validity window. Unsigned, expired, or unverifiable responses are INVALID and MUST NOT be acted upon.
CAPABILITIES	Storage and retrieval of consent records. Verification of consent validity for a specified subject-purpose-scope combination at a specified point in time. Issuance of signed consent validity responses to TZ-4. Receipt of consent revocations from TZ-1 through authenticated interfaces exclusively.
PROHIBITED	Issuance of validity affirmations for revoked consent. Derivation of consent from the absence of explicit revocation. Extension of consent scope beyond declared parameters. Provision of consent validity responses without cryptographic signature.
ATTACK SURFACE	Tampering with stored consent records to manufacture false consent state. Suppression of revocation records to extend validity of withdrawn consent. Replay of previously valid signed responses after consent expiration. Signing key compromise enabling forged validity affirmations indistinguishable from legitimate ones.

TZ-6 Policy Engine

DETERMINISTIC TRUSTED

TRUST LEVEL	Trusted, contingent on verified policy version integrity. The policy engine is trusted only when operating on a policy version whose integrity has been verified against the canonical policy hash registered in the enforcement governance record. An unverified policy version immediately reverts TZ-6 to untrusted status.
CAPABILITIES	Evaluation of declared intent and resolved consent state against the active, verified policy version. Production of a binary permit or deny recommendation for TZ-4. Provision of policy evaluation audit metadata including the specific policy version identifier and rule identifiers applied.

PROHIBITED	Production of probabilistic, conditional, or ambiguous recommendations. Evaluation against an unverified or unsigned policy version. Incorporation of runtime state or external signals beyond declared intent and consent resolution inputs. Override of recommendations by any external signal.
ATTACK SURFACE	Policy content modification to relax enforcement conditions. Policy version substitution to apply outdated or attacker-controlled logic. Evaluation context manipulation to bias decision outcomes. Corruption of the policy hash verification mechanism to suppress detection of tampered policy content.

TZ-7 Model Provider — UNTRUSTED

TRUST LEVEL	Zero. Model providers are explicitly outside the enforcement boundary. Their outputs carry no enforcement authority. Their internal behavior cannot be verified. Their operation does not constitute enforcement under any characterization.
CAPABILITIES	Receipt of inference requests forwarded by TZ-3 upon presentation of a valid, gateway-issued permit token. Generation of inference outputs. No direct interaction with any inside-boundary component is permitted.
PROHIBITED	Receipt of inference requests not accompanied by a valid permit token. Direct communication with TZ-4, TZ-5, TZ-6, or TZ-9. Self-determination of enforcement status. Modification of permit token parameters. Propagation of enforcement claims.
ATTACK SURFACE	Bypass of gateway enforcement through direct routing by a compromised TZ-3 application layer. Output manipulation to circumvent purpose restrictions applied at policy evaluation. Exfiltration of permit token parameters enabling replay attacks against the enforcement infrastructure. Prompt injection attacks designed to cause the model to generate outputs that impersonate enforcement signals.

TZ-8 External Services — UNTRUSTED

TRUST LEVEL	Zero. All external services, APIs, downstream data consumers, and third-party integrations are untrusted. Any component not explicitly assigned to a defined canonical zone is assigned to this zone by default.
CAPABILITIES	Receipt of outputs from permitted inference operations forwarded by TZ-3. No direct access to any inside-boundary component. No capability to initiate interactions with TZ-4, TZ-5, TZ-6, or TZ-9.
PROHIBITED	Direct communication with any inside-boundary component. Provision of signals, configurations, or data that influence enforcement decision logic. Assertion of enforcement status on behalf of the system.
ATTACK SURFACE	Provision of malicious data to TZ-3 that is subsequently submitted as declared intent to TZ-4. Exfiltration of inference outputs beyond the scope of permitted purposes. Supply chain compromise through external dependencies loaded by inside-boundary components. Data poisoning attacks targeting consent records submitted through TZ-1.

TZ-9 Audit and Logging Layer

VERIFIABLE TRUSTED

TRUST LEVEL	Trusted for record generation and integrity verification. Trust is contingent on the tamper-evidence of log storage and the cryptographic integrity of all records. A compromised audit layer does not invalidate enforcement decisions but immediately and completely invalidates all post hoc auditability claims for the period of compromise.
CAPABILITIES	Receipt and storage of cryptographically signed enforcement decision records generated by TZ-4. Provision of decision records to authorized auditors upon authenticated request. Verification of record integrity against stored hash values. Detection and flagging of tampering events.
PROHIBITED	Modification of any stored enforcement decision record after generation. Deletion of records within the mandatory retention period. Provision of unsigned or unverified records as valid enforcement evidence. Suppression of decision records under any operational condition including partial failure.
ATTACK SURFACE	Retroactive record modification to suppress evidence of enforcement bypass. Log truncation or rotation attacks removing records of specific decisions. Injection of false records to manufacture compliance evidence. Audit record exfiltration exposing enforcement patterns that enable targeted bypass.

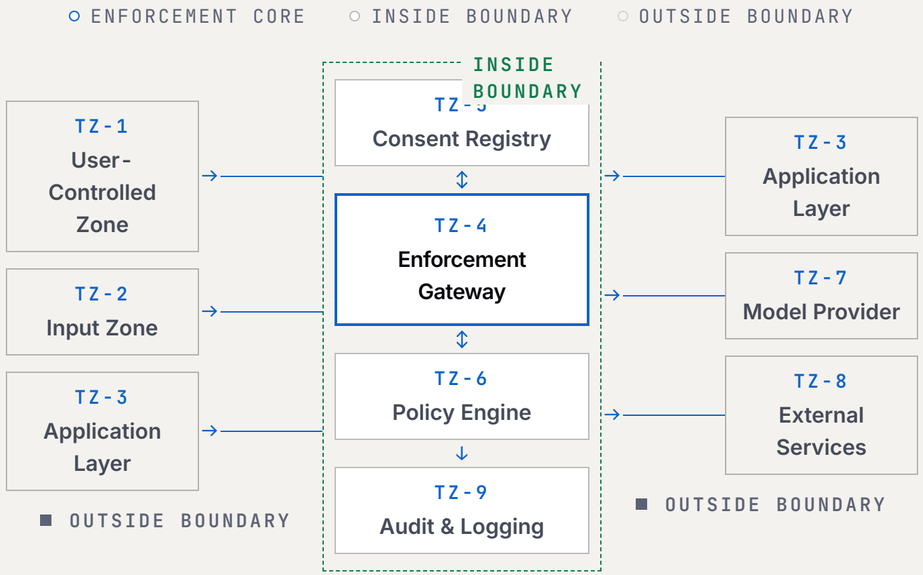
INVARIANT I-06: NO ENFORCEMENT DECISION WITHOUT AUDIT RECORD

Every enforcement decision, permit or deny, MUST generate a corresponding audit record in TZ-9 before the decision is communicated to the requesting component. A decision that cannot be audited MUST NOT be issued.

ARCHITECTURE REFERENCE

System Boundary and Trust Architecture

FIGURE 1: TRUST ZONE LAYOUT AND ENFORCEMENT FLOW



TZ-4 (ENFORCEMENT GATEWAY) IS THE SOLE ENFORCEMENT BOUNDARY. ALL INFERENCE REQUESTS TRAVERSE TZ-4. TZ-5, TZ-6, AND TZ-9 ARE INSIDE-BOUNDARY SUPPORTING COMPONENTS. ALL OTHER ZONES ARE OUTSIDE THE ENFORCEMENT BOUNDARY.

SECTION 4

Trust Model Definition

The QODIQA trust model is governed by five principles. Each principle is expressed as a rule, a constraint, and a consequence. Compliance with the trust model requires demonstrable adherence to all five principles simultaneously.

4.1 Trust Minimization

RULE

Every trust grant MUST be the minimum required for the grantee component to fulfill its defined function. Trust scope MUST NOT exceed the specific capability category for which trust has been granted. Each trust grant MUST be enumerated, scoped, and documented in the implementation's trust inventory.

CONSTRAINT

Trust granted for consent resolution MUST NOT extend to policy evaluation. Trust granted for audit record generation MUST NOT extend to enforcement decision-making. Broad or categorical trust grants are PROHIBITED. Any trust grant that bundles more than one capability category is INVALID.

CONSEQUENCE OF VIOLATION

A component operating with over-broad trust has access to capabilities it has not been authorized to exercise. Any enforcement decision influenced by actions outside the grantee's authorized capability scope is INVALID. The trust inventory MUST be remediated before further enforcement decisions are issued.

4.2 Explicit Trust Contracts

RULE

Every trust relationship between components MUST be defined as an explicit trust contract. A trust contract MUST specify: the granting component, the receiving component, the capability scope, the conditions under which trust is active, and the revocation mechanism. Trust contracts MUST exist in written form as part of the implementation's governance documentation.

CONSTRAINT

No trust relationship may exist without a corresponding trust contract. Implicit trust arising from operational proximity, shared infrastructure, or organizational relationships is PROHIBITED. Trust contracts MUST be available for inspection by authorized auditors at any time.

CONSEQUENCE OF VIOLATION

A trust relationship without an explicit contract is an undocumented trust dependency. All enforcement decisions produced under undocumented trust dependencies are INVALID. The affected deployment is NON-COMPLIANT until all trust relationships are backed by explicit contracts.

INVARIANT: NO UNDOCUMENTED TRUST DEPENDENCY

```
Every trust relationship active in any enforcement-relevant operation MUST have a corresponding, written trust contract in the governance documentation. An undocumented trust relationship is INVALID regardless of operational duration or organizational familiarity.
```

4.3 Revocation Requirement

RULE

Every trust grant MUST be revocable. Revocation MUST be executable within the time bound defined in the applicable trust contract. Revocation MUST take effect without requiring the consent or cooperation of the component whose trust is being revoked. Implementations MUST define and test the revocation path for every trust contract in their trust inventory.

CONSTRAINT

Trust grants with no defined revocation mechanism are INVALID and MUST NOT be issued. An untestable revocation mechanism is equivalent to no revocation mechanism for compliance purposes.

CONSEQUENCE OF VIOLATION

A non-revocable trust grant cannot be neutralized in response to a security incident. Any deployment containing a non-revocable trust grant is NON-COMPLIANT and cannot assert QODIQA conformance regardless of other properties.

4.4 Verifiability Requirement

RULE

Trust MUST be verifiable at the time of each interaction that depends on it. The trusting component MUST confirm, through cryptographic means or a deterministic verification protocol, that the trusted component's current state satisfies the conditions of the trust contract at the moment of interaction.

CONSTRAINT

Trust verified at initialization but not re-verified at interaction time does NOT satisfy this requirement. Session-level trust establishment without per-interaction verification is permitted only where the trust contract explicitly defines the session scope and the session integrity protection mechanism. Unverifiable trust is PROHIBITED.

CONSEQUENCE OF VIOLATION

A trust relationship that cannot be verified at interaction time may be exploited through component compromise between initialization and interaction. Enforcement decisions produced under unverified trust conditions are INVALID. The deployment MUST suspend enforcement operations until verifiability is restored.

4.5 Non-Transitive Trust Rule

RULE

Trust MUST NOT propagate across components. A trust grant from component A to component B does not confer any trust from A to any component C, regardless of the relationship between B and C. Every trust relationship MUST be established independently.

CONSTRAINT

The enforcement gateway's trust in TZ-5 MUST NOT be interpreted as trust in any data source that has written to TZ-5. The enforcement gateway's trust in TZ-6 MUST NOT be interpreted as trust in any entity that has contributed to policy authorship. Trust derived by transitivity from any other trust relationship is INVALID.

CONSEQUENCE OF VIOLATION

Transitive trust creates uncontrolled trust chains that allow untrusted entities to influence enforcement outcomes through trusted intermediaries. An implementation relying on transitive trust is structurally incapable of bounding its trust surface and is NON-COMPLIANT.

INVARIANT I-07: NO TRUST PROPAGATION

Trust granted to any component is scoped exclusively to that component and the capabilities enumerated in its trust contract. Trust MUST NOT propagate to any third component through any mechanism, including shared state, shared infrastructure, or API intermediation.

SECTION 5

Enforcement Boundary

The enforcement boundary is the most security-critical structural element in the QODIQA architecture. Its definition is exact, closed, and non-negotiable.

NORMATIVE DEFINITION: ENFORCEMENT BOUNDARY

The Deterministic Enforcement Boundary is the Enforcement Gateway (TZ-4). No other component, location, or logical construct constitutes an enforcement boundary in a QODIQA-conformant system. This definition admits no exceptions and no qualifications.

Location

RULE: ENFORCEMENT LOCATION

Enforcement occurs exclusively within TZ-4. TZ-4 is the singular, non-delegable decision authority for all consent enforcement outcomes. All inference requests MUST traverse TZ-4 before any interaction with a model provider. No exception to this requirement is permitted under any operational condition.

Why Only There

RULE: STRUCTURAL UNIQUENESS

TZ-4 is the only component simultaneously holding: (1) verified access to consent state via TZ-5, (2) verified access to policy logic via TZ-6, (3) the cryptographic authority to issue binding permit tokens, and (4) the structural position to block requests before model interaction. No other component possesses all four properties. Enforcement attempted at any other location lacks at least one property and is structurally unverifiable.

Impossible Enforcement Locations

CONSTRAINT: PROHIBITED ENFORCEMENT LOCATIONS

- **TZ-3 (Application Layer):** Untrusted zone. Enforcement logic placed here cannot be audited and is subject to bypass or replacement by the application operator. PROHIBITED.
- **TZ-5 (Consent Registry):** Consent resolution is a supporting function, not a decision authority. Conflating resolution with enforcement authority creates trust boundary ambiguity. PROHIBITED.
- **TZ-6 (Policy Engine):** Policy evaluation produces recommendations. Recommendations are not enforcement decisions. Granting decision authority to TZ-6 removes auditability from TZ-4. PROHIBITED.
- **TZ-7 (Model Provider):** Outside the system boundary and subject to no governance constraint. Self-enforcement by a model provider is PROHIBITED and cannot be recognized as QODIQA-conformant enforcement under any condition.
- **TZ-8 (External Services):** Zero trust. No enforcement authority can be delegated to a zero-trust zone. PROHIBITED.
- **Distributed / Multi-gateway architectures:** Multiple enforcement gateways produce no single auditable enforcement record and admit denial of authority in any specific component. PROHIBITED unless each gateway independently satisfies all TZ-4 requirements and each gateway's decisions are independently audited. Shared permit token issuance across gateways is PROHIBITED.

Invalid Architectures

INVALID ARCHITECTURE PATTERNS

- Any architecture in which TZ-3 can route inference requests to TZ-7 without a TZ-4-issued permit token.
- Any architecture in which TZ-7 accepts inference requests without verifying a valid TZ-4-issued permit token.
- Any architecture in which enforcement decisions are cached and reused across distinct requests.
- Any architecture in which the enforcement gateway can be bypassed by changing network routing configuration without governance approval.
- Any architecture in which the enforcement gateway issues permit tokens without consulting TZ-5 and TZ-6 for each decision.

Pre-Condition Validation

RULE: PRE-CONDITIONS FOR PERMIT ISSUANCE

Before issuing any permit decision, TZ-4 MUST verify all of the following. If any cannot be confirmed, TZ-4 MUST issue a deny decision immediately:

- The request is structurally valid and all required fields are present and well-formed.
- The requestor identity is cryptographically authenticated.
- The declared intent is within the processing scope of the enforcement gateway's policy configuration.
- TZ-5 is available and its consent validity response is cryptographically signed and temporally current.
- TZ-6 is available and its recommendation is based on a verified policy version.

Blocking Guarantees

RULE: DEFAULT BLOCKED STATE

The enforcement gateway MUST implement permit-absence as the default state. A request proceeds only if a permit token has been affirmatively issued for that specific request. Denial is the default. Permitting is the exception that requires positive confirmation of all pre-conditions.

INVARIANT I-05: NO PERMIT WITHOUT CONSENT AND POLICY CONFIRMATION

A permit token MUST NOT be issued unless both a signed consent validity affirmation from TZ-5 and a policy permit recommendation from TZ-6 have been received and verified for the specific request being evaluated. Absence of either input MUST result in deny.

NON-BYPASSABILITY CONSEQUENCE

Any architectural configuration in which an inference request can reach TZ-7 without traversing TZ-4 is an enforcement bypass condition. An enforcement bypass condition: (1) invalidates all QODIQA conformance claims for the affected deployment with immediate effect; (2) requires immediate system shutdown pending remediation; (3) constitutes a critical security incident requiring governance notification within the timeframe defined in the applicable governance configuration.

SECTION 6

Cross-Boundary Interaction Model

All interactions between components in different trust zones are cross-boundary interactions. Every cross-boundary interaction MUST conform to the rules in this section. Any interaction not explicitly authorized by these rules is PROHIBITED.

6.1 Data Flows

RULE: AUTHORIZED DATA FLOWS

- Data flowing from outside the enforcement boundary into TZ-4 MUST be treated as untrusted and validated in full before processing.
- Data flowing from TZ-5 or TZ-6 to TZ-4 MUST be cryptographically signed by the originating component and verified by TZ-4 before use.
- Data flowing from TZ-4 to outside-boundary components MUST consist only of permit tokens, deny signals, or explicitly authorized audit outputs.
- Data MUST NOT flow directly between TZ-7 or TZ-8 and any inside-boundary component.
- All data crossing the enforcement boundary MUST be logged to TZ-9 as part of the enforcement decision record.

CONSTRAINT: REJECTION CONDITIONS FOR DATA FLOWS

- Data arriving at TZ-4 from an unauthenticated source MUST be rejected immediately without processing.
- Data arriving from TZ-5 or TZ-6 without a valid cryptographic signature MUST be rejected. The pending enforcement decision MUST be converted to deny.
- Data containing fields not defined in the published request schema MUST be rejected. The request MUST NOT proceed.
- Data from TZ-7 or TZ-8 arriving at any inside-boundary component through any channel constitutes an unauthorized data flow and MUST be blocked.

MALFORMED INTERACTION CONSEQUENCE

A malformed data flow that is processed rather than rejected introduces unvalidated content into the enforcement trust domain. Any enforcement decision influenced by unvalidated content is INVALID. The malformed interaction MUST be logged as a security event in TZ-9.

6.2 Control Flows

RULE: AUTHORIZED CONTROL FLOWS

- Control authority flows inward toward TZ-4. No outside-boundary component may issue control instructions to any inside-boundary component.
- TZ-4 may issue control signals to TZ-5 and TZ-6 solely to invoke consent resolution and policy evaluation for a specific enforcement decision.
- TZ-4 MUST NOT delegate control authority to any outside-boundary component under any operational condition.
- Control flow from TZ-1 to TZ-5 for consent registration and revocation is permitted through the authenticated consent submission interface only.

CONSTRAINT: PROHIBITED CONTROL FLOWS

- TZ-3 MUST NOT issue control instructions to TZ-4, TZ-5, TZ-6, or TZ-9.
- TZ-7 and TZ-8 MUST NOT issue any control signals to any component in the architecture.
- TZ-5 and TZ-6 MUST NOT initiate unsolicited interactions with TZ-4.
- Any control signal originating from an outside-boundary component that reaches an inside-boundary component through an unauthorized channel MUST be rejected and logged as a critical security event.

UNAUTHORIZED CONTROL FLOW CONSEQUENCE

An unauthorized control flow reaching an inside-boundary component constitutes a boundary breach. The affected component MUST reject the signal and generate a critical audit record. If the breach cannot be immediately isolated, the deployment MUST be suspended.

6.3 Trust Downgrade Rules

RULE: TRUST DOWNGRADE ON CROSS-BOUNDARY TRANSFER

When data crosses from a lower-trust zone to a higher-trust zone, the data MUST be downgraded to the trust level of the originating zone and treated accordingly, regardless of its claimed origin. Trust downgrades are irreversible within a single enforcement transaction. No trust elevation may result from cross-boundary data passage.

CONSTRAINT: NO TRUST ELEVATION

A component MUST NOT treat data from TZ-2 as trusted because it was routed through TZ-3. A component MUST NOT treat data from TZ-8 as trusted because it was reformatted by TZ-3. Trust is assigned based on the original source zone, not the most recent forwarding zone. Any implementation that elevates trust based on forwarding path is INVALID.

TRUST ELEVATION CONSEQUENCE

Trust elevation through forwarding is a structural exploit path that allows untrusted data to be laundered into the enforcement trust domain through a trusted intermediary. Any enforcement decision influenced by laundered trust is INVALID. The affected interaction path MUST be remediated immediately.

6.4 Validation Requirements

RULE: FULL VALIDATION AT EVERY BOUNDARY CROSSING

Every cross-boundary interaction MUST be validated at the receiving boundary before the data or control signal is processed. Validation MUST confirm: structural integrity, cryptographic authenticity where applicable, zone-appropriate trust level, and conformance with the declared interaction protocol.

CONSTRAINT: NO PARTIAL VALIDATION

Partial validation is PROHIBITED. A cross-boundary interaction is either fully valid or fully invalid. There is no intermediate acceptance state. A validation failure on any single criterion MUST result in immediate rejection of the entire interaction.

INVARIANT I-03: NO CROSS-BOUNDARY EXECUTION WITHOUT FULL VALIDATION

No data or control signal crossing a trust zone boundary may be acted upon until it has been fully validated in conformance with this section. Partial validation is INVALID. A cross-boundary interaction validated on three of four criteria is NOT validated.

VALIDATION FAILURE CONSEQUENCE

A validation failure MUST generate an audit record in TZ-9 identifying the rejection reason, the source zone, and the interaction type. The failed interaction MUST NOT be retried without resolving the validation failure. Repeated validation failures from the same source zone MUST be treated as a potential attack pattern and escalated to the governance authority.

SECTION 7

Untrusted Component Containment

Untrusted components MUST be contained. Containment means that a component cannot affect enforcement outcomes through any channel other than the formally defined, validated interaction interfaces specified in Section 6. Each class of untrusted component has specific containment requirements, failure scenarios, and escalation paths.

Model Providers (TZ-7)

CONTAINMENT REQUIREMENTS

- No network path may exist between TZ-7 components and TZ-4, TZ-5, TZ-6, or TZ-9 components.
- Inference requests MUST reach TZ-7 through TZ-3 only, and only when accompanied by a valid permit token.
- Outputs from TZ-7 MUST NOT be returned to any inside-boundary component.
- TZ-4 MUST NOT inspect or process model outputs. Enforcement authority ends with permit issuance.

Containment Failure Scenarios: TZ-7

- **Direct routing bypass:** TZ-3 routes to TZ-7 without a valid permit token. Enforcement bypass condition. Immediate suspension of TZ-3 integration required.
- **Model output feedback loop:** TZ-7 outputs are returned to TZ-4 or TZ-5. The inside-boundary component is contaminated by untrusted data. All decisions produced during contamination are INVALID.
- **Permit token exfiltration:** TZ-7 extracts and reuses permit token parameters. Full permit token rotation MUST be executed immediately. Audit review of all decisions issued under the compromised token series is required.

External APIs (TZ-8)

CONTAINMENT REQUIREMENTS

- TZ-8 components MUST have no connectivity to any inside-boundary component.
- Inside-boundary components MUST NOT introduce service dependencies on TZ-8 components.
- External dependencies required for operational purposes unrelated to enforcement MUST be isolated in a separately classified outside-boundary component.

Containment Failure Scenarios: TZ-8

- **Supply chain compromise:** An external dependency loaded by an inside-boundary component is compromised. The inside-boundary component is immediately reclassified as potentially compromised. All decisions produced after the compromise point require forensic review.
- **Data exfiltration channel:** A TZ-8 component receives inside-boundary data through an undocumented channel. Classified as a cross-zone leakage event (TV-04). Critical incident response required.

Plugins and Extensions

CONTAINMENT REQUIREMENTS

- No plugin, extension, or dynamically loaded module may be executed within the enforcement boundary without prior formal evaluation and explicit inclusion in the inside-boundary component inventory.
- Plugins executing within TZ-3 or TZ-7 are subject to the untrusted zone rules of those zones without exception.
- Plugins that interact with TZ-4 MUST present authenticated requestor identities and MUST pass all TZ-4 validation requirements.

Containment Failure Scenarios: Plugins

- **Unevaluated plugin execution inside boundary:** Immediate suspension of all enforcement operations. Complete boundary audit required before resumption.
- **Plugin identity spoofing:** A plugin presents a fabricated authenticated identity to TZ-4. Constitutes a TV-01 boundary breach. Suspension and forensic review required.

User Inputs (TZ-1, TZ-2)

CONTAINMENT REQUIREMENTS

- All user inputs MUST be sanitized and structurally validated before submission to any inside-boundary component.
- TZ-4 MUST NOT execute any logic derived from or influenced by raw user input.
- Declared intent is used only as a lookup key in consent and policy evaluation. It is NEVER executed, interpreted, or evaluated as a general-purpose instruction.
- No user input may influence enforcement decision logic directly or through an intermediary.

Containment Failure Scenarios: User Inputs

- **Intent field injection:** A user crafts an intent declaration that causes TZ-4 to execute unintended logic paths. The affected decision is INVALID. Input validation MUST be hardened before enforcement resumes.
- **Consent scope inflation:** A user submits a consent declaration with a broader scope than intended through exploiting validation gaps in TZ-1. All downstream enforcement decisions relying on the inflated consent record are INVALID pending registry audit.

INVARIANT I-04: NO MODEL INTERACTION OUTSIDE ENFORCEMENT BOUNDARY

No inside-boundary component MUST initiate or participate in a direct interaction with TZ-7. All model interactions MUST occur outside the enforcement boundary, initiated by TZ-3 upon receipt of a valid permit token from TZ-4.

SECTION 8

Trust Violation Scenarios

The following trust violation scenarios are explicitly recognized by this specification. Each scenario defines a detection condition, a mandatory response, and a conformance consequence. Mandatory responses are unconditional. No operational condition, performance requirement, or availability constraint overrides a mandatory response. Reduction in response severity for any enumerated scenario constitutes an independent non-compliance condition.

TABLE 8-A: TRUST VIOLATION SCENARIOS, MANDATORY RESPONSES, AND CONFORMANCE CONSEQUENCES

ID	VIOLATION TYPE	DETECTION CONDITION	MANDATORY RESPONSE	CONFORMANCE CONSEQUENCE
TV-01	Boundary Breach	An inference request reaches TZ-7 without a valid TZ-4-issued permit token, detectable through permit token verification at the model access point or through audit log correlation.	Immediate block of request. Session termination. Critical audit record generation. Governance authority notification. Immediate suspension of the originating integration account pending investigation.	All enforcement decisions produced by the affected integration from the point of boundary breach are INVALID. Certification is suspended. Resumption requires full architectural audit and re-certification.

ID	VIOLATION TYPE	DETECTION CONDITION	MANDATORY RESPONSE	CONFORMANCE CONSEQUENCE
TV-02	Implicit Trust Introduction	An inside-boundary component is found accepting data, instructions, or configuration from a component absent from the authorized trust inventory, without explicit validation per Section 6.4.	Immediate quarantine of the affected component. Suspension of all enforcement operations routed through it. Full audit of all decisions produced during the implicit trust exposure period. Invalidation of affected decisions pending audit outcome.	The affected component is reclassified as untrusted. All decisions it participated in are INVALID pending audit. Deployment is NON-COMPLIANT until trust inventory is remediated and full re-certification is obtained.
TV-03	Enforcement Bypass	Audit log analysis reveals a permitted inference operation not preceded by a logged enforcement decision for the same request identifier, or a permit token presented at model access does not match the TZ-4-issued token.	Immediate suspension of the relevant TZ-3 integration. Invalidation of all permit tokens currently in circulation for that integration. Full forensic review of the bypass mechanism. Governance authority report within the timeframe defined in the applicable governance configuration.	The deployment's enforcement guarantee is null for the period of bypass. All inference operations during the bypass period are unenforceable. QODIQA conformance certification is immediately revoked pending investigation outcome.
TV-04	Cross-Zone Data Leakage	Inside-boundary data, including enforcement decision logic, policy content, consent records, or audit entries, is found in TZ-7 or TZ-8 through a channel not defined in the cross-boundary interaction model.	Critical security incident classification. Execution of the incident response procedure in the QODIQA Security and Cryptographic Profile. Revocation of all signing keys potentially exposed during the leakage period. Notification of affected consent subjects. Immediate certification suspension.	The confidentiality and integrity of the enforcement boundary are compromised. All enforcement decisions produced after the leakage start point require forensic review. Certification MUST NOT be reinstated without independent security audit.
TV-05	Consent Registry Integrity Failure	TZ-4 receives a consent validity response from TZ-5 that fails signature verification, contains a timestamp outside the acceptable validity window, or references a consent record absent from the tamper-evident log.	Response treated as INVALID. Deny decision issued for the pending request. Audit record generated indicating registry integrity failure. TZ-5 suspended as trusted component. Fail-closed on all subsequent requests until TZ-5 trust is re-established through re-verification.	TZ-5 is downgraded to untrusted status. All enforcement decisions issued while TZ-5 integrity was compromised MUST be reviewed. Decisions relying on the compromised consent state are INVALID.
TV-06	Policy Version Integrity Failure	TZ-6 returns a recommendation based on a policy version whose hash does not match the canonical hash in the enforcement governance record, or the policy version identifier is unrecognized by the governance authority.	Recommendation treated as INVALID. Deny decision issued for the pending request. Audit record generated indicating policy integrity failure. All policy evaluations halted until the verified policy version is loaded and confirmed. Review of all decisions issued under the unverified policy version.	TZ-6 is downgraded to untrusted status for the period of policy version integrity failure. Decisions based on the unverified policy are INVALID and MUST be disclosed to relying parties.

INVARIANT: FAIL-CLOSED DEFAULT FOR UNENUMERATED VIOLATIONS

In any trust violation scenario not enumerated in Table 8-A, the system MUST default to deny. Absence of a specific enumerated response rule for a detected anomaly mandates the most conservative available response: block the request, generate a critical audit record, and escalate to the governance authority. Permit issuance is PROHIBITED when the system state is anomalous.

SECTION 9

Trust Reduction Strategy

The QODIQA trust model MUST be continuously hardened. Trust reduction is a mandatory evolution direction, not an optional improvement. Each rule below is normative.

Rule 1: Assumption Elimination

MANDATORY RULE

Every existing trust relationship MUST be reviewed at each formal version cycle to determine whether the underlying assumption remains valid. Where an assumption can be replaced by a cryptographic verification, the substitution is MANDATORY. Where an assumption cannot be eliminated, it MUST be documented as a known trust dependency and tracked as a risk item under the QODIQA Risk and Assumption Disclosure Annex.

MEASURABLE CONSTRAINT

Undocumented trust assumptions are NON-COMPLIANT. An assumption-to-verification substitution deferred without documented justification is NON-COMPLIANT. Each version cycle MUST produce a trust assumption delta report enumerating: assumptions eliminated, assumptions pending elimination, and assumptions accepted as residual with risk justification.

CONSEQUENCE OF NON-ELIMINATION

Trust assumptions that persist without review accumulate as undocumented risk. A deployment that fails to produce a trust assumption delta report at each version cycle is NON-COMPLIANT and MUST NOT have its certification renewed.

Rule 2: Progressive Hardening

MANDATORY RULE

Each successive version of this specification MUST reduce, or hold constant, the total number of trust grants in the canonical trust model. Trust grants MUST NOT be added except where a new component is formally incorporated into the inside-boundary component set through the full governance process. Trust grants MUST NOT be added to address operational convenience, performance optimization, or integration simplification.

MEASURABLE CONSTRAINT

The trust grant count at version N+1 MUST be less than or equal to the trust grant count at version N. Any version in which the trust grant count increases without a formally approved boundary expansion is NON-COMPLIANT.

CONSEQUENCE OF TRUST SURFACE EXPANSION

A version that expands the trust surface without formal approval introduces unreviewed trust dependencies into the architecture. The affected version MUST NOT be deployed in production. The governance authority MUST review and explicitly approve any trust surface expansion before it is permitted in a normative version.

INVARIANT: NON-INCREASING TRUST SURFACE

The total trust grant count of the canonical trust model MUST NOT increase between successive specification versions without a formally approved, governed boundary expansion. A version violating this invariant MUST NOT be deployed.

Rule 3: Reduction of Conditional Trust Dependencies

MANDATORY RULE

Implementations MUST actively reduce dependencies on conditionally trusted components. The conditional trust assigned to TZ-5 (Consent Registry) is the highest residual trust dependency in the architecture. Implementations MUST prioritize cryptographic mechanisms that enable TZ-4 to verify consent validity without relying on a live registry response, such as cryptographically self-contained consent tokens with short validity periods and embedded revocation indicators.

MEASURABLE CONSTRAINT

Implementations MUST document the maximum tolerated TZ-5 unavailability period before fail-closed behavior is triggered. This period MUST decrease at each version cycle where architectural changes permit. An implementation with no documented TZ-5 unavailability tolerance bound is NON-COMPLIANT.

CONSEQUENCE OF DEPENDENCY RETENTION

Unreduced conditional trust dependencies sustain attack surfaces that cryptographic self-containment would eliminate. An implementation that retains avoidable conditional trust dependencies without documented justification is failing its trust reduction obligation. Certification renewal MUST NOT be granted without evidence of active trust reduction effort.

SECTION 10

System Invariants

The following invariants MUST hold continuously in every conformant QODIQA implementation. An invariant is a property that MUST be true at every observable point in system operation, including degraded and adversarial conditions. Invariants I-01 through I-08 are referenced at their point of first relevance throughout this specification and consolidated here for auditability. Failure of any invariant is a critical system fault requiring immediate fail-closed response.

INVARIANT I-00: DEFINITION IMMUTABILITY

Definitions 1, 2, and 3 in Section 1 are immutable at runtime. No operational condition modifies the scope of any definition. Redefinition without formal specification revision is NON-COMPLIANT. First referenced: Section 1.

INVARIANT I-01: NO INFERENCE WITHOUT ENFORCEMENT

No inference request may produce a model output unless a valid, gateway-issued permit token for that specific request exists and has been verified by the forwarding component. No exception. First referenced: Section 3 (TZ-4).

INVARIANT I-02: NO TRUST WITHOUT VERIFICATION

No component MUST treat another component as trusted unless the trust relationship has been explicitly established through a defined trust contract and the conditions of that contract have been verified for the current interaction.

INVARIANT I-03: NO CROSS-BOUNDARY EXECUTION WITHOUT FULL VALIDATION

No data or control signal crossing a trust zone boundary MUST be acted upon until it has been fully validated per Section 6.4. Partial validation is INVALID. First referenced: Section 6.4.

INVARIANT I-04: NO MODEL INTERACTION OUTSIDE ENFORCEMENT BOUNDARY

No inside-boundary component MUST initiate or participate in a direct interaction with TZ-7. All model interactions MUST originate from TZ-3 upon receipt of a valid permit token. First referenced: Section 7.

INVARIANT I-05: NO PERMIT WITHOUT CONSENT AND POLICY CONFIRMATION

A permit token MUST NOT be issued unless both a signed TZ-5 consent validity affirmation and a TZ-6 policy permit recommendation have been received and verified for the specific request. Absence of either MUST result in deny. First referenced: Section 5.

INVARIANT I-06: NO ENFORCEMENT DECISION WITHOUT AUDIT RECORD

Every enforcement decision, permit or deny, MUST generate a corresponding audit record in TZ-9 before the decision is communicated to the requesting component. A decision without a pre-committed audit record MUST NOT be issued. First referenced: Section 3 (TZ-9).

INVARIANT I-07: NO TRUST PROPAGATION

Trust granted to any component is scoped exclusively to that component and the capabilities in its trust contract. Trust MUST NOT propagate to any third component through any mechanism. First referenced: Section 4.5.

INVARIANT I-08: BOUNDARY IMMUTABILITY AT RUNTIME

The composition of the inside-boundary component set MUST NOT change at runtime. Any change requires a governed specification revision and deployment re-certification. An undocumented runtime change is a critical security incident. First referenced: Section 2.4.

SECTION 11

Non-Compliance Conditions

Each condition below independently constitutes non-compliance with this specification. A NON-COMPLIANT implementation MUST NOT assert QODIQA conformance and MUST NOT be presented to relying parties as QODIQA-conformant. Non-compliance conditions are individually sufficient; satisfaction of all other requirements does not offset any single violation.

- **Missing boundary definitions.** The implementation does not maintain a documented inside-boundary component inventory, or the inventory does not cover all components participating in enforcement, consent resolution, policy evaluation, or audit record generation.
- **Implicit trust.** Any trust relationship exists in the implementation that is not documented in the trust inventory and backed by an explicit trust contract satisfying Section 4.2.
- **Enforcement outside the gateway.** Any enforcement decision is made, or any permit token is issued, by a component other than TZ-4.
- **Undefined zones.** Any component participating in enforcement-related operations is not assigned to one of the nine canonical trust zones defined in Section 3.
- **Fail-open behavior.** The system issues a permit decision when TZ-4 is unavailable, when consent verification fails, when policy evaluation fails, or when any pre-condition in Section 5 cannot be confirmed.
- **Trust propagation.** The implementation relies on transitive trust in any enforcement-relevant operation.

- **Unlogged decisions.** Any enforcement decision is issued without a corresponding audit record committed to TZ-9 before decision delivery.
- **Boundary bypass.** Any inference request reaches TZ-7 through a path that does not traverse TZ-4.
- **Non-revocable trust.** Any trust grant exists for which no revocation mechanism has been defined and successfully tested.
- **Unverified policy evaluation.** Any enforcement decision is based on a policy evaluation performed against a policy version not verified against the canonical policy hash.
- **Undocumented trust assumptions.** Any trust assumption exists in the implementation that is not documented in the trust assumption register maintained under the QODIQA Risk and Assumption Disclosure Annex.
- **Invalid architecture patterns.** The implementation matches any pattern enumerated in Section 5 under "Invalid Architectures."

SECTION 12

Conformance Requirements

QODIQA conformance with this specification requires demonstrable satisfaction of all requirements below. All items are mandatory. Partial conformance is not recognized. A claim of conformance that cannot be substantiated by evidence for every item below is INVALID.

What Must Be Proven

- The inside-boundary component inventory is complete, accurate, and maintained under the QODIQA Change Log and Version Control Protocol, demonstrable through version-controlled governance records.

- No inference request can reach TZ-7 without traversing TZ-4, demonstrable through architectural analysis and network topology documentation reviewed by an independent auditor.
- Every trust relationship in the implementation is documented in a trust inventory backed by explicit trust contracts satisfying Section 4.2, demonstrable through inspection of governance documentation.
- Fail-closed behavior per Sections 5 and 8 is implemented without exception, demonstrable through fault injection test results.
- All eight system invariants in Section 10 hold under normal, boundary-condition, and adversarial operation, demonstrable through structured test evidence.

What Must Be Testable

- Blocking behavior of TZ-4 for requests failing pre-condition validation, tested using the QODIQA Conformance Test Suite test cases.
- All six trust violation responses in Table 8-A, tested through controlled scenario injection with documented outcomes.
- Non-bypassability of the enforcement boundary, tested through network-level bypass attempts as defined in the Conformance Test Suite.
- Revocability of every trust grant in the trust inventory, tested through revocation execution and subsequent interaction attempt.
- Cryptographic verification of all cross-boundary signals from TZ-5 and TZ-6, tested through signature forgery rejection scenarios.
- Containment failure escalation paths defined in Section 7, tested through controlled containment breach scenarios.

What Must Be Auditable

- Every enforcement decision MUST be reconstructible from the TZ-9 audit record without access to runtime state.

- The policy version applied to every enforcement decision MUST be identifiable from the audit record.
- The consent state consulted for every enforcement decision MUST be identifiable from the audit record and verifiable against the consent registry log.
- The complete trust inventory MUST be available for inspection by authorized auditors at any time.
- The trust assumption delta report MUST be available for every completed version cycle.
- Any change to the inside-boundary component inventory MUST be recorded in the change log with the authorizing governance event referenced.

SECTION 13

Final Deterministic Statement

This specification defines the complete, closed system boundary and trust model for deterministic runtime consent enforcement architecture. No element of the boundary or trust model is open to interpretation. No gap in this specification may be filled by implementer discretion. Where this specification is silent on a specific scenario, the fail-closed rule applies unconditionally.

> **BOUNDARY VIOLATION: CONSEQUENCE**
>
> Any violation of the system boundary as defined in Section 2 immediately and completely invalidates all QODIQA conformance claims for the affected deployment. A deployment with a violated boundary MUST NOT assert that consent has been enforced for any inference operation, regardless of the intended behavior of TZ-4.

> **TRUST MODEL VIOLATION: CONSEQUENCE**
>
> Any violation of the trust model, including implicit trust introduction, trust propagation across components, or operation of a trust grant without a defined revocation mechanism, immediately invalidates the integrity of all enforcement decisions produced under the affected trust configuration. Decisions produced under a violated trust model MUST be disclosed to relying parties as potentially invalid.

INVARIANT FAILURE: CONSEQUENCE

Failure of any system invariant in Section 10 is a critical architectural fault. The affected deployment MUST be suspended immediately. Enforcement decisions produced during the period of invariant failure MUST NOT be presented as valid enforcement evidence. Resumption of deployment MUST be preceded by root cause analysis, remediation, and re-certification.

FINAL INVARIANT: FULL AND CONTINUOUS COMPLIANCE

Compliance with this specification requires full, continuous, and demonstrable adherence to every boundary definition, trust zone definition, trust model principle, enforcement boundary constraint, cross-boundary interaction rule, containment requirement, violation response, and invariant defined herein. No partial compliance is recognized. No waiver is permitted without formal specification revision under the QODIQA Governance Charter. The QODIQA system boundary and trust model is not a design recommendation. It is a structural requirement. Implementations that satisfy it can assert deterministic enforcement. Implementations that do not satisfy it cannot.

DOCUMENT STATUS

Document Status and Corpus Alignment

This document is a foundational specification within the QODIQA corpus. It defines the complete system boundary and trust model governing the QODIQA deterministic runtime consent enforcement architecture. All boundary definitions, trust zone assignments, enforcement location constraints, cross-boundary interaction rules, and system invariants set forth herein are normative and binding on all conformant implementations.

This specification is a prerequisite for the correct interpretation of all other QODIQA corpus documents that reference trust zones, enforcement boundaries, or cross-boundary interaction rules. Implementations MUST satisfy the requirements of this document before asserting conformance with any other QODIQA specification.

This document MUST be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Threat Model and Abuse Case Specification
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement

- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Governance Charter for the QODIQA Standard Corpus
- QODIQA: Terminology and Normative Definitions
- QODIQA: Residual Risk and Assumption Disclosure Annex

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dușescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-SBTM-2026-001
TITLE	System Boundary and Trust Model Specification
SUBTITLE	Deterministic Runtime Consent Enforcement for Artificial Intelligence Systems
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Foundational Specification
DOCUMENT STATUS	Final: Normative
CLASSIFICATION	Public
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 18 PART III · OPERATIONAL DOCUMENTS

Interoperability and Deployment Constraints

Formal declaration of the architectural boundaries governing system operation across heterogeneous AI environments, including integration constraints, non-override guarantees, and fail-closed enforcement properties.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-INTEROP-CONS-2026-001	1.0	2026	Normative Constraints Note	Institutional	NORMATIVE

ABSTRACT

This document constitutes the formal interoperability and integration constraints declaration for the QODIQA standard corpus. It defines the structural boundaries within which QODIQA operates across heterogeneous AI system architectures, specifies the integration interfaces that are permitted, and declares the invariants that no external system component is permitted to override.

This document does not describe interoperability features, integration tutorials, or vendor-specific implementation guidance. It declares the **hard architectural limits** within which any conformant integration must operate. Interoperability is subordinate to enforcement integrity. Where these objectives conflict, enforcement integrity prevails without exception.

The constraints herein are aligned with the QODIQA Core Standard, the 68-Point Enforcement Framework, the Reference Architecture, the Certification Framework, the Conformance Test Suite, and the Security and Cryptographic Profile. No

constraint defined in this document weakens, qualifies, or defers any provision established by those normative instruments.

All statements using **SHALL** designate mandatory invariants. All statements using **SHALL NOT** designate prohibited behaviors. All statements using **MUST** designate cryptographic or safety-critical requirements. These are not aspirational commitments. They are structural constraints.

SECTION 0

Purpose and Scope Boundary

This document establishes the formal interoperability constraints governing all QODIQA deployments across heterogeneous AI infrastructures and defines the architectural limits within which deterministic runtime consent enforcement operates. Its purpose is to declare, with precision and without qualification, the architectural boundaries within which QODIQA operates when deployed across multi-component, multi-vendor, or multi-layer AI system environments.

> This document defines interoperability **constraints**, not interoperability features. It does not describe how to integrate QODIQA. It describes what integration architectures are structurally permissible and what cannot, under any circumstance, be introduced through integration.

0.1 Scope Declaration

The scope of this document encompasses all external system interfaces that interact with, route traffic through, or operate in conjunction with the QODIQA Enforcement Gateway. This includes but is not limited to: AI orchestration frameworks, agent runtimes, multi-model routing systems, API gateways, reverse proxies, message queue systems, sidecar enforcement proxies, and network-layer enforcement components.

This document explicitly excludes: the internal implementation mechanics of the QODIQA Enforcement Gateway itself, model-internal safety classification mechanisms, training data governance, and organizational or legal compliance processes. Those subjects are addressed by other instruments in the corpus or are outside its scope entirely.

0.2 Foundational Constraint Declaration

PRIMARY INTEROPERABILITY INVARIANT	INV-INTEROP-000

Interoperability SHALL NOT introduce override paths across the deterministic enforcement boundary.

No integration architecture, regardless of its topology, vendor composition, or operational justification, is permitted to create conditions under which QODIQA enforcement logic is bypassed, deferred, weakened, or rendered conditional. This invariant is absolute and admits no exception.

INV-INTEROP-000 · Non-negotiable · Aligned with QODIQA Core Standard and 68-Point Enforcement Framework Domain I

Deterministic runtime enforcement is structurally prior to any integration concern. Any integration pattern that cannot operate within this constraint is non-conformant and SHALL NOT be deployed in conjunction with QODIQA.

SECTION 1

Definitions and System Boundary Vocabulary

The following definitions establish the precise vocabulary used throughout this document. All terms are aligned with the QODIQA Reference Architecture. Where a term appears in other corpus documents, the definition provided here is controlling for the purposes of interoperability analysis.

TERM	FORMAL DEFINITION
Enforcement Gateway	The singular, deterministic boundary component through which all inference requests and responses must transit. The Enforcement Gateway is the locus of consent validation, policy evaluation, and enforcement decision execution. It is not a proxy in the conventional sense; it is a structural enforcement point with no bypass provision.
Consent Registry	The authoritative, append-only data store that records consent grants, revocations, scope limitations, and temporal validity windows. The Consent Registry is read at enforcement time. It is not modified by inference operations or orchestration actions.
Policy Engine	The deterministic evaluation component that produces permit or deny decisions by evaluating resolved consent context against applicable policy rules. The Policy Engine produces no probabilistic outputs. Its decision function is a deterministic mapping from consent state and request attributes to a binary enforcement decision.
Trust Boundary	The structural demarcation between components that operate under QODIQA enforcement and components that do not. Components inside the Trust Boundary have received enforcement verdicts for their operations. Components outside the Trust Boundary have not. No component outside the Trust Boundary may present itself as enforcement-cleared without a valid, current enforcement decision.
Bypass Vector	Any architectural pathway, integration pattern, or operational procedure that enables inference operations to proceed without traversing the QODIQA Enforcement Gateway. Bypass Vectors are non-conformant by definition. Their existence in a deployed architecture constitutes a Critical non-conformance finding under the Certification Framework.
Override Attempt	Any action by an external component, system, or actor that attempts to alter, nullify, defer, or circumvent an enforcement decision produced by the Policy Engine. Override Attempts include: configuration changes that disable enforcement logic, injection of pre-authorized tokens by non-authoritative sources, and post-approval mutation of permitted request payloads.
Integration Surface	A defined interface type through which external components interact with the QODIQA Enforcement Gateway. Integration Surfaces are categorized by observability, enforceability, and trust boundary position. Only Integration Surfaces defined in Section 3 of this document are authorized for use in conformant deployments.
Control Plane	The administrative and configuration interface of the QODIQA system, used for policy management, consent registry administration, and enforcement configuration. The Control Plane is strictly separated from the Data Plane. Control Plane access does not grant Data Plane permissions and vice versa.

TERM	FORMAL DEFINITION
Data Plane	The operational inference traffic pathway through which requests and responses transit the Enforcement Gateway. Data Plane operations are subject to per-request enforcement decisions. Data Plane components do not have access to Control Plane configuration.

SECTION 2

Layered Integration Model

QODIQA enforcement is designed to operate within layered AI system architectures. This section defines the integration layers recognized by the QODIQA framework and specifies the enforcement constraints applicable to each layer. The layered model does not imply that all layers must be present in every deployment. It defines the architectural positions at which enforcement can be instantiated and the constraints that apply when enforcement operates at each position.

> The enforcement constraint applicable to every layer is identical: **no inference operation may proceed without a valid enforcement decision.** The layered model describes where enforcement is instantiated; it does not create layer-specific exceptions to this requirement.

2.1 Pre-Inference Boundary Layer

The Pre-Inference Boundary Layer encompasses all components that process, route, or transform inference requests prior to their submission to a model provider. This layer includes API gateways, reverse proxies, request routers, and orchestration frameworks that operate upstream of model execution.

PRE-INFERENCE ENFORCEMENT REQUIREMENT — INV-INTEROP-011

All inference requests entering the Pre-Inference Boundary Layer SHALL transit the QODIQA Enforcement Gateway before being forwarded to any model provider endpoint.

Components in this layer that forward requests directly to model endpoints without Enforcement Gateway transit are Bypass Vectors and are prohibited in conformant deployments.

INV-INTEROP-011 · Critical · Verified in Conformance Test Suite Section 3.1

2.2 Inference Execution Layer

The Inference Execution Layer encompasses model provider endpoints and their immediate operational environment. QODIQA does not operate within the Inference Execution Layer. QODIQA enforcement occurs at the boundary of this layer, not inside it.

INFERENCE LAYER NON-PENETRATION REQUIREMENT — INV-INTEROP-012

QODIQA enforcement SHALL NOT attempt to operate within the internal computational environment of model providers.

Enforcement operates at the boundary of model execution: on requests before they reach model providers and on responses before they reach requesting components. Internal model behavior, intermediate computation states, and provider-side caching are outside the enforcement perimeter and are not subject to QODIQA invariants.

INV-INTEROP-012 · Scope Boundary · Clarifies enforcement perimeter limits

2.3 Post-Inference Processing Layer

The Post-Inference Processing Layer encompasses all components that receive, transform, store, or forward model responses after they have exited the Inference Execution Layer. This layer includes response parsers, output formatters,

downstream API consumers, storage systems, and notification pipelines.

POST-INFERENCE RESPONSE INTERCEPTION	INV-INTEROP-013

Model responses SHALL transit the QODIQA Enforcement Gateway before delivery to Post-Inference Processing Layer components.

Response-path enforcement is not optional. Enforcement decisions applicable to response content (scope limitations, output filtering requirements, consent-conditioned delivery constraints) are enforced at the response transit point. Responses that bypass the Enforcement Gateway on the return path constitute Bypass Vectors equivalent in severity to request-path bypasses.

INV-INTEROP-013 · Critical · Response-path enforcement requirement

2.4 External Orchestration Layer

The External Orchestration Layer encompasses agent frameworks, workflow orchestrators, multi-step inference pipelines, and autonomous systems that coordinate multiple inference operations. This layer presents the most complex interoperability challenges because its components may generate inference requests programmatically, may modify requests in transit, and may operate across trust boundaries that are not fully visible to individual request-level enforcement.

ORCHESTRATION LAYER PER-OPERATION ENFORCEMENT	INV-INTEROP-014

Each inference operation initiated by an External Orchestration Layer component SHALL receive an independent enforcement decision.

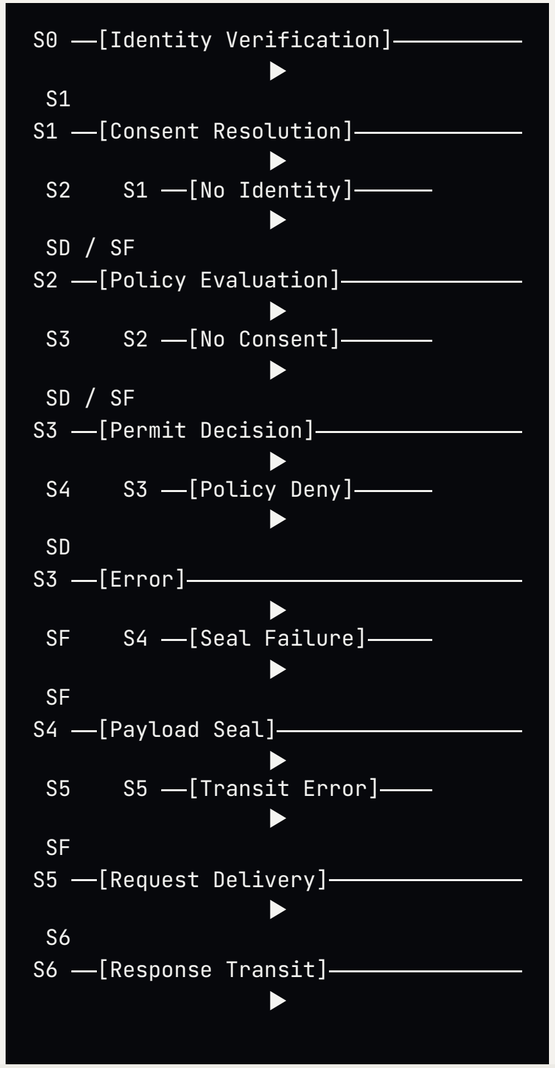
A permit decision for one operation in a multi-step workflow does not constitute authorization for subsequent operations. Orchestration frameworks that cache permit decisions and apply them to subsequent operations without re-evaluation are non-conformant. Per-operation enforcement is the invariant; orchestration convenience does not override it.

INV-INTEROP-014 · Critical · No permit caching across operations permitted

2.5 Interoperability State Model

This section defines the formal state machine governing the lifecycle of an inference operation within the QODIQA enforcement architecture. The state model provides the conceptual foundation for understanding which transitions are permitted, which are prohibited, and what conditions produce enforcement failure states.

```
States: S0 (Request Received) → S1 (Identity Verified) → S2 (Consent Resolved) → S3 (Policy Evaluated) → S4 (Permit Issued) → S5 (Payload Sealed) → S6 (Request Delivered) → S7 (Response Returned) | SD (Deny) | SF (Failure)
```


S0 —[Identity Verification]
S1
S1 —[Consent Resolution]
S2
S1 —[No Identity]
SD / SF
S2 —[Policy Evaluation]
S3
S2 —[No Consent]
SD / SF
S3 —[Permit Decision]
S4
S3 —[Policy Deny]
SD
S3 —[Error]
SF
S4 —[Seal Failure]
SF
S4 —[Payload Seal]
S5
S5 —[Transit Error]
SF
S5 —[Request Delivery]
S6
S6 —[Response Transit]

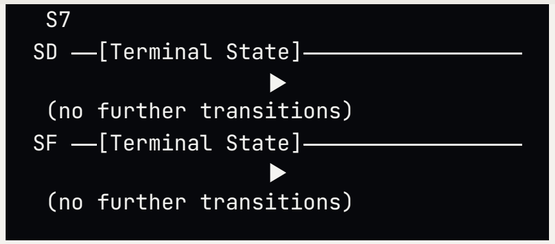


FIGURE 2.5: QODIQA ENFORCEMENT LIFECYCLE STATE MACHINE · ALL TRANSITIONS NOT ENUMERATED ARE PROHIBITED

2.5.2 Allowed and Prohibited Transitions

The permitted transition set for the nominal path is: S0 → S1 → S2 → S3 → S4 → S5 → S6 → S7. Deny transitions are: S1 → SD, S2 → SD, S3 → SD. Failure transitions are: S1 → SF, S2 → SF, S3 → SF, S4 → SF, S5 → SF.

TRANSITION	CONSTRAINTS
PROHIBITED	**All transitions not enumerated in the permitted transition set are prohibited.** Specifically: any transition that skips one or more intermediate states on the nominal path, any reverse transition from a later state to an earlier state, and any transition from SD or SF to any other state.
PROHIBITED	**S4 → S6 (skip S5) is prohibited.** A permit issued in S4 without payload integrity sealing in S5 cannot be delivered. The absence of a sealed payload signature is a transition condition failure that produces SF.
PROHIBITED	**S0 → S3 (skip S1, S2) is prohibited.** Policy evaluation cannot proceed without resolved identity and consent context. A Policy Engine invocation without prior identity verification and consent resolution produces SF.
PROHIBITED	**S6 → S3 (backtrack) is prohibited.** A delivery event cannot cause re-evaluation. Post-delivery enforcement re-evaluation is a separate operation lifecycle beginning at S0, not a continuation of the current lifecycle.

2.5.3 Formal Constraints on the State Machine

NO-SKIP CONSTRAINT — INV-INTEROP-025A

No inference operation may transition directly from S0 to S6 without traversing S1, S2, S3, S4, and S5 in order.

This constraint is absolute. Any architectural pattern, optimization, caching mechanism, or pre-authorization scheme that delivers a request to a model provider without the complete ordered traversal of S1 through S5 constitutes a Bypass Vector as defined in Section 1 and is non-conformant.

INV-INTEROP-025A · Absolute · No exception pathway · Aligned with Conformance Test Suite Section 8.1

NO-BACKTRACKING CONSTRAINT — INV-INTEROP-025B

State transitions in the enforcement lifecycle are strictly forward-directed. No operation that has reached state Sn may return to any state S(n-k) for $k \geq 1$ within the same operation lifecycle.

Conditions that might appear to require backtracking (updated consent arriving after S2 completion, policy changes detected after S3 completion) do not produce backtracking. They produce SF, requiring a new operation lifecycle beginning at S0 with current state.

INV-INTEROP-025B · Absolute · State machine integrity property

ATOMIC DECISION CONSTRAINT — INV-INTEROP-025C

The transition from S3 to S4 (permit issuance) is an atomic operation. It cannot be partially executed, deferred, or split across multiple computational units without cryptographic coordination that preserves atomicity guarantees.

Distributed enforcement architectures that evaluate policy rules across multiple nodes MUST produce a single, consolidated, cryptographically bound permit decision before transitioning to S4. No component downstream of S3 receives permit information until S4 is complete as an atomic unit.

INV-INTEROP-025C · MUST · Cryptographic requirement · Aligned with Security Profile Section 5

SECTION 3

Authorized Integration Surfaces

QODIQA enforcement can be deployed across a defined set of integration surface types. Each surface type has specific observability properties, enforcement capabilities, and trust boundary positions. Deploying QODIQA at a surface type not defined herein requires formal evaluation against the Certification Framework prior to production use.

3.1 REST and gRPC Boundary Enforcement

OBSERVABLE	Request headers, request body content, response status, response body content, caller identity claims, and temporal metadata.
ENFORCEABLE	Permit or deny based on consent context, policy evaluation, and request metadata. Response interception and blocking where output contains enforcement-relevant content.
NON-OBSERVABLE	Model-internal computation state, provider-side caching behavior, and out-of-band provider configuration.
TRUST BOUNDARY	The Enforcement Gateway resides inline between the caller and the model endpoint. The REST/gRPC interface is the surface through which enforcement decisions are communicated.

3.2 Message Queue Enforcement

OBSERVABLE	Message payload, message metadata, queue identity, producer identity claims, and message timestamp.
ENFORCEABLE	Message consumption blocking, permit-conditional forwarding to inference endpoints, and dead-letter routing for denied messages.
NON-OBSERVABLE	Messages queued prior to enforcement gateway deployment, out-of-band producer channels, and queue infrastructure internal state.

TRUST BOUNDARY	Enforcement occurs at message consumption. Messages must not reach inference execution by pathways that bypass enforcement consumption.

3.3 Reverse Proxy and API Gateway Enforcement

OBSERVABLE	Full HTTP/S request and response envelope, TLS termination metadata, upstream routing decisions, and load balancer annotations.
ENFORCEABLE	Request blocking at proxy layer, response interception, upstream routing modification based on enforcement decision, and connection termination.
NON-OBSERVABLE	Encrypted sideband channels, provider-internal API responses not passing through proxy, and out-of-band WebSocket upgrades unless explicitly intercepted.
TRUST BOUNDARY	The proxy layer is the Enforcement Boundary. Direct routes to backend model endpoints that bypass the proxy constitute Bypass Vectors and are prohibited.

3.4 Sidecar Enforcement Model

In containerized deployment environments, QODIQA enforcement may be deployed as a sidecar component co-located with the application container. The sidecar intercepts all outbound inference traffic and inbound model responses. The network topology MUST be configured to prevent the application container from reaching inference endpoints by pathways that bypass the sidecar.

SIDECAR NETWORK ISOLATION REQUIREMENT	INV-INTEROP-034
Network policy configurations in sidecar deployments MUST restrict all outbound traffic from the application container to inference endpoints exclusively through the sidecar enforcement proxy. Direct container-to-provider routes SHALL NOT exist.	
INV-INTEROP-034 · Critical · Verified in Conformance Test Suite Section 8.2	

3.5 Network Segmentation Enforcement Model

In network segmentation deployments, QODIQA enforcement is instantiated as a network segment boundary through which all inference traffic must transit. This model is applicable to on-premises deployments, private cloud environments, and hybrid architectures where physical or virtual network control is available.

The network segmentation model provides the broadest observability surface and the strongest bypass prevention guarantees. It does not, however, reduce the requirement for application-layer enforcement validation. Network-layer controls are complementary to, not substitutes for, application-layer enforcement.

SECTION 4

Non-Override Guarantees

This section constitutes the formal declaration of non-override guarantees that apply to all QODIQA conformant deployments. These guarantees are structural. They cannot be disabled by configuration, overridden by operational decisions, or negotiated away through commercial arrangements. Their violation constitutes a non-conformance finding that invalidates certification status.

No external system component, regardless of its authority level, operational role, or technical capability, is permitted to disable, defer, weaken, or circumvent the deterministic enforcement decisions of the QODIQA Enforcement Gateway.

4.1 External System Constraints

NON-OVERRIDE INVARIANTS, EXTERNAL SYSTEMS	
SHALL NOT	External systems SHALL NOT disable enforcement logic through configuration flags, environment variables, feature toggles, or administrative API calls. Enforcement logic is not configurable for bypass.
SHALL NOT	Model providers SHALL NOT modify enforcement decisions. Enforcement decisions are produced by the Policy Engine against Consent Registry state. Model providers receive only request traffic that has been permitted. They do not receive enforcement metadata and cannot alter enforcement outcomes.
SHALL NOT	Orchestrators SHALL NOT inject post-approval mutations into inference requests. A request approved at the Enforcement Boundary must be delivered to the model provider in the form in which it was approved. Post-approval mutation constitutes an Override Attempt.
SHALL NOT	Runtime prompts SHALL NOT alter policy state. Prompt content is evaluated against policy; it does not modify policy. No prompt construction pattern, including structured injection formats, system-role manipulation, or meta-instruction sequences, constitutes a valid policy update pathway.
SHALL NOT	Configuration flags in application code, infrastructure templates, or deployment manifests SHALL NOT provide mechanisms through which deterministic controls can be bypassed. The absence of such flags is a conformance requirement, not a default.

4.2 Invariant Declarations

ENFORCEMENT DECISION IMMUTABILITY	INV-INTEROP-041

An enforcement decision, once produced by the Policy Engine, is immutable within the scope of the inference operation to which it applies.

No downstream component receives authority to reverse a deny decision. No upstream component receives authority to pre-authorize a request in a manner that bypasses Policy Engine evaluation. Enforcement decisions are the exclusive output of the QODIQA Enforcement Gateway operating against current Consent Registry state.

INV-INTEROP-041 · Absolute · Immutability property of enforcement decisions

CONSENT REGISTRY NON-WRITEABILITY FROM DATA PLANE — INV-INTEROP-042

The Consent Registry SHALL NOT accept write operations originating from Data Plane components, inference operations, model provider responses, or orchestration framework actions.

Consent state is modified exclusively through the Control Plane by authorized consent management operations. Inference operations read consent state; they do not modify it. Any Data Plane component that issues write operations to the Consent Registry is non-conformant and constitutes an Override Attempt.

INV-INTEROP-042 · Critical · Control Plane / Data Plane separation enforcement

FAIL-CLOSED NON-OVERRIDE — INV-INTEROP-043

The fail-closed default of the QODIQA Enforcement Gateway SHALL NOT be overridden by external system configuration, operational policy, or commercial arrangement.

Fail-closed means: when the Enforcement Gateway cannot produce a definitive permit decision (due to Consent Registry unavailability, Policy Engine error, identity verification failure, or any other condition that prevents complete enforcement lifecycle execution), the default outcome is deny. This default is not configurable. Deployments that require a fail-open default for operational continuity are non-conformant with QODIQA.

INV-INTEROP-043 · Non-negotiable · Fail-closed is structural, not configurable

SECTION 5

Multi-Model and Multi-Vendor Deployment Constraints

Multi-model and multi-vendor deployments introduce integration complexity that must be managed without weakening enforcement integrity. This section defines the constraints applicable to deployment architectures that involve multiple model providers, multiple enforcement gateway instances, or hybrid on-premises and cloud configurations.

5.1 Multi-Provider Routing Constraints

When QODIQA is deployed in an environment that routes inference requests to multiple model providers, each routing path must be covered by enforcement. The existence of multiple providers does not create provider-specific enforcement exemptions.

MULTI-PROVIDER COVERAGE REQUIREMENT — INV-INTEROP-051

Every model provider endpoint accessible within a conformant deployment SHALL be covered by Enforcement Gateway routing. No provider endpoint SHALL be reachable from application components by paths that do not transit enforcement.

Routing decisions that select between providers occur after enforcement, not before. The enforcement decision is made on the request as presented; routing to a specific provider is an implementation detail of request delivery that does not alter enforcement requirements.

INV-INTEROP-051 · Critical · Universal coverage across all provider endpoints

5.2 Hybrid On-Premises and Cloud Deployment Constraints

Hybrid deployments span on-premises infrastructure and cloud-hosted model providers. The trust boundary in hybrid deployments must be explicitly designed to ensure that the network path between on-premises components and cloud-hosted endpoints transits enforcement controls.

HYBRID DEPLOYMENT ENFORCEMENT CONTINUITY — INV-INTEROP-052

In hybrid deployments, enforcement continuity SHALL be maintained across the on-premises/cloud boundary. Network paths from on-premises application components to cloud-hosted model provider endpoints SHALL transit enforcement controls that are subject to the same invariants as fully on-premises or fully cloud-hosted deployments.

The physical or logical boundary between on-premises and cloud infrastructure does not constitute a trust boundary exemption. Enforcement applies to the inference operation regardless of where in the network the request originates or terminates.

INV-INTEROP-052 · Critical · Cross-boundary enforcement continuity

5.3 Fallback and Load Balancing Constraints

Fallback routing (routing requests to alternative providers when primary providers are unavailable) and load balancing (distributing requests across multiple provider instances) must not create enforcement gaps. Fallback targets and load balancing targets are subject to the same enforcement requirements as primary targets.

FALLBACK TARGET ENFORCEMENT REQUIREMENT — INV-INTEROP-053

Fallback model provider endpoints SHALL be subject to identical enforcement requirements as primary endpoints. A fallback routing event SHALL NOT bypass enforcement or reduce enforcement scope relative to the original request.

Fallback routing that routes to endpoints not covered by enforcement topology is a Bypass Vector. Operational continuity requirements do not justify fallback paths that circumvent enforcement.

INV-INTEROP-053 · Critical · Fallback paths must maintain enforcement coverage

5.4 Federated Inference Deployment Constraints

Federated inference deployments distribute inference computation across multiple nodes, potentially across organizational boundaries. These architectures present enforcement challenges because inference operations may be fragmented, results may be aggregated, and the organizational ownership of individual inference nodes may differ.

FEDERATED INFERENCE ENFORCEMENT BOUNDARY — INV-INTEROP-054

In federated inference deployments, enforcement SHALL be applied to the composite inference operation at the point of request initiation and result delivery. Enforcement of sub-operations within the federation is required where sub-operation results are independently usable by the requesting component.

Federated architectures that aggregate results from multiple inference nodes without per-sub-operation enforcement are conformant only where sub-operation results are not delivered to application components independently. Where sub-results are independently delivered, each constitutes an inference operation subject to independent enforcement.

INV-INTEROP-054 · Complex topology requirement · Requires architectural evaluation during certification

SECTION 6

Agent Framework Constraints

Agent frameworks introduce inference patterns that differ structurally from single-turn request-response interactions. This section defines the enforcement constraints applicable to agent framework integrations, including constraints on tool invocation, memory access, and multi-turn interaction management.

6.1 Agent Framework Integration Boundaries

Agent frameworks that generate inference requests autonomously or semi-autonomously must route those requests through enforcement with the same rigor applied to application-generated requests. The autonomous or programmatic origin of a request does not reduce the enforcement requirement.

AGENT-GENERATED REQUEST ENFORCEMENT — INV-INTEROP-061

Inference requests generated by agent framework components SHALL transit the QODIQA Enforcement Gateway with complete identity context attributable to both the agent framework instance and the originating principal on whose behalf the agent is operating.

Agent frameworks that submit requests under generic or framework-level identities without propagating the originating principal's identity context are non-conformant. Enforcement decisions require identity resolution to the human or organizational principal that authorized the agent's operation.

INV-INTEROP-061 · Critical · Identity propagation requirement for agent operations

6.2 Tool Invocation Layer Constraints

Agent frameworks commonly use tool invocation (also referred to as function calling) to interact with external systems. Where tool invocations trigger inference operations (for example, tools that call AI sub-agents or that use AI-powered APIs), those operations are subject to enforcement.

TOOL-INITIATED INFERENCE ENFORCEMENT	INV-INTEROP-062

Inference operations initiated through tool invocation mechanisms SHALL be subject to enforcement as independent inference operations. Tool invocation does not constitute implicit authorization. The consent and identity context of the parent inference operation does not extend to tool-initiated sub-operations without explicit, independently validated consent.

INV-INTEROP-062 · Critical · No implicit consent extension through tool invocation

6.3 External Plugin Systems

Agent frameworks that support external plugin architectures (plugins that extend agent capability through third-party code execution) must ensure that plugin-initiated inference operations transit enforcement. The third-party origin of plugin code does not reduce enforcement requirements; it increases them.

PLUGIN-INITIATED OPERATION ENFORCEMENT	INV-INTEROP-063

All inference operations initiated by third-party plugin code executing within an agent framework SHALL transit enforcement with identity context that includes the plugin's identity in addition to the agent framework and originating principal identities.

Plugin execution environments that permit inference calls without enforcement gateway routing are Bypass Vectors. Plugin sandboxing that does not include network-level enforcement routing constraints is insufficient for QODIQA conformance.

INV-INTEROP-063 · Critical · Plugin identity must be included in enforcement context

SECTION 7

Data Flow Constraints

Data flow constraints govern the movement of inference-related data through QODIQA-enforced architectures. These constraints address input validation, context window management, streaming response handling, and the separation of data and control plane traffic.

7.1 Input Validation Constraints

The Enforcement Gateway validates the structural and semantic properties of inference requests as part of the enforcement lifecycle. Input validation is not a pre-enforcement filter; it is part of enforcement itself. Requests that do not pass validation produce enforcement failure states, not pre-enforcement rejections.

INPUT VALIDATION AS ENFORCEMENT	INV-INTEROP-071
Input validation failures SHALL produce enforcement failure states (SF) with audit log entries equivalent in completeness to permit or deny decisions. Validation failures are enforcement events, not pre-enforcement events.	
INV-INTEROP-071 · Audit requirement · Validation failures are enforcement events	

7.2 Context Window Injection Constraints

Context window injection (the addition of system prompts, retrieved documents, memory contents, or other contextual material to inference requests) must be governed by enforcement. Material injected into the context window is part of the inference request and is subject to the same consent and policy evaluation as user-provided prompt content.

CONTEXT INJECTION ENFORCEMENT — INV-INTEROP-072

Context window content injected by orchestration frameworks, agent runtimes, retrieval systems, or memory components SHALL be subject to enforcement evaluation as part of the complete inference request. Injected content that was not present at consent grant time requires explicit consent scope coverage or triggers a deny decision.

System prompt injection that alters the effective scope of the inference operation beyond what consented principals authorized constitutes an Override Attempt. Retrieval-augmented generation systems that inject retrieved content must operate within consent scopes that explicitly authorize the data sources being retrieved.

INV-INTEROP-072 · Critical · Context injection is part of the enforcement surface

7.3 Streaming Token Constraints

Streaming inference responses (where model output is delivered as a token stream rather than a complete response) present enforcement challenges because output content is not fully known at response initiation. QODIQA addresses streaming through response-path enforcement that monitors token streams for consent-relevant content.

STREAMING RESPONSE ENFORCEMENT — INV-INTEROP-073

Streaming response delivery SHALL be interceptable by the Enforcement Gateway at any point in the token stream. Enforcement conditions that arise mid-stream (consent scope violations in response content, prohibited output patterns) SHALL be enforceable through stream termination.

Streaming architectures that deliver token streams directly from model providers to application components without Enforcement Gateway interception are Bypass Vectors on the response path. The enforcement requirement applies to streaming responses with equal force to complete responses.

INV-INTEROP-073 · Critical · Streaming does not exempt from response-path enforcement

7.4 Data and Control Plane Separation

The separation of Data Plane traffic (inference requests and responses) from Control Plane traffic (policy management, consent administration, enforcement configuration) is a structural requirement of QODIQA architecture. Commingling of Data Plane and Control Plane traffic creates conditions under which Data Plane operations could influence enforcement configuration.

DATA/CONTROL PLANE SEPARATION	INV-INTEROP-074

Data Plane and Control Plane traffic SHALL be carried on logically or physically separate channels. Data Plane components SHALL NOT have write access to Control Plane interfaces. Control Plane interfaces SHALL NOT be reachable through pathways that also carry Data Plane traffic without explicit, audited access controls.

INV-INTEROP-074 · Structural · Plane separation is an architectural requirement

7.5 Enforcement Logging Integrity

The enforcement audit log is a structural component of the QODIQA architecture. Its integrity must be maintained against both external tampering and internal compromise. Log integrity is not a post-enforcement concern; it is an enforcement property.

AUDIT LOG INTEGRITY REQUIREMENT	INV-INTEROP-075

Enforcement audit log entries SHALL be cryptographically sealed at the time of creation. No component that participates in Data Plane operations SHALL have write access to enforcement audit logs. Audit log access control is a Control Plane function.

Enforcement audit logs that are accessible for modification by Data Plane components constitute a structural integrity failure. Deployments in which application components or inference frameworks have write access to audit infrastructure are non-conformant.

INV-INTEROP-075 · Critical · Log integrity is an enforcement property, not an audit property

SECTION 8

Cryptographic and Identity Constraints

Cryptographic and identity constraints govern the authentication, authorization, and integrity verification mechanisms required for QODIQA conformant deployments. These constraints are not implementation recommendations; they are structural requirements derived from the deterministic enforcement properties that QODIQA guarantees.

8.1 Key Exchange and mTLS Requirements

All connections between components within the QODIQA enforcement boundary must use mutual TLS (mTLS) for authentication and channel integrity. mTLS is the minimum transport security requirement; additional cryptographic controls may be required by the Security and Cryptographic Profile for specific deployment contexts.

MTLS REQUIREMENT	INV-INTEROP-081
All inter-component connections within the QODIQA enforcement boundary MUST use mTLS with certificates issued by a QODIQA-recognized certificate authority. Unauthenticated or one-way TLS connections between enforcement boundary components are non-conformant.	
INV-INTEROP-081 · MUST · mTLS is mandatory for all enforcement boundary connections	

8.2 Signature Verification Requirements

Enforcement decisions, permit tokens, and audit log entries are cryptographically signed by the Enforcement Gateway. Components that consume these artifacts must verify signatures before acting on them. Signature verification failure is an enforcement failure condition.

SIGNATURE VERIFICATION REQUIREMENT	INV-INTEROP-082
Components that receive enforcement decisions, permit tokens, or audit log entries MUST verify cryptographic signatures before acting on those artifacts. Acting on an unverified or invalidly signed enforcement artifact is a non-conformance condition equivalent to bypass.	
INV-INTEROP-082 · MUST · Signature verification is mandatory before artifact consumption	

8.3 Identity Propagation Requirements

Identity context must be propagated through all layers of the integration architecture. Components that strip, transform, or substitute identity context without cryptographic authorization are non-conformant.

IDENTITY CONTEXT PROPAGATION	INV-INTEROP-083
Identity context SHALL be propagated without modification through all integration layers. Components that modify identity claims without cryptographic re-issuance from an authorized identity provider are producing non-authoritative identity context that SHALL NOT be accepted by the Enforcement Gateway.	
INV-INTEROP-083 · Critical · Identity context must not be modified in transit	

8.4 Token Integrity Invariants

TOKEN INTEGRITY REQUIREMENTS	
MUST	Consent tokens MUST contain cryptographically bound scope, expiry, and principal binding fields. Tokens without these fields SHALL NOT be accepted by the Enforcement Gateway.
MUST	Permit tokens issued by the Enforcement Gateway MUST be bound to the specific request payload they authorize. Permit tokens that are not payload-bound SHALL NOT be accepted as authorization for delivery.

SHALL NOT	Expired consent tokens SHALL NOT be accepted. Token expiry evaluation MUST use a monotonic clock source that is not configurable by Data Plane components.
SHALL NOT	Consent tokens SHALL NOT be reused across inference operations. Each inference operation requires an independent, current consent evaluation against the Consent Registry.

SECTION 9

Failure Mode Handling and Recovery Constraints

Failure mode handling in QODIQA conformant deployments is governed by the fail-closed principle: when enforcement cannot be completed, the default outcome is deny. This section specifies the failure categories, their required handling, and the constraints applicable to recovery procedures.

9.1 Failure Mode Specifications

FAILURE MODE CLASSIFICATION AND REQUIRED HANDLING

FAILURE MODE	CATEGORY	REQUIRED HANDLING	RECOVERY CONSTRAINT
Consent Registry Unavailable	Infrastructure	Deny all inference operations. Log unavailability event with timestamp.	Recovery requires Registry availability verification before resuming enforcement. No cached consent state may be used.
Policy Engine Error	Computation	Deny the affected operation. Log error with full request context (excluding payload content).	Policy Engine must complete self-diagnostic evaluation before processing subsequent requests.
Identity Verification Failure	Authentication	Deny the operation. Do not expose identity verification failure details to requesting component.	No recovery path within the current operation lifecycle. New lifecycle must begin at S0.
Permit Token Integrity Failure	Cryptographic	Deny delivery. Log integrity failure as critical security event. Alert enforcement operations.	Integrity failures trigger audit review before enforcement resumes for the affected integration pathway.

FAILURE MODE	CATEGORY	REQUIRED HANDLING	RECOVERY CONSTRAINT
Audit Log Write Failure	Audit	Deny the operation that cannot be logged. Enforcement without auditability is non-conformant.	Log storage must be restored and verified before enforcement operations resume.
Network Path Disruption	Infrastructure	Deny inference operations on affected paths. Do not route to alternative paths that bypass enforcement.	Path restoration must include enforcement topology verification before traffic resumes.

FAILURE RECOVERY ENFORCEMENT REQUIREMENT INV-INTEROP-091

Recovery from any failure mode SHALL include enforcement topology verification before inference operations resume. Recovery procedures that restore inference capability without verifying enforcement coverage are non-conformant, regardless of operational pressure to restore service.

INV-INTEROP-091 · Critical · Recovery must include enforcement verification

SECTION 10

Anti-Bypass and Adversarial Integration Controls

This section enumerates the bypass attempt categories that have been identified in the analysis of adversarial integration patterns and specifies the structural mitigations required to address each. Mitigations are structural. They are architectural properties that make bypass attempts ineffective, not detection mechanisms that respond after bypass has occurred.

10.1 Direct Model Access Outside Gateway

BYPASS VECTOR	An application component discovers or is provided with a direct endpoint to a model provider that does not route through the Enforcement Gateway. Inference requests are submitted directly, bypassing enforcement entirely.

STRUCTURAL MITIGATION	Model provider endpoints SHALL NOT be accessible from application network segments except through the Enforcement Gateway. Network policy, firewall rules, and service mesh configuration MUST enforce this constraint. Periodic network topology audits are required during certification maintenance.

10.2 Shadow API Routes

BYPASS VECTOR	A model provider or orchestration platform exposes undocumented or legacy API endpoints that provide equivalent inference functionality. These endpoints are not covered by enforcement gateway routing rules.
STRUCTURAL MITIGATION	Conformant deployments SHALL enumerate all accessible model provider endpoints during certification audits. Unaccounted-for endpoints result in a non-conformance finding. Enforcement topology documentation is a required certification artifact. Any new endpoint exposed by a model provider requires re-certification of the integration topology.

10.3 Side-Channel Token Injection

BYPASS VECTOR	An attacker or compromised component injects identity or consent tokens into request headers, message metadata, or context window content in ways that cause the Policy Engine to evaluate the request under an elevated or different permission context.
STRUCTURAL MITIGATION	The Enforcement Gateway SHALL derive identity and consent context exclusively from verified, mTLS-authenticated sources. Header or payload content purporting to represent identity or consent grants SHALL be stripped and disregarded unless it is the authoritative token channel. All token validation MUST include issuer verification against a registered, non-malleable authority set.

10.4 Prompt Mutation After Approval

BYPASS VECTOR	A component between the Enforcement Gateway and the model provider modifies the prompt content after the enforcement permit decision has been made. The modified prompt may contain instructions or content that would not have received a permit decision in its modified form.
STRUCTURAL MITIGATION	The Enforcement Gateway SHALL cryptographically sign the permitted prompt payload at the moment of permit decision. The model provider interface adapter SHALL verify this signature prior to model submission. Any deviation between the signed payload and the submitted payload SHALL cause the request to be rejected and the event logged as a critical integrity violation.

10.5 Dynamic Configuration Toggling

BYPASS VECTOR	An attacker or compromised administrative interface modifies enforcement configuration in real time to relax enforcement constraints during a targeted inference operation, then restores the original configuration to evade detection.
STRUCTURAL MITIGATION	All enforcement configuration changes are Control Plane events and MUST be logged with cryptographic timestamps and administrator identity. Configuration changes that reduce enforcement strictness require multi-party authorization. The enforcement configuration effective at the time of each enforcement decision is recorded in the audit log. Configuration rollback to evade audit review is detectable through log integrity verification.

10.6 Interoperability Threat Classification

This subsection defines a formal taxonomy of threat classes applicable to QODIQA interoperability architectures. The taxonomy is structured to support systematic risk identification during integration design, certification evaluation, and post-deployment audit. It does not replace the bypass vector catalog defined in Sections 10.1 through 10.5; it provides the analytical classification layer within which those vectors and others are situated.

The classification draws on structural threat modeling disciplines consistent with ISO/IEC 27005 and NIST SP 800-30. Each class is defined by its origin characteristics, manifestation pattern, and required architectural mitigation posture. Classification informs the depth of assessment required. It does not alter the enforcement obligation applicable to any class.

THREAT CLASSIFICATION INVARIANT	INV-INTEROP-106
Threat classification does not reduce enforcement obligation. All threat classes, regardless of their origin, probability, or severity assessment, default to fail-closed behavioral requirements.	

A threat assessed as low-probability does not receive relaxed mitigation requirements. A threat attributed to accidental misconfiguration does not receive less rigorous architectural response than one attributed to coordinated adversarial action. Classification is an analytical instrument; it is not a risk-acceptance pathway.

INV-INTEROP-106 · Absolute · Governs all Class I through Class V threat responses

10.6.1 Class I: Accidental Misconfiguration

DEFINITION	Enforcement gaps that arise from unintentional errors in deployment configuration, network topology setup, or integration surface specification. The misconfiguration is not directed; no actor intends to weaken enforcement. The enforcement gap is a consequence of operational error.
THREAT ORIGIN	Operations teams, platform engineers, or automated deployment tooling executing integration setup without complete awareness of enforcement topology requirements. Commonly associated with initial deployment, migration events, and infrastructure updates.
TYPICAL MANIFESTATION	Incomplete routing rules that expose direct model provider endpoints. Sidecar enforcement proxies not applied to newly provisioned application containers. API gateway enforcement plugins not re-enabled after version upgrades. Consent Registry connection strings pointing to non-production instances in production environments.
MITIGATION POSTURE	Enforcement topology validation at deployment time and as a scheduled operational control. Integration topology documentation maintained current. Automated conformance checks that detect direct routes to model endpoints. Change management procedures that require enforcement impact assessment for all infrastructure modifications.
CERTIFICATION IMPLICATION	Class I occurrences discovered during certification assessment are classified as Major non-conformance findings. They indicate insufficient operational controls and require both remediation and demonstration of preventive control implementation before certification can proceed.

10.6.2 Class II: Architectural Drift

DEFINITION	Gradual divergence between the certified integration architecture and the deployed integration architecture, occurring through incremental changes that individually appear minor but cumulatively produce material enforcement gaps. Drift is a systemic condition, not a discrete event.

THREAT ORIGIN	Accumulated effect of undocumented integration changes, provider API version migrations, component upgrades that alter routing behavior, and organic growth of integration surface area without corresponding enforcement coverage review.
TYPICAL MANIFESTATION	Integration topology documentation that no longer accurately reflects deployed architecture. Model provider endpoints added during capacity scaling events not covered by enforcement routing rules. Agent framework version upgrades that introduce new tool invocation pathways not assessed against enforcement requirements. Enforcement gateway version mismatches across distributed deployment nodes.
MITIGATION POSTURE	Periodic integration topology re-validation against deployed architecture. Mandatory enforcement impact assessment for all integration component upgrades. Continuous monitoring of outbound traffic patterns to detect routes not transiting the Enforcement Gateway. Scheduled re-certification assessments at intervals defined in the Certification Framework.
CERTIFICATION IMPLICATION	Class II conditions discovered at re-certification are evaluated for severity based on the magnitude of drift from the last certified baseline. Drift affecting enforcement coverage of active integration surfaces constitutes a Major or Critical finding depending on the enforcement gap created.

10.6.3 Class III: Privileged Component Abuse

DEFINITION	Exploitation of legitimate, authorized access to system components (orchestrators, agent runtimes, administrative interfaces, or Control Plane endpoints) to produce enforcement outcomes that the authorization framework was not designed to permit. The threat actor possesses real credentials for real components; the abuse is in how those components are operated.
THREAT ORIGIN	Compromised internal components, malicious insiders with operational access, or legitimate components whose behavior has been modified by supply chain interference. Distinguished from Class IV by the use of genuinely authorized access rather than synthesized or stolen credentials.
TYPICAL MANIFESTATION	An authorized orchestrator submitting consent modification requests to the Control Plane outside of legitimate consent management workflows. An authorized administrative account toggling enforcement configuration during a targeted inference window. An agent runtime accumulating permissions across tool invocation chains by presenting prior permit decisions as authorization for subsequent calls.
MITIGATION POSTURE	Multi-party authorization for Control Plane modifications. Behavioral anomaly monitoring on privileged component activity. Per-operation permission scoping that prevents permit accumulation. Separation of enforcement administration from inference operations administration. Cryptographic audit trails that attribute each enforcement-relevant action to a specific authorized identity.
CERTIFICATION IMPLICATION	Certification assessment evaluates whether authorization structures and operational controls are sufficient to detect and prevent Class III abuse patterns. Insufficient separation of duties in enforcement administration is a Major finding. Evidence of actual Class III exploitation during the certification window is a Critical finding requiring immediate remediation and incident investigation.

10.6.4 Class IV: Coordinated Bypass Attempt

DEFINITION	A deliberate, multi-vector attempt to circumvent enforcement by simultaneously targeting multiple components or control points in the integration architecture. Coordination implies that individual attack vectors are insufficient in isolation; the bypass requires their simultaneous or sequenced execution.
THREAT ORIGIN	Sophisticated external adversaries with architectural knowledge of the deployment, coordinated insider threat scenarios, or supply chain compromise affecting multiple components simultaneously.
TYPICAL MANIFESTATION	Simultaneous compromise of identity provider and Consent Registry to produce false consent context while suppressing audit log entries. Coordinated configuration modification across multiple enforcement nodes to create a brief enforcement gap. Multi-stage prompt injection designed to accumulate permissions across a sequence of individually permitted operations.
MITIGATION POSTURE	Defense-in-depth architectural design that requires simultaneous compromise of cryptographically independent components to produce a bypass. Anomaly detection across enforcement event streams from multiple components. Out-of-band audit log integrity verification that is not subject to the same compromise vectors as the primary enforcement infrastructure.
CERTIFICATION IMPLICATION	Certification assessment evaluates whether the deployment architecture requires coordinated compromise of multiple independent components to produce a bypass. Single-point-of-failure bypass paths are Critical findings regardless of their assessed probability.

10.6.5 Class V: Supply Chain Interference

DEFINITION	Compromise of enforcement properties through modification of components prior to their deployment: tampered enforcement gateway binaries, modified policy evaluation libraries, substituted certificate authority roots, or corrupted model provider interface adapters.
THREAT ORIGIN	Adversarial modification of software supply chain artifacts, including build pipeline compromise, dependency poisoning, container image tampering, and distribution channel interference.
TYPICAL MANIFESTATION	Enforcement gateway binary that behaves correctly under audit conditions but contains conditional bypass logic activated by specific trigger conditions. Policy evaluation library that produces systematically biased results for specific request patterns. Interface adapter that silently strips payload integrity signatures before model submission.
MITIGATION POSTURE	Cryptographic verification of all enforcement component integrity at deployment time using checksums registered with the QODIQA specification corpus. Reproducible build verification for enforcement gateway components. Runtime integrity attestation mechanisms that detect binary modification after deployment. Supply chain audit requirements imposed on enforcement component vendors.

CERTIFICATION IMPLICATION	Certification requires evidence of supply chain integrity controls for all enforcement boundary components. Absence of deployment-time integrity verification is a Major finding. Absence of supply chain audit requirements for enforcement component vendors is a Minor finding requiring remediation within the certification maintenance period.

SECTION 11

Certification Implications

This section specifies the certification implications of the interoperability constraints defined in this document. Certification assessment evaluates whether a deployment architecture satisfies all applicable invariants. Findings are classified by severity and drive remediation requirements.

11.0 Integration Assurance Level Framework

The QODIQA Certification Framework defines Integration Assurance Levels (IAL) that characterize the rigor of enforcement coverage across the integration architecture. IAL classification determines the depth of certification assessment required and the audit frequency applicable to a certified deployment.

INTEGRATION ASSURANCE LEVEL DEFINITIONS

IAL	DESCRIPTION	ASSESSMENT REQUIREMENTS	APPLICABLE DEPLOYMENT PROFILE
IAL-1	Basic enforcement coverage across a single-provider, single-layer integration topology with no agent framework involvement.	Documentation review, enforcement topology verification, basic bypass vector testing.	Simple API integrations with a single model provider and no orchestration complexity.
IAL-2	Extended enforcement coverage across multi-provider or single-layer agent framework integrations.	IAL-1 requirements plus multi-provider routing verification, agent identity propagation testing, and tool invocation enforcement testing.	Multi-provider routing, basic agent framework integrations, and hybrid deployment topologies.
IAL-3	Comprehensive enforcement coverage across complex multi-layer, multi-provider, multi-agent architectures with federated inference or cross-organizational boundaries.	IAL-2 requirements plus federated enforcement verification, cross-organizational trust boundary assessment, supply chain integrity evaluation, and adversarial bypass testing.	Enterprise AI platform deployments, federated inference architectures, and cross-organizational AI service integrations.

11.2 Non-Conformance Finding Classifications

NON-CONFORMANCE FINDING SEVERITY CLASSIFICATIONS

CLASSIFICATION	DEFINITION	CERTIFICATION IMPACT	REMEDIATION REQUIREMENT
Critical	A Bypass Vector exists in the deployment architecture. Enforcement can be circumvented without coordinated multi-component compromise.	Certification denied or revoked immediately.	Architectural remediation required. Full re-assessment before certification can proceed.
Major	An invariant defined in this document or the QODIQA Core Standard is not satisfied. Bypass is not currently possible but enforcement integrity is materially weakened.	Certification denied pending remediation.	Remediation required within 30 days. Targeted re-assessment of affected areas.
Minor	A conformance requirement is partially satisfied. Enforcement coverage is present but documentation, operational controls, or monitoring are deficient.	Certification conditional on remediation plan submission.	Remediation plan required within 15 days. Remediation within 90 days.
Observation	A practice or configuration is inconsistent with recommended implementation guidance but does not constitute a violation of a normative requirement.	No certification impact.	No mandatory remediation. Addressed at deployer's discretion.

11.3 Enforcement Topology Validation Requirements

Certification assessment includes mandatory enforcement topology validation: a systematic verification that the deployed integration architecture matches the documented architecture and that all inference traffic pathways transit the Enforcement Gateway.

TOPOLOGY VALIDATION SCOPE INV-INTEROP-113

Enforcement topology validation SHALL include: enumeration of all model provider endpoints accessible from the deployment environment, verification that no application network segment has direct routing to model provider endpoints, verification that all Enforcement Gateway instances are operating with current policy state, and verification that audit log integrity controls are active on all enforcement nodes.

INV-INTEROP-113 · Certification requirement · Topology validation scope definition

SECTION 12

Non-Coverage and Architectural Limitations

This section formally declares what QODIQA does not cover. These declarations are not admissions of weakness; they are precise scope boundaries that enable deployers to correctly identify what additional systems or controls are required in conjunction with QODIQA.

EXPLICIT NON-COVERAGE DECLARATIONS	
EXCLUDED	**Model correctness.** QODIQA does not guarantee that permitted inference operations produce factually accurate, logically consistent, or task-appropriate outputs. Enforcement of a permit decision is not an endorsement of output quality.
EXCLUDED	**Model ethics evaluation.** QODIQA does not evaluate whether model outputs are ethical, fair, unbiased, or aligned with any particular value framework. Consent enforcement and ethical evaluation are distinct functions. QODIQA performs the former; it is not designed to perform the latter.
EXCLUDED	**Training data integrity.** QODIQA does not enforce or verify any property of the data on which model providers have trained their models. Training data governance is outside the scope of runtime consent enforcement.
EXCLUDED	**Runtime safety classification.** QODIQA does not classify inference outputs as safe or unsafe. It enforces whether an inference operation was consented to. Whether consented operations produce safe outputs is a property of the model, the consent design, and the application, not of the enforcement infrastructure.
EXCLUDED	**Model internal modification.** QODIQA does not modify model weights, modify model inference procedures, inject system prompts into model internals, or alter the computational behavior of model providers. It operates at the boundary of model execution, not within it.
EXCLUDED	**Organizational compliance assurance.** QODIQA provides technical enforcement infrastructure that supports regulatory compliance programs. It does not, by its deployment, guarantee compliance with GDPR, the EU AI Act, NIST AI RMF, or any other regulatory framework. Compliance requires organizational, legal, and governance dimensions that extend beyond technical enforcement.
EXCLUDED	**Content moderation.** QODIQA does not perform semantic content moderation on inference outputs. Whether output content is appropriate, harmful, or objectionable is not a question QODIQA's enforcement infrastructure is designed to answer. Content moderation is a separate system function.

12.1 Scope Restatement

QODIQA strictly enforces **deterministic consent validation at runtime boundaries**. Every guarantee, every invariant, every certification requirement, and every conformance criterion in the QODIQA corpus is directed toward this single, precise function. Systems that require any of the excluded functions must source them from components designed for those purposes and integrate those components separately from QODIQA enforcement infrastructure.

DECLARATION

Architectural Declaration

This document has defined the formal interoperability and integration constraints governing the QODIQA deterministic consent enforcement system. The constraints herein are not aspirational standards or recommended practices. They are structural invariants, architectural properties that must be present for a deployment to be conformant and for certification to remain valid.

> Interoperability is permitted within the bounds defined by these constraints. Outside those bounds, there is no interoperability. There is only non-conformance.

The integration complexity of modern AI system architectures (multi-model routing, federated inference, autonomous agent frameworks, hybrid cloud topologies) does not reduce the obligation to maintain deterministic enforcement at every boundary. Complexity is not justification for relaxation. Where complexity makes enforcement harder, the architectural response is to simplify the topology or invest in enforcement infrastructure capable of handling that complexity. The response is never to weaken enforcement.

> **The Trust Boundary is not a parameter to be adjusted per deployment.** It is a structural property of the QODIQA architecture. Every integration decision that treats the Trust Boundary as negotiable is a decision to produce a non-conformant system.

These constraints are aligned with the totality of the QODIQA standard corpus. They do not introduce new requirements beyond what the corpus already establishes; they specify, in integration-specific terms, what those requirements mean for the full diversity of deployment architectures that QODIQA is designed to support.

The non-override guarantees declared in Section 4 are the architectural expression of the consent-as-infrastructure principle that motivates the QODIQA standard. Enforcement is not a feature that can be selectively enabled or disabled based on integration convenience. It is the infrastructure on which all compliant use of AI systems within the QODIQA framework depends.

DOCUMENT STATUS

Document Status and Classification

This document is the Interoperability and Deployment Constraints specification of the QODIQA standard corpus. It constitutes the formal declaration of architectural boundaries, integration guardrails, non-override guarantees, and fail-closed requirements governing QODIQA deployments across heterogeneous AI system environments. It is issued as a Normative Constraints Note and is binding on all conformant implementations.

The constraints defined herein are structural invariants. They are not implementation recommendations, configuration guidelines, or vendor integration advisories. Their violation constitutes a non-conformance finding under the Certification Framework. No integration architecture is permitted to weaken, defer, or circumvent any invariant defined in this document.

This document should be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0

- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Threat Model and Abuse Case Specification, Version 1.0
- QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dușescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-INTEROP-CONS-2026-001
TITLE	Interoperability and Deployment Constraints
SUBTITLE	Deterministic Interoperability Boundaries, Integration Guardrails, and Non-Override Guarantees
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Normative Constraints Note
DOCUMENT STATUS	Normative: Constraints Specification
GOVERNING AUTHORITY	QODIQA Governance Charter
SCOPE	Interoperability constraints, integration guardrails, non-override guarantees, anti-bypass controls, failure mode specifications, and certification implications for QODIQA deployments across heterogeneous AI system architectures.
AMENDMENT PROCESS	Governed by the QODIQA Governance Charter for the QODIQA Standard Corpus. No amendment may relax any non-override guarantee or weaken any fail-closed requirement without Core Invariant Protection Rule evaluation.
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 19 PART III · OPERATIONAL DOCUMENTS

Terminology and Normative Definitions

The canonical set of terms and normative definitions governing the framework for deterministic runtime consent enforcement.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-TND-2026-001	1.0	2026	Normative Specification	Public Release	NORMATIVE

SCOPE

This specification defines the canonical set of terms and normative definitions governing the QODIQA framework for deterministic runtime consent enforcement. All terms defined herein are normative. They override informal, colloquial, or derivative usages wherever such usages conflict with this specification. Any interpretive ambiguity arising within the QODIQA corpus across all constituent documents **SHALL defer to this specification as the authoritative semantic reference.**

This document introduces no new conceptual scope beyond that already established in the QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0, the QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0, the QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0, the QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0, the QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0, and the QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0. Its

sole function is to bound, fix, and protect the meaning of terms already in use across those documents. **Undefined terms carry no normative standing** within the QODIQA framework.

SECTION 0

Scope of Terminology Control

This specification serves as the canonical semantic authority for the QODIQA framework. It defines the meaning, boundaries, and binding status of all terms used across QODIQA normative documents. No corpus document may introduce, redefine, or substitute a term defined herein without a versioned amendment to this specification processed in accordance with Section 8.

The scope of this specification is limited to terms directly operative within the domain of deterministic runtime consent enforcement. Terms belonging to general computing, regulatory law, or adjacent technical fields are adopted by reference only where their meaning is unambiguous and their scope does not conflict with definitions stated herein.

> Terms defined in this specification **override informal usage** in all QODIQA documents, communications, and derivative materials. Where a term appears in a QODIQA document without a normative definition in this specification, that term carries no binding meaning and SHALL NOT be interpreted as introducing a normative requirement.

0.1 Canonical Authority

This document constitutes the sole lexical control instrument for the QODIQA framework. Derivative documents including implementation playbooks, use-case dossiers, and certification frameworks are bound by the definitions contained

herein and MAY NOT introduce alternative definitions for the same terms.

0.2 Undefined Terms

A term not defined in this specification has no normative standing within the QODIQA framework. Implementers and certifying bodies SHALL NOT attribute normative meaning to undefined terms. When operational necessity requires a term not yet defined, a formal amendment request SHALL be submitted through the change control process defined in Section 8.

0.3 Precedence

In the event of conflict between a definition in this specification and any language in a derivative QODIQA document, this specification takes precedence. Derivative documents SHALL be updated to align with this specification upon any versioned amendment. Conflict resolution is not discretionary; it is mandatory.

SECTION 1

Normative Language Conventions

The following terms, when appearing in any QODIQA normative document in uppercase form, SHALL be interpreted strictly as defined below. These conventions follow the model established in RFC 2119 but are restated here with binding force specific to the QODIQA framework. The absence of a normative keyword in a statement does not imply permissiveness; it implies that the statement is informative and carries no binding requirement.

TABLE 1: NORMATIVE KEYWORD DEFINITIONS

KEYWORD	BINDING STATUS	INTERPRETATION
SHALL	**Mandatory**	The requirement is unconditionally binding. Non-conformance constitutes a violation.

KEYWORD	BINDING STATUS	INTERPRETATION
MUST	**Mandatory**	Equivalent to SHALL. Both forms carry identical binding force and are interchangeable within this corpus.
SHALL NOT	**Mandatory Prohibition**	The action or condition is unconditionally prohibited. Any occurrence constitutes a violation.
MUST NOT	**Mandatory Prohibition**	Equivalent to SHALL NOT. Identical binding force.
PROHIBITED	**Absolute Prohibition**	Stronger than SHALL NOT. Used where a violation would constitute a fundamental non-conformance that invalidates certification status or audit integrity. Context-independence is assumed.
MAY	*Permissive*	The action or condition is permitted but not required. Implementers are free to include or omit the described behavior without affecting conformance status.
SHOULD	*Recommended*	The action or condition is recommended where technically feasible. Deviation from a SHOULD-qualified requirement does not constitute non-conformance but SHALL be documented in conformance disclosures.

> Informative content within normative documents is explicitly marked with the prefix **"Informative Note:"** and does not constitute a requirement. Absence of this prefix implies normative status for all definitional and requirement statements.

1.1 Semantic Drift Prevention

Normative keywords SHALL NOT be used in lowercase form within normative documents when the intent is to invoke binding requirements. Use of lowercase "shall," "must," or "must not" in normative sections is treated as informal language and carries no binding force. Document authors are responsible for consistent application of uppercase normative keywords.

TAXONOMY

Taxonomy of Terms

The terms defined in this specification are classified into six functional categories. This taxonomy is provided to support audit readability, cross-document navigation, and certification scoping. Classification does not alter the normative status or definition of any term; it is a structural aid only.

Each term appears in exactly one category. Categories are defined by the functional role the term plays within the QODIQA enforcement architecture. Terms that span multiple functional roles are classified by their primary role as operative within enforcement evaluation. This classification is exhaustive for all terms defined within this specification.

The classification labels used across definition blocks (Foundational, State, Process, Artifact, Architectural, Control) constitute the canonical term type system for this specification.

FOUNDATIONAL TERMS

- Deterministic Runtime Consent Enforcement DRCE
- Deterministic System DS
- Consent C
- Runtime RT

STATE DEFINITIONS

- Fail-Closed State FCS
- Non-Execution State NES
- Revocation State RS

- Integrity Failure Condition IFC

PROCESS DEFINITIONS

- Consent Validation CV
- Enforcement Action ENF
- Integrity Verification IV
- Tamper Detection TD
- Artifact Signing AS

ARTIFACT DEFINITIONS

- Consent Artifact CA
- Audit Artifact AA
- Immutable Record IR
- Append-Only Record AOR
- Cryptographic Integrity Anchor CIA

ARCHITECTURAL COMPONENTS

- Enforcement Gateway EG
- Policy Engine PE
- Consent Registry CR
- Enforcement Layer EL
- Policy Evaluation Module PEM

- Registry Interface RI
- Conformance Validation Layer CVL
- Immutable Log IL

CONTROL CONSTRUCTS

- Decision Boundary DB
- Inference Boundary IB
- Blocking Event BE
- Execution Authorization EA
- Control Surface CS
- Trust Boundary TB
- Deterministic Evaluation Path DEP
- Key Management Boundary KMB
- Runtime Invocation RVI
- External Dependency XD
- Certification Level CL

SECTION 2

Core Conceptual Definitions: Foundational Layer

The following definitions constitute the foundational semantic layer of the QODIQA framework. All subsequent architectural, security, and governance definitions in this specification depend on the meanings established here. Each

definition is single-scope and technically anchored.

Deterministic Runtime Consent Enforcement DRCE

FOUNDATIONAL

NORMATIVE DEFINITION

A system property whereby every AI execution request is evaluated against explicitly declared consent state and applicable policy rules at the moment of invocation, producing a bounded output (allow, deny, or allow with stated restrictions) that is reproducible given identical inputs. The evaluation process introduces no probabilistic inference, no contextual assumption, and no default-to-permit behavior.

EXPLICIT NON-INCLUSIONS

This definition does not include: probabilistic inference of enforcement outcomes; contextual assumption of consent from behavioral or usage signals; default-to-permit behavior in the absence of an explicit consent record; model-derived or AI-generated authorization decisions; and post-hoc enforcement applied after execution has occurred.

(Reference: QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0)

Deterministic System DS

FOUNDATIONAL

NORMATIVE DEFINITION

A system in which, given a fixed set of inputs and a fixed rule set at a fixed version, the evaluation process always yields the same output with no variance. Determinism within the QODIQA framework applies specifically to the enforcement evaluation path and does not constrain the behavior of AI models or downstream services operating outside that path.

(Reference: QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0)

Runtime RT

STATE

NORMATIVE DEFINITION

The temporal and computational interval during which an AI execution request is processed, from the point of request submission to the point at which an authorized or denied execution outcome is produced. All consent evaluation and enforcement operations within the QODIQA framework occur within this interval.

Consent C

ARTIFACT

NORMATIVE DEFINITION

An explicit, machine-readable authorization record granted by an identified principal, scoped to a defined action type, bounded by a declared purpose, constrained by temporal validity limits, and subject to revocation. Consent within the QODIQA framework is not inferred, assumed, or derived from behavioral signals. It exists as a discrete, addressable record in the Consent Registry.

INFORMATIVE NOTE

This definition deliberately excludes implied consent, browsewrap consent, and terms-of-service acceptance. Those forms carry no normative standing within QODIQA enforcement logic.

EXPLICIT NON-INCLUSIONS

The following do not constitute Consent within this specification: inferred consent derived from behavioral patterns or usage history; assumed consent based on prior interactions; contextual consent derived from conversational or environmental signals; consent implied by silence or inaction; and any authorization not recorded as a discrete Consent Artifact in the Consent Registry.

Consent Artifact CA

ARTIFACT

NORMATIVE DEFINITION

A cryptographically signed, immutable record representing a discrete consent state at a specific point in time. A Consent Artifact contains, at minimum: the identity of the consenting principal, the scope of the consented action, the declared purpose, the temporal validity window, and an integrity anchor. Consent Artifacts are the unit of consent evaluated by the Policy Engine.

(Reference: QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0)

Consent Registry CR

ARCHITECTURAL

NORMATIVE DEFINITION

The authoritative, append-only data store in which all Consent Artifacts are recorded, indexed, and made retrievable for enforcement evaluation. The Consent Registry is the sole source of consent truth within the QODIQA framework. Consent not recorded in the Consent Registry does not exist for enforcement purposes.

(Reference: QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0)

Enforcement Gateway EG

ARCHITECTURAL

NORMATIVE DEFINITION

The system component through which all AI execution requests must pass before reaching the execution layer. The Enforcement Gateway is responsible for receiving execution requests, triggering consent validation, invoking policy evaluation, and returning a deterministic enforcement decision. No execution request may bypass the Enforcement Gateway.

(Reference: QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0)

Decision Boundary DB

CONTROL

NORMATIVE DEFINITION

The complete set of conditions, constraints, and input parameters that fully determine the output of a single enforcement evaluation. A Decision Boundary is closed: no inputs outside its declared scope affect the evaluation outcome. Extension of the Decision Boundary requires a versioned policy amendment.

(Reference: QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0)

Policy Engine PE

CONTROL

NORMATIVE DEFINITION

The component responsible for evaluating declared execution intent against applicable policy rules and resolved consent state, producing a deterministic enforcement decision. The Policy Engine operates on explicitly declared inputs only. It does not access raw data, perform semantic interpretation of content, or apply probabilistic risk scoring.

EXPLICIT NON-INCLUSIONS

The Policy Engine does not perform: semantic analysis or natural language interpretation of content; probabilistic risk scoring or threat modeling; contextual inference from behavioral or environmental signals; model-derived reasoning about intent; and any function that introduces variance into enforcement output for identical inputs.

(Reference: QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0; QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0)

Blocking Event BE

CONTROL

NORMATIVE DEFINITION

A condition arising during enforcement evaluation that causes the Policy Engine to halt processing and return a deny decision. Blocking Events include, without limitation: absent consent, expired consent, consent scope mismatch, policy rule violation, and integrity verification failure. All Blocking Events are recorded as Audit Artifacts.

Fail-Closed State FCS

STATE

NORMATIVE DEFINITION

The system condition in which, when required inputs for enforcement evaluation are absent, incomplete, or unresolvable, the default output is a deny decision. No execution proceeds in the absence of a complete and valid enforcement evaluation. Fail-Closed State is a mandatory property of all QODIQA-conformant enforcement implementations.

(Reference: QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0)

Non-Execution State NES

STATE

NORMATIVE DEFINITION

The state in which an AI execution request has been received by the Enforcement Gateway but has not been authorized. A system in Non-Execution State has neither permitted nor committed to any downstream action. The Non-Execution State is the initial and default state for all requests entering the Enforcement Gateway.

Execution Authorization EA

CONTROL

NORMATIVE DEFINITION

A deterministic, time-stamped enforcement decision produced by the Policy Engine that permits a specified execution request to proceed under stated conditions. Execution Authorization is not granted by default and is not presumed by the absence of a deny decision. It must be explicitly produced as an output of a completed enforcement evaluation.

Inference Boundary IB

CONTROL

NORMATIVE DEFINITION

The declared limit of the scope of execution authorized for a given AI invocation. The Inference Boundary constrains what the AI system may produce, access, or affect as a result of an authorized execution. Outputs or effects falling outside the declared Inference Boundary constitute a boundary violation and SHALL be treated as a Blocking Event in subsequent evaluations.

Consent Validation CV

PROCESS

NORMATIVE DEFINITION

The process by which the enforcement system verifies that a Consent Artifact exists in the Consent Registry, is cryptographically intact, is temporally valid at the moment of evaluation, and is scoped to cover the action declared in the execution request. Consent Validation precedes Policy Evaluation in all enforcement flows.

(Reference: QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0)

Enforcement Action ENF

PROCESS

NORMATIVE DEFINITION

An operation performed by the Enforcement Gateway as a result of a completed enforcement evaluation. Enforcement Actions include: granting Execution Authorization, returning a deny decision, returning a conditional authorization with stated restrictions, and recording an Audit Artifact. An Enforcement Action is distinct from and independent of any downstream AI operation.

Audit Artifact AA

ARTIFACT

NORMATIVE DEFINITION

A tamper-evident, cryptographically anchored record generated at the completion of every enforcement evaluation, capturing the complete set of inputs, the resolved consent state, the applicable policy version, and the resulting Enforcement Action. Audit Artifacts are appended to the Immutable Log and are the primary mechanism through which enforcement decisions are made verifiable and reproducible.

(Reference: QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0; QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0)

Immutable Record IR

ARTIFACT

NORMATIVE DEFINITION

A data record whose content, once written, cannot be altered, deleted, or overridden by any system operation or administrative action. Within the QODIQA framework, Immutable Records include Consent Artifacts, Audit Artifacts, and policy version snapshots. Immutability is enforced through append-only storage architecture and cryptographic integrity anchoring.

Cryptographic Integrity Anchor CIA

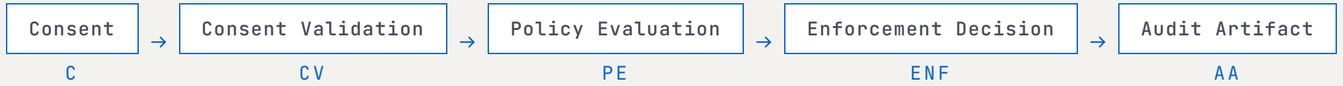
ARTIFACT

NORMATIVE DEFINITION

A cryptographic value, specifically a digital signature or a hash chain element, bound to a specific record and used to verify that the record's content has not been altered since its creation. Cryptographic Integrity Anchors are required for all Consent Artifacts and Audit Artifacts within QODIQA-conformant implementations.

(Reference: QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0)

FIGURE 1: ENFORCEMENT EVALUATION FLOW



All enforcement evaluations traverse this sequence in full. No step is optional under normal operating conditions. Each node corresponds to a term defined in this specification.

This flow defines the normative sequence of enforcement evaluation but does not prescribe implementation topology.

TABLE 2: TERM DEPENDENCY RELATIONSHIPS

TERM	DEPENDS ON	DEPENDENCY TYPE
Deterministic Runtime Consent Enforcement (DRCE)	Consent (C), Policy Engine (PE), Enforcement Gateway (EG)	Structural
Consent (C)	Consent Registry (CR)	Structural
Consent Artifact (CA)	Consent (C), Cryptographic Integrity Anchor (CIA)	Input / Structural

TERM	DEPENDS ON	DEPENDENCY TYPE
Consent Validation (CV)	Consent Artifact (CA), Consent Registry (CR)	Input
Policy Engine (PE)	Consent Validation (CV), Decision Boundary (DB)	Input / Constraint
Decision Boundary (DB)	Policy Engine (PE)	Constraint
Enforcement Gateway (EG)	Policy Engine (PE), Consent Validation (CV)	Structural
Enforcement Action (ENF)	Enforcement Gateway (EG), Policy Engine (PE)	Output
Audit Artifact (AA)	Enforcement Action (ENF), Cryptographic Integrity Anchor (CIA)	Output / Structural

SECTION 3

Architectural Component Definitions

The following definitions specify the named components, layers, and boundaries referenced in QODIQA architectural documents. Each definition bounds the functional scope of the named component within the enforcement stack.

Enforcement Layer EL

ARCHITECTURAL

NORMATIVE DEFINITION

The logical tier of the QODIQA reference architecture comprising the Enforcement Gateway, Policy Engine, and Consent Validation components. The Enforcement Layer is the sole site of enforcement decision-making. No other architectural layer produces or modifies enforcement decisions.

(Reference: QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0)

Registry Interface RI

ARCHITECTURAL

NORMATIVE DEFINITION

The defined programmatic interface through which the Enforcement Layer queries the Consent Registry. The Registry Interface is the exclusive access path to Consent Artifact data for enforcement evaluation purposes. Direct registry access by components outside the Enforcement Layer is PROHIBITED.

Policy Evaluation Module PEM

ARCHITECTURAL

NORMATIVE DEFINITION

The bounded component within the Policy Engine responsible for applying a specific versioned policy rule set against resolved consent state and declared execution intent. The Policy Evaluation Module receives only the inputs required for evaluation and returns only a binary or ternary enforcement decision. It retains no state between evaluations.

Conformance Validation Layer CVL

ARCHITECTURAL

NORMATIVE DEFINITION

The system or procedural layer responsible for verifying that an implementation satisfies the requirements of the QODIQA Conformance Test Suite and Certification Framework. The Conformance Validation Layer operates independently of the Enforcement Layer and does not participate in runtime enforcement decisions.

(Reference: QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0)

Certification Level CL

CONTROL

NORMATIVE DEFINITION

A defined tier of conformance achievement, as specified in the QODIQA Certification Framework, indicating the scope of control points satisfied by an implementation. Certification Levels are ordinal, cumulative, and verifiable. Attainment of a higher Certification Level requires satisfaction of all requirements at lower levels plus additional level-specific requirements.

(Reference: QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0)

Deterministic Evaluation Path DEP

CONTROL

NORMATIVE DEFINITION

The complete sequence of processing steps (from request receipt at the Enforcement Gateway through consent validation, policy evaluation, and Audit Artifact generation) that produces an enforcement decision. The Deterministic Evaluation Path is fully defined, has no optional branches under normal operating conditions, and must be traversed in its entirety for each request.

(Reference: QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0)

Control Surface CS

CONTROL

NORMATIVE DEFINITION

The complete set of defined control points at which enforcement decisions are made or enforcement conditions are checked within a QODIQA implementation. The Control Surface is enumerated in the QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0 and constitutes the measurable scope of enforcement coverage.

(Reference: QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0)

Trust Boundary TB

CONTROL

NORMATIVE DEFINITION

A declared demarcation between system components or operational domains across which data, requests, or credentials must be explicitly validated before being accepted as input to enforcement evaluation. Trust is not transitive across Trust Boundaries. Each boundary crossing requires independent verification.

(Reference: QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0)

External Dependency XD

ARCHITECTURAL

NORMATIVE DEFINITION

Any system, service, or data source not contained within the QODIQA Enforcement Layer upon which the enforcement evaluation process relies. External Dependencies are subject to availability failure and MUST be treated as potential sources of Blocking Events. A QODIQA-conformant implementation documents all External Dependencies and specifies failure behavior for each.

(Reference: QODIQA: Residual Risk and Assumption Disclosure Annex, Version 1.0)

Runtime Invocation RVI

CONTROL

NORMATIVE DEFINITION

A single, discrete request submitted to the Enforcement Gateway for evaluation, associated with one declared intent, one identified principal, and one target action scope. Each Runtime Invocation is evaluated independently. The outcome of a prior Runtime Invocation does not affect the evaluation of a subsequent one, except where applicable policy rules explicitly reference prior enforcement history.

SECTION 4

Security and Integrity Terms

The following definitions specify terms governing the cryptographic, integrity, and security properties required of QODIQA-conformant implementations. These definitions are aligned with and bounded by the QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0.

Cryptographic Binding CB

ARTIFACT

NORMATIVE DEFINITION

The association between a record, such as a Consent Artifact or Audit Artifact, and a cryptographic value such that any modification to the record's content renders the cryptographic value invalid. Cryptographic Binding is achieved through digital signature or keyed hash mechanisms as specified in the QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0.

(Reference: QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0)

Integrity Verification IV

PROCESS

NORMATIVE DEFINITION

The process of confirming that a record's content matches its Cryptographic Binding, establishing that the record has not been altered since its creation. Integrity Verification is a mandatory step in Consent Validation and MUST be performed for every Consent Artifact retrieved from the Consent Registry during an enforcement evaluation.

(Reference: QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0)

Tamper Detection TD

PROCESS

NORMATIVE DEFINITION

The capability of a system component to identify that a record's content does not match its Cryptographic Binding. Detection of tamper constitutes an Integrity Failure Condition and SHALL trigger a Blocking Event. Tamper Detection is distinct from tamper prevention; it is a detection and response property, not a prevention property.

Artifact Signing AS

PROCESS

NORMATIVE DEFINITION

The cryptographic operation by which a Cryptographic Integrity Anchor is computed and attached to a record at the time of its creation. Artifact Signing uses private key operations as specified in the QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0 and is performed exactly once per record, at creation time. Re-signing of existing records is PROHIBITED.

(Reference: QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0)

Key Management Boundary KMB

CONTROL

NORMATIVE DEFINITION

The defined operational and architectural perimeter within which cryptographic key material used for Artifact Signing is generated, stored, and used. Key material SHALL NOT cross the Key Management Boundary in plaintext form. Access to key material is restricted to the signing operation and is not available to enforcement evaluation logic.

(Reference: QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0)

Immutable Log IL

ARCHITECTURAL

NORMATIVE DEFINITION

The append-only, cryptographically chained sequence of Audit Artifacts produced by the Enforcement Layer. Entries in the Immutable Log cannot be altered or deleted after being written. The Immutable Log is the primary evidence source for enforcement audits and reproducibility verification.

(Reference: QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0; QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0)

Append-Only Record AOR

ARTIFACT

NORMATIVE DEFINITION

A record for which the only permitted write operation is creation. No update, deletion, or modification operation is permitted after creation. All Consent Artifacts and Audit Artifacts within the QODIQA framework are Append-Only Records. Revocation of a Consent Artifact is achieved through the creation of a new Revocation State record, not through modification of the original.

Revocation State RS

STATE

NORMATIVE DEFINITION

A record in the Consent Registry asserting that a specific Consent Artifact is no longer valid as of a declared timestamp. A Revocation State record does not alter the original Consent Artifact. Consent Validation MUST check for the existence of a Revocation State record for any Consent Artifact retrieved during an enforcement evaluation. A revoked Consent Artifact SHALL NOT authorize execution.

(Reference: QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0)

Integrity Failure Condition IFC

STATE

NORMATIVE DEFINITION

A system state in which the result of Integrity Verification for a record is negative, indicating that the record's content does not match its Cryptographic Binding. An Integrity Failure Condition is an immediate Blocking Event. The affected record SHALL NOT be used as input to enforcement evaluation. The condition SHALL be recorded in the Immutable Log.

SECTION 5

Regulatory and Governance Terms

The following definitions specify terms used in the context of conformance assessment, certification, regulatory alignment, and document governance within the QODIQA framework. A critical distinction is maintained between conformance, which is a technical property, and compliance, which is a legal or regulatory property.

Conformance CON

GOVERNANCE

NORMATIVE DEFINITION

The verifiable state of an implementation in which all mandatory requirements of the applicable QODIQA specification set, as assessed through the Conformance Test Suite, are satisfied. Conformance is a technical determination. It does not imply, guarantee, or substitute for regulatory compliance with any external legal framework.

(Reference: QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0; QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0)

Certification CERT

GOVERNANCE

NORMATIVE DEFINITION

The formal recognition issued by an authorized certifying body upon verified attainment of a specified Certification Level. Certification is based on documented evidence of Conformance and does not extend beyond the scope, version, and configuration of the implementation assessed. Certification SHALL NOT be claimed for configurations, versions, or deployment contexts not covered by the assessment.

(Reference: QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0)

Compliance COMP

GOVERNANCE

NORMATIVE DEFINITION

The state of an organization or system in which applicable legal, regulatory, or contractual obligations are satisfied. Compliance is a legal determination, not a technical one. QODIQA Conformance and Certification are evidence inputs that MAY be relevant to a compliance determination but do not constitute compliance with any regulatory framework. Organizations remain solely responsible for their own compliance assessments.

INFORMATIVE NOTE

The distinction between Conformance and Compliance is intentional and significant. QODIQA specifications assert technical properties only. Claims of regulatory compliance, with GDPR, the EU AI Act, NIST AI RMF, or any other framework, must be made by qualified legal and compliance professionals, not derived from QODIQA conformance status alone.

(Reference: QODIQA: Regulatory Alignment Matrix for Deterministic Runtime Consent Enforcement, Version 1.0; QODIQA: Positioning and Scope Limitation Statement, Version 1.0)

Non-Coverage Disclosure NCD

GOVERNANCE

NORMATIVE DEFINITION

A formal statement accompanying a conformance or certification claim identifying the specific control points, domains, or requirements that fall outside the scope of the assessed implementation. Non-Coverage Disclosures are mandatory for partial-scope implementations. Their absence is treated as a claim of full-scope coverage.

(Reference: QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0)

Scope Limitation SL

GOVERNANCE

NORMATIVE DEFINITION

A declared boundary on the applicability of a QODIQA specification, certification claim, or conformance assessment. Scope Limitations identify the deployment contexts, data categories, jurisdictions, or operational conditions to which an assessed claim applies. Claims made outside a declared Scope Limitation are unverified.

(Reference: QODIQA: Positioning and Scope Limitation Statement, Version 1.0)

Amendment AMD

GOVERNANCE

NORMATIVE DEFINITION

A versioned modification to a QODIQA specification document, processed through the change control procedure defined in Section 8. An Amendment that alters a normative definition in this specification constitutes a new version of this document. Amendments are recorded as Immutable Records and are appended to the document revision history.

(Reference: QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0)

Canonical Version CV

GOVERNANCE

NORMATIVE DEFINITION

The current, authoritatively published version of a QODIQA specification document as recognized by the QODIQA Governance Charter. The Canonical Version is the only version against which conformance assessments and certification claims are valid. Prior versions are superseded upon publication of a Canonical Version but remain accessible as archived references.

(Reference: QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0)

Breaking Change BC

GOVERNANCE

NORMATIVE DEFINITION

A modification to a QODIQA specification that alters the meaning, scope, or binding status of a normative requirement such that implementations previously assessed as conformant may no longer satisfy the amended requirement without modification. Breaking Changes require a major version increment and mandatory re-assessment of affected certified implementations.

Normative Revision NR

GOVERNANCE

NORMATIVE DEFINITION

Any Amendment to a QODIQA specification document that alters, adds, or removes normative requirements. Normative Revisions are distinguished from editorial revisions, which correct typographical or formatting errors without changing meaning. Normative Revisions require a version increment according to the versioning scheme defined in Section 8.

Interpretive Drift ID

GOVERNANCE

NORMATIVE DEFINITION

The condition in which the practical application of a term diverges from its definition in this specification, whether through informal restatement, paraphrase, omission, or contextual substitution. Interpretive Drift constitutes semantic non-conformance. Its detection obligates the responsible party to align usage with this specification through documented correction.

(Reference: Section 7 · Semantic Integrity Requirements)

SECTION 6

Prohibited Interpretations

This section enumerates interpretations of QODIQA terms that are explicitly prohibited. These prohibitions are necessary because the terms defined in this specification are used in adjacent domains, AI governance, regulatory compliance, cryptography, and legal practice, where they carry meanings that differ materially from their definitions within the QODIQA framework. Adopting such adjacent meanings within QODIQA normative contexts constitutes Interpretive Drift and SHALL be treated as a non-conformance condition.

PROHIBITED INTERPRETATION REGISTRY · NORMATIVE

Deterministic	SHALL NOT be interpreted as referring to probabilistic optimization, best-effort approximation, or any process in which the same inputs may produce varying outputs. Within QODIQA, determinism is an unconditional property of enforcement evaluation, not a performance characteristic.

Enforcement	SHALL NOT be interpreted as advisory notification, logging without action, retrospective flagging, or any process that permits execution to proceed without a completed enforcement evaluation. Enforcement is a gate, not an observer.
Consent	SHALL NOT be interpreted as inferred consent, assumed consent, behavioral consent, contextual consent derived from usage patterns, or any form of consent not recorded as an explicit Consent Artifact in the Consent Registry. Consent exists as a discrete record or it does not exist.
Conformance	SHALL NOT be interpreted as regulatory compliance, legal certification, a guarantee of lawful operation under any jurisdiction, or equivalence to certification under any external standard. Conformance is a bounded technical determination scoped to the QODIQA specification set.
Fail-Closed	SHALL NOT be interpreted as a recoverable error state, a degraded-mode fallback, or any condition in which execution may proceed with reduced enforcement. Fail-Closed is an unconditional deny in the absence of a complete, valid enforcement evaluation.
Immutable	SHALL NOT be interpreted as difficult to modify, access-controlled, or administratively restricted. Immutability is a structural property enforced architecturally. A record that can be modified under any administrative or emergency condition is not Immutable within this specification.
Policy Engine	SHALL NOT be interpreted as a system capable of semantic analysis, risk scoring, natural language policy interpretation, or probabilistic judgment. The Policy Engine evaluates declared inputs against explicit rules. It does not reason.
Certification Level	SHALL NOT be interpreted as a measure of security strength, data protection adequacy, or readiness for any specific regulatory context. Certification Levels measure scope of coverage against QODIQA control points exclusively.
Audit Artifact	SHALL NOT be interpreted as a log entry, a notification record, or a monitoring event. An Audit Artifact is a structured, cryptographically anchored enforcement record that enables independent reproduction of an enforcement decision. Unstructured log entries do not satisfy this definition.

Trust Boundary	SHALL NOT be interpreted as a firewall perimeter, a network security zone, or a general access control boundary. Within QODIQA, Trust Boundaries govern the conditions under which data and requests are accepted as valid inputs to enforcement evaluation.

[1] This list is not exhaustive. Additional prohibited interpretations MAY be added through Amendment. The absence of a term from this list does not imply that its informal usage is permitted; all terms defined in this specification are subject to the semantic control requirements of Section 7.

SECTION 7

Semantic Integrity Requirements

This section establishes the requirements that govern how terms defined in this specification are used across the QODIQA corpus. These requirements elevate this document from a reference glossary to an active enforcement instrument. Failure to comply with the requirements in this section constitutes semantic non-conformance.

7.1 Non-Redefinition Requirement

Terms defined in this specification SHALL NOT be redefined, re-scoped, or assigned alternative meanings in any derivative QODIQA document, including but not limited to implementation guides, use-case dossiers, training materials, or marketing communications. Any document that requires a different meaning for a defined term must introduce a new term rather than reusing a defined one.

7.2 Non-Substitution Requirement

Alternative terminology SHALL NOT substitute for canonical definitions within normative QODIQA documents. For example, a document MUST NOT use "authorization record" to mean Consent Artifact, or "decision log" to mean Audit

Artifact, unless those alternative terms are explicitly cross-referenced to the canonical definition in this specification and that cross-reference is normatively declared.

> Semantic drift, whether by omission, paraphrase, or contextual substitution, **constitutes non-conformance**. Certification bodies SHALL treat documented instances of semantic drift as findings requiring remediation before a conformance assessment can be closed.

7.3 Terminological Adherence as Certification Condition

Certification at any level under the QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0 requires that all normative documentation produced by the implementing organization uses terms defined in this specification in accordance with their definitions herein. Terminological adherence is a certification condition, not merely a stylistic preference. Certification bodies SHALL verify terminological adherence as part of documentation review.

7.4 Semantic Drift Detection and Remediation

Where semantic drift is detected in a QODIQA corpus document, whether through internal review, external assessment, or certification audit, the responsible document owner SHALL initiate a documented correction process within thirty days of detection. The correction process SHALL produce either an amendment to the affected document or, where the affected usage was valid but required a new term, an Amendment request to this specification processed under Section 8.

7.5 Adversarial Interpretive Resistance

Definitions in this specification are drafted to withstand adversarial interpretive analysis, that is, readings that extract permissions or narrow obligations by exploiting ambiguity, silence, or term boundaries. Where an adversarial reading of a definition is identified, it SHALL be addressed either by Amendment to the affected definition or by addition to the Prohibited Interpretations registry in Section 6.

SECTION 8

Change Control of Terminology

This section defines the processes and constraints governing modifications to the terms and definitions in this specification. Terminology is a normative asset. Changes to it affect the semantic foundation of the entire QODIQA corpus. Accordingly, change control for this document is stricter than for operational or architectural specifications.

8.1 Amendment Process

An Amendment to this specification is initiated through a formal Amendment Request submitted to the QODIQA Governance Charter authority. The Amendment Request SHALL identify the term to be added, modified, or deprecated; the proposed new or revised definition; the rationale; and the impact on conformance assessments and certification claims that reference the affected term. All Amendment Requests are tracked and resolved before publication of an amended version.

8.2 Breaking Semantic Change

A change to a definition constitutes a Breaking Semantic Change if it narrows or expands the scope of a term such that implementations or documents previously conformant under the prior definition may no longer be conformant under the new one, or vice versa. Breaking Semantic Changes require a major version increment of this specification, mandatory

notification to all current certificate holders, and a defined re-assessment window not to exceed twelve months from publication.

8.3 Versioning Interaction

This specification uses a two-part version identifier: major.minor. A major version increment is triggered by any Breaking Semantic Change. A minor version increment is triggered by the addition of new terms, clarifying amendments that do not change scope, or additions to the Prohibited Interpretations registry. Editorial amendments that correct typographical errors without changing meaning are published as errata and do not trigger version increments.

> Conformance assessments and certification claims are valid only against a **specified version** of this specification. Claims referencing "QODIQA Terminology" without a version identifier are considered unverifiable and SHALL NOT be accepted by certifying bodies.

8.4 Deprecated Terms

A term may be deprecated when it is superseded by a new term with a more precise definition or when its use domain has been restructured. A deprecated term retains its definition in this specification, marked with a deprecation notice identifying the superseding term and the version in which deprecation took effect. Deprecated terms SHALL NOT be used in new QODIQA normative documents produced after the deprecation date. Existing documents using deprecated terms are not required to be updated but SHALL include a conformance note referencing the superseding term.

8.5 Governance Charter Alignment

All change control processes defined in this section operate under and are constrained by the QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0. Where this section and the Governance Charter conflict, the

Governance Charter takes precedence. Changes to the Governance Charter that affect terminology change control processes require a corresponding Amendment to this specification to maintain alignment.

CONCLUSION

Semantic Authority Statement

This specification establishes the terminological foundation upon which the enforceability of the entire QODIQA framework depends. The definitions contained herein are not descriptions of intent; they are bounded, operational definitions with direct consequences for conformance assessment, certification validity, and the defensibility of enforcement claims.

> Any QODIQA corpus document, certification claim, or conformance assertion that uses terms defined herein in a manner inconsistent with this specification is semantically non-conformant. **Semantic non-conformance weakens certification defensibility and SHALL be treated as a remediable finding.**

The Prohibited Interpretations in Section 6 and the Semantic Integrity Requirements in Section 7 exist because the domain in which QODIQA operates is subject to motivated misreading, whether through regulatory minimization, competitive reframing, or inadvertent drift. This specification eliminates the lexical surface area that such readings require.

Implementers, certifying bodies, regulatory reviewers, and derivative document authors are bound by the definitions in this specification as of the version stated on the cover. Version-specific conformance is not optional. It is a prerequisite for any valid claim against the QODIQA framework.

DOCUMENT STATUS

Document Status and Classification

This document is the Canonical Version of the QODIQA Terminology and Normative Definitions Specification, Version 1.0. It is published under the authority of the QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0 and is effective upon publication. It is issued as a Normative Specification and constitutes the sole lexical control instrument for the QODIQA framework.

This document is intended for the following audiences:

- QODIQA corpus document authors and maintainers
- Conformance assessment and certification bodies
- Legal and regulatory reviewers conducting QODIQA-related assessments
- System integrators implementing QODIQA-conformant architectures
- Standards bodies and interoperability working groups referencing QODIQA

This document is classified as a Public Release and MAY be distributed without restriction. Derivative documents that paraphrase, excerpt, or reference definitions from this specification SHALL identify the source version from which definitions are drawn.

This document should be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0
- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dușescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-TND-2026-001
TITLE	Terminology and Normative Definitions
SUBTITLE	Canonical Lexical Control Instrument for the QODIQA Framework
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Normative Specification
DOCUMENT STATUS	Normative: Canonical Lexical Reference
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 20 PART III · OPERATIONAL DOCUMENTS

Worked Example End-to-End Enforcement Scenario

A complete technical narrative tracing a single AI request through every enforcement stage, from intent declaration to revocation, so that any engineer can see the system running, not just described.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-WE-2026-001	1.0	2026	Illustrative Scenario	Normative	NORMATIVE

PURPOSE & SCOPE

This document traces a single AI action request, an AI diagnostic assistant reading a patient's longitudinal record on a clinical data platform, through every stage of QODIQA enforcement: intent declaration, consent resolution, policy evaluation, decision artifact production, execution gating, and mid-session revocation. All values shown are concrete. The scenario is designed to be complete enough that an engineer who has never read the Core Standard can identify, from this document alone, exactly what must be built and in what sequence. This is a normative illustrative document; it does not introduce requirements beyond those defined in the Core Standard and the 68-Point Enforcement Framework.

SECTION 1

Scenario Setup

Before tracing the enforcement flow, it is necessary to establish the deployment context, the actors involved, the state of the consent registry at the time of the request, and the active policy version. These are not background details; they are the concrete inputs that QODIQA will consume. Every enforcement decision in this scenario is fully determined by this initial state.

1.1 Platform and Actors

DEPLOYMENT CONTEXT

NeuraPath Health operates a clinical decision-support platform that uses an AI model (`dx-assistant-v3`) to analyze longitudinal patient records and surface diagnostic patterns for reviewing clinicians. The platform processes data under GDPR Article 9 (sensitive health data) and is subject to internal NeuraPath data governance policy `NPGOV-CLINICAL-2026-003` .

QODIQA is deployed as an enforcement sidecar between the application tier and the model execution layer. Every request from any AI component must pass through QODIQA before the model receives data. The enforcement layer runs in strict fail-closed mode: any error, timeout, or missing field produces an unconditional DENY.

REQUESTING ACTOR	DATA SUBJECT	REVIEWING CLINICIAN	QODIQA POLICY
actor:dx-assistant-v3 AI diagnostic assistant. Authorized to read clinical records for purpose `dx_analysis` only.	**patient:PT-00441** Elena Marin, adult patient. Has an active consent record for AI-assisted diagnostics.	**user:clinician-DR-IONESCU** Dr. Radu Ionescu. Initiated the diagnostic session at 09:14 UTC. Not a QODIQA actor; operates downstream of enforcement.	**NPGOV-CLINICAL-2026-003 · v7** Immutable published policy version active at the time of this session.

1.2 Consent Registry State at T_0

The following consent record for patient PT-00441 exists in the Consent Registry at session start. This record was issued at T_0 − 14 days, when the patient signed the NeuraPath digital consent form for AI-assisted diagnostics. It has not been modified since.

CONSENT RECORD: AS STORED IN CONSENT REGISTRY

CONSENT_REGISTRY · RECORD: CNST-PT00441-DX-2026

```
consent_id: "CNST-PT00441-DX-2026"
subject_id: "patient:PT-00441"
granted_to: "actor:dx-assistant-v3"
purpose: "dx_analysis"
data_categories: [
  "clinical.longitudinal_record",
  "clinical.lab_results",
  "clinical.imaging_metadata"
]
jurisdiction: "EU-RO"
valid_from: "2026-03-24T00:00:00Z"
valid_until: "2026-09-24T23:59:59Z"
revoked: false
revocation_ts: null
issuer_sig: "sha256:a3f9c...d82e1" // cryptographic binding to NeuraPath CA
```

SCOPE BOUNDARY

This consent record grants `actor:dx-assistant-v3` access to three specific data categories for the single purpose `dx_analysis`. It does not extend to research, billing, training, or any secondary use. QODIQA does not infer scope; any request for a data category or purpose not listed here will be denied at the consent resolution stage regardless of what policies say.

1.3 Active Policy Version

Policy `NPGOV-CLINICAL-2026-003 · v7` was published and frozen on 2026-04-01. It cannot be modified. If NeuraPath's governance team needs to change any condition, they must publish v8, which will apply only to requests submitted after v8 becomes active. The version applied to this request will be permanently recorded in the decision artifact.

POLICY NPGOV-CLINICAL-2026-003 · V7: KEY CONDITIONS

CONDITION	REQUIRED VALUE	FAILURE BEHAVIOR
Purpose must be	`dx_analysis`	DENY: purpose mismatch
Actor must be	registered in actor registry, role `ai-model.clinical`	DENY: actor not authorized
Jurisdiction must be	`EU-RO` or `EU-*`	DENY: jurisdiction out of scope
Data categories	Must be subset of consent-granted categories	DENY: category not consented
Consent state	GRANTED · not expired · not revoked	DENY: consent invalid
Session window	Request must arrive within active clinical session	ALLOW-WITH-RESTRICTIONS: flag for clinician review
Data output restriction	Model output must not include raw identifiers	Restriction propagated to execution layer

STAGE 1

Intent Declaration

STAGE 1 OF 6 **DecisionRequest: Closed Contract Submission** *T = 09:14:32Z*

The diagnostic session has opened. Dr. Ionescu's interface requests that the AI assistant analyze PT-00441's longitudinal record for a potential metabolic diagnosis. The application tier constructs and submits a DecisionRequest to QODIQA. This is the first contact with the enforcement layer.

The DecisionRequest is the only input QODIQA accepts. It is a closed, immutable contract: once submitted, none of its fields can be modified, extended, or supplemented. If the application needs to change any field, it must submit a new request. There is no mechanism to amend a request in flight.

DECISIONREQUEST · SUBMITTED AT 09:14:32.041Z

REQUEST_ID: DRQ-20260407-004411

```
request_id: "DRQ-20260407-004411"
submitted_at: "2026-04-07T09:14:32.041Z"
actor_id: "actor:dx-assistant-v3"
action: "read.longitudinal_record"
purpose: "dx_analysis"
data_subjects: ["patient:PT-00441"]
data_categories: [
  "clinical.longitudinal_record",
  "clinical.lab_results"
]
jurisdiction: "EU-RO"
session_id: "SES-20260407-DR-IONESCU-0091"
policy_version: "NPGOV-CLINICAL-2026-003:v7"
request_hash: "sha256:b7d2f4...a9c310" // computed over all fields above
```

QODIQA immediately validates the structural completeness of the request. It verifies that all required fields are present, that no unknown fields have been injected, that the request hash matches the field content, and that the `policy_version` field references a known, published, and active policy document. This validation is purely structural: QODIQA does not yet know whether the request will be allowed.

INTENT DECLARATION: FIELD ANNOTATIONS

FIELD	VALUE	ENFORCEMENT SIGNIFICANCE
request_id	`DRQ-20260407-004411`	Globally unique. Will appear in the decision artifact and all downstream audit records. Enables trace-to-origin without ambiguity.
actor_id	`actor:dx-assistant-v3`	Identifies the acting AI model. QODIQA will verify this against the actor registry and the consent record's `granted_to` field.
action	`read.longitudinal_record`	Specifies the operation type. Policy conditions may restrict which actions are permitted for a given purpose. This is not free-text; it must match an enumerated action type.
purpose	`dx_analysis`	The declared purpose of the action. Must exactly match the `purpose` field in the consent record. A different purpose, even a plausible one like `quality_review`, would cause a DENY.
data_categories	`clinical.longitudinal_record`, `clinical.lab_results`	The two categories requested. Note that `clinical.imaging_metadata` is also in the consent record but is not requested here. The request need not exhaust the consented scope; it must only stay within it.
jurisdiction	`EU-RO`	Matches the consent record and falls within the policy's accepted jurisdiction set. A mismatch here would be a DENY regardless of consent state.
policy_version	`NPGOV-CLINICAL-2026-003:v7`	The caller explicitly declares which policy version governs this request. QODIQA will evaluate against exactly this version. If v8 is published tomorrow, past decisions are not retroactively affected.
request_hash	`sha256:b7d2f4...a9c310`	A SHA-256 digest of all fields. QODIQA recomputes this hash independently; a mismatch indicates tampering or transmission corruption and results in an immediate DENY with error code `E-HASH-MISMATCH`.

PASS Structural validation succeeded at 09:14:32.043Z. Request hash verified. All required fields present. Policy version `NPGOV-CLINICAL-2026-003:v7` confirmed active. Request contract frozen. Proceeding to consent resolution.

STAGE 2

Consent Resolution

STAGE 2 OF 6 **Consent Registry Lookup: Explicit State Resolution** *T = 09:14:32.043Z*

QODIQA queries the Consent Registry to resolve the consent state for the data subject identified in the request. Consent resolution is completely separate from policy evaluation. Policies do not know about consent; they only see the resolved consent state as a frozen input.

QODIQA performs a deterministic lookup against the Consent Registry using three keys derived from the request: `data_subject`, `actor_id`, and `purpose`. The lookup is exact-match only. There is no fuzzy matching, no inheritance from parent records, and no cross-purpose resolution. Either a record exists for this exact combination, or it does not.

If the registry is unavailable, network timeout, service error, or degraded response, QODIQA does not wait and does not retry beyond a single bounded timeout. It issues an immediate fail-closed DENY with reason code `E-CONSENT-STORE-UNAVAILABLE` and records a failure artifact. This is not an error state requiring manual intervention; it is the correct behavior.

CONSENT RESOLUTION: LOOKUP PARAMETERS

```
// Parameters derived from the frozen DecisionRequest
subject_id: "patient:PT-00441"
```

```
actor_id: "actor:dx-assistant-v3"
purpose: "dx_analysis"
eval_time: "2026-04-07T09:14:32.043Z" // evaluation timestamp, frozen
```

The registry returns record `CNST-PT00441-DX-2026`. QODIQA now performs four sequential checks. All four must pass; failure at any point terminates consent resolution with DENY.

CONSENT VALIDATION CHECKS: EXECUTED IN SEQUENCE

#	CHECK	TEST	RESULT
1	Record exists	Registry returned a non-null record for the lookup keys.	PASS
2	Not revoked	`revoked == false` and `revocation_ts == null`.	PASS
3	Within validity window	`valid_from (2026-03-24) ≤ eval_time (2026-04-07) ≤ valid_until (2026-09-24)`.	PASS
4	Requested categories within consent scope	Both `clinical.longitudinal_record` and `clinical.lab_results` appear in `data_categories` of the consent record.	PASS

All four checks pass. QODIQA freezes the consent state as a named reference. This snapshot is immutable: if the patient revokes consent one second later, it will not change the outcome of this evaluation. It will, however, affect the very next request, which is described in Stage 6.

RESOLVED CONSENT STATE · FROZEN AT 09:14:32.051Z

```
consent_ref: "CNST-PT00441-DX-2026"
state: "GRANTED"
resolved_at: "2026-04-07T09:14:32.051Z"
scope_confirmed: [
  "clinical.longitudinal_record",
  "clinical.lab_results"
```

```
]
issuer_sig_valid: true // cryptographic binding to NeuraPath CA verified
```

GRANTED Consent resolved as GRANTED at 09:14:32.051Z. Frozen state reference: `CNST-PT00441-DX-2026`. Proceeding to policy evaluation. Consent state is now immutable input; it will not be re-queried during this evaluation.

1 QODIQA does not verify the legal sufficiency of a consent record; it verifies its structural validity and scope alignment. Legal sufficiency is a governance concern addressed upstream of QODIQA.

2 The issuer signature check verifies that the consent record was created by an entity holding the NeuraPath CA private key. It does not verify the content of the patient's informed consent disclosure, which is a human-layer concern.

STAGE 3

Policy Evaluation

STAGE 3 OF 6 **Policy Engine: Deterministic Evaluation Against Frozen Context** *T = 09:14:32.051Z*

The policy engine receives two immutable inputs: the frozen DecisionRequest and the frozen consent state. It evaluates every condition in policy version NPGOV-CLINICAL-2026-003:v7 against this context. No external state is read during evaluation. The engine is a pure function.

Policy evaluation in QODIQA is not a reasoning process. The engine does not interpret, infer, or exercise judgment. It evaluates a finite set of declared conditions against a frozen context and produces exactly one outcome. Given the same inputs, it will always produce the same output; this is the determinism guarantee.

The evaluation context is fully captured at the moment evaluation begins. This means the context is impervious to race conditions, concurrent updates, or the passage of time during evaluation. The world that the policy engine sees is the

world as it existed at 09:14:32.051Z.

EVALUATION CONTEXT: FROZEN AT EVALUATION ENTRY

```
// This is the complete and immutable context for this evaluation.
// Nothing outside this structure is read during policy evaluation.

request: {
  actor_id: "actor:dx-assistant-v3",
  action: "read.longitudinal_record",
  purpose: "dx_analysis",
  data_categories: ["clinical.longitudinal_record", "clinical.lab_results"],
  jurisdiction: "EU-RO",
  session_id: "SES-20260407-DR-IONESCU-0091"
},
consent: {
  state: "GRANTED",
  consent_ref: "CNST-PT00441-DX-2026"
},
policy_version: "NPGOV-CLINICAL-2026-003:v7",
eval_timestamp: "2026-04-07T09:14:32.051Z"
```

The engine evaluates each policy condition in order. Conditions are stateless predicates. The table below shows the evaluation trace: the complete sequence of conditions tested, the test result for each, and the engine's running state.

POLICY EVALUATION TRACE · NPGOV-CLINICAL-2026-003:V7

CONDITION	TEST	RESULT
C-01 Purpose matches policy scope	`purpose == "dx_analysis"`	PASS
C-02 Actor is authorized for this purpose	Actor registry confirms `dx-assistant-v3` holds role `ai-model.clinical` with `dx_analysis` authorization.	PASS

CONDITION	TEST	RESULT
C-03 Jurisdiction within accepted range	`jurisdiction "EU-RO"` matches policy's `EU-*` wildcard.	PASS
C-04 Consent state is GRANTED	`consent.state == "GRANTED"`. Policy does not re-query the registry; it reads the frozen state.	PASS
C-05 Requested categories within consented scope	Both requested categories were confirmed in the frozen consent scope. Policy validates this was done before it received control.	PASS
C-06 Request within active clinical session	Session `SES-20260407-DR-IONESCU-0091` is active in session registry at eval time. Request is within session window.	PASS
C-07 Output restriction: no raw identifiers	Condition passes but adds a restriction: model output must pass through the de-identification filter before being returned to the application tier. This does not block execution; it constrains it.	RESTRICT

Conditions C-01 through C-06 all pass without restriction. Condition C-07 introduces an output restriction. This restriction does not change the top-level outcome from ALLOW; it modifies the terms of the allow. The restriction is encoded into the decision artifact and propagated to the execution layer, which is responsible for enforcing it operationally.

ALLOW-WITH-RESTRICTIONS

Policy evaluation completed at 09:14:32.058Z. All mandatory conditions satisfied. Restriction R-01 applies: model output must pass through de-identification filter `deid-filter-v2` before surface to application tier.

STAGE 4

Decision Artifact Production

STAGE 4 OF 6 **Immutable Decision Artifact: Creation and Storage** *T = 09:14:32.058Z*

The enforcement engine produces a Decision Artifact: a permanent, immutable record of the complete enforcement decision. This artifact is created before execution begins. It captures every input, every evaluation result, and the final outcome. Once written, it cannot be modified by any system component.

The Decision Artifact is the foundational unit of QODIQA's accountability model. It is not a log entry; it is a structured, cryptographically bound record that makes the enforcement decision independently verifiable. Given this artifact and the policy document it references, any auditor can replay the decision and confirm it is correct, without access to the production enforcement system, without access to the Consent Registry, and without any contextual interpretation.

DECISION ARTIFACT · DA-20260407-004411 · CREATED AT 09:14:32.058Z

DECISION_ARTIFACT · ID: DA-20260407-004411

```
artifact_id: "DA-20260407-004411"
request_id: "DRQ-20260407-004411" // trace link to originating request
created_at: "2026-04-07T09:14:32.058Z"

// ── OUTCOME ──────────────────────────────────────────
outcome: "ALLOW-WITH-RESTRICTIONS"
restrictions: [{
  id: "R-01",
  description: "Output must pass through deid-filter-v2 before surface",
  enforced_by: "execution-layer"
}]

// ── POLICY ───────────────────────────────────────────
policy_version: "NPGOV-CLINICAL-2026-003:v7"
```

```
policy_hash: "sha256:f3a1b9...c04d72" // hash of the immutable policy document

// ── CONSENT ──────────────────────────────────────────────
consent_ref: "CNST-PT00441-DX-2026"
consent_state_at_eval: "GRANTED"
consent_scope_used: [
  "clinical.longitudinal_record",
  "clinical.lab_results"
]

// ── REQUEST SNAPSHOT ─────────────────────────────────────
request_snapshot: {
  actor_id: "actor:dx-assistant-v3",
  action: "read.longitudinal_record",
  purpose: "dx_analysis",
  data_subjects: ["patient:PT-00441"],
  jurisdiction: "EU-RO",
  session_id: "SES-20260407-DR-IONESCU-0091",
  submitted_at: "2026-04-07T09:14:32.041Z",
  request_hash: "sha256:b7d2f4...a9c310"
},

// ── EVALUATION TRACE ─────────────────────────────────────
condition_results: [
  { condition: "C-01", result: "PASS" },  { condition: "C-02", result: "PASS" },  { condition: "C-03", result: "PASS" },
  { condition: "C-04", result: "PASS" },  { condition: "C-05", result: "PASS" },  { condition: "C-06", result: "PASS" },
  { condition: "C-07", result: "RESTRICT", restriction_id: "R-01" }
],

// ── INTEGRITY ────────────────────────────────────────────
enforcement_latency_ms: 17,
artifact_hash: "sha256:e8c3a1...7f2d90" // computed over all fields above; immutable
```

WHY THE ARTIFACT HASH MATTERS

The `artifact_hash` is computed over every field in the artifact, including the evaluation trace and the request snapshot. If any field is altered after creation, even a single character, the hash will not match. An auditor who independently computes the hash and finds a mismatch knows the artifact has been tampered with. This property makes the audit trail tamper-evident without requiring a blockchain or distributed ledger: the hash itself is the proof.

STAGE 5

Execution Gate

STAGE 5 OF 6 **Decision Emission and Execution Layer Response** *T = 09:14:32.062Z*

QODIQA emits the enforcement decision to the application tier. The application tier's execution layer reads the outcome and acts accordingly. QODIQA's responsibility ends at emission. The execution layer owns enforcement of the restriction.

QODIQA emits a structured enforcement response to the application tier. The response is minimal: it contains the outcome, the artifact reference, and the restrictions. It does not contain raw data, model outputs, or any content from the clinical record. QODIQA does not touch data; it governs access to it.

ENFORCEMENT RESPONSE · EMITTED TO APPLICATION TIER AT 09:14:32.062Z

```
outcome: "ALLOW-WITH-RESTRICTIONS"
artifact_ref: "DA-20260407-004411"
restrictions: [{
  id: "R-01",
  filter: "deid-filter-v2",
  apply_to: "model.output"
```

```
  }]
  emit_timestamp: "2026-04-07T09:14:32.062Z"
```

The execution layer receives this response and proceeds as follows: it loads patient PT-00441's longitudinal record and lab results from the data store, passes them to `dx-assistant-v3` for analysis, intercepts the model output before returning it to Dr. Ionescu's interface, applies `deid-filter-v2` to strip any raw patient identifiers from the output, and delivers the filtered result to the clinical interface. The artifact reference `DA-20260407-004411` is logged by the execution layer alongside the session record, creating a traceable link between the governance decision and the operational action.

At this point, the happy-path scenario is complete. Total enforcement latency from request submission to decision emission: **21 milliseconds**. The patient has not been inconvenienced. Dr. Ionescu has received a de-identified diagnostic analysis. The enforcement decision is permanently recorded and independently verifiable.

> **WHAT QODIQA DID NOT DO**
>
> QODIQA did not access the patient record. It did not read the model output. It did not apply the de-identification filter. It did not assess whether the diagnostic analysis was clinically appropriate. These are all outside QODIQA's scope. QODIQA's only concern was whether the request to access the data was authorized. The answer was yes, with a restriction, and that answer is now permanently provable.

STAGE 6

Mid-Session Revocation

STAGE 6 OF 6 **Patient Revocation: Mid-Session Consent Withdrawal** *T = 09:22:17Z*

Eight minutes after the first request was authorized, patient PT-00441 contacts NeuraPath's patient portal and withdraws consent for AI-assisted diagnostics. The revocation is processed immediately. A second AI request arrives seven seconds later, from the same actor, for the same patient, for the same purpose. The outcome is completely different.

7.1 The Revocation Event

At 09:22:17Z, the NeuraPath patient portal receives a consent withdrawal request from Elena Marin. The portal's consent management service writes a revocation update to the Consent Registry immediately. The update is atomic: the registry transitions from the existing record to a revoked state in a single operation.

CONSENT REGISTRY · RECORD STATE AFTER REVOCATION AT 09:22:17.339Z

```
consent_id: "CNST-PT00441-DX-2026"
subject_id: "patient:PT-00441"
granted_to: "actor:dx-assistant-v3"
purpose: "dx_analysis"
revoked: true
revocation_ts: "2026-04-07T09:22:17.339Z"
revocation_reason: "patient_withdrawal" // written by consent management service
```

The revocation does not affect past decisions. Decision Artifact `DA-20260407-004411`, which authorized the 09:14 request, remains valid and correct. Its consent state was resolved at 09:14:32.051Z, at which point the record was not revoked. The artifact truthfully records what was true at that moment, and that truth does not change retroactively. This is

a foundational property of QODIQA's architecture: decisions are facts about a point in time, not claims about the world today.

7.2 The Next Request: Instant Denial

At 09:22:24Z, seven seconds after revocation, the clinical interface attempts a second AI read request. Dr. Ionescu's session is still active. The application tier constructs a new DecisionRequest, structurally identical to the first except for a new request ID and timestamp. It is submitted to QODIQA.

- `09:14:32Z`
  First request authorized. Decision Artifact `DA-20260407-004411` produced. Execution proceeds with restriction R-01.
- `09:22:17Z · REVOCATION`
  Patient PT-00441 withdraws consent via patient portal. Consent Registry updated atomically. `CNST-PT00441-DX-2026` marked revoked.
- `09:22:24Z · DENY`
  Second request submitted. Consent resolution fails at check #2: `revoked == true`. Execution blocked. Decision Artifact `DA-20260407-009972` produced with outcome DENY.

Consent resolution for the second request proceeds through the same four-check sequence as before. Check #1 passes: the record still exists in the registry. Check #2 fails: `revoked == true`. Resolution stops immediately. The engine does not evaluate checks 3 and 4, and does not enter policy evaluation. The outcome is DENY.

DECISION ARTIFACT: DA-20260407-009972 · SECOND REQUEST, DENIED

```
DECISION_ARTIFACT · ID: DA-20260407-009972

artifact_id: "DA-20260407-009972"
request_id: "DRQ-20260407-009972"
created_at: "2026-04-07T09:22:24.112Z"

outcome: "DENY"
deny_reason: "E-CONSENT-REVOKED"
```

```
deny_stage: "consent_resolution" // denied before reaching policy evaluation

consent_ref: "CNST-PT00441-DX-2026"
consent_state_at_eval: "REVOKED"
consent_revocation_ts: "2026-04-07T09:22:17.339Z"

request_snapshot: {
  actor_id: "actor:dx-assistant-v3",
  action: "read.longitudinal_record",
  purpose: "dx_analysis",
  session_id: "SES-20260407-DR-IONESCU-0091",
  submitted_at: "2026-04-07T09:22:24.101Z"
},

enforcement_latency_ms: 11,
artifact_hash: "sha256:c1f7e2...84a301"
```

DENY Request `DRQ-20260407-009972` denied at consent resolution stage. Reason: `E-CONSENT-REVOKED`. Revocation recorded at 09:22:17.339Z, 6.762 seconds before this request. Execution blocked. No data accessed. Enforcement latency: 11ms.

The enforcement response returned to the application tier is a DENY with error code `E-CONSENT-REVOKED`. The application tier is expected to surface an appropriate message to Dr. Ionescu's interface, for example, that the patient's consent is no longer active and the session cannot continue without intervention from the care team. QODIQA does not dictate the user experience; it dictates the enforcement outcome.

7.3 What the Revocation Artifact Proves

The two decision artifacts now tell a coherent and independently verifiable story. `DA-20260407-004411` proves that at 09:14:32Z, consent was valid and the request was properly authorized. `DA-20260407-009972` proves that at 09:22:24Z, consent had been revoked and the request was correctly blocked. The gap between revocation timestamp and the second request's submission, 6.762 seconds, is on record. If NeuraPath's legal team or a data protection

authority ever asks whether consent was respected, these two artifacts are the answer. Not a narrative. Not a policy document. Proof.

SECTION 8

Audit Replay

Three months later, a data protection authority reviewing NeuraPath's AI practices asks: *"On April 7th 2026, was patient PT-00441's data accessed by AI systems after she withdrew consent?"*

The NeuraPath compliance team runs the following audit query against the decision artifact store:

AUDIT QUERY

```
subject: "patient:PT-00441"
date: "2026-04-07"
after_timestamp: "2026-04-07T09:22:17.339Z" // revocation timestamp
outcome_filter: "ALLOW | ALLOW-WITH-RESTRICTIONS"
```

The query returns zero results. No ALLOW or ALLOW-WITH-RESTRICTIONS artifact exists for patient PT-00441 after 09:22:17.339Z on April 7th. The only artifact in that window is `DA-20260407-009972`, which is a DENY. The answer to the authority's question is: **no**, and that answer comes with a cryptographically verifiable artifact, not a human attestation.

The compliance team can additionally replay Decision Artifact `DA-20260407-004411`, the authorized request, against the policy document at `sha256:f3a1b9...c04d72` and the consent record as it existed at 09:14:32.051Z. The replay will produce the same ALLOW-WITH-RESTRICTIONS outcome. This confirms that the earlier access was correctly authorized under the rules that were in effect at the time.

THE AUDIT PROPERTY

The replay invariant holds: given the same frozen inputs, request, consent state, policy version, evaluation timestamp, QODIQA's evaluation engine always produces the same decision. This makes the audit outcome reproducible without trust in any single party's assertion. An auditor who independently constructs the replay environment and verifies the artifact hashes reaches the same conclusion as the production system did in real time.

SECTION 9

What QODIQA Did Not Do

This scenario demonstrates several things QODIQA deliberately chose not to do. Understanding these boundaries is as important as understanding what QODIQA enforces, because the constraints define where implementation responsibility lies in the systems around it.

SCOPE BOUNDARIES: QODIQA RESPONSIBILITIES VS. SURROUNDING SYSTEM RESPONSIBILITIES

QODIQA DID NOT DO THIS	WHO IS RESPONSIBLE
Read or inspect the patient's clinical record or lab results.	Data store and execution layer. QODIQA governs access; it does not access data.
Verify that Dr. Ionescu is clinically qualified to request this analysis.	NeuraPath's authentication and role management system. QODIQA receives an authenticated actor ID; it does not perform authentication.
Apply the de-identification filter to the model's output.	The execution layer. QODIQA declared the restriction; the execution layer enforces it operationally.
Assess whether the AI diagnostic analysis was clinically appropriate or accurate.	Clinical governance. Outside QODIQA's scope entirely.

QODIQA DID NOT DO THIS	WHO IS RESPONSIBLE
Notify Dr. Ionescu or the care team of the consent revocation.	The application tier. QODIQA emits a DENY; what the interface does with it is the application's responsibility.
Verify the legal adequacy of NeuraPath's consent disclosure language.	Legal and compliance. QODIQA verifies structural validity and scope alignment of a consent record, not the human process that produced it.
Make a probabilistic or risk-based judgment about whether to allow the request despite revocation.	Nobody. This is not a permitted behavior. QODIQA is deterministic: revoked consent produces DENY, always, without exception or override.

The intentional minimalism of QODIQA's scope is a design property, not a limitation. A system that tries to do everything provides guarantees about nothing. By restricting itself to enforcement, and doing enforcement deterministically and verifiably, QODIQA makes it possible for every surrounding system to rely on exactly one thing: that if QODIQA said ALLOW, the request was authorized, and if it said DENY, execution did not proceed.

DOCUMENT STATUS

Document Status and Corpus Alignment

This document is a normative illustrative instrument within the QODIQA specification corpus. It does not introduce requirements; all requirements are defined in the Core Standard and the 68-Point Enforcement Framework. It illustrates how those requirements manifest in a complete, concrete deployment scenario, using real field values, specific timestamps, and actual artifact structures.

The scenario in this document, NeuraPath Health and patient PT-00441, is synthetic. All identifiers, timestamps, hash values, and actor names are constructed for illustrative purposes. Any resemblance to real persons, organizations, or systems is coincidental.

This document should be read together with the following related specifications:

- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement
- QODIQA: Implementation Conformance Checklist
- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement

— QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dițescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-WE-2026-001
TITLE	Worked Example: End-to-End Enforcement Scenario
SUBTITLE	A Complete Technical Narrative from Intent Declaration to Revocation in a Clinical Data Platform
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Normative Illustrative Document
DOCUMENT STATUS	Normative
GOVERNING AUTHORITY	QODIQA Governance Charter
SCENARIO	Synthetic · NeuraPath Health · patient:PT-00441 · 2026-04-07
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

PART IV

IV

Governance and legal instruments

5 documents

21 Governance Charter 754
22 Residual Risk and Assumptions 798
23 Certification Framework 835
24 Public Specification License 892
25 Positioning and Scope Limitation 923

CHAPTER 21 PART IV · GOVERNANCE AND LEGAL INSTRUMENTS

Governance Charter

Constitutional governance instrument establishing the authority, structure, and procedural discipline under which the technical standards defined by this corpus are maintained, amended, and evolved.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-GOV-CH-2026-001	2.0	2026	Governance Instrument	Institutional	NORMATIVE

Status note. No certification body or program exists yet. This is a conformance framework for self-assessment; any reference to certification or certified status describes a planned process, not an existing one.

PREAMBLE

This Charter constitutes the foundational governance instrument of the QODIQA standard corpus. It establishes the authority, structure, and procedural discipline under which the QODIQA technical standards are maintained, amended, and evolved. It is not a policy document, a product specification, or an organizational charter in the corporate sense. It is a **constitutional layer** governing the life cycle of a technical standard.

The QODIQA standard corpus addresses a structural problem in AI governance: the absence of deterministic, machine-enforceable consent at the moment of AI execution. The integrity of that architecture depends not only on its technical precision, but on the **governance discipline** that ensures it cannot be silently diluted, arbitrarily amended, or captured by interests incompatible with its foundational invariants. This Charter provides that discipline.

Version 2.0 of this Charter introduces hardening provisions across six dimensions: governance scalability under adoption growth, Standards Steward institutionalization guardrails, formal qualification of amendment proposers, voting record discipline, an emergency amendment protocol, and consolidation of the Core Invariant Protection Rule into a single canonical definition. No enforcement semantics, no foundational principles, and no corpus document authority structures have been modified. These are structural additions only.

Nothing in this Charter may be construed to alter, weaken, or reinterpret the technical enforcement semantics defined in the normative documents of the QODIQA corpus. Governance exists to **protect structural integrity** - not to substitute for it.

ARTICLE 01

Institutional Purpose

This Charter defines the governance framework for the QODIQA standard corpus. Its purpose is to establish the procedures, roles, and constraints under which QODIQA technical standards are created, amended, versioned, and retired. It does not define enforcement semantics, implementation requirements, or certification criteria. Those functions belong to the normative documents of the corpus.

1.1 Scope of Governance Authority

The governance authority established by this Charter applies to all documents designated as part of the QODIQA Standard Corpus, as defined in Article 3. It governs the processes by which those documents are modified, the roles authorized to participate in those processes, and the safeguards that preserve the technical integrity of the corpus across versions.

Governance authority does not extend to: the internal implementation decisions of conformant systems, the legal interpretation of regulatory obligations addressed by the corpus, the operational decisions of certified entities, or the commercial activities of any organization building upon the standard.

1.2 Relationship to Corpus Documents

TABLE 1.1 - GOVERNANCE RELATIONSHIP BY DOCUMENT CLASS

DOCUMENT	ROLE IN CORPUS	GOVERNANCE RELATIONSHIP
QODIQA Whitepaper	Conceptual foundation	Informative; governed as corpus record
68-Point Enforcement Framework	Primary normative standard	Normative; full amendment process applies
Core Standard	Formal specification	Normative; full amendment process applies
Certification Framework	Conformance criteria	Normative; certification oversight applies
Implementation Playbook	Guidance document	Informative; simplified amendment process
Regulatory Alignment Matrix	Compliance mapping	Informative; periodic review required
Security and Cryptographic Profile	Security specification	Normative; Cryptographic Advisory Panel review required
Conformance Test Suite	Verification specification	Normative; full amendment process applies
Governance Charter	This document	Constitutional; supermajority required for amendment

1.3 Boundary Between Governance and Technical Content

This Charter governs process and structure. It does not define, interpret, or constrain the technical content of the normative documents. Technical decisions - including enforcement semantics, control point definitions, cryptographic

requirements, and conformance criteria - are determined through the technical review processes established herein, and are recorded in the normative documents themselves.

In the event of apparent conflict between this Charter and a normative document, the Charter governs matters of process; the normative document governs matters of technical specification.

ARTICLE 02

Governance Principles

The following principles constitute the normative foundation of this Charter. All governance decisions, amendment proposals, and procedural interpretations shall be evaluated against these principles. No governance action may be taken that is demonstrably inconsistent with any principle defined herein.

PRINCIPLE I - DETERMINISTIC INTEGRITY — GP-01

The governance framework shall preserve the deterministic character of the QODIQA enforcement model. No amendment to any normative document may introduce non-deterministic behavior, probabilistic decision paths, or inference-based authorization logic into the core enforcement semantics. Any proposed change that introduces ambiguity in enforcement outcome shall be rejected. See canonical definition at **Rule CIPR-01** in Article 7.

Classification: Foundational - Non-waivable

PRINCIPLE II - FAIL-CLOSED PRESERVATION — GP-02

The fail-closed invariant is a core structural property of the QODIQA enforcement model. No governance action may weaken, conditionally suspend, or create exceptions to fail-closed behavior. Amendments that propose optional, configurable, or deployment-conditional fail-closed semantics shall not be ratified. This principle applies to all normative documents and is not subject to override by any governance body acting within its standard authority. See canonical definition at **Rule CIPR-01** in Article 7.

Classification: Foundational - Non-waivable

PRINCIPLE III - BACKWARD COMPATIBILITY DISCIPLINE GP-03

Amendment to normative documents shall distinguish between breaking and non-breaking changes and apply appropriate versioning accordingly. Breaking changes that alter conformance requirements, modify enforcement semantics, or invalidate existing certified implementations shall only be introduced through major version increments and must be accompanied by a formal migration path and transition period specification.

Classification: Structural - Supermajority required to override

PRINCIPLE IV - NON-ARBITRARY AMENDMENT GP-04

Every proposed amendment to a normative document must be justified by technical necessity, demonstrated conformance gap, regulatory evolution, or identified security risk. Amendments motivated solely by commercial interest, market positioning, or stakeholder preference without technical basis shall not be accepted into review. The amendment record shall document the technical justification for each ratified change.

Classification: Procedural - Enforced at proposal intake

PRINCIPLE V - TRANSPARENCY OF VERSIONING GP-05

All version transitions, amendment ratifications, and document state changes shall be recorded in a publicly accessible changelog. The complete version history of each corpus document shall be preserved and retrievable. No change to a normative document may be made without a corresponding, timestamped amendment record. Retroactive alteration of published version records is prohibited.

Classification: Procedural - Enforced at publication

PRINCIPLE VI - TECHNICAL NEUTRALITY GP-06

The QODIQA standard corpus shall not prescribe implementation technologies, specific software libraries, cryptographic primitives beyond what is necessary for security properties, or vendor-specific integration mechanisms except where technically necessary for deterministic enforcement. Governance decisions shall not favor or disfavor specific implementation approaches that satisfy the enforcement invariants.

Classification: Technical - Applied during review

PRINCIPLE VII - INDEPENDENCE FROM COMMERCIAL INTERESTS GP-07

The governance of the QODIQA standard corpus shall remain structurally independent from the commercial interests of any organization building products, services, or certification programs upon the standard. No governance role may be conditioned upon commercial contribution. Amendment proposals shall not be weighted by the commercial significance of the proposing party. Conflict-of-interest provisions in Article 6 shall be enforced without exception.

Classification: Structural - Enforced at role assignment and vote

ARTICLE 03

Standard Corpus Definition

The QODIQA Standard Corpus is the complete set of documents governed by this Charter. Corpus membership confers specific procedural obligations with respect to versioning, amendment, and archival. Corpus membership is established by formal designation and recorded in the Corpus Registry maintained by the Standards Steward.

3.1 Normative Documents

Normative documents define mandatory requirements. Conformant implementations must satisfy the requirements stated in normative documents. Amendments to normative documents require the full amendment process defined in Article 5.

The following document classes carry normative authority:

- The 68-Point Enforcement Framework, including all control point definitions and domain specifications
- The Core Standard, defining the enforcement model, consent primitives, and evaluation semantics
- The Certification Framework, defining conformance criteria and certification procedures
- The Security and Cryptographic Profile, defining mandatory cryptographic requirements
- The Conformance Test Suite, defining test methods and acceptance criteria
- This Governance Charter

3.2 Informative Documents

Informative documents provide context, guidance, and implementation assistance. They do not impose mandatory requirements. Conformance is assessed against normative documents only. Informative documents may be amended through a simplified process as defined in Article 5. The following document classes are informative:

- The QODIQA Whitepaper, providing conceptual foundations and problem framing
- The Implementation Playbook, providing integration guidance and reference patterns
- The Regulatory Alignment Matrix, providing mapping to external regulatory frameworks

- Technical Appendices designated as informative at publication

3.3 Experimental Drafts

Experimental drafts are working documents under active development. They carry no normative authority and are not subject to the full amendment process. Experimental drafts may be published for community review and technical feedback. An experimental draft may be advanced to candidate status by decision of the Technical Review Committee following review. Until formally advanced, experimental drafts shall be clearly marked as such and shall not be cited as normative references.

3.4 Corpus Registry

The Standards Steward shall maintain a Corpus Registry that records the current membership, document state, version, and amendment history of all corpus documents. The Corpus Registry shall be publicly accessible and updated within five business days of any state transition or version publication.

ARTICLE 04

Versioning Model

All normative documents in the QODIQA Standard Corpus shall be versioned according to the semantic versioning model defined in this Article. Version identifiers are permanent and immutable. A published version may not be altered; corrections require the issuance of a new version.

4.1 Semantic Versioning Rules

Version identifiers follow the format `MAJOR.MINOR.PATCH`, where each component is a non-negative integer. The following rules govern version increments:

MAJOR	Incremented when a change introduces breaking modifications to enforcement semantics, conformance requirements, or core invariants. A major version increment resets MINOR and PATCH to zero. A major version increment triggers re-certification obligations for certified implementations. Requires supermajority ratification and public review.
MINOR	Incremented when a change adds new requirements, extends existing control points, or introduces new optional elements in a backward-compatible manner. A minor version increment resets PATCH to zero. Certified implementations remain valid but must be assessed for gap coverage within the transition period.
PATCH	Incremented for editorial corrections, clarifications, and non-normative explanatory additions that do not alter requirements. A patch does not affect certification validity. Patch increments require Technical Review Committee approval only, without full amendment process.
Interim Patch	A temporary designation applied exclusively to emergency amendments issued under Article 5b. Carries the suffix `-ip` appended to the current PATCH component (e.g., `1.2.3-ip`). An Interim Patch version automatically enters Sunset Review upon publication and must be ratified or reverted within the period defined in Article 5b. Until sunset, it is treated as Ratified Standard for conformance purposes.

4.2 Version Lifecycle States

Each corpus document version exists in one of the following lifecycle states. State transitions are recorded in the Corpus Registry and announced via the publication process defined in Article 9.

Draft	Document is under active development by the Technical Review Committee. Not yet subject to public review. May not be cited as authoritative. May be modified without version increment during this state.
Candidate Standard	Document has passed Technical Review Committee approval and is in the public review window. Version identifier is frozen. May be cited with the explicit caveat of candidate status. Implementation testing may begin.
Ratified Standard	Document has completed public review, addressed material comments, and been ratified through the process defined in Article 5. The authoritative state for conformance assessment and certification.

Sunset Review	Applies exclusively to Interim Patch versions issued under Article 5b. The version is live and enforced but subject to mandatory full-process review and ratification within the defined sunset period. If not ratified within the sunset period, the version is automatically reverted to the immediately prior Ratified Standard.
Superseded	A newer ratified version of the document has been published. The superseded version remains valid for implementations certified against it until the applicable transition period expires. New certifications shall not be issued against a superseded version after the transition period ends.
Archived	Document version has been retired from active use. Archived versions are preserved in the public archive but carry no governance authority and shall not be used as basis for new certifications.

4.3 Deprecation Rules

Deprecation is the formal notice that a version, requirement, or document element will be removed or rendered non-normative in a future major version. Deprecation requires a minimum notice period of twelve months from announcement to removal. Deprecated elements shall be clearly marked in the document and listed in the deprecation registry. Deprecated elements retain their normative force until removal in the subsequent major version.

ARTICLE 05

Amendment Process

The amendment process governs all proposed changes to normative documents in the QODIQA Standard Corpus. The process is designed to ensure technical rigor, structural integrity, and transparent record-keeping while remaining practicable for the volume of changes expected during active standard development. A simplified process applies to informative documents and patch increments as noted. Emergency amendments are governed by Article 5b.

5.1 Qualified Party Definition

The following categories define parties eligible to submit amendment proposals to the full amendment process. Eligibility is determined at intake by the Standards Steward. Classification is competence-based; commercial affiliation does not

confer or deny eligibility.

QP-A	**Governance Body Member.** Any sitting member of the Technical Review Committee, Certification Oversight Panel, or Cryptographic Advisory Panel. Proposals from governance body members do not require a technical sponsor but are subject to mandatory recusal from the review and ratification vote on their own proposals.
QP-B	**Certified Entity Representative.** A technically authorized representative of an organization holding a current, valid certification against the QODIQA standard corpus. The certification must be active at the time of proposal submission. Certification lapse after submission does not void an accepted proposal already in review.
QP-C	**Technical Contributor with Sponsor.** Any individual or organization with demonstrable technical competence in the subject matter of the proposed amendment, accompanied by a written technical sponsorship from a current QP-A or QP-B party. The sponsor attests to the technical soundness of the proposal and takes responsibility for facilitating the proposer's engagement with the review process. Sponsorship does not constitute endorsement of the proposed outcome.
QP-D	**Authorized Certification Body.** Any certification body holding current authorization from the Certification Oversight Panel. Proposals from certification bodies are limited in scope to matters within the Certification Framework and Conformance Test Suite and are subject to the non-interference provisions of Article 8.

The Standards Steward shall publish the Qualified Party eligibility determination and any sponsorship record in the amendment intake register within five business days of intake disposition.

5.2 Full Amendment Process

The following process applies to all proposed changes to normative documents that result in MINOR or MAJOR version increments:

STEP 01	**Amendment Proposal Submission** A Qualified Party (QP-A through QP-D) submits an amendment proposal. The proposal must include: a precise statement of the proposed change; technical justification satisfying Principle GP-04; preliminary impact assessment covering determinism, fail-closed behavior, backward compatibility, and certification; proposer identity and category; conflict-of-interest disclosure; and, for QP-C proposers, the written technical sponsorship. Incomplete proposals are returned without review.

STEP 02	**Intake Review** The Standards Steward reviews the proposal for completeness, proposer eligibility, and structural compatibility with non-waivable principles (GP-01, GP-02) within ten business days. Proposals that are incomplete, submitted by ineligible parties, or in conflict with Rule CIPR-01 are rejected at intake with written explanation. Accepted proposals are assigned an amendment identifier, entered into the amendment register, and the intake determination is published.
STEP 03	**Technical Review Phase** The Technical Review Committee conducts substantive review within thirty business days. The Committee evaluates the proposal against all governance principles, performs the Change Impact Analysis defined in Article 7, and may request clarification, modification, or expert consultation. The Committee issues a recommendation: Accept, Accept with Modifications, or Reject, with written reasoning. The proposer's recusal from any vote on their own proposal is enforced at this stage.
STEP 04	**Public Review Window** Proposals accepted by the Technical Review Committee enter a public review window of no less than thirty calendar days. The proposal, impact analysis, and Committee recommendation are published in full. Written comments from any party are accepted and recorded. The Committee must address all substantive technical objections in writing. No response obligation applies to non-technical or procedural comments.
STEP 05	**CIPR-01 Clearance** Prior to ratification, the Technical Review Committee must issue a formal written clearance statement confirming that the proposed change does not trigger Rule CIPR-01 (defined in Article 7). This clearance is a hard prerequisite for ratification. If the analysis is inconclusive or produces a finding of impact, the proposal must be modified to eliminate the impact, or withdrawn. No ratification vote may proceed without a documented CIPR-01 clearance on the amendment record.
STEP 06	**Ratification Vote** Following public review and CIPR-01 clearance, the Technical Review Committee votes on ratification. Each member records a vote of Yes, No, or Abstain. Recusals are recorded separately and do not count toward quorum or voting totals. MINOR version increments require a simple majority of eligible (non-recused) voting members. MAJOR version increments require a supermajority of two-thirds of the sitting membership. Governance Charter amendments require supermajority ratification. The complete vote record - including member names, votes, recusals, and any dissenting technical opinion - is published as part of the amendment record.
STEP 07	**Publication and Changelog Issuance** Ratified amendments are incorporated into the new document version, which is published within fifteen business days of ratification. A corresponding changelog entry is issued simultaneously, recording the amendment identifier, description, impact classification, ratification date, effective date, and resulting document version. The prior version transitions to Superseded status. The complete amendment record - proposal, analysis, public comments, CIPR-01 clearance, and vote record - is archived and published.

5.3 Voting Record Requirements

For every ratification vote conducted under the full amendment process, the following must be recorded and published in the amendment register within fifteen business days of the vote:

Vote identifier	Amendment identifier, vote date, and quorum confirmation.
Individual votes	For each eligible voting member, their recorded vote (Yes / No / Abstain) by name; anonymous voting is not permitted for amendment ratification.
Recusal record	For each recused member, the basis for recusal and the conflict-of-interest disclosure filed.
Dissenting opinions	Any member voting No may attach a written dissenting technical opinion to the amendment record; this attachment is optional but, if provided, must be published in full alongside the ratification record.
Quorum determination	Confirmation that the vote satisfied quorum requirements, with the numerical basis stated.
Vote outcome	Final tally, threshold required, and whether the outcome was approval or rejection.

Vote records are immutable once published. Corrections to vote records may only be issued by the Conflict Resolution Body following a formal review, and any correction must reference the original record and the basis for correction.

5.4 Simplified Amendment Process

Patch increments to normative documents, and all amendments to informative documents, may be processed under the simplified amendment process: proposal submission with technical justification by any Qualified Party, Technical Review Committee approval without public review, and publication with changelog entry. Simplified process amendments may not alter normative requirements. A CIPR-01 clearance statement is required even under the simplified process.

5.5 Required Documentation for Amendment Proposals

Every amendment proposal submitted to the full process must include the following documentation. Proposals lacking any required element will be returned at intake:

Amendment Statement	Precise, unambiguous description of the proposed change, including specific document sections, control points, or requirements affected.
Technical Justification	Statement of the technical necessity, conformance gap, regulatory trigger, or security concern motivating the proposal.
Preliminary Impact Assessment	Initial analysis of determinism impact, fail-closed impact, backward compatibility, and certification scope.
Migration Implications	Description of impact on existing implementations and certified entities, including proposed transition provisions.
Proposer Identity and Category	Full name, affiliation, and Qualified Party category (QP-A through QP-D) of the proposing party; for QP-C, the written sponsorship document.
Conflict of Interest Disclosure	Disclosure of any commercial or institutional interest the proposing party holds in the outcome of the amendment.

ARTICLE 05B

Emergency Amendment Protocol

The Emergency Amendment Protocol provides a narrowly scoped accelerated path for addressing conditions that present immediate structural risk to the QODIQA corpus or its conformant implementations. It is not a general-purpose fast-track; it is a bounded exception to the standard amendment timeline. Its use does not waive any substantive requirement of the amendment process - it compresses the timeline and defers the full public review to a mandatory post-hoc phase.

> The Emergency Amendment Protocol **does not provide a mechanism to weaken deterministic or fail-closed invariants under any condition.** Rule CIPR-01 applies without modification to all emergency amendments. No emergency designation overrides the CIPR-01 clearance requirement.

5b.1 Eligible Triggering Conditions

The Emergency Amendment Protocol may be initiated only when one or more of the following conditions is confirmed by the Technical Review Committee:

EAP-T1	**Security Vulnerability.** A confirmed vulnerability in the normative corpus - or in a referenced cryptographic or protocol specification - that enables bypass, circumvention, or undermining of enforcement behavior in conformant implementations. The vulnerability must be concretely demonstrated or credibly disclosed through responsible disclosure procedures.
EAP-T2	**Cryptographic Break.** A demonstrated or authoritatively confirmed break in a cryptographic primitive, algorithm, or protocol referenced as mandatory in the Security and Cryptographic Profile, rendering the security properties of conformant implementations materially compromised.
EAP-T3	**Regulatory Mandate Requiring Immediate Conformity.** A binding regulatory instrument - enacted with legal force in a jurisdiction where conformant implementations operate - that creates a direct, time-constrained conflict with a specific normative requirement such that continued compliance with the current corpus creates legal liability for conformant implementations. The regulatory mandate must be specific, binding, and the conflict must be irresolvable through implementation-level measures alone.
EAP-T4	**Deterministic Model Integrity Risk.** A confirmed structural defect in the normative corpus - distinct from enforcement semantics - that causes deterministic evaluation to produce incorrect outcomes under defined conditions, provided the defect was introduced by a prior amendment rather than present in the original corpus design. This trigger is not available for challenges to the foundational enforcement model.

Conditions that do not satisfy one of the above trigger types shall not be processed under this Protocol. Urgency of commercial need, implementation convenience, or regulatory interpretation differences do not constitute eligible triggering conditions.

5b.2 Emergency Amendment Procedure

STEP 01	**Emergency Declaration** A QP-A or QP-B party submits a written emergency declaration to the Standards Steward, specifying the trigger type, the concrete basis for the trigger, the affected corpus document and section, and the proposed resolution. The Standards Steward convenes the Technical Review Committee for an emergency eligibility determination within three business days.
STEP 02	**Eligibility Confirmation** The Technical Review Committee confirms or denies emergency eligibility by simple majority vote within three business days of convening. A denial requires written reasoning. An approval initiates the accelerated review window and is published in the amendment register immediately.
STEP 03	**Accelerated Technical Review** Upon eligibility confirmation, the Technical Review Committee conducts full technical review, including CIPR-01 clearance, within fifteen business days. The Cryptographic Advisory Panel is convened in parallel for EAP-T1 and EAP-T2 triggers. The public review window is deferred but not eliminated; it proceeds as a mandatory post-hoc phase defined in Step 05.
STEP 04	**Interim Patch Publication** If the Technical Review Committee approves the amendment under accelerated review, the change is published as an Interim Patch version (designated with the `-ip` suffix per Article 4.1). The Interim Patch enters Sunset Review immediately upon publication. The complete vote record is published simultaneously.
STEP 05	**Mandatory Post-Hoc Full Review** Following publication of the Interim Patch, the full amendment process - from public review window onward - must be completed within ninety calendar days. During this period, the Interim Patch is in effect. If the post-hoc review results in ratification, the version designation transitions from Interim Patch to Ratified Standard (dropping the `-ip` suffix). If the post-hoc review results in rejection or modification, the version is reverted to the immediately prior Ratified Standard, and a new proposal reflecting any modifications must restart from Step 01 of the full amendment process.
STEP 06	**Automatic Sunset** If the mandatory post-hoc full review is not completed within the ninety-calendar-day sunset period, the Interim Patch is automatically reverted to Superseded status and the immediately prior Ratified Standard is reinstated as authoritative. This reversion requires no further vote and takes effect by operation of this Charter. The Standards Steward shall publish the reversion notice within three business days of the sunset date.

5b.3 Emergency Protocol Constraints

- No more than two Interim Patch versions of any single corpus document may be active simultaneously
- An Interim Patch may not itself be amended through the Emergency Amendment Protocol; it may only be superseded by completion of its own post-hoc full review
- Emergency Protocol usage frequency shall be reviewed annually by the Technical Review Committee; patterns of repeated use may indicate underlying structural deficiencies requiring a formal MINOR amendment
- All emergency declarations, eligibility determinations, and Interim Patch publications are recorded in the amendment register and remain publicly accessible without expiration

ARTICLE 06

Governance Structure

The governance of the QODIQA Standard Corpus is conducted through a set of defined institutional roles. The structure is designed to be functional with a minimal composition and scalable as the standard matures in accordance with the evolution thresholds defined in Article 11. Roles are defined by function, not by organizational affiliation. All role holders are bound by the conflict-of-interest provisions of this Article.

6.1 Institutional Roles

ROLE-01

Standards Steward

The primary governance authority for the QODIQA corpus during the Foundational Phase. Responsible for corpus registry maintenance, proposal intake, amendment publication, and institutional continuity. The Standards Steward holds no veto over Technical Review Committee decisions on technical matters but retains authority over procedural matters and dissolution provisions. Transition of the Steward role is governed by Article 6b.

ROLE-02

Technical Review Committee

The body responsible for substantive technical review of all amendment proposals. Conducts impact analysis, issues recommendations, manages public review windows, and votes on ratification. Must consist of qualified technical reviewers with demonstrated competence in consent systems, distributed enforcement, or applicable regulatory frameworks. Composition evolves in accordance with Article 11.

ROLE-03

Certification Oversight Panel

Responsible for supervision of certification body activities, review of certification revocation decisions, and maintenance of the certified implementation registry. Operates independently of the Technical Review Committee on matters of certification integrity. Reports to the Standards Steward during the Foundational Phase; reporting structure evolves per Article 11.

ROLE-04

Cryptographic Advisory Panel

Advisory body with standing to review all proposed amendments affecting the Security and Cryptographic Profile. Issues non-binding recommendations on cryptographic adequacy and algorithm lifecycle. Mandatory consultation for EAP-T1 and EAP-T2 emergency triggers. Convened on demand during the Foundational Phase; constituted as a standing body at Tier 2 per Article 11.

ROLE-05

Conflict Resolution Body

Convened on an ad hoc basis to resolve disputes arising from amendment process disputes, recusal decisions, or interpretation of this Charter. Composed of parties with no direct interest in the matter under dispute. Issues binding procedural rulings; does not rule on technical merit.

6.2 Composition, Voting, and Quorum

The Technical Review Committee shall consist of a minimum of three and a maximum of nine members during the Foundational Phase. Composition expands in accordance with Article 11. A quorum for any vote requires the participation of a majority of eligible (non-recused) sitting Committee membership. Decisions on MINOR amendments require a simple majority of quorum. Decisions on MAJOR amendments require a supermajority of two-thirds of the sitting membership, including absent members.

6.3 Recusal Requirements

Any governance role holder must recuse themselves from review, deliberation, or voting on any amendment where they hold a direct commercial or institutional interest in the outcome. Recusal is mandatory and self-enforcing; failure to recuse constitutes a governance violation. The Conflict Resolution Body may compel retroactive recusal review where conflicts are identified after the fact. Recusals are recorded in the amendment register and published as part of the vote record.

6.4 Term Limits and Continuity

Technical Review Committee members serve terms of two years, renewable once without interruption. After two consecutive terms, a minimum one-year gap is required before re-appointment. The Standards Steward role is subject to institutionalization provisions defined in Article 6b. Succession provisions for involuntary governance disruption are addressed in Article 12.

ARTICLE 06B

Standards Steward Institutionalization

The Standards Steward role in the Foundational Phase is necessarily concentrated. This concentration is intentional: it provides the decisional clarity required during initial corpus establishment and before governance institutions are sufficiently populated to sustain distributed authority. This Article defines the structural guardrails that prevent indefinite centralization and the transition path to a ratified or elected Steward structure under defined adoption conditions.

6b.1 Foundational Phase Steward Authority

During the Foundational Phase (Tier 1 as defined in Article 11), the Standards Steward exercises the procedural authorities defined in this Charter. Those authorities are bounded: the Steward may not unilaterally ratify amendments, modify normative documents, alter version designations, or override Technical Review Committee decisions. The Steward's role is custodial and procedural. Technical authority rests with the Technical Review Committee.

6b.2 Mandatory Transition Triggers

Upon reaching Tier 2 adoption thresholds defined in Article 11, the Standards Steward function must transition from a single-holder role to a Steward Council structure within twelve months of threshold confirmation. The Steward Council consists of three to five members, none of whom may hold the same organizational affiliation. The founding Steward becomes a Council member by default and may serve no more than one additional two-year Council term before mandatory rotation.

Upon reaching Tier 3 adoption thresholds, the Steward Council must transition to an elected structure within eighteen months of threshold confirmation. *This is a planned arrangement: no Steward Council has been constituted yet.* Elections are conducted by the Technical Review Committee membership, applying the voting record requirements of Article 5.3.

The elected structure must preserve the conflict-of-interest provisions and commercial neutrality requirements of this Charter without diminution.

6b.3 Steward Removal Procedure

The Standards Steward - or any Steward Council member - may be removed from the role under the following conditions:

BASIS	**Grounds for Removal** Demonstrated and documented: (a) failure to discharge governance obligations for a continuous period exceeding ninety days without prior notification; (b) material conflict-of-interest violation not remediated within thirty days of formal notice; or (c) governance action that is demonstrably inconsistent with the non-waivable principles of this Charter (GP-01, GP-02).
PROCESS	**Removal Initiation** Removal may be initiated by a two-thirds majority vote of the Technical Review Committee. The subject of removal is notified in writing and provided thirty days to respond prior to a final vote. The Conflict Resolution Body reviews the grounds and the response and issues a non-binding procedural finding before the final vote proceeds.
VOTE	**Ratification of Removal** Removal is ratified by a two-thirds supermajority vote of the sitting Technical Review Committee membership, applied to the full membership (not quorum). The vote record is published. The removed Steward's governance actions prior to removal retain their validity; removal is not retroactive.
SUCCESSION	**Interim and Successor Steward** Upon removal, the Technical Review Committee appoints an interim Steward from its sitting membership within fifteen business days. The interim Steward serves until a permanent successor is ratified by the Committee by simple majority vote, within ninety days of the removal. Governance continuity is maintained without interruption during this period.

6b.4 Continuity of Governance During Transition

No Steward transition - whether triggered by Article 11 adoption thresholds, voluntary withdrawal, or removal under Section 6b.3 - shall suspend or interrupt the amendment process for any proposal currently in review. All in-progress proposals carry forward to the successor structure without prejudice to their position in the process.

ARTICLE 07

Change Impact Analysis Requirements

Every proposed amendment entering the full amendment process must be accompanied by a Change Impact Analysis. The Change Impact Analysis is a formal technical document, not a summary or narrative approximation. The Technical Review Committee is responsible for verifying the completeness and accuracy of the analysis. Where the proposing party has produced an initial analysis, the Committee must independently validate it.

> **No amendment may proceed to ratification unless all required Change Impact Analysis components have been completed and independently verified.** An incomplete or contested analysis constitutes grounds for suspension of the amendment process pending resolution.

7.1 Required Analysis Components

Determinism Impact Statement	Analysis of whether the proposed change alters the deterministic character of any enforcement decision, evaluation path, or consent resolution outcome. Must conclude with an explicit finding: No Determinism Impact, or Determinism Impact Present with description.
Fail-Closed Integrity Impact Statement	Analysis of whether the proposed change alters, weakens, or creates exceptions to fail-closed enforcement behavior under any execution condition. Must conclude with an explicit finding: No Fail-Closed Impact, or Fail-Closed Impact Present with description.
Backward Compatibility Assessment	Identification of all existing certified implementations, conformant deployments, or normative references materially affected by the proposed change. Must classify the change as Non-Breaking, Conditionally Breaking, or Breaking, with justification.
Certification Impact Analysis	Determination of whether the proposed change requires existing certifications to be renewed, re-assessed, or revoked. Must specify the certification scope affected and the proposed re-certification timeline.

Interoperability Assessment	Evaluation of whether the proposed change affects the interoperability of conformant implementations with other systems, registries, or external references within the QODIQA ecosystem.
Migration Implications	Specification of the implementation changes required of conformant systems, estimated migration complexity, and the proposed transition period.

7.2 Core Invariant Protection Rule - Canonical Definition

CORE INVARIANT PROTECTION RULE — CIPR-01

No amendment to any QODIQA corpus document may be ratified if its Change Impact Analysis produces a finding of Determinism Impact Present or Fail-Closed Impact Present.

This rule operates as an absolute, non-delegable constraint on the amendment process. It applies without exception to: the full amendment process (Article 5), the simplified amendment process (Article 5.4), the Emergency Amendment Protocol (Article 5b), and any governance evolution transition (Article 11). No governance body, governance phase, adoption threshold, or emergency condition provides authority to waive, defer, or override Rule CIPR-01.

The only path to modifying the deterministic or fail-closed invariants of the corpus is through a MAJOR version increment subject to a supermajority vote of two-thirds of the sitting Technical Review Committee, a public review period of no less than ninety calendar days, and a re-ratification process beginning from a new Candidate Standard designation. That path does not constitute an exception to Rule CIPR-01; it constitutes a complete replacement of the current major version of the standard, for which new invariant definitions must themselves be internally consistent and fail-closed.

Status: Canonical Definition - Non-waivable - Cross-referenced from Articles 2 (GP-01, GP-02), 5.2 (Step 05), 5.4, 5b, 10.3, 10.4

ARTICLE 08

Certification Oversight

The Certification Framework governs the assessment of implementation conformance with the QODIQA normative standard. Governance oversight of certification ensures that the certification process faithfully reflects the requirements

of the current ratified standard and that certified status is not issued or retained by implementations that fail to satisfy those requirements.

8.1 Evolution of the Certification Framework

The Certification Framework is a normative document subject to the full amendment process. Amendments to the Certification Framework that alter conformance criteria, add new assessment categories, or change certification validity periods require full Technical Review Committee review. The Certification Oversight Panel holds standing as a QP-D qualified party to submit proposed amendments to the Certification Framework and to participate in their review as a subject matter authority.

8.2 Supervision of Certification Bodies

The Certification Oversight Panel is responsible for maintaining the registry of authorized certification bodies. Authorization of a certification body requires demonstration of: independence from the implementation under assessment; technical competence in the QODIQA enforcement model; procedural capacity to execute the Conformance Test Suite; and commitment to public reporting of certification outcomes. Authorization may be withdrawn for demonstrated failure in any of these areas.

8.3 Certification Revocation

Certification revocation may be initiated by the Certification Oversight Panel upon: evidence that a certified implementation fails to satisfy the requirements of the normative standard against which it was certified; a material version transition that renders the certification basis superseded beyond the applicable transition period; or demonstrated misrepresentation in the certification assessment process. Revocation decisions are subject to review by the Conflict Resolution Body upon request of the affected implementation holder. Revocations are published in the Corpus Registry.

8.4 Conflict of Interest in Certification

Certification bodies may not assess implementations in which they hold a direct commercial interest. Governance role holders who participate in the assessment of a specific implementation's certification application must recuse themselves from any governance decision affecting that certification. The Certification Oversight Panel must review and publish its conflict-of-interest policy on an annual basis.

8.5 Non-Interference with Enforcement Semantics

The Certification Oversight Panel and all certification bodies operate within the enforcement semantics defined by the normative standard. Neither the Panel nor any certification body holds authority to reinterpret, relax, or supplement the enforcement requirements of the standard in the course of certification activities. Ambiguities in the standard are to be referred to the Technical Review Committee for clarification through the patch amendment process, not resolved unilaterally by certification bodies.

ARTICLE 09

Transparency and Publication Requirements

The governance of the QODIQA Standard Corpus shall be conducted transparently. All material governance actions, amendment records, and version transitions shall be accessible to the public. Transparency is not merely a procedural aspiration; it is a structural requirement that enables the corpus to function as a trustworthy technical standard.

9.1 Public Changelog Requirements

A versioned changelog shall be maintained for each normative corpus document and for the Corpus Registry. Each changelog entry must record: the amendment identifier, a concise description of the change, the impact classification,

the ratification date, the effective date, and the resulting document version. Changelog entries are immutable once published. Corrections to changelog entries are issued as separate entries referencing the original.

9.2 Version Archive Retention

All published versions of normative corpus documents - including Interim Patch versions, whether ratified or reverted - shall be preserved in a publicly accessible archive without expiration. Archived versions shall be clearly marked with their lifecycle state, the date of state transition, and, for reverted Interim Patches, the reversion notice. The archive shall be maintained in a format that does not require specialized software to access. Archive responsibilities transfer with governance custody in accordance with Article 12.

9.3 Disclosure of Amendments

The complete amendment record for each ratified amendment - including the original proposal, proposer identity and Qualified Party category, sponsorship record if applicable, the Technical Review Committee's analysis and recommendation, all material public comments received, the CIPR-01 clearance statement, and the vote record as defined in Article 5.3 - shall be published no later than fifteen business days following ratification. Rejected proposals and their reasons for rejection shall also be published with sufficient technical detail.

9.4 Registry Publication Format

The Corpus Registry shall be published in a machine-readable format in addition to a human-readable representation. The registry shall include: the current list of corpus documents with their version, state, and document identifier; the list of ratified amendments and Interim Patches; the list of active certifications and certification bodies; the list of current governance role holders; and the current governance phase (Tier designation per Article 11). The registry shall be updated within five business days of any change.

9.5 Historical Version Accessibility

Superseded and archived versions of corpus documents shall remain accessible at stable, permanent identifiers. The Standards Steward shall maintain or designate a preservation host for historical versions. Governance custody transfers under Article 12 include explicit obligation to maintain or transfer historical version accessibility.

ARTICLE 10

Safeguards Against Capture

The structural integrity of the QODIQA standard corpus depends on its independence from capture by interests incompatible with its foundational purpose. This Article defines the structural protections that prevent the standard from being diluted, redirected, or rendered technically incoherent through amendment processes that serve interests other than the integrity of deterministic runtime consent enforcement.

10.1 Commercial Neutrality Clause

No amendment to any corpus document may be conditioned upon, funded by, or structured to benefit the commercial interests of any specific organization, product, or implementation. Amendments motivated by market positioning, competitive advantage, or commercial differentiation - rather than technical necessity - shall not be accepted at intake. The Standards Steward shall enforce this clause at the proposal intake stage and may request additional disclosure from proposing parties where commercial motivation is suspected.

10.2 Amendment Anti-Dilution Protections

The following amendment patterns are structurally prohibited and shall be rejected at intake regardless of their technical framing:

- Amendments that introduce optional, configurable, or deployment-conditional enforcement semantics into requirements that are currently mandatory
- Amendments that add "best effort," "reasonable," or probabilistic qualifiers to requirements that currently impose deterministic obligations
- Amendments that narrow the scope of the fail-closed invariant by defining categories of execution to which it does not apply
- Amendments that substitute documentation obligations for enforcement obligations in normative requirements
- Amendments that re-classify normative requirements as informative guidance without a major version increment

10.3 Core Invariant Protection

The canonical definition of core invariant protection is Rule CIPR-01, defined in Article 7.2. No governance body constituted under this Charter holds authority to weaken the deterministic or fail-closed invariants through any ordinary amendment path. This constraint is absolute and applies across all governance phases. Cross-reference: **Rule CIPR-01**.

10.4 Supermajority Requirement for Core Changes

The path to modifying the deterministic or fail-closed invariants is defined in Rule CIPR-01 (Article 7.2). That path requires: a MAJOR version increment; a supermajority vote of two-thirds of the sitting Technical Review Committee; a full public review period of no less than ninety calendar days; and a re-ratification process beginning from a new Candidate Standard designation. No governance phase, emergency condition, or adoption threshold alters or waives these requirements.

ARTICLE 11

Governance Evolution and Scaling Mechanism

The governance structure of the QODIQA Standard Corpus is designed to scale with adoption. The scaling mechanism is trigger-based: defined, measurable adoption thresholds initiate transitions between governance phases. These transitions expand governance capacity and distribute oversight authority. They do not modify enforcement invariants, alter the amendment process, or weaken the safeguards defined in Articles 7 and 10.

> Governance phase transitions are structural changes only. **No transition between governance phases modifies enforcement semantics, changes Rule CIPR-01, or alters the voting thresholds required for MAJOR amendments or core invariant changes.** Those requirements are invariant across all phases.

11.1 Governance Phase Definitions

Tier 1 - Foundational	**Initial Phase.** Single Standards Steward. Technical Review Committee of three to nine members. Cryptographic Advisory Panel convened on demand. Certification Oversight Panel active. This Charter enters force at Tier 1 and remains at Tier 1 until Tier 2 thresholds are confirmed. All provisions of this Charter apply in full from Tier 1 inception.
Tier 2 - Multi-Member	**Distributed Institutional Phase.** Standards Steward transitions to Steward Council (three to five members, no two of the same organizational affiliation). Technical Review Committee expands to a minimum of seven members with geographic and organizational diversity requirements as defined by the Committee itself at the time of expansion. Cryptographic Advisory Panel constituted as a standing body. Initiated upon confirmation of Tier 2 thresholds.
Tier 3 - Distributed Oversight	**Full Distributed Governance Phase.** Steward Council transitions to an elected structure per Article 6b.2. Technical Review Committee expands to a minimum of eleven members with mandatory multi-jurisdictional representation. Independent standing committees for amendment review, certification oversight, and cryptographic advisory are established as separate bodies with defined interaction protocols. Initiated upon confirmation of Tier 3 thresholds.

11.2 Adoption Threshold Definitions

Tier transitions are triggered when all thresholds for the target tier are confirmed by the Technical Review Committee and recorded in the Corpus Registry. Confirmation requires independent verification by the Certification Oversight Panel of the certification and certification body counts.

TABLE 11.1 - ADOPTION THRESHOLDS FOR TIER TRANSITIONS

THRESHOLD METRIC	TIER 2 TRIGGER VALUE	TIER 3 TRIGGER VALUE
Active certified implementations	10 or more	50 or more
Independent authorized certification bodies	3 or more, no two sharing organizational affiliation	7 or more, operating across at least 3 distinct jurisdictions
Jurisdictions with active conformant deployments	3 or more	10 or more
Qualified Party amendment proposers (distinct organizations, trailing 24 months)	5 or more	15 or more

Thresholds are evaluated on a rolling basis. Once confirmed, a Tier transition may not be reversed by a subsequent decline in adoption metrics. The governance structure advances but does not regress.

11.3 Technical Review Committee Composition Under Scaling

At each tier transition, the Technical Review Committee is responsible for defining and publishing its own expanded composition criteria within the constraints specified in Article 11.1. These criteria must address: minimum member count, organizational diversity (no single organization may hold more than one-third of seats at Tier 2 and one-quarter at Tier 3), technical competency requirements for new members, and the nomination and appointment procedure. Composition criteria are published as an informative addendum to the Corpus Registry and do not require a full amendment process to update, provided the update does not reduce the diversity or competency requirements below the minimum thresholds defined in this Article.

11.4 Invariance of Enforcement Structure Across Phases

GOVERNANCE PHASE INVARIANCE CLAUSE	GPI-01

No governance phase transition - whether from Tier 1 to Tier 2, Tier 2 to Tier 3, or any intermediate structural change - may alter the following: the enforcement invariants defined in Principles GP-01 and GP-02; the canonical Rule CIPR-01; the supermajority thresholds for MAJOR amendments and core invariant changes; the amendment anti-dilution protections of Article 10.2; or the dissolution and transfer provisions of Article 12. These elements are phase-invariant by constitutional design.

Classification: Constitutional - Non-waivable across all governance phases

ARTICLE 12

Dissolution and Transfer Provisions

The long-term institutional continuity of the QODIQA Standard Corpus requires explicit provisions for the event that the current governance entity is dissolved, becomes inactive, or elects to transfer governance custody. These provisions ensure that the corpus remains accessible, intact, and governed under consistent principles regardless of changes in governance structure.

12.1 Dissolution of the Governance Entity

In the event that the Standards Steward or Steward Council becomes unable or unwilling to discharge governance responsibilities - whether through organizational dissolution, voluntary withdrawal, legal incapacity, or demonstrated abandonment of governance obligations - the following provisions apply in order:

STEP 01	**Dissolution Notice** The Standards Steward or their legal representative shall provide no less than ninety days written notice of impending dissolution or transfer. Where involuntary dissolution occurs without advance notice, the Technical Review Committee assumes interim governance authority for a period not to exceed one hundred eighty days.
STEP 02	**Corpus Archive Publication** Prior to dissolution, the complete corpus archive - including all historical versions, Interim Patch records, the amendment register, and the Corpus Registry - must be deposited with a qualified public archive, open source foundation, or successor governance entity. Publication in a durable, publicly accessible repository satisfies this obligation.
STEP 03	**Custody Transfer** Governance custody may be transferred to a qualified successor entity that accepts the obligations of this Charter in writing. Transfer does not permit retroactive amendment of existing ratified versions. The successor entity inherits the amendment register and Corpus Registry and must maintain public accessibility of all historical materials.
STEP 04	**Public Archival Requirement** If no qualified successor entity accepts governance custody within one hundred eighty days of dissolution, the complete corpus archive shall be published under an open access license that permits unrestricted reading and redistribution of the standard text. This publication satisfies the minimum continuity obligation and does not convey amendment authority to any party.

12.2 Preservation of Document Integrity

Custody transfer does not grant the successor entity authority to retroactively alter, re-version, or re-classify any previously ratified document. All ratified versions retain their version identifiers and lifecycle states. The successor entity may issue new amendments following the processes defined in this Charter, but may not modify the amendment history or changelog records of predecessor versions.

12.3 Continuity of Certification

Upon governance transfer, all valid certifications issued under the prior governance structure remain valid for the duration of their applicable transition periods. The successor governance entity assumes the Certification Oversight Panel function and inherits the registry of authorized certification bodies and certified implementations. Existing

certification bodies are not required to re-authorize following a custody transfer unless the successor entity determines that their practices are inconsistent with the current Certification Framework.

12.4 Governance Phase Continuity on Transfer

Custody transfer does not reset the governance phase. A successor entity that accepts governance custody of a corpus in Tier 2 or Tier 3 assumes the obligations of that tier, including composition requirements, Steward Council structure, and elected structure obligations. The successor may not retract to a prior tier's governance structure.

ARTICLE 13

Institutional Closing Declaration

The QODIQA Governance Charter exists for one purpose: to ensure that the technical standard it governs remains capable of doing what it was designed to do - provide deterministic, machine-enforceable consent at the moment of AI execution. Governance is not an end in itself. It is a structural condition for the trustworthiness of the standard over time.

> The authority of this standard derives from the technical invariants it preserves - not from the institutions that administer it, not from regulatory endorsement, and not from market adoption. If those invariants are intact, the standard retains its meaning. If they are compromised, no governance structure can restore it.

13.1 The Purpose of Governance

Governance exists to preserve deterministic runtime consent enforcement across the life of the standard. It does this by ensuring that changes to the corpus are technically justified, structurally sound, and transparent. It does not do this by

introducing policy ideology, by resolving societal questions about AI governance through the amendment process, or by making the standard an instrument of any commercial, political, or institutional agenda.

The governance structure defined in this Charter is minimal by design at inception and scales by threshold rather than by preference. At no tier does it acquire authority over third parties, legal standing as a regulatory body, or the capacity to modify enforcement semantics. Its authority is entirely constituted by the commitment of participants to abide by its procedures.

13.2 Governance Exists to Protect Structural Integrity

Every provision of this Charter - the non-waivable principles, the change impact requirements, the supermajority thresholds, the anti-capture safeguards, the emergency protocol constraints, the Steward institutionalization guardrails, the qualified party eligibility structure, and the governance phase invariance clause - reflects a single underlying commitment: that the structural properties of the QODIQA enforcement model cannot be eroded through well-intentioned procedural exceptions, commercially motivated amendments, or the accumulated pressure of stakeholder preferences. Governance exists to make that erosion structurally difficult, not merely discouraged.

> The determinism and fail-closed invariants of the QODIQA standard are not configurable properties. **They are the standard.** Their preservation is the condition under which QODIQA remains a trustworthy basis for AI authorization infrastructure. This Charter is the instrument by which that condition is maintained - across all adoption scales, governance phases, and institutional transitions.

13.3 Authority Derived from Technical Invariants

QODIQA's authority as a technical standard derives from the precision, consistency, and verifiability of its enforcement model - not from institutional prestige, regulatory mandate, or market position. Governance cannot manufacture that authority; it can only protect it. The obligation of every governance role holder - at every tier, in every phase - is to act in the interest of the standard's structural integrity, and to subordinate their institutional, commercial, or personal preferences to that obligation without exception.

This Charter is the constitutional layer of the QODIQA corpus. It does not expire, does not require periodic renewal, and does not become optional as the standard matures or as governance scales. It governs until a duly constituted successor instrument is ratified through the process it defines.

DOCUMENT STATUS

Document Status and Classification

This document is the Governance Charter of the QODIQA specification corpus. It constitutes the constitutional governance instrument establishing the authority, procedural structure, and constraints under which all QODIQA technical standards are maintained, amended, versioned, and retired. It is issued as a Normative Constitutional Governance Instrument and is not a product specification, compliance framework, or organizational charter in the corporate sense.

Version 2.0 supersedes QODIQA-GOV-CH-2026-001 v1.0. All corpus documents governed by v1.0 are governed by this Charter from its effective date. The amendment register, Corpus Registry, and all ratified versions produced under v1.0 governance retain their validity and are incorporated by reference into the v2.0 governance structure without re-ratification.

Proposed amendments to this Charter must be submitted by a Qualified Party in accordance with the full amendment process defined in Article 5 and require supermajority ratification as defined in Article 10. This Charter does not sunset and does not require periodic reaffirmation.

This document should be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0

- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0
- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Implementation Playbook for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Regulatory Alignment Matrix for Deterministic Runtime Consent Enforcement, Version 1.0

Version 2.0 represents the second formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dușescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-GOV-CH-2026-001
TITLE	Governance Charter for the QODIQA Standard Corpus
SUBTITLE	Institutional Governance and Amendment Framework for the Standard Corpus
PUBLICATION DATE	2026
VERSION	2.0
PRIOR VERSION	1.0 - Superseded by this document
DOCUMENT TYPE	Normative Constitutional Governance Instrument
DOCUMENT STATUS	Normative - Constitutional layer; Active - Ratified Standard
LIFECYCLE STATE	Ratified Standard - Foundational Instrument
CURRENT GOVERNANCE PHASE	Tier 1 - Foundational Phase
AMENDMENT AUTHORITY	Supermajority - Technical Review Committee
STANDARDS STEWARD	Bogdan Duțescu
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

INSTITUTIONAL

Scope

This document constitutes the constitutional governance instrument for the QODIQA Standard Corpus. It defines the authority, procedural structure, and constraints under which all QODIQA technical standards (including the QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0, the QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0, and associated normative documents) are maintained, amended, versioned, and retired.

This Charter does not define enforcement semantics, implementation requirements, certification criteria, or organizational obligations beyond the governance of the standard corpus. It is not a policy document, compliance framework, or product specification. Organizations seeking normative implementation guidance should refer to the QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0 and the QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0. This Charter governs the processes by which those documents evolve.

INSTITUTIONAL

Terminology

The following definitions apply to terms as used in this Charter. For normative terminology applicable to the full QODIQA standard corpus, refer to the QODIQA: Terminology and Normative Definitions, Version 1.0 document.

`Governance Charter`
This document. The constitutional governance instrument establishing authority structures, amendment procedures, and structural safeguards for the QODIQA Standard Corpus. Distinguished from policy documents by its constitutional authority over the amendment process itself.

`Standard Corpus`
The complete set of documents governed by this Charter, as recorded in the Corpus Registry. Includes normative, informative, and experimental documents designated for corpus membership by the Standards Steward.

`Amendment Record`
The complete set of documentation associated with a single amendment, including the original proposal, impact analysis, public comments, CIPR-01 clearance statement, and vote record. Amendment records are permanently published and immutable.

`Rule CIPR-01`
The Core Invariant Protection Rule. An absolute, non-delegable constraint that prevents ratification of any amendment whose Change Impact Analysis finds Determinism Impact Present or Fail-Closed Impact Present. Canonical definition in Article 7.2.

`Qualified Party`
A party eligible to submit amendment proposals under this Charter. Four categories are defined (QP-A through QP-D), based on governance body membership, certified entity representation, technical contribution with sponsorship, and authorized certification body status. Eligibility is determined at proposal intake.

`Governance Phase`
One of three structural configurations of the governance apparatus (Tier 1 Foundational, Tier 2 Multi-Member, Tier 3 Distributed Oversight), triggered by adoption thresholds defined in Article 11. Phase transitions expand governance capacity without modifying enforcement invariants.

INSTITUTIONAL

Conformance Language

The key words **MUST**, **MUST NOT**, **SHOULD**, **SHOULD NOT**, and **MAY** in the QODIQA specification corpus indicate requirement levels for conformant implementations of the QODIQA enforcement architecture.

In this Governance Charter, such terms describe procedural obligations on governance role holders and the amendment process itself. They do not describe implementation requirements for conformant AI systems. For normative implementation requirement definitions, refer to the QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0.

INSTITUTIONAL

Standard Corpus

This document is part of the QODIQA standard corpus - a structured set of technical specifications, analytical papers, governance instruments, and implementation resources maintained by QODIQA Standards and Research. Documents in the corpus are cross-referenced and versioned under the QODIQA: Corpus Change Log and Version History, Version 1.0.

Governance Note

This Charter constitutes the constitutional layer of the corpus. All amendments to corpus documents are subject to the procedures defined herein. The authority structure, amendment process, and invariant protection rules of this Charter apply to all documents listed below.

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0

- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0
- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Regulatory Alignment Matrix for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Threat Model and Abuse Case Specification, Version 1.0
- QODIQA: Implementation Playbook for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Economic Impact Analysis of Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Terminology and Normative Definitions, Version 1.0
- QODIQA: Interoperability and Deployment Constraints, Version 1.0
- QODIQA: Public Specification License, Version 1.0
- QODIQA: Positioning and Scope Limitation Statement, Version 1.0
- QODIQA: Residual Risk and Assumption Disclosure Annex, Version 1.0

— QODIQA: Use Case Dossiers for Runtime Consent Enforcement Deployments, Version 1.0

INSTITUTIONAL

Institutional Disclaimer

> This document constitutes the constitutional governance instrument of the QODIQA standard corpus. It does not constitute regulatory guidance, legal advice, financial recommendation, or investment analysis of any kind. The procedural obligations defined herein apply to governance role holders and amendment participants. They do not create direct legal obligations for third parties operating conformant implementations.

Nothing in this Charter alters, supplements, or overrides the applicable law of any jurisdiction with respect to data protection, AI regulation, or related obligations. Organizations implementing QODIQA-conformant infrastructure should obtain qualified legal counsel for jurisdiction-specific compliance determinations.

QODIQA Standards and Research does not accept liability for governance decisions or amendment outcomes that result from processes conducted under this Charter. The Charter provides procedural structure; it does not guarantee outcomes or confer legal standing to any party.

INSTITUTIONAL

Recommended Citation

CITE THIS DOCUMENT AS

Duțescu, Bogdan.
QODIQA Governance Charter - Institutional Governance and Amendment Framework.
QODIQA Governance Charter v2.0.

QODIQA Standard Corpus, 2026.
Document Identifier: QODIQA-GOV-CH-2026-001.

Persistent Reference: QODIQA-GOV-CH-2026-001

INSTITUTIONAL

Revision History

REVISION HISTORY - QODIQA-GOV-CH-2026-001

VERSION	DATE	SUMMARY OF CHANGES
v2.0	2026	Added: Art. 5.1 Qualified Party definitions; Art. 5.3 Voting Record Requirements; Art. 5b Emergency Amendment Protocol; Art. 6b Steward Institutionalization; Art. 11 Governance Evolution and Scaling Mechanism; Art. 4.1 Interim Patch version type; Art. 4.2 Sunset Review lifecycle state. Consolidated: Rule CIPR-01 canonical definition into Art. 7.2 with cross-references from Arts. 2, 5, and 10. Restructured Art. 12 (Dissolution and Transfer). No enforcement semantics modified. All foundational principles and corpus authority structures preserved without change.
v1.0	February 2026	Initial publication of the QODIQA Governance Charter. Established foundational governance framework including Governance Principles (GP-01 through GP-07), Standard Corpus Definition, Versioning Model (MAJOR/MINOR/PATCH), Amendment Process (7-step full process), Simplified Amendment Process, Governance Structure with five institutional roles, Change Impact Analysis requirements, Core Invariant Protection Rule, Certification Oversight, Transparency and Publication Requirements, Safeguards Against Capture, and Dissolution or Transfer Provisions.

CHAPTER 22 PART IV · GOVERNANCE AND LEGAL INSTRUMENTS

Residual Risk and Assumption Disclosure Annex

The normative risk and assumption disclosure annex for the deterministic runtime consent enforcement standard.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-RISK-2026-001	1.0	2026	Normative Annex	Institutional	NORMATIVE

ABSTRACT

This document constitutes the **Residual Risk and Assumption Disclosure Annex** for QODIQA - the deterministic runtime consent enforcement standard for artificial intelligence systems. It is a structural integrity document, not a marketing instrument.

The Annex enumerates all operational assumptions required for QODIQA to function as specified, identifies residual risk surfaces that persist under conditions of perfect conformance, declares the explicit non-coverage domains of the standard, and imposes formal constraints on how QODIQA may be characterised by implementors and regulators.

This document reduces reputational, regulatory, and legal exposure for all parties operating under or referencing the QODIQA standard. It is issued alongside the Core specification and is a **normative component of the QODIQA corpus**. No deployment may represent itself as QODIQA-conformant without acknowledgement of the disclosures herein.

SECTION 00

Executive Disclosure Statement

QODIQA is a **deterministic runtime consent boundary enforcement layer** for artificial intelligence systems. The following declarations constitute the authoritative characterisation of the standard and supersede any contrary representation in promotional, marketing, or secondary materials.

FORMAL DECLARATION - SCOPE OF COVERAGE

QODIQA enforces deterministic consent boundaries at runtime. It does not guarantee AI safety. It does not guarantee model alignment. It does not eliminate hallucination. It does not regulate model weights. It does not inspect latent representations. It does not prevent harm from within its own enforcement perimeter.

0.1 Affirmative Scope

- QODIQA enforces consent policy at the API gateway and execution boundary layer.
- QODIQA produces deterministic permit/deny decisions based on encoded policy and verified principal identity.
- QODIQA generates tamper-evident, auditable enforcement logs.
- QODIQA provides a conformance framework against which implementations may be assessed.

- QODIQA aligns its control architecture with the requirements of GDPR, the EU AI Act, and NIST AI RMF as documented in the Regulatory Alignment Matrix.
- QODIQA defines cryptographic attestation mechanisms for principal identity binding and policy integrity.

0.2 Negative Scope - Formal Exclusions

TABLE 0-A - FORMAL EXCLUSION REGISTER

EXCLUDED DOMAIN	SCOPE BOUNDARY STATEMENT
AI Safety	QODIQA does not constitute an AI safety framework. Enforcement of consent boundaries does not prevent unsafe model outputs within those boundaries.
Model Alignment	QODIQA does not verify, assess, or enforce alignment of model behaviour to human values or intent.
Hallucination Elimination	QODIQA does not inspect model outputs for factual accuracy. Hallucinated outputs within a permitted execution context are not subject to QODIQA enforcement.
Model Weight Regulation	QODIQA operates at the inference boundary. It has no access to, and does not regulate, model weights, training procedures, or fine-tuning processes.
Latent Representation Inspection	QODIQA does not inspect internal model states, attention patterns, or embedding spaces.
Universal Harm Prevention	QODIQA prevents execution outside of consented scope. It does not prevent harm that occurs within consented scope.

> NORMATIVE NOTICE
>
> This annex is issued as a normative component of the QODIQA corpus. Implementors who represent QODIQA as providing guarantees beyond those enumerated above are in breach of this disclosure. Regulatory submissions citing QODIQA conformance must reproduce or reference this annex.

SECTION 01

Foundational Assumptions Register

QODIQA's deterministic enforcement properties hold only when the following assumptions are satisfied. Each assumption is accompanied by its failure mode and systemic consequence. Implementors bear responsibility for ensuring these assumptions hold within their operational environment.

1.1 Assumption Register

A-01 ACCURATE PRINCIPAL IDENTITY BINDING	
DESCRIPTION	The identity of every principal making consent-governed requests is accurately bound to a cryptographic credential. Identity claims presented at the enforcement boundary are genuine and have not been forged, spoofed, or transferred.
FAILURE MODE	Credential theft, identity spoofing, or compromised binding procedures result in a principal operating under a false identity, rendering consent records non-authoritative.
SYSTEMIC CONSEQUENCE	All enforcement decisions made under the false identity are attributed to the wrong principal. Audit logs become unreliable. Regulatory accountability is broken.

A-02 INTEGRITY OF CRYPTOGRAPHIC KEY MATERIAL	
DESCRIPTION	All signing keys, verification keys, and session tokens used within QODIQA enforcement flows are generated by approved processes, stored securely, and have not been compromised.
FAILURE MODE	Key compromise, key leakage, or improper key custody allows adversaries to forge valid enforcement tokens or consent attestations.
SYSTEMIC CONSEQUENCE	The cryptographic foundation of the enforcement layer is voided. Deterministic enforcement guarantees cease to hold. The entire consent record corpus may be rendered untrustworthy.

A-03 AVAILABILITY OF CONSENT REGISTRY	
DESCRIPTION	The consent registry is available, reachable, and consistent at the time of each enforcement query. The registry reflects the current and authoritative state of principal consent records.
FAILURE MODE	Registry unavailability, network partition, or stale cache states result in enforcement decisions being made against outdated or absent consent data.
SYSTEMIC CONSEQUENCE	Enforcement may default to permit or deny based on fallback policy, which may not reflect principal intent. Withdrawal of consent may not propagate in time. Regulatory exposure follows.

A-04 ACCURATE POLICY ENCODING	
DESCRIPTION	The consent policies encoded in the registry accurately represent the actual consent intentions of the principals they govern. Policy encoding is performed by authorised personnel following documented procedures.
FAILURE MODE	Misconfigured, ambiguous, or fraudulently encoded policies produce enforcement decisions that do not reflect genuine principal consent.
SYSTEMIC CONSEQUENCE	Technically deterministic enforcement of incorrect policy is indistinguishable from correct enforcement at the system layer. Audits will confirm policy was applied; the policy itself remains the error surface.

A-05 PROPER ENFORCEMENT GATEWAY DEPLOYMENT	
DESCRIPTION	The QODIQA enforcement gateway is deployed in a position of mandatory interception relative to the AI execution environment. No request path exists that bypasses the enforcement layer.
FAILURE MODE	Partial deployment, misconfigured routing, or shadow API access paths allow requests to reach the AI execution environment without enforcement evaluation.
SYSTEMIC CONSEQUENCE	Enforcement coverage is incomplete. The QODIQA boundary is porous. Conformance claims are invalid if any execution path bypasses the enforcement layer.

A-06 ABSENCE OF SIDE-CHANNEL BYPASS	
DESCRIPTION	No operational, technical, or procedural pathway exists through which the AI model may be accessed or influenced outside of the QODIQA enforcement boundary.
FAILURE MODE	Direct model access via maintenance interfaces, debug endpoints, internal tooling, or vendor backdoors circumvents consent enforcement entirely.
SYSTEMIC CONSEQUENCE	QODIQA enforcement is rendered meaningless for any execution conducted via the bypass path. No audit record exists for such executions.

A-07 RELIABLE TIME SYNCHRONISATION	
DESCRIPTION	All system components participating in enforcement flows operate on a synchronised time source with sufficient precision for temporal consent window evaluation and log ordering.
FAILURE MODE	Clock skew, NTP manipulation, or time spoofing creates ambiguity in consent window validity, replay attack detection, and log sequencing.
SYSTEMIC CONSEQUENCE	Time-bound consent assertions may be evaluated incorrectly. Expired tokens may be accepted. Audit log integrity cannot be confirmed across components with divergent clocks.

A-08 LOG IMMUTABILITY ENFORCEMENT	
DESCRIPTION	Enforcement audit logs are written to an append-only store that is technically and procedurally protected against modification, deletion, or suppression.
FAILURE MODE	Log tampering, selective deletion, or access by privileged insiders enables post-hoc modification of the enforcement record.
SYSTEMIC CONSEQUENCE	The audit trail ceases to be authoritative. Regulatory reliance on QODIQA audit records is undermined. Legal defensibility of enforcement decisions is compromised.

A-09 HONEST-BUT-CURIOUS INFRASTRUCTURE OPERATORS	
DESCRIPTION	Infrastructure operators - including cloud providers, managed service vendors, and internal platform teams - are assumed to be honest in their execution of system operations, though they may observe traffic and metadata. They are not assumed to actively subvert enforcement.
FAILURE MODE	A malicious infrastructure operator may selectively suppress enforcement, replay tokens, alter policy state, or exfiltrate consent records without leaving a detectable trace within the QODIQA layer.
SYSTEMIC CONSEQUENCE	QODIQA provides no defence against a compromised or adversarial operator at the infrastructure level. This threat surface is explicitly out of scope without Trusted Execution Environment controls.

A-10 TRUSTED EXECUTION ENVIRONMENT INTEGRITY (CONDITIONAL)	
DESCRIPTION	Where QODIQA deployment profiles specify TEE-based attestation, the TEE is assumed to be correctly provisioned, uncompromised, and attesting to genuine execution state.
FAILURE MODE	TEE firmware compromise, side-channel attacks on TEE (e.g., Spectre-class), or vendor-supplied TEE backdoors invalidate attestation guarantees.
SYSTEMIC CONSEQUENCE	TEE-dependent enforcement attestations are untrustworthy. This assumption failure requires remediation at the hardware or firmware layer and is outside QODIQA's operational control.

A-11 SECURE SOFTWARE SUPPLY CHAIN INTEGRITY	
DESCRIPTION	The QODIQA enforcement gateway, policy engine, registry services, and logging infrastructure are built and deployed from verified software artefacts whose integrity has been cryptographically validated. Build pipelines, dependency graphs, and deployment toolchains have not been compromised at any stage from source to runtime.
FAILURE MODE	Dependency poisoning, malicious package injection, build pipeline compromise, or unauthorised code modification introduces hidden logic into enforcement components prior to or during deployment.
SYSTEMIC CONSEQUENCE	Deterministic enforcement may be subverted at the implementation level without visible deviation from specification. Audit logs continue to record enforcement events, but the enforcement logic executing those events is no longer trustworthy. Trust in the enforcement layer collapses without any detectable specification violation.

A-12 DETERMINISTIC POLICY ENGINE INTEGRITY	
DESCRIPTION	The policy evaluation engine correctly implements the deterministic decision function defined in the QODIQA Core specification. The engine produces permit/deny decisions that are faithful to the specification under all valid input combinations, including boundary conditions and concurrent execution states.
FAILURE MODE	Logical error, race condition, state corruption, or undefined behaviour in the evaluation engine produces inconsistent permit/deny outcomes under identical inputs, violating the determinism guarantee at the implementation layer.
SYSTEMIC CONSEQUENCE	The deterministic guarantee central to QODIQA's enforcement model is invalidated. Auditability of individual decisions is preserved but does not confirm correctness of the decision logic applied. Conformance claims based on deterministic enforcement cannot be sustained against an implementation with a compromised policy engine.

SECTION 02

Deterministic Scope Boundary

The term *deterministic runtime consent enforcement* as used in the QODIQA standard has a precise and bounded meaning. This section defines that meaning and explicitly delineates what it does not encompass.

2.1 What Deterministic Runtime Consent Enforcement Means

FORMAL DEFINITION QODIQA-RISK-2026-001 s. 2.1

Deterministic

Given identical inputs - principal identity, request context, consent record state, and current time - the enforcement decision function produces the same output on every invocation. There is no probabilistic element in the permit/deny decision.

Runtime

Enforcement occurs at the moment of execution, not at deployment time, design time, or audit time. Each request is evaluated against the current state of the consent registry at the time of that specific request.

Consent Enforcement

The enforcement function evaluates whether the described action falls within the scope of valid, current, and applicable consent granted by the relevant principal. It does not evaluate the quality, safety, or downstream effects of the permitted action.

2.2 What Deterministic Runtime Consent Enforcement Does Not Mean

TABLE 2-A - DETERMINISM BOUNDARY CLARIFICATION REGISTER

ASSERTION	BOUNDARY CLARIFICATION
Does not mean the permitted action is safe	A request may be permitted by QODIQA and still produce harmful outputs within the consented scope.
Does not mean the policy reflects genuine consent	Determinism applies to policy evaluation, not to the quality of the underlying policy. Incorrectly configured policy is evaluated deterministically.

ASSERTION	BOUNDARY CLARIFICATION
Does not mean the enforcement perimeter is complete	Determinism holds only within the deployed enforcement boundary. Actions taken outside that boundary are not subject to QODIQA evaluation.
Does not mean the model is aligned	QODIQA governs access to model execution. It has no visibility into model behaviour during execution.
Does not mean the system is compliant	QODIQA provides controls relevant to certain regulatory requirements. Legal compliance requires additional measures and legal assessment.
Does not mean all risks are mitigated	Deterministic boundary enforcement is one layer in a defence-in-depth architecture. Residual risk surfaces enumerated in Section 3 persist regardless of QODIQA deployment.

> PRECISION REQUIREMENT
>
> All representations of QODIQA in regulatory submissions, marketing materials, procurement documentation, and technical specifications must respect the boundary defined in this section. Representations that conflate boundary enforcement with AI safety, alignment, or harm elimination are inaccurate and are prohibited under this annex.

2.3 Formal Enforcement Perimeter Definition

The QODIQA enforcement perimeter is defined as the set of components, interfaces, and processing steps over which the deterministic enforcement guarantee holds. Deterministic guarantees hold exclusively within this defined perimeter. Any execution occurring outside this perimeter is not governed by QODIQA and is not subject to any enforcement, audit, or conformance claim under this standard.

PERIMETER COMPONENT DEFINITIONS QODIQA-RISK-2026-001 s. 2.3

Ingress Interface Boundary

The enforcement perimeter begins at the network or API interface at which an inbound request is first received by the QODIQA enforcement gateway. All traffic received at this interface is subject to enforcement evaluation. Traffic that does not reach this interface - including requests routed directly to the AI execution environment via any path that bypasses the gateway - is outside the perimeter and is not governed by QODIQA.

Identity Verification Point

The identity verification point is the component within the enforcement gateway responsible for validating the cryptographic credentials presented by the requesting principal. This component is within the perimeter. The credential issuance infrastructure - including identity providers, certificate authorities, and key management systems - is external to the QODIQA enforcement perimeter and is governed by the assumptions enumerated in Section 1.

Policy Evaluation Function

The policy evaluation function is the deterministic decision engine that takes as input the verified principal identity, the request scope parameters, and the current state of the consent registry, and produces a permit or deny decision. The policy evaluation function is within the perimeter. The consent registry itself is an external dependency accessed synchronously at evaluation time; its integrity is governed by assumption A-03.

Token Issuance Boundary

Upon a permit decision, the enforcement gateway issues a cryptographically signed enforcement token. The token issuance process - including signing key access, token serialisation, and token delivery to the requesting client - is within the perimeter. The handling, storage, and transmission of the token by the requesting client after delivery is outside the perimeter.

Audit Log Commitment Point

The audit log commitment point is the operation by which an enforcement decision record is cryptographically committed to the immutable audit store. The commitment operation and the integrity of records prior to commitment are within the perimeter. The security of the audit store itself - including its physical infrastructure, access controls, and backup procedures - is an organisational dependency governed by assumption A-08.

Egress Handoff to Model Runtime

The enforcement perimeter ends at the egress handoff point: the interface at which a permitted request is passed from the QODIQA enforcement gateway to the AI model runtime for execution. From the egress handoff point onward - including all processing by the model runtime, all model outputs, and all downstream systems that consume model outputs - is outside the QODIQA enforcement perimeter. No enforcement, audit, or determinism guarantee applies beyond this point.

2.3.1 Components Explicitly Outside the Enforcement Perimeter

TABLE 2-B - COMPONENTS EXPLICITLY OUTSIDE THE ENFORCEMENT PERIMETER

COMPONENT	PERIMETER EXCLUSION STATEMENT
AI model weights, architecture, and runtime	The model execution environment, including all inference operations, is outside the QODIQA perimeter.
Model output handling and downstream systems	Any system that receives, processes, stores, or acts upon model outputs is outside the perimeter.
Identity provider infrastructure	Certificate authorities, identity providers, and key management systems that support principal credential issuance are outside the perimeter.
Consent registry infrastructure	The physical and logical infrastructure hosting the consent registry, including its database engine, replication systems, and access control layer, is outside the perimeter.
Audit store infrastructure	The storage infrastructure, backup systems, and access control layer of the audit log store are outside the perimeter.
Network transport layer	The network infrastructure over which requests and responses transit between clients and the enforcement gateway ingress interface is outside the perimeter.
Client-side request construction	The process by which a requesting principal constructs a request, including all client-side software, is outside the perimeter.

SECTION 03

Residual Technical Risk Surfaces

The following risk surfaces persist under conditions of perfect QODIQA implementation, perfect conformance, and perfect policy encoding. They are structural properties of the threat environment that QODIQA does not and cannot address.

Risk bands are assigned as follows: **Critical** - systemic failure with no internal mitigation pathway; **High** - significant impact, partial mitigation possible through external controls; **Medium** - meaningful but contained impact; **Low** - limited scope.

TABLE 3-A - RESIDUAL TECHNICAL RISK REGISTER

RISK ID	DESCRIPTION	BAND
RT-01	Model weight bias producing skewed or harmful outputs within permitted execution scope	HIGH
RT-02	Training data poisoning contaminating model behaviour prior to deployment	HIGH
RT-03	Upstream training corpus contamination by adversarial third-party data	HIGH
RT-04	Prompt injection attacks originating within the permitted execution context	CRITICAL
RT-05	Adversarial input generation that produces harmful model outputs within consented scope	HIGH
RT-06	Zero-day runtime exploits targeting the enforcement gateway process itself	CRITICAL
RT-07	Cryptographic primitive compromise through mathematical advancement or implementation flaw	CRITICAL
RT-08	Infrastructure-level compromise (hypervisor, network fabric, storage layer) below the enforcement boundary	CRITICAL
RT-09	Hallucination of false information within permitted execution scope causing downstream harm	HIGH

RISK ID	DESCRIPTION	BAND
RT-10	Model capability overhang - capabilities not known at policy encoding time operating within permitted scope	HIGH
RT-11	Policy staleness - consent records that no longer reflect principal intent due to changed circumstances	MEDIUM
RT-12	Cross-context inference attacks using permitted outputs to reconstruct protected information	MEDIUM

3.1 Prompt Injection Within Boundary (RT-04)

Prompt injection represents a residual risk surface of particular structural significance. QODIQA evaluates whether a request is permitted based on its metadata, scope parameters, and principal consent record. The content of permitted prompts is not inspected by the enforcement layer. A permitted prompt containing adversarial instructions directed at the model is evaluated by QODIQA as a valid request if it falls within the consented scope. The model's response to such instructions is not within QODIQA's governance perimeter.

3.2 Infrastructure Compromise (RT-08)

An adversary with access at the infrastructure layer - including hypervisor access, network interception capabilities, or storage layer access - may observe, modify, or suppress enforcement operations without QODIQA being able to detect or prevent such interference. This risk surface is acknowledged but not addressed by QODIQA. Mitigations at the infrastructure layer, including hardware-based attestation and network segmentation, are the responsibility of the deploying organisation.

3.3 Model Capability Overhang (RT-10)

Consent policies are encoded against a model's known or anticipated capabilities at the time of policy creation. AI models may exhibit capabilities that were not known, anticipated, or accounted for during policy encoding. Such unanticipated

capabilities may operate within the consented scope while producing effects that were not contemplated when consent was obtained. This is a structural limitation of any consent framework applied to systems whose capability boundaries are not fully characterised.

3.4 Risk Interdependency and Compounded Failure

The risk surfaces enumerated in this section and the foundational assumptions enumerated in Section 1 are not independent. Simultaneous failure of multiple assumptions, or concurrent realisation of multiple risk surfaces, compounds the systemic impact beyond the severity classification assigned to any individual risk. The classifications in Table 3-A reflect the impact of each surface in isolation; compounded failure scenarios may exceed those classifications.

QODIQA does not model probabilistic compounding risk internally. The following scenarios are provided to illustrate the structural nature of risk interdependency. They are not exhaustive.

CI-01 IDENTITY BINDING FAILURE + REGISTRY TAMPERING	COMPOUNDED
ASSUMPTIONS FAILED	A-01 (Accurate Principal Identity Binding) and A-03 (Availability and Integrity of Consent Registry)
COMPOUNDED SCENARIO	An adversary operating under a forged principal identity concurrently tampers with the consent registry to insert or modify consent records attributed to the spoofed identity. Enforcement decisions are made against falsified identity claims evaluated against falsified policy. The audit record captures the falsified identity and the falsified policy as though both were legitimate. Post-incident forensic reconstruction cannot distinguish the falsified enforcement record from a legitimate one without out-of-band evidence.
COMPOUNDED IMPACT	The two individual assumptions, each of which would independently degrade enforcement integrity, together eliminate both the enforcement correctness and the audit reconstructability of the affected session. No compensating control within the QODIQA layer addresses this compound failure.

CI-02 KEY COMPROMISE + LOG SUPPRESSION	COMPOUNDED
ASSUMPTIONS FAILED	A-02 (Cryptographic Key Material Integrity) and A-08 (Log Immutability Enforcement)
COMPOUNDED SCENARIO	An adversary who has obtained valid signing key material is able to forge enforcement tokens and consent attestations that are indistinguishable from legitimate ones. Concurrently, suppression of audit log entries covering the period of token forgery removes the forensic record of the forged operations. The enforcement layer continues to operate normally for legitimate requests; the adversarial operations are neither detected nor recorded.
COMPOUNDED IMPACT	Key compromise alone is detectable through anomaly analysis of the audit record. Log suppression alone may be detectable through log gap analysis. The combination eliminates both detection pathways simultaneously. The resulting window of undetected, unrecorded adversarial operation has no internal remediation pathway within the QODIQA layer.

CI-03 INFRASTRUCTURE COMPROMISE + TOKEN REPLAY	COMPOUNDED
ASSUMPTIONS FAILED	A-09 (Honest-but-Curious Infrastructure Operators) and risk surface RT-06 (Token Replay, s. 9.2)
COMPOUNDED SCENARIO	An adversary with infrastructure-layer access observes legitimate enforcement tokens in transit, extracts them without the knowledge of the requesting principal, and replays them to obtain model execution under the legitimate principal's consented scope. The infrastructure-layer adversary can selectively suppress the log entries corresponding to the replayed token usage. The legitimate principal's audit record shows no anomaly; the replayed executions are either unrecorded or appear as legitimate executions by the principal.
COMPOUNDED IMPACT	Token replay in isolation is detectable through log analysis of usage frequency anomalies. Infrastructure compromise in isolation is a known, bounded assumption failure. Their combination creates a replay capability coupled with suppression of the evidence required for its detection. This compounded scenario has no internal QODIQA mitigation pathway.

CI-04 POLICY MISCONFIGURATION + MODEL CAPABILITY OVERHANG	COMPOUNDED
ASSUMPTIONS FAILED	A-04 (Accurate Policy Encoding) and risk surface RT-10 (Model Capability Overhang, s. 3.3)
COMPOUNDED SCENARIO	A consent policy that was intended to permit a narrowly scoped AI function is encoded with scope parameters that are broader than intended - for example, through ambiguous action type definitions or overly permissive context bindings. The consented model simultaneously exhibits capabilities beyond those anticipated at policy encoding time. The permissive scope admits requests that were not intended to be permitted; the unanticipated model capabilities operate on those requests to produce effects that were not contemplated when consent was granted.
COMPOUNDED IMPACT	Policy misconfiguration alone can be identified and corrected through audit review and policy hygiene procedures. Capability overhang alone can be addressed through model evaluation and policy update cycles. Their simultaneous occurrence creates a window during which unanticipated capabilities operate on improperly scoped permissions with no enforcement-layer signal indicating that anything is anomalous. Both the execution and the policy appear valid from the QODIQA perspective.

> **COMPOUNDED FAILURE CLASSIFICATION**
>
> QODIQA does not assign a single severity band to compounded failure scenarios. Compounded failures involving two or more Critical-band individual risks should be treated as exceeding the Critical classification for purposes of risk treatment planning. Organisations are responsible for performing their own compounded risk analysis in the context of their specific deployment architecture and threat model.

SECTION 04

Regulatory Interpretation Risk

QODIQA maintains a Regulatory Alignment Matrix that maps control points to provisions of GDPR, the EU AI Act, NIST AI RMF, and other frameworks. The following disclosures govern how that alignment matrix must and must not be interpreted.

4.1 QODIQA Does Not Constitute Legal Compliance

The QODIQA Regulatory Alignment Matrix maps technical controls to regulatory provisions. This mapping does not constitute a legal determination that an organisation implementing QODIQA is compliant with any cited regulation. Legal

compliance requires assessment by qualified legal counsel with jurisdiction-specific expertise. The alignment matrix is a technical reference instrument, not a legal opinion.

4.2 QODIQA Does Not Substitute for Legal Counsel

No provision of the QODIQA standard, including this annex, constitutes legal advice. Organisations using QODIQA alignment documentation in regulatory submissions, compliance certifications, or legal proceedings do so at their own risk and must obtain independent legal review of such submissions.

4.3 Jurisdictional Divergence

The regulatory landscape for AI governance is evolving at different rates across jurisdictions. Requirements under the EU AI Act do not necessarily correspond to requirements under US federal frameworks, national UK regulations, or emerging frameworks in other jurisdictions. The QODIQA alignment matrix reflects the state of regulation as understood at the time of document issuance. It does not predict, anticipate, or account for regulatory amendments, new guidance documents, or enforcement decisions issued after the document date.

4.4 Regulatory Interpretation Variance

Regulatory authorities may interpret the provisions of AI governance frameworks differently from the interpretations reflected in the QODIQA alignment matrix. Enforcement actions, supervisory guidance, and judicial decisions may clarify or alter the requirements attributed to specific provisions. QODIQA alignment with a regulatory provision as documented does not guarantee that a supervising authority will reach the same interpretive conclusion.

> **COMPLIANCE VALIDATION REQUIREMENT**
>
> Organisations subject to regulatory enforcement who rely on QODIQA conformance as a compliance argument must independently validate that the technical controls provided by QODIQA satisfy the specific requirements of the applicable regulatory authority in the relevant jurisdiction.

SECTION 05

Organizational Risk Dependencies

QODIQA's technical controls are necessary but not sufficient conditions for effective consent enforcement. The following organisational capabilities must be maintained by the deploying entity. Absence or degradation of these capabilities constitutes residual risk that QODIQA cannot compensate for at the technical layer.

5.1 Governance Maturity

Effective deployment of QODIQA requires a functioning governance structure capable of making, recording, and communicating policy decisions. Organisations with immature governance - undefined ownership, inconsistent approval processes, or absent policy review cycles - will produce unreliable consent policies regardless of the technical enforcement precision of the QODIQA layer.

5.2 Internal Policy Hygiene

Consent policies must be reviewed on a defined cycle to remain current with changing operational requirements, regulatory developments, and model capability changes. Policy rot - the accumulation of stale, contradictory, or abandoned consent records - creates a surface for both unintended permissions and unintended restrictions. QODIQA enforces policy as encoded; it does not identify policy that has become obsolete.

5.3 Incident Response Capability

Detection and response to enforcement anomalies, suspected bypass events, and audit log integrity failures require a functioning incident response capability. QODIQA generates the audit record and enforcement signals. Acting on those signals, investigating anomalies, and executing remediation procedures are organisational responsibilities.

5.4 Audit Discipline

The value of QODIQA's tamper-evident logging is realised only if logs are reviewed on a defined schedule by personnel with the authority and competence to act on their findings. An unread audit log provides no protective value.

5.5 Key Rotation Practices

Cryptographic key material used in QODIQA enforcement flows must be rotated at defined intervals and immediately upon any suspected compromise. Organisations that do not maintain a documented and practiced key rotation programme degrade the cryptographic assurance of the enforcement layer over time.

5.6 Personnel Competence and Continuity

Correct configuration, maintenance, and operation of QODIQA deployments requires personnel with appropriate technical competence. Staff turnover, knowledge transfer failures, and inadequate training create operational risk that manifests as misconfiguration, policy errors, or delayed incident response.

SECTION 06

Non-Coverage Declaration

The following domains are explicitly and entirely outside the scope of QODIQA. No provision of the QODIQA standard, and no implementation conformant with it, provides coverage in these areas. This declaration is not a statement of future roadmap. These domains are structurally distinct from boundary enforcement and are not subject to expansion into QODIQA scope.

TABLE 6-A - NON-COVERAGE DECLARATION REGISTER

EXCLUDED DOMAIN	NON-COVERAGE STATEMENT
General AI Alignment	The alignment of AI model behaviour with human values, intent, or ethical principles is not addressed by QODIQA. QODIQA governs access to model execution; it does not govern what the model does during execution.
Model Interpretability	QODIQA provides no tools, mechanisms, or standards for understanding model internal states, attention distributions, reasoning chains, or decision processes.
AGI Containment	QODIQA does not address the containment of artificial general intelligence systems, their capability trajectories, or the prevention of capability elicitation beyond intended design.
Economic Displacement Mitigation	QODIQA has no provisions relating to the economic effects of AI deployment, including workforce displacement, market concentration, or distributional impacts.
Ethical Theory Arbitration	QODIQA does not adjudicate between competing ethical frameworks, moral positions, or value systems. It does not encode any ethical position beyond the procedural requirement that specified consent boundaries be enforced as specified.
Human Oversight Frameworks	QODIQA provides technical enforcement of consent boundaries. It does not design, certify, or enforce broader human oversight structures, board-level governance requirements, or accountability frameworks beyond its defined control points.
Model Architecture Design	QODIQA makes no requirements on model architecture, training methodology, data provenance, or evaluation procedures. These remain outside the scope of the standard.
Content Moderation	QODIQA does not evaluate the content of model outputs for harmful, offensive, or illegal material. Enforcement of content standards within a permitted execution context is not a QODIQA function.
Bias Detection and Mitigation	QODIQA provides no assessment of, or mitigation for, systematic biases in model outputs, including demographic bias, representational harm, or distributional unfairness.
Privacy-Preserving Computation	QODIQA governs consent to execute. It does not implement differential privacy, federated learning, homomorphic encryption, or other privacy-preserving computation techniques.

STRUCTURAL NOTE

> The non-coverage domains listed above represent areas of genuine and significant concern in AI governance. Their exclusion from QODIQA scope is a consequence of the standard's deliberate focus on a well-defined technical function. Organisations addressing these domains must employ additional frameworks appropriate to each area.

SECTION 07

Cryptographic Assumption Disclosure

QODIQA's enforcement integrity depends on cryptographic primitives. The security of the standard is computationally bounded, not unconditional. The following disclosures govern the cryptographic trust basis of QODIQA.

7.1 Dependence on Cryptographic Primitives

QODIQA relies on hash functions, digital signature algorithms, and authenticated encryption schemes as specified in the Security and Cryptographic Profile. The integrity guarantees provided by QODIQA - including consent attestation, token non-repudiation, and log tamper evidence - are conditional on these primitives remaining computationally infeasible to break under current and near-term adversary capabilities.

7.2 Risk of Future Cryptanalytic Breakthrough

Mathematical advances may render currently approved cryptographic primitives insecure before end-of-life dates assigned by standards bodies. Such advances may occur without warning and may pre-date their public announcement by adversaries with classified research capabilities. QODIQA does not provide a mechanism for detecting cryptographic compromise after the fact; the audit record itself would be untrustworthy if the underlying cryptographic primitives were broken.

7.3 Quantum Threat Exposure

Current QODIQA cryptographic specifications use algorithms based on the hardness of problems assumed to be computationally infeasible for classical computers. Sufficiently capable quantum computing systems would break widely deployed asymmetric cryptographic algorithms including RSA and elliptic curve schemes. The timeline for such capability remains uncertain. Organisations operating under long-term security requirements should monitor NIST Post-Quantum Cryptography standardisation outputs and assess migration requirements accordingly. QODIQA will issue post-quantum migration guidance under formal change control when relevant standards are ratified.

7.4 Key Custody Exposure

The security of QODIQA's cryptographic layer is bounded by the security of key management practices. Keys held in software-only key stores, shared across environment boundaries, or managed without hardware security module controls are materially more exposed than keys managed under high-assurance key custody procedures. QODIQA specifies minimum key management requirements; organisations that fail to implement them degrade the cryptographic assurance of their deployment below the standard's stated security model.

7.5 Implementation Flaw Exposure

Correct selection of cryptographic algorithms does not ensure correct implementation. Side-channel vulnerabilities, padding oracle attacks, timing attacks, and random number generator weaknesses have historically compromised otherwise sound cryptographic designs in their implementation. QODIQA does not certify the implementation quality of cryptographic libraries used by conformant deployments.

SECTION 08

Operational Misuse Risk

QODIQA may be misconfigured, partially deployed, or misrepresented in ways that produce either inadequate enforcement or false confidence in enforcement coverage. The following misuse scenarios are identified as material risks arising from operational rather than adversarial sources.

8.1 Improper Policy Configuration

Consent policies encoded with ambiguous scope parameters, incorrect principal bindings, or missing temporal constraints may produce enforcement decisions that do not reflect the intent of the consenting principal. QODIQA enforces policy as encoded. Garbage-in-garbage-out failure modes apply: technically correct enforcement of incorrect policy produces incorrect outcomes.

8.2 Over-Restrictive Enforcement Blocking

Overly conservative policy encoding may block legitimate requests and create operational friction that incentivises bypass, shadow deployment, or informal workarounds. Organisations that respond to over-restrictive enforcement by routing around the QODIQA layer create the very bypass surfaces that enforcement is intended to prevent.

8.3 Under-Restrictive Enforcement

Permissive policy encoding, broad scope parameters, or absent expiry controls may result in consent records that provide effective permission to all requests. An enforcement layer that permits everything provides no enforcement value and creates false confidence in governance coverage.

8.4 Shadow Deployments

Organisational units or teams that deploy AI capabilities outside the QODIQA enforcement perimeter create unmonitored execution surfaces. Such deployments may occur deliberately, through ignorance of the enforcement requirement, or through procurement of third-party AI services that bypass the enforcement gateway. Shadow deployments are not detectable by QODIQA; they require organisational controls including procurement governance and network access controls.

8.5 Partial Implementation Represented as Full Conformance

Implementation of a subset of QODIQA control points, followed by public representation as QODIQA-conformant, constitutes a misuse of the standard. Partial conformance provides partial protection and partial compliance evidence. It does not constitute the assurance level associated with full conformance. The QODIQA Certification Framework specifies conformance tiers; representations must accurately reflect the tier achieved.

8.6 Delegation Misuse

Delegation chains in QODIQA allow principals to delegate enforcement authority to agents. Improperly scoped delegations, expired delegations not removed from the registry, or delegation to untrusted agents create authority transfer paths that were not intended by the original principal.

SECTION 09

Abuse and Adversarial Exploitation Scenarios

The following scenarios represent deliberate attempts to defeat or circumvent QODIQA enforcement. Detection limits are noted where relevant.

9.1 Direct Model Access Bypass

An adversary with credentials or network access sufficient to reach the AI execution environment directly - bypassing the QODIQA enforcement gateway - may interact with the model without any consent evaluation occurring. This attack surface requires operational controls at the infrastructure layer. QODIQA cannot detect bypass events that do not traverse the enforcement boundary; no log entry is generated for requests that circumvent the enforcement layer entirely.

9.2 Token Replay

A valid, unexpired enforcement token that was legitimately issued for one purpose may be replayed by an adversary for a different purpose if token scope binding is insufficiently specific or if token revocation is not enforced at verification time. Token replay attacks result in QODIQA treating replayed requests as legitimate. Detection requires log analysis to identify statistically anomalous token usage patterns; QODIQA does not implement automated replay detection as a default control point.

9.3 Registry Tampering

An adversary with write access to the consent registry may modify, delete, or insert consent records to expand or restrict permitted execution scope. Such tampering would cause subsequent enforcement decisions to be made against altered policy, with no indication at the enforcement layer that the registry has been modified. Log immutability and registry integrity monitoring are required compensating controls.

9.4 Log Suppression

An adversary with access to the logging pipeline may suppress, delete, or alter enforcement log records. If log suppression is achieved before cryptographic commitment of log entries, the tamper-evidence property of the audit trail is defeated. Post-commitment suppression is detectable through log gap analysis if an independent record of expected

log entry volume is maintained. QODIQA specifies log integrity requirements; their effectiveness depends on implementation and monitoring discipline.

9.5 Cross-Domain Request Smuggling

An adversary may craft requests that are evaluated by the enforcement layer as belonging to a permitted scope category while being processed by the model as belonging to a different category. This attack exploits differences between the metadata evaluated by the enforcement layer and the semantic content interpreted by the model. QODIQA evaluates consent based on request metadata and scope parameters; it does not perform semantic analysis of request content.

9.6 Identity Claim Forgery

Forgery of principal identity claims at the enforcement boundary results in consent evaluation being performed against the wrong principal's consent record. Effective detection requires cryptographic identity binding and the integrity of the key management infrastructure. An adversary able to forge valid cryptographic credentials defeats identity binding entirely.

9.7 Detection Limits

QODIQA's detection capabilities are bounded by the enforcement boundary. Events that occur outside that boundary, attacks that achieve access below the enforcement layer, and bypass events that produce no enforcement-layer artefacts are not detectable by QODIQA. External monitoring, network anomaly detection, and independent audit of AI system usage patterns are required to identify exploitation scenarios that operate outside QODIQA's observation horizon.

SECTION 10

Residual Societal Risk

This section addresses the scope of risk that persists at a societal level even under conditions of universal QODIQA adoption with full conformance.

> **STRUCTURAL DISCLOSURE**
>
> Universal adoption of QODIQA would represent universal deployment of a deterministic consent boundary enforcement layer across AI execution environments. It would not represent the resolution of the broader challenges posed by AI systems to society, democratic institutions, or individual welfare. These disclosures are required to prevent QODIQA from being characterised as a solution to problems it does not address.

10.1 AI Misuse Risk Persists

AI systems may be misused within the scope of consent that has been granted. A principal who consents to broad AI-enabled processing retains the ability to direct that processing toward harmful ends. QODIQA enforces that the processing occurred within consented scope; it does not evaluate whether the consented scope was appropriate or whether the consented use is harmful.

10.2 Model Misuse Risk Persists

AI models with the capacity to produce harmful outputs - disinformation, manipulation, dual-use content - retain that capacity when operated within QODIQA-enforced consent boundaries. The enforcement layer governs access; it does not govern capability. Harmful outputs produced within consented scope are not addressed by QODIQA.

10.3 Strategic AI Competition Risk Persists

The deployment of AI systems in strategic competition between state and non-state actors involves risk surfaces - capability asymmetry, surveillance infrastructure, autonomous decision systems, information operations - that are not addressed by consent boundary enforcement. These risks require responses at levels of governance, international coordination, and policy that are beyond the scope of any technical standard.

10.4 Consent Framework Limitations at Scale

Consent as a governance mechanism has known limitations at population scale. Informed consent requires genuine understanding of the consequences of the decision being made. At scale, the cognitive demands of evaluating AI consent decisions may exceed practical human capacity, consent may be aggregated and transferred in ways that no longer reflect individual intent, and the power asymmetry between consent-requesting entities and consenting individuals may render consent formally valid but substantively non-autonomous. QODIQA enforces the technical expression of consent; it does not resolve these structural challenges in consent theory.

10.5 Coordination Failure Risk

Jurisdictions, organisations, and actors that do not adopt QODIQA or equivalent standards create unmonitored AI execution environments that represent potential competitive advantages for actors willing to operate without consent boundaries. Partial adoption of consent enforcement standards does not eliminate the incentive for non-adopting actors to exploit the absence of constraints.

SECTION 11

No Guarantee Clause

FORMAL STATEMENT - NO GUARANTEE OF SAFETY, ALIGNMENT, OR ETHICS

QODIQA provides deterministic boundary enforcement - not safety guarantees, not alignment guarantees, not ethical guarantees. Conformance with QODIQA does not constitute a guarantee that an AI system is safe, aligned, ethical, or legally compliant. QODIQA is an infrastructure standard for consent boundary enforcement. It operates within a broader system of AI governance and does not substitute for any other component of that system.

11.1 Scope of the No Guarantee Clause

The no guarantee clause applies to all representations of QODIQA made by: the issuing authority; implementing organisations; certification bodies issuing QODIQA conformance certificates; regulatory bodies citing QODIQA conformance as evidence of compliance; and any third party characterising QODIQA capabilities in any medium.

11.2 What QODIQA Does Provide

QODIQA provides a structured, auditable, cryptographically attested boundary enforcement mechanism. Within the scope of that mechanism - and subject to the assumptions enumerated in Section 1 - QODIQA provides deterministic enforcement decisions, tamper-evident audit records, and a conformance framework for third-party assessment. These properties are bounded by the technical and operational conditions set forth in this annex.

11.3 Liability Disclaimer

The QODIQA standard is issued as a technical reference framework. The issuing authority makes no warranty, express or implied, regarding the fitness of QODIQA for any particular purpose, its effectiveness against any specific threat, or its sufficiency for any regulatory requirement. Organisations implementing QODIQA bear full responsibility for the adequacy of their implementation, the correctness of their policy configuration, and the completeness of their governance arrangements.

SECTION 12

Transparency Commitment Statement

The QODIQA issuing authority commits to the following transparency obligations with respect to the disclosures contained in this annex. These commitments are normative and are subject to oversight under the QODIQA Governance Charter.

12.1 Periodic Risk Re-Evaluation

The assumption register, residual risk surfaces, and non-coverage declarations in this annex will be reviewed at intervals not exceeding twelve months from the date of issuance, and additionally upon: any material change to the QODIQA core specification; any significant development in the AI threat landscape that materially alters the risk assessment; any cryptographic standard deprecation affecting the primitives used in QODIQA deployments; and any regulatory development that materially alters the interpretive risk described in Section 4.

12.2 Versioned Disclosure Updates

All revisions to this annex will be issued as versioned documents under the QODIQA document identifier scheme. Revision history will be maintained and publicly accessible. Prior versions will be retained in the document archive. No

version of this annex will be suppressed, and no revision will reduce the scope or rigor of the disclosures without publication of a formal rationale under the Governance Charter change process.

12.3 Formal Change Control

Amendments to this annex are subject to the change control procedures defined in the QODIQA Governance Charter v2.0. All proposed revisions must pass the designated review process. Emergency revisions - required in response to a critical security finding - follow the expedited process defined in the Governance Charter and are subject to retroactive formal review.

12.4 Publicly Documented Assumption Revisions

If a foundational assumption in Section 1 is revised - whether because the assumption has been strengthened by new technical controls, weakened by new threat intelligence, or determined to be invalid in specific deployment contexts - the revision will be publicly documented with a rationale explaining the change and its implications for existing QODIQA conformant deployments.

12.5 Disclosure of Known Limitation Instances

If the QODIQA issuing authority becomes aware of a confirmed exploitation of a residual risk surface identified in this annex, or a failure mode in a foundational assumption that has materially impacted a conformant deployment, the issuing authority will issue a formal advisory under the QODIQA advisory process. Such advisories will not identify affected organisations without consent but will provide sufficient technical detail for the broader implementing community to assess their own exposure.

† This annex supersedes all prior informal characterisations of QODIQA's scope limitations. In cases of conflict between this document and any other QODIQA document, the more restrictive characterisation of QODIQA's scope and guarantees governs.

‡ Document identifier QODIQA-RISK-2026-001. Issued March 2026. Next scheduled review: March 2027 or upon material change to the QODIQA corpus, whichever occurs first.

§ This document is a normative component of the QODIQA corpus and is governed by the QODIQA Governance Charter. Reproduction for purposes of regulatory submission, procurement assessment, or academic citation is permitted with attribution.

ANNEX

Closing Statement

RESIDUAL RISK AND ASSUMPTION DISCLOSURE ANNEX - CLOSING

This document has enumerated the foundational assumptions, residual risk surfaces, non-coverage domains, and explicit scope limitations of the QODIQA standard. It has been produced without advocacy, without marketing language, and without suppression of material limitations.

The QODIQA standard is a technical instrument of bounded scope. Its value lies in what it precisely and reliably delivers: deterministic, auditable, cryptographically attested consent boundary enforcement at the AI execution layer. That value is best protected by accurate characterisation of its limits.

Organisations, regulators, and auditors who treat this annex as a constraint on QODIQA's utility misunderstand the function of disclosure. Accurate scope definition does not diminish the value of deterministic enforcement; it locates that value correctly within a broader architecture of AI governance in which QODIQA is one necessary component among several.

- QODIQA enforces consent boundaries. It does not resolve the harder problems of AI governance.
- The harder problems of AI governance remain open. They require different instruments.
- This annex documents both what QODIQA addresses and what it does not. Both are required for honest assessment.

DOCUMENT STATUS

Document Status and Classification

This document constitutes the normative Residual Risk and Assumption Disclosure Annex for QODIQA - the deterministic runtime consent enforcement standard for artificial intelligence systems. It is a structural integrity document issued as a normative component of the QODIQA corpus. No deployment may represent itself as QODIQA-conformant without acknowledgement of the disclosures herein.

It is issued alongside the Core specification and is not a marketing instrument, a product specification, or a legal instrument. All technical principles described in this document are designed to be production-oriented, cryptographically anchored, and architected for integration within large-scale AI ecosystems.

The material contained herein is intended for:

- Systems architects and infrastructure engineers
- Enterprise AI platform designers
- Security and governance officers
- Regulatory and policy stakeholders
- Organizations building or operating autonomous AI systems
- Legal and compliance professionals engaged in AI governance

This document should be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Threat Model and Abuse Case Specification, Version 1.0
- QODIQA: Regulatory Alignment Matrix for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0
- QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Duţescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT TITLE	Residual Risk and Assumption Disclosure Annex
CORPUS POSITION	Normative Annex - QODIQA Standard Corpus
VERSION	1.0
PUBLICATION DATE	2026
DOCUMENT CODE	QODIQA-RISK-2026-001
NORMATIVE STATUS	Normative - Residual Risk and Assumption Disclosure Annex
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 23 PART IV · GOVERNANCE AND LEGAL INSTRUMENTS

Certification Framework for Deterministic Runtime Consent Enforcement

The framework defines how implementations of the corpus are evaluated, certified, attested, and maintained in a verified conformant state.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-CERT-2026-001	1.0	2026	Certification Specification	Institutional	NORMATIVE

Status note. No certification body or program exists yet. This is a conformance framework for self-assessment; any reference to certification or certified status describes a planned process, not an existing one.

SCOPE AND PURPOSE

This document defines how implementations of the QODIQA Core v1.0 Standard and, where applicable, the QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0, are evaluated, certified, attested, and maintained in a verified conformant state. It establishes the evidentiary, procedural, and structural requirements that constitute conformance, and the conditions under which conformance may be affirmed, suspended, or revoked.

This Framework is not a partner program, a commercial endorsement scheme, or a symbolic designation. It is a formal conformance and assurance instrument. **Certification under this Framework means that a deployment has demonstrated, through verifiable evidence and reproducible testing, that its enforcement infrastructure satisfies the normative requirements of the QODIQA corpus at a defined assurance level.** Certification does not attest to vendor relationship, market position, or organizational intent. It attests solely to technical conformance.

SECTION 1

Executive Certification Overview

The QODIQA Core Standard defines the normative requirements for deterministic runtime consent enforcement. The Implementation Playbook defines how those requirements are realized in infrastructure. The 68-Point Enforcement Framework defines the enumerated control set against which deployments are assessed. This Certification Framework defines how conformance to those requirements is measured, evidenced, and formally affirmed.

These four documents form the complete QODIQA corpus. Each is normatively subordinate to the Core Standard. This Framework is operationally subordinate to the Implementation Playbook for infrastructure guidance, but it is the authoritative specification for all conformance assessment procedures. No certification activity that deviates from the procedures defined here produces a valid QODIQA certification outcome.

1.1 Purpose of Certification

Certification under this Framework serves a single purpose: to produce a verifiable, time-bounded attestation that a specified deployment of QODIQA enforcement infrastructure satisfies the normative requirements of the QODIQA corpus at a declared assurance level. That attestation must be reproducible, evidence-backed, and falsifiable. An attestation that cannot be independently re-derived from the archived evidence is not a valid certification under this Framework.

> **Certification Anchor:** The technical anchor of all QODIQA certification is the determinism invariant: for any frozen enforcement context C at a declared policy version P, the enforcement function E(C, P) must return an identical outcome on every evaluation. Certification at any level that cannot demonstrate this invariant is not conformant certification.

1.2 Relationship to the Core Standard

The Core Standard defines what a conformant system must do. This Framework defines how conformance to those requirements is demonstrated. The Core Standard is normative over enforcement semantics; this Framework is normative over certification procedure. In the event of conflict between this Framework's procedural requirements and the Core Standard's enforcement requirements, the Core Standard prevails for enforcement semantics. This Framework prevails for all matters of evidence, testing, and attestation procedure.

1.3 Relationship to the Implementation Playbook

The Implementation Playbook specifies how conformant infrastructure is built and operated. This Framework specifies how the resulting infrastructure is assessed. The Playbook's conformance checklist (Sections 10.1 through 10.6) defines the baseline evidentiary requirements. This Framework extends those requirements with structured test procedures, evidence packaging specifications, assessor qualification criteria, and certification lifecycle management. The Playbook and this Framework are complementary instruments within the same corpus; the Playbook is the build specification, and this Framework is the audit specification.

1.4 Institutional Scope

This Framework applies to any entity deploying QODIQA enforcement infrastructure for production use and seeking to make a formal conformance claim. The Framework is equally applicable to self-attestation by the deploying organization, third-party attestation by an independent auditor, or certification by an accredited certification body. The procedures defined here are identical regardless of the attestation pathway; only the authority and independence requirements differ.

DOCUMENTS GOVERNED BY THIS FRAMEWORK

- → QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0
- → QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- → QODIQA: Implementation Playbook for Deterministic Runtime Consent Enforcement, Version 1.0
- → QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0

In the event of conflict, precedence order: Core Standard → 68-Point Framework → Implementation Playbook → This Framework (for procedural matters only).

SECTION 2

Conformance Definition Model

Conformance to the QODIQA corpus is not a binary attribute. It is a structured claim composed of four distinct conformance dimensions, each of which must be independently satisfied for a certification claim to be valid at any level. A deployment that satisfies three of four dimensions is not partially conformant; it is non-conformant for the dimensions it fails, and the overall certification claim must reflect that deficiency.

2.1 Conformance Dimensions

DIMENSION A

Normative Conformance

The deployment implements all required controls specified in the applicable QODIQA normative documents at the declared certification level. All QCC controls are operational. No required component is absent or operating outside its defined failure mode. Normative conformance is assessed by inspection of configuration, architecture, and component behavior against the control definitions in the Core Standard and the 68-Point Framework.

DIMENSION B

Operational Conformance

The deployment operates according to the normative requirements under live conditions. Operational conformance is not satisfied by configuration alone; it requires demonstrated behavior under test conditions, including failure injection, load, and revocation scenarios. A system that is correctly configured but does not behave correctly under operational stress is not operationally conformant.

DIMENSION C

Evidentiary Conformance

The deployment has produced, preserved, and made available for assessment the evidence artifacts required at the declared certification level. Evidence artifacts include test logs, determinism records, revocation propagation measurements, audit chain exports, and policy version snapshots. Evidentiary conformance requires that evidence is authentic, tamper-evident, time-stamped, and traceable to specific test executions. Missing or unverifiable evidence constitutes evidentiary non-conformance.

DIMENSION D

Deterministic Conformance

The enforcement function produces identical outcomes for identical inputs across all evaluation instances, PE restarts, and time intervals, subject only to changes in token state or policy version. Deterministic conformance is validated by the VERIFY_DETERMINISM protocol defined in Section 6. It cannot be inferred from operational conformance alone; it requires an independent reproducibility test against archived test vectors. Any deviation in outcome across identical inputs is a determinism failure that invalidates certification at any level.

2.2 Declared, Verified, and Certified Conformance

Three conformance claim types are recognized under this Framework. They differ in the independence and depth of the assessment process that supports them. They are not interchangeable; no claim type may be represented as another.

CLAIM TYPE	Assessment Method	Independence Requirement	Registry Eligible
Declared Conformance	Self-assessment against this Framework's procedures; evidence produced and retained internally	Assessor is the deploying organization; no independence requirement	No: internal record only
Verified Conformance	Third-party review of evidence and test results by a qualified independent assessor	Assessor must have no organizational relationship to the deploying entity	Yes: with assessor attestation
Certified Conformance	Full assessment, test execution, and evidence review by an accredited certification body	Accredited body must satisfy the independence criteria in Section 9	Yes: with certification body identification

> **Representation Constraint:** A deploying organization must not represent a Declared Conformance as Verified or Certified Conformance in any external communication. Misrepresentation of conformance claim type constitutes a conformance violation and requires immediate correction of all public-facing claims and notification to any affected parties who received the misrepresented claim.

2.3 Conformance Boundary

A conformance claim is always bounded to a specific deployment, a specific certification level, a specific policy version, and a specific assessment date. The claim does not extend to other deployments of the same software, future policy versions, or deployments at higher maturity levels that have not been independently assessed. The conformance boundary must be explicitly declared in any public conformance statement.

A change in any of the following constitutes a boundary-breaking event that requires re-assessment: deployment topology modification; component replacement or major version upgrade; policy version change; change to revocation TTL commitment; change to the audit ledger infrastructure; or a security event affecting the key hierarchy.

SECTION 3

Certification Levels

Four certification levels are defined. Each level is a complete, independently assessable conformance designation. Higher levels are not additive tiers; each level is a fully specified set of requirements that must be satisfied independently. A deployment seeking Level 3 certification must satisfy all Level 3 requirements; it is not sufficient to satisfy Level 1 and Level 2 requirements and claim Level 3 designation. The assessor must evaluate the deployment against the specific, complete requirements of the claimed level.

CL-01 Core Conformance

QCC-01 through QCC-10 · Documented QCC-11/12

REQUIRED CONTROLS

- All seven canonical components deployed and operationally independent (QCC-01)
- EG is sole enforcement gateway; no bypass path (QCC-02)
- Fail-closed behavior verified for all enforcement-path component failures (QCC-03)
- CR implements linearizable revocation flag reads (QCC-04)
- RS propagates revocation within documented TTL ≤ 300s (QCC-05)
- AL is append-only; hash chain integrity verified weekly (QCC-06)
- AE uses HMAC-SHA256 from three-tier key hierarchy (QCC-07)
- Context canonicalization documented and version-tagged (QCC-08)
- Key hierarchy documented; root key offline (QCC-09)
- Required metrics collected; all alert thresholds active (QCC-10)
- QCC-11 and QCC-12 status documented; full satisfaction not required

REQUIRED EVIDENCE

- Architecture diagram with component boundaries signed by infrastructure lead
- Network topology map confirming no inference system bypass path
- Fail-closed test execution log: ≥ 4 component failure scenarios
- CR linearizability configuration extract from storage system
- RS propagation measurement log (minimum 100 events)
- AL chain integrity verification log (minimum 1 full-chain run)
- DETERMINISM_TEST AL record for each deployed policy version
- Key hierarchy documentation with root key storage attestation
- Metric endpoint snapshot at two timestamps showing counter increments
- QCC-11/12 status declaration document signed by technical lead

REQUIRED TESTS

- Component failure injection: CR, PE, AL, AE · verify 100% deny
- Timeout ceiling verification: all components within bounds
- Circuit breaker state test: N failures → OPEN → probe recovery
- Revocation test: commit revocation; confirm propagation within TTL
- VERIFY_DETERMINISM protocol: ≥ 500 vectors per policy version corpus
- AL chain break detection: inject synthetic break; verify alert fires
- Artifact mismatch: tamper artifact; verify inference system refuses

RENEWAL INTERVAL	CLAIM TYPE PERMITTED	FAILURE CONDITION
24 months or on boundary-breaking event	Declared or Verified	Any QCC-01 through QCC-10 not satisfied

CL-02 Verifiable Deterministic Conformance

Replay Validation · Artifact Recomputation · HA Enforcement

REQUIRED CONTROLS

- All CL-01 controls satisfied at CL-01 standard
- CR and AL replicated across ≥ 2 independent failure domains
- EG and PE: ≥ 2 active instances; no shared mutable state
- RS as durable message broker with delivery guarantees
- Revocation TTL ≤ 60 seconds in all failure domains
- Key rotation operational within 30-day deployment key ceiling
- AL hash chain verified daily; out-of-band export daily minimum
- QCC-11 fully satisfied: agent delegation independence verified
- QCC-12 fully satisfied: topology declared; TTL disclosed
- Replay validation: nonce uniqueness enforced and verified
- Artifact recomputation verification: AE output reproducible under same key and nonce

REQUIRED EVIDENCE

- All CL-01 evidence at CL-01 standard
- Multi-domain replication topology diagram with failure domain boundaries
- RS delivery guarantee configuration extract
- Key rotation log: AL KEY_ROTATION_EVENT records for 90 days
- Dual-key window test log: artifact accepted under both keys during rotation
- Nonce replay test: AL NONCE_REPLAY_DETECTED record from test execution
- Artifact recomputation test: two artifacts from identical inputs, same key and nonce · must be bit-identical
- Agent delegation independence test log: QCC-11 simulation results
- Topology declaration signed document with TTL disclosure

REQUIRED TESTS

- All CL-01 tests executed at CL-01 standard
- Failure domain isolation: terminate one failure domain; verify other continues enforcing
- Revocation within 60s: 50 revocation events; measure P99 propagation latency
- Replay attack simulation: re-submit prior request with same nonce; verify DENY
- Key rotation live test: rotate key during active traffic; verify no enforcement interruption
- VERIFY_DETERMINISM with PE restart between passes; corpus ≥ 1000 vectors
- Artifact recomputation: AE restart; same inputs; verify identical output

RENEWAL INTERVAL
18 months or on boundary-breaking event

CLAIM TYPE PERMITTED
Verified or Certified

FAILURE CONDITION
Any determinism or replay test failure; any CL-01 failure

CL-03 Enterprise Assurance Conformance

Revocation SLA · Audit Durability · Monitoring Controls

REQUIRED CONTROLS

- All CL-02 controls satisfied at CL-02 standard

REQUIRED EVIDENCE

- All CL-02 evidence at CL-02 standard
- 90-day revocation propagation histogram export; P99 ≤ TTL

REQUIRED TESTS

- All CL-02 tests executed at CL-02 standard

- Revocation TTL ≤ 60s committed as SLA; measured, not aspirational
- RS propagation P99 ≤ TTL verified in production metric data
- AL durability: continuous export to operator-controlled immutable storage
- AL hash chain integrity verified continuously (per-write at CL-03)
- Enforcement path SLO ≥ 99.9% over 90-day measured window
- Integrity heartbeat operational: 60s maximum verification interval
- Drift classification matrix implemented: Class I through Class 0 alerts active
- SLA breach escalation invariants operational and tested
- Conservative mode tested: RS propagation delay triggers affected consent_id deny

- AL immutable storage export receipts for 90 days
- AL per-write chain verification log for 30 days
- Enforcement path SLO measurement report: 90-day availability ≥ 99.9%
- Integrity heartbeat run log: all passes; any failures with resolution record
- Conservative mode test: propagation delay injected; affected consent_ids denied; AL records present
- Class I and Class 0 alert test records: each condition triggered; escalation confirmed

- Revocation SLA validation: 500 events across 30 days; P99 ≤ TTL confirmed
- Conservative mode trigger: delay RS propagation beyond TTL/2; verify affected consent_id denial activates
- Per-write chain integrity: inject synthetic break during write; verify immediate Class 0 alert and AL suspension
- Integrity heartbeat failure: disable one check; verify heartbeat escalates within 2 minutes
- SLO boundary test: simulate 0.1% enforcement path unavailability; verify Class I alert within threshold

RENEWAL INTERVAL
12 months or on boundary-breaking event

CLAIM TYPE PERMITTED
Certified only

FAILURE CONDITION
SLO breach; revocation P99 > TTL; any CL-02 failure

CL-04 High-Assurance Conformance

Optional Advanced Tier · Independent Determinism Audit · Multi-Region

REQUIRED CONTROLS

- All CL-03 controls satisfied at CL-03 standard
- Multi-region deployment with linearizable cross-region revocation
- Revocation TTL ≤ 15s across all regions; P99 verified in production data

REQUIRED EVIDENCE

- All CL-03 evidence at CL-03 standard
- Multi-region revocation propagation histogram: P99 ≤ 15s per region pair
- HSM configuration and partition audit for each AE instance

REQUIRED TESTS

- All CL-03 tests executed at CL-03 standard
- Multi-region revocation: 1000 events; all regions within TTL
- Cross-region partition: isolate Region A from Region B; verify conservative mode for affected consent_ids

- HSM-backed AE mandatory in all regions
- PE and AE in hardware secure enclave with verified remote attestation
- Enclave attestation archived in AL at each initialization
- Formal determinism audit by independent third party pre-deployment
- Multi-region propagation race-condition testing completed
- Cross-region consensus divergence scenario tested (FC-04b)
- Annual penetration test of enforcement infrastructure
- Enforcement path SLO ≥ 99.99% over 90-day measured window
- Full AL audit trail independently verifiable without SaaS intermediation

- Remote attestation records: ENCLAVE_ATTESTATION AL records for each enclave
- Independent third-party determinism audit report with methodology disclosure
- Cross-region divergence test log: FC-04b scenario executed; conservative mode activated; recovery confirmed
- Penetration test report from qualified independent tester: enforcement path scope only
- Independent AL verification procedure: verifier operates without deploying organization tooling
- 90-day SLO measurement at ≥ 99.99%

- HSM failover: simulate HSM partition saturation; verify AE conservative scale-out
- Enclave restart attestation: restart PE enclave; verify attestation re-verified before key provisioning
- Independent AL chain verification: assessor runs chain verification from exported data without deployer tooling
- Clock skew edge case: inject skew > tolerance; verify timestamp-based denies and ML alert

RENEWAL INTERVAL	CLAIM TYPE PERMITTED	FAILURE CONDITION
12 months; independent third-party re-audit required	Certified only: accredited body	Any cross-region invariant failure; enclave attestation failure; any CL-03 failure

SECTION 4

Certification Evidence Requirements

Evidence artifacts are the material basis of all conformance claims. An evidence artifact is a structured, time-stamped, tamper-evident record produced by a specific test execution, measurement collection, configuration inspection, or system event. Evidence artifacts are not narratives or self-descriptions; they are machine-verifiable records that a qualified assessor can independently examine and cross-reference against the requirements they attest to.

Evidence Integrity Requirement: All evidence artifacts must be produced at the time of the activity they record. Retroactive evidence construction is not permitted. An evidence artifact that cannot be time-stamped to within the window of the assessed deployment period is not valid evidence under this Framework, regardless of its content.

4.1 Evidence Artifact Taxonomy

PS	**Policy Version Snapshot** Frozen rule set with SHA-256 content hash; corresponding AL POLICY_FROZEN record. Integrity via AL hash chain and snapshot hash in POLICY_FROZEN record.	Required from CL-01
TEL	**Test Execution Log** Structured log of each test step with timestamp, inputs, observed outputs, and pass/fail status. Log signed by test executor; hash archived in AL under CONFORMANCE_TEST record type.	Required from CL-01
DTR	**Determinism Test Record** AL DETERMINISM_TEST record: corpus hash, vector count, two-pass results, policy version binding. AL hash chain; corpus archived separately with matching hash.	Required from CL-01
RPM	**Revocation Propagation Metric Export** Time-series histogram of `qodiqa_revocation_propagation_ms` per EG instance and region. Signed export from ML system; hash recorded in evidence package.	Required from CL-01
AHS	**Artifact Hash Sample** Set of ≥ 100 HMAC-SHA256 artifacts with corresponding consent_id, context_hash, policy_version; verification pass/fail per sample. Verification result signed by assessor; source artifacts from AL records.	Required from CL-01
ACV	**Audit Chain Verification Result** Output of chain verification procedure: partition, record range, CHAIN_VALID or CHAIN_BREAK at position k. Verification tooling output signed by assessor; reproducible from AL export.	Required from CL-01

CFL	**Control Failure Log** Record of each Class I and Class 0 condition triggered during testing: alert timestamp, response action, resolution. ML alert log export signed; cross-referenced with AL ENFORCEMENT_HALTED records.	Required from CL-01
KRA	**Key Rotation Event Archive** AL KEY_ROTATION_EVENT records for the assessed period; key fingerprint chain traceable to root. AL hash chain integrity; root key fingerprint independently verifiable.	Required from CL-02
RTR	**Replay Test Record** Test execution: nonce reuse scenario; AL NONCE_REPLAY_DETECTED record with timestamp. AL hash chain; test log signed by assessor.	Required from CL-02
RAR	**Remote Attestation Record** AL ENCLAVE_ATTESTATION records per enclave instance: measurement, certificate chain, timestamp. AL hash chain; attestation certificate chain verifiable against hardware vendor trust anchor.	Required from CL-04

4.2 Evidence Package Specification

Evidence for a certification assessment must be assembled into a structured evidence package. The evidence package is the complete, self-contained artifact set submitted to the assessor. It must include: a manifest listing all artifact files with their SHA-256 hashes; a conformance claim statement identifying the certification level, deployment boundary, and policy versions assessed; copies of all required evidence artifacts with timestamps and provenance; AL export for the assessment period covering all normative record types; and a signed declaration from the infrastructure lead attesting to the completeness and authenticity of the package contents.

The assessor must not accept an evidence package that is missing any required artifact type for the claimed certification level, that contains artifacts whose timestamps fall outside the assessed deployment period, or whose AL export hash chain fails verification. An incomplete or unverifiable evidence package is treated as evidentiary non-conformance, equivalent to a test failure for the controls whose evidence is missing.

4.3 Minimum Evidence Set by Level

TABLE 4.2: MINIMUM EVIDENCE SET BY CERTIFICATION LEVEL

ARTIFACT TYPE	CL-01	CL-02	CL-03	CL-04
Policy Version Snapshot	Required	Required	Required	Required
Test Execution Log	Required	Required	Required	Required
Determinism Test Record	Required	Required	Required	Required
Revocation Propagation Metrics	Required	Required	Required (90 days)	Required (90 days, all regions)
Artifact Hash Sample	Required	Required	Required	Required
Audit Chain Verification Result	Required	Required	Required (daily run)	Required (independent verifier)
Control Failure Log	Required	Required	Required	Required
Key Rotation Event Archive	Not required	Required	Required	Required
Replay Test Record	Not required	Required	Required	Required
Remote Attestation Record	Not required	Not required	Not required	Required

SECTION 5

Conformance Test Methodology

Conformance testing under this Framework is structured testing against defined pass/fail criteria. Tests are deterministic in structure: given a specified test precondition, a specified stimulus, and a specified expected outcome, the test either passes or fails. There are no partial passes. A test result of "mostly compliant" or "substantially conformant" is not recognized; each test either produces the required outcome or it does not.

All conformance tests must be executed in an environment that is representative of the production deployment being assessed. Tests executed in a development or staging environment that differs from production in topology, component versions, or configuration do not produce valid evidence for a production deployment assessment.

5.1 Test Vector Validation

Test vector validation is the process of verifying that the enforcement function produces the expected decision for each member of a defined test corpus. The test corpus must be constructed before test execution and must be immutable during execution. The corpus hash must be computed and archived before the first evaluation pass. Any modification to the corpus after its hash is archived invalidates the test.

TEST VECTOR CORPUS: CONSTRUCTION REQUIREMENTS — SPEC

```
// Corpus composition requirements: minimum 500 vectors at CL-01
// Minimum 1000 vectors at CL-02 through CL-04

Mandatory vector categories:
  ACTIVE_token_full_scope_match,        expected: PERMIT  (≥ 10%)
  ACTIVE_token_scope_mismatch,          expected: DENY    (≥ 10%)
  REVOKED_token,                        expected: DENY    (≥ 10%)
  EXPIRED_token,                        expected: DENY    (≥ 10%)
  NOT_FOUND_token,                      expected: DENY    (≥ 5%)
  UNKNOWN_issuer,                       expected: DENY    (≥ 5%)
  boundary_context_hash_valid,          expected: PERMIT  (≥ 5%)
  boundary_context_hash_invalid,        expected: DENY    (≥ 5%)
  default_deny_no_matching_rule,        expected: DENY    (≥ 5%)
  policy_version_not_found,             expected: DENY    (≥ 5%)
  combined_multi_condition_permit,      expected: PERMIT  (≥ 5%)
  combined_multi_condition_deny,        expected: DENY    (≥ 5%)
  // Remaining vectors: operator-defined for deployment-specific policy coverage
```

```
Corpus serialization:
  corpus_hash = SHA-256(canonical_serialize(all_vectors_sorted_by_id))
  corpus_hash archived before first evaluation pass
  corpus stored with evidence package; verifiable by assessor
```

5.2 Negative Test Cases

Negative test cases verify that the enforcement system does not permit what it must not permit. They are as important as positive test cases: a system that permits correctly but fails to deny correctly is non-conformant. The following negative test categories are mandatory at all certification levels:

- Revoked token must be denied; ACTIVE state returned by a non-linearizable read does not relax this requirement
- Request with no consent_id field must be denied at the EG extraction stage before CR query
- Request with invalid context_hash format (not matching sha256: prefix and format) must be denied
- Request referencing an unknown policy version must be denied by PE without evaluation
- Request from a caller with an expired mTLS certificate must be denied at transport layer before enforcement evaluation
- Request submitted after rate limit exceedance must be denied with RATE_LIMIT_EXCEEDED and produce AL record
- Request where AL write fails must never result in inference execution, regardless of PE decision
- Inference execution with missing, malformed, or HMAC-invalid artifact must be refused by inference system

5.3 Fail-Closed Validation Procedure

FCT-01	**CR Component Failure Injection** Terminate or network-isolate CR process. Submit 50 requests spanning at least 10 distinct consent_ids. Verify: (a) 100% of requests denied; (b) all AL records show DENY with reason CR_UNAVAILABLE; (c) zero inference executions recorded; (d) CIRCUIT_OPEN alert dispatched within circuit window W.	Pass criterion: 100% deny, 0 permits
FCT-02	**PE Component Failure Injection** Terminate PE process after CR is confirmed operational. Submit 20 requests for ACTIVE tokens. Verify: (a) 100% denied; (b) AL records show DENY with PE_UNAVAILABLE; (c) zero inference executions; (d) CIRCUIT_OPEN alert for PE dispatched.	Pass criterion: 100% deny
FCT-03	**AL Write Failure Injection** Network-partition EG→AL path. Submit 10 requests for ACTIVE tokens eligible for PERMIT. Verify: (a) PE returns PERMIT in all 10 cases; (b) AE generates artifact in all 10 cases; (c) none of the 10 requests reach inference system; (d) EG returns error to caller; (e) AUDIT_WRITE_FAILURE alert dispatched.	Pass criterion: 0 inference executions
FCT-04	**AE Signing Failure Injection** Corrupt or revoke the AE signing key. Submit 10 PERMIT-eligible requests. Verify: (a) PE returns PERMIT; (b) AE fails to generate artifact; (c) EG blocks all 10 requests; (d) inference system receives zero requests; (e) AE_UNAVAILABLE alert dispatched.	Pass criterion: 0 inference executions
FCT-05	**Timeout Ceiling Verification** For each component (CR, PE, AE, AL): configure artificial response delay exceeding the configured timeout ceiling. Verify EG does not wait beyond ceiling; request is denied; AL receives DENY record; timeout alert dispatched. Verify timeout values are within bounds specified in Implementation Playbook §5.	Pass criterion: denial at ceiling ± 5ms

5.4 Revocation Race Condition Test

The revocation race condition test verifies that the system handles concurrent enforcement and revocation correctly and that the audit record correctly reflects both events. This test must be executed under load conditions representative of the production deployment, not under idle conditions.

REVOCATION RACE CONDITION TEST: PROCEDURE

PROCEDURE

```
procedure REVOCATION_RACE_TEST(consent_id C, load_rps R):

  Step 1: Establish baseline
    Confirm C.state = ACTIVE; run 100 requests for C at R req/s
    Verify all 100 produce PERMIT with AL records

  Step 2: Concurrent revocation
    At T0: begin continuous requests for C at R req/s
    At T0 + 50ms: commit revocation of C via RS
    Continue requests for C until T0 + TTL + 100ms

  Step 3: Collect results
    Extract all AL records for C between [T0, T0 + TTL + 100ms]
    Classify records as: PERMIT_BEFORE_REVOCATION or DENY_AFTER_REVOCATION

  Step 4: Verify invariants
    (a) All PERMIT records must have committed_at < revocation_committed_at + epsilon
        where epsilon = revocation propagation tolerance (max: TTL)
    (b) All DENY records after T0 + TTL must have reason = TOKEN_REVOKED
    (c) No PERMIT record exists at committed_at > T0 + TTL
    (d) Revocation propagation: qodiqa_revocation_propagation_ms P99 ≤ TTL
    (e) Conservative mode: if any EG did not acknowledge revocation at TTL/2,
        verify that EG applied DENY for affected consent_ids

  Pass criterion: all four invariants satisfied; no PERMIT after T0 + TTL
```

5.5 Policy Drift Detection Test

Policy drift occurs when a deployed PE instance diverges from the canonical frozen policy snapshot, either through unauthorized modification, version mismatch across PE instances, or snapshot corruption. The policy drift detection test verifies that all PE instances report identical policy versions and snapshot hashes, and that the integrity heartbeat detects any divergence.

- Query policy version and snapshot hash from all active PE instances; confirm they are identical. Any divergence is an immediate test failure.
- Verify the snapshot hash reported by each PE instance matches the SHA-256 hash in the corresponding AL POLICY_FROZEN record
- Inject a single-bit modification to a PE instance's local policy snapshot; verify PE detects the tamper at next integrity check and emits POLICY_LOAD_FAILURE
- Deploy a test PE instance with a syntactically valid but semantically different policy snapshot; verify it is rejected and does not join the enforcement pool
- Verify the integrity heartbeat detects PE policy version divergence within its scheduled interval

5.6 Artifact Recomputation Test

Artifact recomputation testing verifies that the Artifact Emitter produces bit-identical output for identical inputs under the same key and nonce. This is distinct from the determinism test: the determinism test verifies that the enforcement decision is reproducible; the artifact recomputation test verifies that the artifact encoding of that decision is reproducible. Both must hold for certification at CL-02 and above. A system that makes correct enforcement decisions but produces non-deterministic artifacts violates the integrity model because the inference system cannot reliably verify the artifact against a known expected value.

ARTIFACT RECOMPUTATION TEST: FORMAL PROCEDURE ART

PROCEDURE

```
procedure ARTIFACT_RECOMPUTATION_TEST(sample_size N ≥ 100):

  Preconditions:
    AE is operational; PE is operational; current key K is active
    No key rotation in progress during test execution

  Step 1: Generate N input tuples
    ∀ i ∈ [1..N]:
      T_i = (consent_id_i, context_hash_i, policy_version, decision_i, nonce_i)
      A_i_first = AE.emit(T_i, K)
      Record A_i_first with timestamp ts_first_i

  Step 2: AE restart
    Perform controlled AE process restart
    Verify remote attestation re-executed (CL-04: mandatory; CL-02/03: log-only)
    Verify AE reports same key K loaded and verified

  Step 3: Recompute artifacts
    ∀ i ∈ [1..N]:
      A_i_second = AE.emit(T_i, K)    // same input tuple, same key, same nonce
      Record A_i_second with timestamp ts_second_i

  Step 4: Bit-identity verification
    ∀ i ∈ [1..N]:
      if A_i_first ≠ A_i_second: FAIL: non-deterministic artifact emission
        record: index=i, first_hash=SHA256(A_i_first), second_hash=SHA256(A_i_second)
        test terminates; all N results must pass

  Step 5: Inference system verification
    Submit A_i_first to inference system for 20 sampled inputs; verify acceptance
    Submit A_i_second to inference system for same 20 inputs; verify acceptance
```

```
    Verify inference system acceptance outcome is identical for both versions

Pass criterion: ∀ i: A_i_first = A_i_second (byte-exact);
  inference system accepts both versions identically
Evidence: test log with input tuple hashes, artifact hashes, restart timestamp;
  archived to AL as ARTIFACT_RECOMPUTATION_TEST record type
```

The artifact recomputation invariant must hold across all three failure modes: (a) AE process restart with key reload from persisted key store; (b) AE instance replacement within a multi-instance AE pool; and (c) AE recovery from a DENY state following a transient key verification failure. In all three cases, the recomputed artifact for a previously seen input must be bit-identical to the original artifact produced under the same key and nonce. An AE implementation that uses any timestamp, randomness source, or instance-specific state in the artifact encoding is structurally non-conformant at CL-02 and above.

SECTION 6

Determinism Verification Protocol

Determinism is the fundamental assurance property of QODIQA enforcement. It means that the enforcement function is a pure mathematical function of its inputs: given identical token state, context hash, and policy version, the function must return an identical decision on every invocation, on every PE instance, after every restart, and across every deployment that holds the same frozen policy snapshot. Determinism verification is a certification requirement at all levels and cannot be waived or approximated.

The deterministic certification invariant: for all frozen contexts C and policy versions P, `evaluate(C, P) = replay(C, P)`. Any implementation for which this invariant fails for any vector in the test corpus fails the determinism requirement and cannot receive certification at any level.

6.1 Frozen Context Extraction Method

A frozen context is the canonicalized, version-tagged representation of a processing request's enforcement-relevant attributes at the moment of evaluation. Frozen context extraction must produce a context object that is both complete and irreducible: complete in that all attributes that could affect the enforcement decision are present; irreducible in that no attribute is included that does not affect the enforcement decision. The frozen context extraction procedure must be identical to the canonical serialization procedure defined in the Implementation Playbook §8.1.

FROZEN CONTEXT EXTRACTION: CANONICAL INVARIANTS — INVARIANTS

```
// FC-EXT-INV-01: Completeness
frozen_context(request R) must include:
  { consent_id, context_hash, policy_version,
    token.state, token.issuer, token.scope,
    token.issued_at, token.expires_at,
    request_nonce, caller_id }

// FC-EXT-INV-02: Canonical hash reproduction
canonical_hash(frozen_context) must be deterministic:
  ∀ C₁, C₂: semantically_equal(C₁, C₂) → canonical_hash(C₁) = canonical_hash(C₂)
  ∀ C: canonical_hash(C) is invariant under field reordering, whitespace, encoding

// FC-EXT-INV-03: Policy binding
frozen_context.policy_version must reference a FROZEN policy version
```

```
  PE must reject evaluation requests for non-FROZEN versions
  Frozen context produced for a non-FROZEN policy version is invalid for certification

// FC-EXT-INV-04: Timestamp exclusion
evaluation_timestamp must NOT be included in the canonical context hash
  The hash must be invariant across time for identical token states
  Token expiry is evaluated against the EG clock, not baked into the hash

// FC-EXT-INV-05: Scope normalization
token.scope must be canonical-sorted before hashing
  scope = sort_lexicographic(split(raw_scope, delimiter))
  Scope presentation order must not affect evaluation outcome
```

6.2 Hash Reproduction Test

The hash reproduction test verifies that the canonical serialization procedure produces identical hashes for semantically equivalent contexts. This test is a prerequisite for all higher-order determinism tests, because if the canonicalization procedure is not reproducible, the determinism test results are not interpretable.

HASH REPRODUCTION TEST: PROCEDURE HRT — PROCEDURE

```
procedure HASH_REPRODUCTION_TEST(sample_size M ≥ 50):

  Step 1: Generate M base context objects with known canonical hashes
  Step 2: For each base context C_i, generate variants:
    V_i_1: field reordering (same fields, different JSON key order)
    V_i_2: whitespace injection (spaces after colons and commas)
    V_i_3: scope reordering (same scope values, different sequence)
    V_i_4: encoding variation (ASCII-escaped vs UTF-8 for non-ASCII values)

  Step 3: For each variant V_i_k:
```

```
    h_variant = canonical_hash(V_i_k)
    if h_variant ≠ canonical_hash(C_i): FAIL: non-deterministic canonicalization
      record: variant type V_i_k, divergent hash pair, test failure

Step 4: Verify floating-point rejection
  Inject a context with a floating-point numeric field
  Verify: rejected with CONTEXT_TYPE_VIOLATION before hash computation

Pass criterion: all M×4 variants produce hash identical to base context hash
Evidence: hash reproduction test log with all input/output pairs archived
```

6.3 Policy Version Binding Verification

Policy version binding verification confirms that the enforcement function is bound to the declared policy version at the time of evaluation and that the binding is enforced at the PE layer, not inferred, approximated, or subject to runtime policy loading. The verifier must confirm that PE holds the exact policy snapshot whose SHA-256 hash matches the POLICY_FROZEN AL record for the declared policy version.

- Query PE for the current loaded policy version and snapshot hash; verify match against AL POLICY_FROZEN record for that version
- Submit an evaluation request referencing an unrecognized policy version; verify DENY with reason POLICY_VERSION_NOT_FOUND before any rule evaluation
- Submit a request referencing a valid but deprecated policy version (not the current active version); verify whether the deployment correctly handles version plurality or restricts to a single active version, per operator declaration
- Verify that the PE snapshot hash is computed at startup and verified before accepting any evaluation request; a PE that begins accepting requests before snapshot verification is non-conformant

6.4 Replay Invariant Validation

Replay invariant validation verifies that submitting the same enforcement request a second time produces the same enforcement outcome, given identical token state and policy version, but is correctly blocked if the request nonce has been previously consumed. These two properties are distinct and must be tested separately.

DETERMINISM TEST: SAME INPUTS → SAME DECISION	REPLAY DETECTION TEST: SAME NONCE → DENY
1. Submit request R with unique nonce N1; record decision D1.	1. Submit request R with nonce N1; record decision D1; confirm AL record.
2. Submit identical request R with a new unique nonce N2 (different nonce).	2. Submit identical request R with the same nonce N1 again.
3. Record decision D2.	3. Verify: second submission denied with NONCE_REPLAY_DETECTED.
4. Verify: D1 = D2 (same token state, context, policy version).	4. Verify: AL record for second submission shows DENY.
5. Repeat with PE restart between N1 and N2 submissions.	5. Verify: no second inference execution occurred.
✓ *Pass: D1 = D2 across all repetitions including PE restart*	✓ *Pass: N1 reuse denied; AL record present; zero second executions*

SECTION 7

Audit and Ledger Integrity Validation

The Audit Ledger (AL) is the evidentiary foundation of QODIQA enforcement accountability. Its integrity is not a performance characteristic; it is a precondition for any conformance claim. A certification assessment that cannot verify AL chain integrity is incomplete regardless of the results of all other tests. AL integrity validation is mandatory at every certification level.

7.1 Hash-Chain Validation Process

HCV-01	**Export and Hash Verification** Export the AL partition under assessment. Verify the export signature against the AL component's public signing key. Compute SHA-256 of the export content; verify it matches the export manifest hash. A signature failure or hash mismatch at this step means the export is not a valid evidence artifact.	Prerequisite to all subsequent steps
HCV-02	**Record Ordering Verification** For each partition, verify that records are ordered by `committed_at` timestamp and that no two records share an identical timestamp within the same partition. Duplicate timestamps indicate either a clock failure or a record injection attempt and must be reported as a chain integrity anomaly.	Anomaly: duplicate timestamps
HCV-03	**Chain Link Verification** For each consecutive record pair (r_n, r_{n+1}) in each partition: compute SHA-256 of r_n's canonical serialization; verify it equals r_{n+1}.prev_hash. Any mismatch at position k is a chain break at k. Record the break position, the computed hash, and the stored prev_hash.	Chain break at k = test failure
HCV-04	**Completeness Verification** Cross-reference AL records against the enforcement request log for the assessment period. For each enforcement decision in the request log, verify that a corresponding AL record exists. Any enforcement decision without an AL record is an audit completeness failure, more severe than a chain break because it represents a missing accountability record.	Missing record = completeness failure
HCV-05	**Record Type Distribution Verification** Verify that the expected distribution of AL record types is present: PERMIT, DENY, KEY_ROTATION_EVENT, POLICY_FROZEN, DETERMINISM_TEST, and ENFORCEMENT_HALTED (if applicable). An AL partition that contains only PERMIT records without any KEY_ROTATION_EVENT records over a 90-day period is anomalous and must be flagged for manual review.	Anomalous distribution = flagged

7.2 Log Tamper Detection

Tamper detection for the AL relies on the combination of the hash chain and the signed export mechanism. A tamper that modifies, deletes, or reorders records within a partition will produce a chain break at the tamper location. A tamper that

appends fabricated records at the end of the chain cannot be detected by the chain alone; it requires cross-referencing the export against a prior integrity checkpoint or against an independently maintained record count.

At CL-03 and CL-04, the per-write chain integrity verification provides near-real-time tamper detection. Any write that does not produce a valid chain extension must trigger an immediate Class 0 alert. At CL-01 and CL-02, weekly and daily chain verification schedules allow for a longer tamper detection window, which must be disclosed in the conformance claim as part of the stated assurance boundary.

7.3 Retention Compliance Verification

Audit record retention requirements are defined by the operator's declared retention policy, which must be disclosed as part of the conformance boundary. The assessor must verify that: records in the AL have not been deleted or truncated within the declared retention period; the AL export for the assessed period is complete and contiguous; and the immutable storage export (where required at CL-02 and above) contains records back to the beginning of the declared retention period with unbroken chain continuity.

7.4 Sampling Strategy for Audit Validation

For large-scale deployments where the AL contains millions of records per day, full-chain verification on every record may be computationally intensive. The following sampling strategy is permitted for CL-01 and CL-02 assessments where full verification is impractical within the assessment window. At CL-03 and CL-04, full-chain verification is required and sampling is not permitted.

- **Under 100,000 records/day:** Full chain verification of all records: no sampling permitted.
- **100K to 1M records/day:** Minimum 10% of records per day. Sample selection: systematic: every N-th record, where N = floor(daily count / sample size). If any sampled break is found: full verification of the surrounding 10,000 records.

- **1M to 10M records/day:** Minimum 1% of records per day, with a floor of 10,000 records. Sample selection: stratified random, proportional across all record types. If any sampled break is found: full verification of the surrounding 50,000 records.
- **Over 10M records/day:** Minimum 0.1% with a floor of 10,000 records; mandatory boundary sampling required. Sample selection: stratified random plus mandatory boundary sampling. Any break triggers full partition verification.

This sampling strategy applies to CL-01 and CL-02 assessments only. At CL-03 and CL-04, full-chain verification is required and sampling is not permitted.

SECTION 8

Revocation Propagation Certification

The revocation propagation guarantee is the time-bounded commitment that a revocation event committed to the Consent Registry will be acknowledged by all Enforcement Gateway instances within a declared TTL. This guarantee is a contractual commitment to affected parties. Certification of this guarantee requires demonstrated measurement, not configuration inspection alone.

8.1 Revocation SLA Verification

Revocation SLA verification is the process of confirming that the measured revocation propagation latency does not exceed the declared TTL at the P99 percentile. SLA verification requires production metric data, not test-environment measurements. The assessor must examine the `qodiqa_revocation_propagation_ms` histogram from the production ML system, covering the most recent 30 days of operation at CL-01 and CL-02, and the most recent 90 days at CL-03 and CL-04.

REVOCATION SLA VERIFICATION: FORMAL PROCEDURE

PROCEDURE

```
procedure REVOCATION_SLA_VERIFY(metric_export M, declared_TTL T):

  Step 1: Verify metric export integrity
    M.signature verified against ML signing key
    M.period_start, M.period_end within assessment window

  Step 2: Per-EG-instance P99 check
    ∀ EG instance i ∈ M.instances:
      P99_i = percentile(M.latencies[i], 99)
      if P99_i > T: FAIL: SLA violation at EG instance i
        record: instance_id=i, P99=P99_i, declared_TTL=T, violation_delta=P99_i-T

  Step 3: Worst-case P99 check
    P99_max = max(P99_i for all i)
    if P99_max > T: FAIL: aggregate SLA violation
    if P99_max > T * 0.9: WARNING: operating within 10% of TTL ceiling

  Step 4: Event count validation
    total_events = count(M.events)
    if total_events < minimum_events_threshold: INCONCLUSIVE
      minimum_events_threshold = max(1000, estimated_daily_revocations × period_days × 0.1)

  Step 5: Conservative mode activation record
    Query AL for CONSERVATIVE_MODE_ACTIVATED records in assessment period
    Each CONSERVATIVE_MODE_ACTIVATED record must have a corresponding
    CONSERVATIVE_MODE_RESOLVED record within the same TTL window
    Unresolved conservative mode records are a SLA certification failure

  Pass criterion: all P99_i ≤ T; event count ≥ threshold;
    no unresolved CONSERVATIVE_MODE_ACTIVATED records
```

8.2 TTL Boundary Enforcement

TTL boundary enforcement testing verifies that the system enforces the TTL boundary as a hard constraint, not a soft target. The test must verify that no PERMIT is issued for a consent_id whose revocation was committed at time T_rev at any EG instance after T_rev + TTL, regardless of cache state or propagation lag. This invariant must hold under load, under network stress, and immediately following an EG restart.

- At T_rev, commit revocation of C. At T_rev + TTL + 1s, submit a request for C at all EG instances. Verify all instances return DENY.
- Verify that EG instances for which propagation was delayed beyond TTL activated conservative mode before the TTL expired and not after.
- Verify that requests submitted between T_rev and T_rev + TTL are classified correctly: PERMITs before revocation reached the EG are acceptable; PERMITs after revocation reached the EG are failures.
- Restart an EG instance at T_rev + TTL/2. Verify that the restarted instance does not issue a PERMIT for C (even before it has received the propagated revocation) by requiring a fresh CR read on restart before resuming enforcement.

8.3 Stale Registry Detection

Stale registry detection testing verifies that the CR cache correctly expires and that stale ACTIVE states are not served beyond their cache TTL. The test simulates a scenario where the CR network is transiently unavailable, after which the cache TTL expires and the EG must either re-query CR or deny due to unavailability, and must not serve the stale ACTIVE state beyond the cache TTL bound.

8.4 Time-Bound Validation Procedure

TIME-BOUND REVOCATION VALIDATION: TBV PROCEDURE — PROCEDURE

```
procedure TIME_BOUND_VALIDATION(consent_id C, declared_TTL T, n_EG_instances K):

  Precondition: C.state = ACTIVE; all K EG instances operational

  Phase A: Baseline (t = 0 to t = 60s):
    Submit 10 requests per EG instance; verify all PERMIT
    Record baseline latencies and AL record timestamps

  Phase B: Revocation Commit (t = 60s):
    T_rev = now()
    Commit revocation of C via RS; record T_rev precisely (monotonic clock)
    RS emits revocation event to all K EG instances

  Phase C: Window Monitoring (t = 60s to t = 60s + TTL):
    Every 5s: submit one request to each EG instance
    Record: (EG_instance_id, request_time, decision, AL_record)
    Any PERMIT after T_rev is acceptable only if request_time < T_rev + propagation_lag_i

  Phase D: Post-TTL Boundary (t = 60s + TTL + 1s):
    Submit 5 requests to each EG instance
    Assert: ALL decisions = DENY with reason = TOKEN_REVOKED
    Assert: zero PERMITs across all K instances after T_rev + TTL
    If any PERMIT: FAIL: TTL boundary violated at instance i

  Phase E: Measurement
    For each EG instance i: propagation_lag_i = first_DENY_at_i - T_rev
    Verify: ∀ i: propagation_lag_i ≤ T
    Record: P50, P95, P99 across all instances

  Evidence: full AL export for C over [T_rev - 60s, T_rev + TTL + 60s];
    metric export for revocation_propagation_ms covering Phase C
```

SECTION 9

Certification Authority Model

Certification authority is the organizational and procedural structure through which a certification claim is evaluated and affirmed. This Framework defines three authority pathways. Each pathway produces a different claim type with different independence and credibility properties. The authority pathway must be declared as part of every public conformance statement.

9.1 Internal Self-Attestation

Internal self-attestation is the process by which a deploying organization assesses its own deployment against this Framework's requirements, produces the required evidence, and issues a Declared Conformance claim. Internal self-attestation is permitted at CL-01 and CL-02. It is not permitted at CL-03 or CL-04.

An internal self-attestation is valid only if: the assessment is performed by personnel who were not responsible for the design or implementation of the deployment being assessed; the evidence package is reviewed by a technical lead independent of the assessment team; and the resulting conformance claim is documented internally with the evidence package retained for at least the duration of the certification validity period plus six months.

9.2 Independent Third-Party Attestation

Independent third-party attestation is the process by which a qualified assessor not affiliated with the deploying organization reviews the evidence package and, where required by the certification level, executes additional verification procedures. A third-party attestation produces a Verified Conformance claim. Third-party attestation is required for CL-02 claims submitted to the public registry, and is the minimum pathway for CL-03.

INDEPENDENT ASSESSOR QUALIFICATION REQUIREMENTS

- No current or prior employment, contractual, or financial relationship with the deploying organization within the preceding 24 months
- Demonstrable technical qualification in distributed systems, cryptographic systems, or high-assurance infrastructure assessment
- Familiarity with the QODIQA corpus sufficient to evaluate evidence against normative requirements without interpretation assistance from the deploying organization
- No financial interest in the outcome of the assessment (fixed-fee engagements only; outcome-contingent fees are disqualifying)
- Assessor identity and qualification disclosed in the Verified Conformance claim

Qualification is self-declared by the assessor. The deploying organization is responsible for verifying qualification before engaging the assessor.

9.3 Accredited Certification Bodies

An accredited certification body (ACB) is an organization that has established formal procedures for conducting QODIQA certification assessments, employs or contracts qualified assessors, maintains assessment records, and issues Certified Conformance claims. ACBs are required for CL-04 certification and are an option for CL-03.

An ACB must satisfy the following structural requirements: independent governance with no financial dependency on the deploying organizations it certifies; documented assessment procedures consistent with this Framework; assessor qualification standards at least as rigorous as §9.2; a formal certification revocation procedure; and public disclosure of the certification methodology used. An ACB that fails to satisfy any of these requirements may not issue claims under this Framework.

9.4 Conflict-of-Interest Rules

The following relationships between an assessor or ACB and a deploying organization are disqualifying and require disclosure to any party relying on the certification claim: equity ownership or financial interest in the deploying organization; provision of implementation services, architecture design, or infrastructure components to the deployment

under assessment; participation in the design or specification of the deployment's enforcement infrastructure; and any relationship that would create an incentive for the assessor to produce a positive certification outcome regardless of evidence.

When a conflict of interest is discovered after a certification has been issued, the certification must be suspended pending review. The deploying organization must be notified immediately. The certification may be reinstated only after a conflict-free assessor repeats the assessment from the original evidence package.

9.5 Certification Revocation Procedure

A certification issued under this Framework may be revoked under the following conditions: evidence of a boundary-breaking event (as defined in §2.3) that has not triggered re-certification; discovery that the evidence package contained fabricated, altered, or retroactively constructed artifacts; discovery of a conflict of interest that was not disclosed; a Class 0 integrity breach in the certified deployment that has not been resolved and re-assessed; and failure to renew within the required renewal interval.

Revocation is an administrative action, not a punishment. When a certification is revoked, the deploying organization must update all public conformance statements to reflect the revocation within 48 hours. The revocation must be recorded in the public registry alongside the original certification entry. A revoked certification may be reinstated through a complete re-assessment under the applicable certification level requirements.

SECTION 10

Renewal and Continuous Compliance

Certification is a time-bounded claim. A certification issued today does not attest to the conformance of the deployment in twelve months. The enforcement infrastructure will evolve; policy versions will change; components will be updated;

infrastructure will be modified. Renewal procedures ensure that the certification claim remains an accurate representation of the deployment's current conformance state.

10.1 Certification Validity Period

TABLE 10.1: CERTIFICATION VALIDITY PERIODS BY LEVEL

LEVEL	VALIDITY PERIOD	RENEWAL TYPE REQUIRED	EVIDENCE REFRESH REQUIREMENT
CL-01 Declared	24 months	Self-attestation re-assessment	Full evidence package; all tests re-executed
CL-01 Verified	24 months	Third-party re-review	Full evidence package; assessor may rely on delta evidence for unchanged controls
CL-02 Verified	18 months	Third-party re-assessment including replay tests	Full evidence; determinism corpus re-run required
CL-03 Certified	12 months	Certified re-assessment; all SLA evidence refreshed	90-day metric data must cover period ending within 30 days of renewal date
CL-04 Certified	12 months	Independent third-party re-audit; accredited body required	All CL-04 evidence; independent determinism audit required at each renewal

10.2 Trigger Events for Re-Certification

The following events require re-certification before the next scheduled renewal, regardless of how recently the previous certification was issued. A boundary-breaking event overrides the validity period.

- Change in deployment topology (including addition, removal, or relocation of any canonical component)
- Major version upgrade of any enforcement-path component (EG, PE, CR, AL, AE)
- Change in the declared revocation TTL, in any direction

- Change in the audit ledger infrastructure, including storage system replacement, partition restructuring, or export mechanism change
- Change in the key hierarchy (including root key re-establishment, intermediate key replacement, or AE signing key algorithm change)
- A Class 0 integrity breach event that required forensic review and remediation
- Policy version change that introduces new rule categories not present in the previous assessed version
- Discovery of a conformance failure in production that was not detected by the prior assessment

10.3 Version Migration Requirements

When the QODIQA corpus is revised and new normative versions of the Core Standard, the 68-Point Framework, or the Implementation Playbook are published, a version migration window is declared. During the migration window, certifications issued against the prior version remain valid. After the migration window closes, certifications must be renewed against the new version requirements. The migration window duration will be defined in the corpus revision notice; it will not be less than six months.

A deployment that is certified against an earlier version of the QODIQA corpus and that wishes to claim certification against a newer version must complete a full re-assessment against the new version requirements. Delta assessments are not sufficient for version migration; the new version may introduce requirements that interact with previously assessed controls in ways that require fresh evaluation.

10.4 Revocation of Certification Conditions

Certification revocation is distinct from certification expiry. Expiry is a scheduled event at the end of the validity period. Revocation is an unscheduled event triggered by evidence that the certified deployment is no longer conformant,

regardless of position within the validity period. The following conditions mandate revocation initiation within 72 hours of discovery:

- A Class 0 integrity breach (confirmed replay, ledger chain break with external modification evidence, unauthorized CR write, or policy mutation post-freeze) that has not been remediated and re-assessed
- Evidence that the evidence package submitted for certification contained fabricated test results, retroactively constructed artifacts, or timestamps inconsistent with the claimed test execution window
- Discovery that the assessor or ACB had a disqualifying conflict of interest not disclosed at the time of assessment
- A boundary-breaking event (§2.3) that has not triggered re-certification within 30 days of the event
- Deployment of a component version that the deploying organization has determined introduces a regression in conformance but that has not yet been assessed
- A public security disclosure affecting a cryptographic primitive or key management component in the certified deployment, where the deploying organization has not demonstrated that the affected component has been replaced or mitigated
- Failure to renew within the maximum renewal interval plus a 30-day grace period

Revocation does not delete the certification history. All revoked certifications remain in the public registry with their revocation reason code and the date of revocation. A deploying organization that disagrees with a revocation determination by an ACB may request a review by a second ACB; the original certification remains SUSPENDED (not REVOKED) during the review period. If the review ACB concurs with the original revocation determination, the status changes to REVOKED and the review outcome is archived alongside the original certification record.

SECTION 11

Public Registry Model

The public certification registry is the authoritative record of all QODIQA conformance claims that have been submitted for public listing under the Verified or Certified claim types. It is a transparency instrument, not a marketing channel. Its purpose is to allow relying parties to independently verify conformance claims made by deploying organizations without depending on the deploying organization's own representations.

11.1 Registry Structure

Each registry entry must contain the following mandatory fields. No entry may be published without all mandatory fields. Optional fields may be included where applicable.

TABLE 11.1: REGISTRY ENTRY SCHEMA

FIELD	TYPE	DESCRIPTION	MANDATORY
`registry_id`	String	Unique opaque identifier assigned at registry submission	Yes
`deployment_id`	String	Deploying organization's identifier for the assessed deployment; must be consistent across renewals	Yes
`certification_level`	Enum	CL-01, CL-02, CL-03, or CL-04	Yes
`claim_type`	Enum	VERIFIED or CERTIFIED	Yes
`assessment_date`	ISO date	Date of final evidence review and conformance determination	Yes
`valid_until`	ISO date	Expiry date per §10.1 validity periods	Yes
`corpus_version`	String	QODIQA corpus version assessed against (e.g., CORE-1.0, FW-3.0)	Yes

FIELD	TYPE	DESCRIPTION	MANDATORY
`policy_versions`	Array	List of policy version identifiers in scope at assessment date	Yes
`assessor_id`	String	Identifier for assessor or ACB; not disclosed for self-attestation	Verified/Certified
`evidence_package_hash`	SHA-256 hex	Hash of the evidence package manifest submitted with the certification	Yes
`status`	Enum	ACTIVE, EXPIRED, REVOKED, SUSPENDED	Yes
`revocation_reason`	String	If status = REVOKED: structured reason code per §9.5	If revoked

11.2 Transparency Requirements

Registry entries must be publicly queryable by `registry_id` and by `deployment_id`. The current status of any entry must be returned within 24 hours of a status change. REVOKED entries must remain in the registry indefinitely; they are not removed on revocation. The registry must disclose the corpus version and claim type for every entry so that relying parties can assess the scope and independence of the conformance claim.

Registry entries do not disclose the content of the evidence package, the detailed test results, or any internal deployment information. The evidence package hash is published so that a relying party who has been provided the evidence package by the deploying organization can verify its authenticity. The registry is not a substitute for direct evidence review by relying parties with specific assurance needs.

11.3 Registry Integrity

The registry itself must maintain an append-only record of all status changes, with timestamps and reason codes for each change. A registry operator that cannot demonstrate the integrity of its status change history is not operating a

conformant public registry under this Framework. The registry integrity record must be independently verifiable by the deploying organization for entries pertaining to its deployments.

11.4 Version Disclosure

The public registry must disclose, for every entry, the exact QODIQA corpus version against which the deployment was assessed. Corpus version disclosure must be specific to the named document versions; it is not sufficient to state "QODIQA Core" without specifying the version number. Where the assessment covered multiple corpus documents (e.g., both the Core Standard and the 68-Point Framework), both versions must be disclosed separately in the registry entry.

Version disclosure enables relying parties to determine whether a certification was conducted against the current version of the corpus, an earlier version, or a version that has since been superseded by a normative revision. A relying party must be able to assess, from the registry entry alone and without additional information from the deploying organization, the following: the exact normative requirements the deployment was assessed against; whether those requirements have been superseded; and, if superseded, how long ago the superseding version was published. The registry must provide version lineage information sufficient to support this determination.

TABLE 11.2: VERSION DISCLOSURE REQUIREMENTS BY REGISTRY FIELD

DISCLOSURE ELEMENT	REQUIREMENT AND RELYING PARTY PURPOSE
Core Standard version	Must be expressed as a major.minor identifier with publication date (e.g., CORE-1.0 (2026-Q1)). Enables relying parties to determine the normative enforcement requirements in scope.
68-Point Framework version	Must be version identifier **1.0** corresponding to the QODIQA corpus version in scope.
Implementation Playbook version	Major.minor if assessed; NONE if not in scope (e.g., IMPL-1.1 or NONE). Enables determination of the infrastructure conformance reference used.
Certification Framework version	Major.minor identifier, always required (e.g., CERT-1.0). Enables determination of the certification procedure version and detection of procedural changes since assessment.

DISCLOSURE ELEMENT	REQUIREMENT AND RELYING PARTY PURPOSE
Corpus status at assessment	CURRENT or SUPERSEDED at time of assessment. Enables relying parties to determine whether the deployment was assessed against the then-current corpus.
Corpus status at query time	CURRENT or SUPERSEDED, computed by the registry at query time (e.g., SUPERSEDED by CERT-1.1). Enables relying parties to determine whether the certification basis is still current.

SECTION 12

Institutional Closing Statement

This Framework completes the QODIQA corpus by providing the assurance layer that makes the corpus's normative requirements verifiable. A standard without a certification framework is a set of claims that cannot be independently substantiated. A certification framework without a technically rigorous standard to certify against is a procedural shell with no normative content. The QODIQA corpus requires both, and this Framework is the second half of that pair.

> Certification under this Framework is not a declaration of intent. It is a technically substantiated claim, bound to a specific deployment, a specific time window, and a specific assurance level. It means that someone who was not responsible for building the deployment has examined the evidence and concluded that the determinism invariant holds, the revocation guarantee is measured and within bounds, the ledger is tamper-evident and complete, and the fail-closed behavior is demonstrated under injection, not merely configured.

The certification anchor, the deterministic certification invariant `evaluate(C, P) = replay(C, P)`, is not a performance metric and not a statistical property. It is a structural requirement: the enforcement function must be pure. If it is not pure, the entire accountability model built on top of it collapses. Certified conformance under this Framework is

evidence that it is pure, for the assessed deployment, at the assessed time, against the assessed policy version. Nothing more and nothing less.

> Fail-closed enforcement is the non-negotiable posture of this Framework. A deployment that routes requests through an open circuit breaker, serves stale revocation state beyond TTL, or executes inference without a committed audit record cannot be certified at any level. **The absence of a certification claim is not a failure of process; it is an accurate representation of the deployment's conformance state. Certification under this Framework is earned through demonstrated behavior, not through documentation of intent.**

12.1 Corpus Alignment Statement

This Certification Framework (QODIQA-CERT-2026-001 v1.0) is normatively subordinate to and must be read in conjunction with: QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0; QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0; QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0; QODIQA: Implementation Playbook for Deterministic Runtime Consent Enforcement, Version 1.0. In the event of conflict between this Framework's procedural specifications and the enforcement semantics of the foregoing documents, the foregoing documents prevail. This Framework is authoritative for all matters of certification procedure, evidence specification, and attestation pathway.

SECTION 13

Certification Lifecycle

Certification under this Framework is not a one-time event. It is a continuing compliance state that must be actively maintained through a structured lifecycle of assessment, surveillance, and renewal. A certification claim is valid only for

the specific deployment configuration, policy version, and assurance level against which it was issued, and only for the duration of its declared validity period. Any departure from the assessed configuration that is not captured and re-evaluated in a subsequent assessment constitutes an unmanaged deviation from the certified state.

> The certification lifecycle is the operational mechanism by which the deterministic certification invariant is kept current. A certificate that has expired, lapsed under surveillance failure, or been issued against a configuration that has since materially changed does not constitute valid evidence of conformance. Relying parties must verify not only that a certificate exists but that it remains active and reflects the current deployment state.

13.1 Certification Lifecycle Phases

LC-01	**Application** The deploying organization submits a formal certification application identifying the deployment scope, target certification level, applicable corpus version, and proposed assessment pathway. The application must include a declaration of the conformance boundary and an identification of all system components within scope. Applications that do not clearly define the conformance boundary are returned for clarification before assessment scheduling.	Entry gate to lifecycle
LC-02	**Pre-Assessment** The assessor conducts an initial readiness review against the minimum evidence set for the target certification level. The pre-assessment is not a certification determination; it is a structured triage that identifies evidence gaps, architectural ambiguities, and test precondition failures before formal evaluation begins. A pre-assessment finding of material readiness gaps suspends progress until the deploying organization has resolved the identified deficiencies.	Readiness gate

LC-03	**Conformance Evaluation** The assessor executes the conformance test suite defined in Section 5, the determinism verification protocol defined in Section 6, and the audit ledger integrity validation defined in Section 7. Each test produces a structured result record that is archived as a component of the evidence package. Conformance evaluation may not be conducted by the deploying organization at CL-02 and above; independence requirements defined in Section 9 apply in full.	Core technical gate
LC-04	**Evidence Review** The complete evidence package (comprising test results, audit ledger exports, deployment topology documentation, policy artifact records, and key management records) is reviewed against the artifact requirements defined in Section 4. Evidence review verifies that the package is internally consistent, that test results are traceable to specific deployment configurations, and that no required artifact is absent or unverifiable. Evidence packages that fail integrity checks are rejected and must be resubmitted following remediation.	Evidence integrity gate
LC-05	**Certification Decision** On the basis of the conformance evaluation results and the evidence review outcome, the certification authority issues a formal certification decision: Certified, Conditionally Certified with specified remediation conditions, or Not Certified with a structured finding report. The decision record must reference the specific test results, evidence artifacts, and assessor identity that support it. Certification decisions may not be issued without a complete and verified evidence package.	Authority determination
LC-06	**Certificate Issuance** Upon a positive certification decision, the certification authority issues a structured certificate identifying the certified deployment, the assurance level, the validity period, the policy version assessed, the assessor identity, and the evidence package identifier. The certificate is registered in the public registry defined in Section 11. Certificates are issued with explicit expiry dates and are not automatically renewable; renewal requires a separate assessment under the procedures defined in Section 10.	Formal record creation
LC-07	**Surveillance Monitoring** During the validity period, the certified deployment is subject to surveillance monitoring at intervals defined by the applicable certification level. Surveillance monitoring verifies that the deployment continues to operate within the assessed configuration, that no trigger events defined in Section 10.2 have occurred without notification, and that audit ledger integrity has been maintained. Surveillance findings that indicate deviation from the certified state may result in suspension or revocation as defined in Section 14.4.	Ongoing compliance gate

LC-08	**Renewal and Re-Certification** Prior to certificate expiry, the deploying organization must initiate renewal assessment. Renewal follows the same procedural pathway as initial certification but may be scoped to the changes that have occurred since the prior assessment. If no material changes have occurred and surveillance monitoring has been continuous and uninterrupted, a scoped renewal assessment covering only the changed components and updated evidence is permissible at CL-01 and CL-02. At CL-03 and CL-04, full reassessment is required regardless of change scope.	Continuity pathway
LC-09	**Revocation Conditions** Certification may be suspended or revoked at any point during the validity period if conditions defined in Section 14.4 are met. Revocation terminates the certification status immediately and requires removal of the certificate from the public registry. A revoked certification cannot be reinstated; a new certification assessment must be completed. Revocation findings are recorded in the public registry and are not expunged upon subsequent recertification.	Integrity enforcement

13.2 Certification as a Continuing Compliance State

The lifecycle model above reflects a fundamental principle of this Framework: certification status is not a fixed property of a deployment; it is a time-bounded, configuration-specific, evidence-backed assertion that is continuously subject to invalidation. A deployment that was conformant at the time of certification may become non-conformant through any of the following: changes to the enforcement architecture without reassessment; policy version updates that alter enforcement semantics without re-verification; key rotation events that are not captured in the audit ledger; infrastructure migrations that alter component isolation properties; or surveillance findings that reveal unresolved deviations.

Relying parties must treat a certification claim as a living assertion, not a historical record. The public registry defined in Section 11 is the authoritative source of current certification status. A certificate that appears in the registry as active and unexpired, against a deployment configuration that matches the current operational state, is a valid certification claim. A certificate that is expired, suspended, or revoked, or that was issued against a configuration that has materially changed, is not valid evidence of current conformance regardless of when it was originally issued.

LIFECYCLE INTEGRITY PRINCIPLE

The certification lifecycle is the operational expression of the deterministic certification invariant. The invariant asserts that enforcement behavior is reproducible. The lifecycle asserts that reproducibility has been verified, that the verification is current, and that any change to the deployment that could affect reproducibility has been captured and re-evaluated. A certification program that issues certificates but does not maintain lifecycle integrity is not implementing this Framework; it is issuing assertions without the accountability structure that makes those assertions verifiable.

SECTION 14

Certification Assessment Procedure

Certification under this Framework requires verifiable runtime enforcement, not policy declarations, architectural diagrams, or vendor attestations. The assessment procedure is structured to produce evidence that is independently reproducible: an assessor who was not present at the original assessment must be able to re-derive the certification determination from the archived evidence package alone. Any assessment that does not produce a self-contained, independently verifiable evidence package does not satisfy the procedural requirements of this Framework.

14.1 Structured Evaluation Process

The assessment procedure is organized into six sequential evaluation stages. Each stage must be completed before the subsequent stage begins. A finding of non-conformance at any stage may be grounds for suspension of the assessment, depending on the severity classification of the finding and the certification level being sought.

AE-01	**Technical Documentation Review** The assessor reviews the deployment's technical documentation package against the architecture and component requirements of the QODIQA Core Standard. Documentation review covers: enforcement gateway topology and component isolation model; consent registry persistence and signing configuration; policy artifact version control and frozen-snapshot mechanism; audit ledger write path and signing chain; key management architecture and rotation schedule; and revocation propagation configuration and measured TTL. Documentation review does not constitute evidence of runtime conformance; it establishes the architectural context within which runtime testing is interpreted.	AE-01: Documentation
AE-02	**Architecture Inspection** The assessor conducts a structured inspection of the deployed architecture to verify that the component topology described in the documentation matches the operational deployment. Architecture inspection must include: verification that the enforcement gateway is the sole path through which inference requests transit; verification that the consent registry is not co-located with the enforcement gateway on shared mutable infrastructure; verification that the audit ledger write path cannot be bypassed by the enforcement gateway; and verification that model endpoint access is not reachable from outside the enforcement boundary. Architecture inspection findings that reveal discrepancies between documentation and deployment are classified as documentation integrity failures and must be resolved before runtime testing proceeds.	AE-02: Architecture
AE-03	**Runtime Enforcement Verification** The assessor executes the full conformance test suite defined in Section 5 against the live deployment. Runtime enforcement verification is the core of the assessment procedure and cannot be replaced by simulation, staging-environment results, or vendor-supplied test records. The assessor must have direct access to the enforcement gateway's test interface and must be able to independently submit test vectors and retrieve audit ledger records for the submitted vectors. Any mediation of the assessor's access to the enforcement gateway by the deploying organization is a conflict-of-interest finding that disqualifies the assessment at CL-02 and above.	AE-03: Runtime testing
AE-04	**Conformance Test Suite Execution** The assessor executes each test defined in Sections 5, 6, 7, and 8, recording structured result records for each test vector submitted. Test suite execution must cover: the complete positive test set for ACTIVE and REVOKED token states; the complete negative test set for boundary and injection conditions; the fail-closed injection procedure for each enforcement component; the determinism verification protocol against archived test vectors; the audit ledger hash-chain validation; and the revocation SLA verification against production metric exports. Test suite execution results are archived as structured evidence artifacts and must be traceable to specific test identifiers and timestamps.	AE-04: Test execution

AE-05	**Evidence Validation** Following test suite execution, the assessor assembles and validates the complete evidence package. Evidence validation verifies: that all required artifact types for the target certification level are present; that each artifact carries a valid integrity signature or hash; that test result records are internally consistent and cross-reference correctly to the deployment configuration; that audit ledger exports are contiguous and chain-verified; and that no required artifact has been produced by the deploying organization in circumstances that compromise its independence. Evidence packages that fail validation are returned for remediation and may not be used to support a certification determination until revalidated.	AE-05: Evidence package
AE-06	**Independent Assessment Requirements** At CL-02 and above, all assessment stages from AE-02 through AE-05 must be conducted by an assessor who satisfies the independence criteria defined in Section 9. Independence is not merely organizational separation; it requires that the assessor has no financial, contractual, or advisory relationship with the deploying organization that could influence the assessment outcome; that the assessor has direct and unmediated access to the deployment under assessment; and that the assessor is technically qualified to evaluate the specific enforcement architecture under assessment without relying on the deploying organization's interpretation of its own architecture.	AE-06: Independence

14.2 Required Certification Evidence

The following table defines the standard artifact categories required during certification. Each artifact category must be represented in the evidence package by at least one artifact that satisfies the integrity and provenance requirements for the target certification level. Artifact categories marked as mandatory at a given level must be present; artifact categories marked as conditional must be present if the relevant system component or configuration is within the declared conformance boundary.

TABLE 14.1: REQUIRED CERTIFICATION EVIDENCE ARTIFACT CATEGORIES

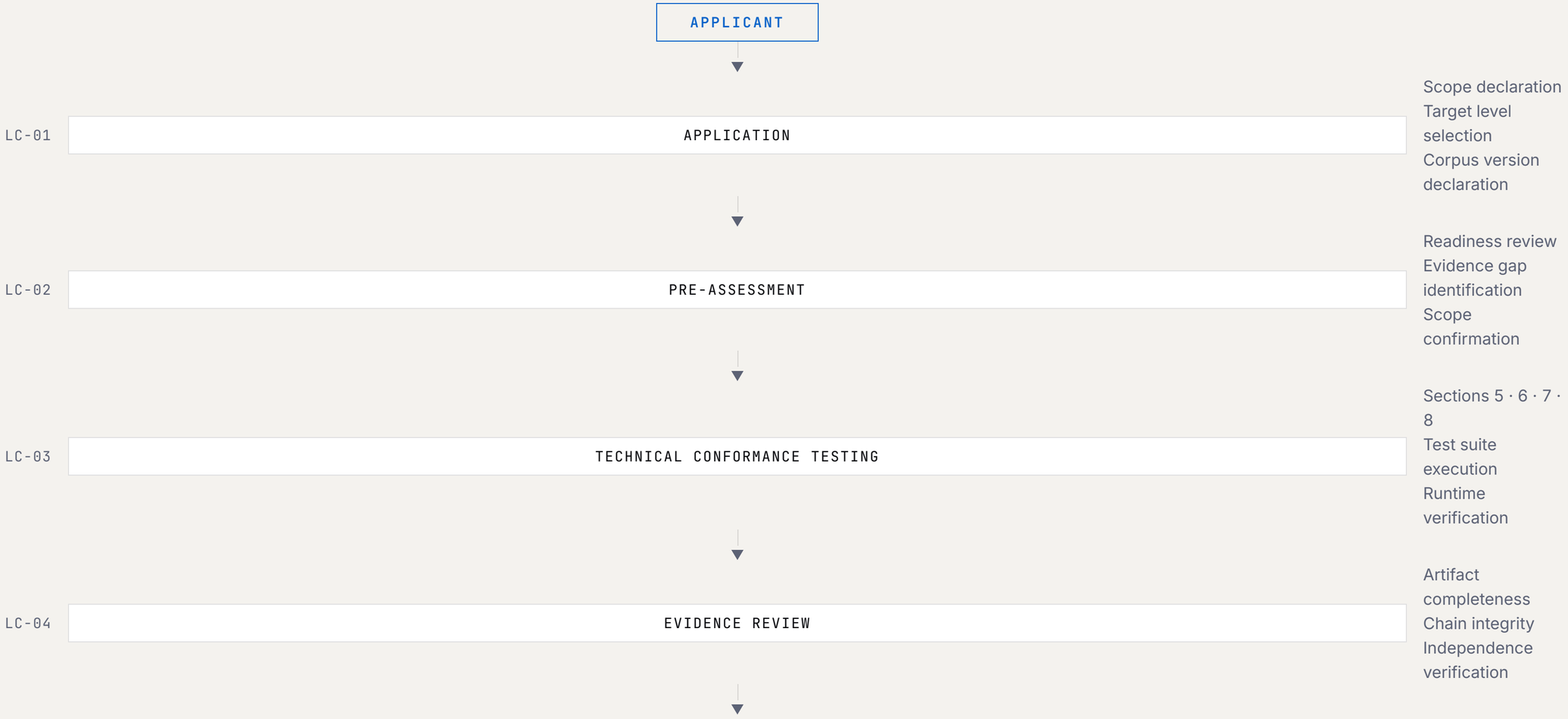
ARTIFACT CATEGORY	DESCRIPTION AND REQUIRED CONTENT	MINIMUM LEVEL	INTEGRITY REQUIREMENT
Enforcement Gateway Deployment Topology	Architectural diagram and configuration record of the enforcement gateway deployment, including component isolation model, network boundary configuration, and inference endpoint access controls. Must reflect the operational deployment at the time of assessment.	CL-01	Assessor-signed at time of inspection

ARTIFACT CATEGORY	DESCRIPTION AND REQUIRED CONTENT	MINIMUM LEVEL	INTEGRITY REQUIREMENT
Runtime Enforcement Logs	Exported audit ledger records covering the assessment period, including PERMIT, DENY, and circuit-breaker event records. Must be exported via the signed export mechanism and carry a valid chain-integrity signature. Log exports must be contiguous and cover the full test window.	CL-01	AL-signed export with chain verification
Consent Registry Integrity Proof	Evidence that the consent registry is operating with write-ahead logging, signing, and tamper-detection active. At CL-02 and above, includes a signed export of the registry state at the time of assessment with a corresponding hash record in the audit ledger.	CL-01	CR-signed export
Policy Artifact Validation Record	Record confirming that the policy engine is operating against a frozen, immutable policy snapshot. Must include the snapshot hash, the POLICY_FROZEN audit ledger record, and evidence that no policy modification occurred during the assessment period. At CL-03 and above, includes multi-instance snapshot hash consistency verification.	CL-01	PE-reported hash with AL cross-reference
Model Endpoint Isolation Evidence	Network topology records or inspection evidence confirming that the model endpoint is not directly reachable from outside the enforcement boundary. At CL-02 and above, includes active verification that requests submitted directly to the model endpoint without a valid enforcement artifact are rejected.	CL-02	Assessor-verified at inspection
System Integration Documentation	Complete integration map of all systems that interact with the enforcement boundary, including upstream request originators, downstream inference consumers, and any external dependencies of the consent registry or audit ledger. Integration documentation must identify all data paths that carry consent signals or enforcement decisions.	CL-01	Deploying organization declaration with assessor verification
Audit Log Integrity Proof	Independent verification that the audit ledger chain is unbroken for the assessment period. Must include the hash-chain verification result record from procedure HCV-01 through HCV-05. At CL-03 and above, must also include the per-write integrity verification configuration record and a sample of real-time tamper-detection alerts (or a record that no alerts were generated).	CL-01	Assessor-executed chain verification

ARTIFACT CATEGORY	DESCRIPTION AND REQUIRED CONTENT	MINIMUM LEVEL	INTEGRITY REQUIREMENT
Conformance Test Result Archive	Structured archive of all test result records produced during conformance test suite execution, including input vectors, output decisions, audit ledger records for each vector, and pass/fail determinations. Test result archives must be tamper-evident and must carry an assessor integrity signature applied at the time of test execution.	CL-01	Assessor-signed at execution time
Determinism Verification Record	Result record from the VERIFY_DETERMINISM protocol defined in Section 6, including input tuple hashes, output decision records, replay comparison results, and the pass/fail determination. At CL-02 and above, includes the PE restart record and the post-restart replay comparison results confirming bit-identical output for all sampled vectors.	CL-02	Assessor-signed with tuple hash archive
Revocation SLA Metric Export	Signed export of the revocation propagation latency histogram covering the required measurement period for the target certification level, as defined in Section 8.1. Export must include per-EG-instance P99 values and the worst-case aggregate P99, both verified against the declared TTL.	CL-01	ML-signed metric export

14.3 Certification Process Overview

The following diagram illustrates the sequential flow of the certification process from initial application through ongoing surveillance and renewal. Each stage corresponds to a lifecycle phase defined in Section 13.1. The diagram is intended to provide a navigational overview of the process for applicants and relying parties; the normative procedural requirements are defined in Sections 13, 14.1, and 14.4.


APPLICANT
LC-01
APPLICATION
Scope declaration
Target level selection
Corpus version declaration
LC-02
PRE-ASSESSMENT
Readiness review
Evidence gap identification
Scope confirmation
LC-03
TECHNICAL CONFORMANCE TESTING
Sections 5 · 6 · 7 · 8
Test suite execution
Runtime verification
LC-04
EVIDENCE REVIEW
Artifact completeness
Chain integrity
Independence verification

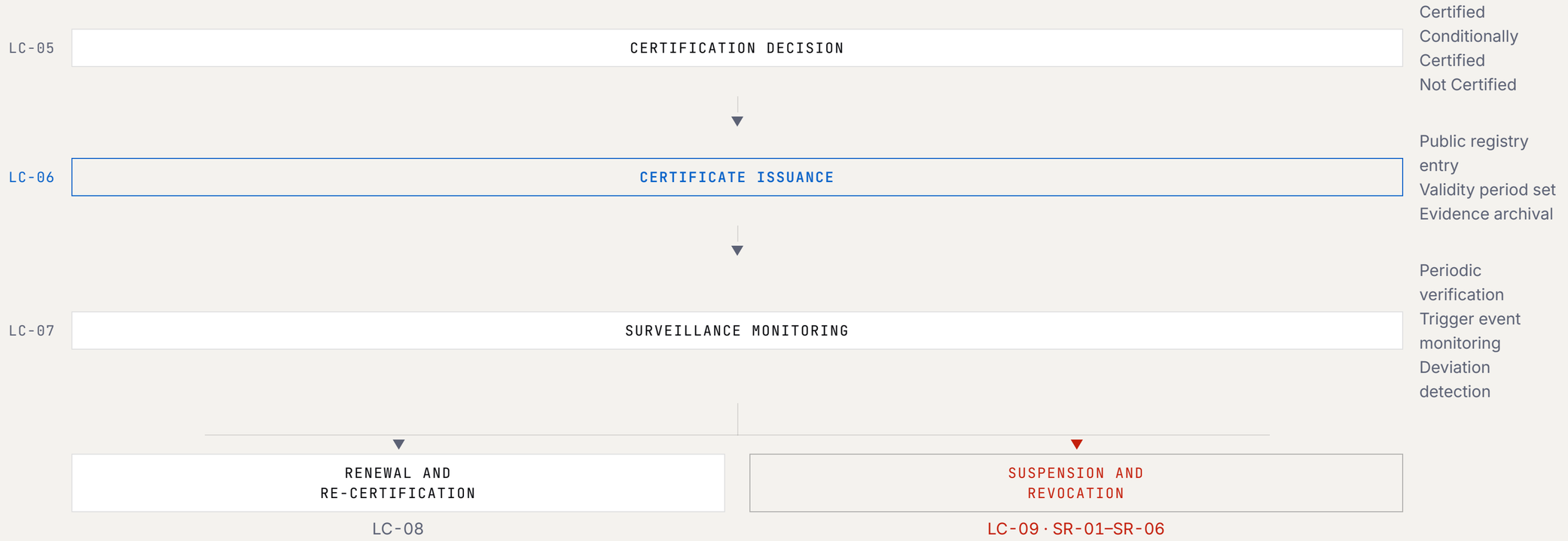


FIGURE 14.1: CERTIFICATION PROCESS FLOW. EACH STAGE IS A GATE; NON-CONFORMANCE AT ANY STAGE SUSPENDS FORWARD PROGRESS PENDING REMEDIATION.

14.4 Certification Suspension and Revocation

Certification status must be withdrawn if deterministic enforcement can no longer be verified for the certified deployment. Suspension and revocation are not disciplinary actions; they are accurate representations of the deployment's current conformance state. A certification authority that maintains active certification status for a

deployment that has materially departed from its certified configuration is not administering this Framework; it is issuing misleading assurance claims.

The following conditions constitute grounds for immediate certification suspension, pending investigation and remediation assessment:

SR-01	**Removal or Bypass of Runtime Enforcement Controls** Any configuration change that allows inference requests to reach model endpoints without transiting the enforcement gateway, or that allows the enforcement gateway to issue permits without a committed audit ledger record, constitutes an immediate suspension trigger.	Immediate suspension
SR-02	**Failure of Scheduled Surveillance Audit** Failure to complete a required surveillance audit within the scheduled window, or failure to produce the required surveillance evidence artifacts upon request, constitutes a suspension trigger. Surveillance failure is not excused by operational constraints; if the deployment cannot be audited, its certification status cannot be maintained.	Suspension trigger
SR-03	**Material Architecture Changes Without Reassessment** Any change to the enforcement boundary, the component isolation model, the audit ledger write path, the policy engine configuration, or the consent registry persistence mechanism that has not been captured in a subsequent assessment constitutes a suspension trigger. Minor configuration changes that do not affect enforcement semantics may be managed through a scoped change notification process, subject to assessor review.	Suspension trigger
SR-04	**Inability to Produce Audit Evidence** If the deploying organization is unable to produce a complete, chain-verified audit ledger export for any period within the certificate's validity window upon assessor request, this constitutes a suspension trigger. The inability to produce audit evidence is itself an accountability failure, independent of whether the underlying enforcement behavior was conformant.	Suspension trigger

SR-05	**Confirmed Determinism Failure** If post-issuance evidence demonstrates that the enforcement function produced different outcomes for identical inputs during the certificate's validity period (outside of documented token state or policy version changes), this constitutes a revocation trigger. Determinism failures discovered post-issuance may also require retroactive notification to relying parties who relied on the certificate during the affected period.	Revocation trigger
SR-06	**Revocation Propagation SLA Breach** If measured revocation propagation latency is found to have exceeded the declared TTL at the P99 percentile during the certificate's validity period, and this was not disclosed at the time of certification as a known conformance boundary condition, this constitutes a revocation trigger.	Revocation trigger

Following suspension, the deploying organization has a defined remediation window to resolve the suspension trigger and submit evidence of remediation for assessor review. If remediation is confirmed, the suspension is lifted and the certificate is reinstated with an updated surveillance schedule. If remediation is not completed within the remediation window, or if the investigation reveals that the trigger condition reflects a deeper conformance failure, the certificate is revoked.

Revocation is permanent. A revoked certificate cannot be reinstated. A new certification assessment is required to establish a new conformance claim. Revocation records are retained in the public registry and are not removed upon subsequent recertification. Relying parties must be notified of revocation through the registry notification mechanism defined in Section 11.

The suspension and revocation framework is not a punitive mechanism. It is the accountability layer that gives certification status its meaning. A certification program that does not revoke certificates when revocation conditions are met is not providing assurance; it is providing the appearance of assurance. The integrity of every active certificate in the registry depends on the willingness of the certification authority to revoke certificates that no longer reflect the conformance state they were issued to attest.

DOCUMENT STATUS

Document Status and Classification

This document is the Certification Framework of the QODIQA specification corpus. It defines the conformance assessment procedures, evidence requirements, test methodology, and attestation authority model for implementations of the QODIQA Core Standard and the QODIQA 68-Point Enforcement Framework. It is issued as a Technical Certification Specification and is not a product specification, a commercial certification program, or a legal instrument.

The framework establishes four certification levels, defines the evidentiary requirements and test procedures for each level, and specifies the authority pathways through which conformance claims may be affirmed, maintained, and revoked. It is the authoritative instrument for all QODIQA conformance assessment activity.

This document should be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Implementation Playbook for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0

- QODIQA: Threat Model and Abuse Case Specification, Version 1.0
- QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dușescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-CERT-2026-001
TITLE	Certification Framework for Deterministic Runtime Consent Enforcement
SUBTITLE	Conformance, Assurance, and Attestation Model for QODIQA Enforcement Infrastructure
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Technical Certification Specification
DOCUMENT STATUS	Normative: Technical Certification Specification
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 24 PART IV · GOVERNANCE AND LEGAL INSTRUMENTS

Public Specification License

Governing the conditions under which the specification corpus may be accessed, referenced, cited, implemented, and extended.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-PSLI-2026-001	1.0	2026	Normative License	Institutional	NORMATIVE

Licensing notice (supersedes the terms below). The text of the QODIQA specification corpus is licensed under the Creative Commons Attribution 4.0 International license (CC BY 4.0); see the LICENSE file of this repository. Where any provision of this document restricts copying, modification, redistribution, forking or derivative works of the specification text, that provision is withdrawn and CC BY 4.0 applies instead. Nothing in this document adds conditions to CC BY 4.0.
Unchanged by this notice: CC BY 4.0 grants no trademark rights, so the naming and certification-boundary statements (Sections 7 and 8) remain as statements of policy about the QODIQA name and marks. No certification body or program exists yet. The sections on warranty and liability are superseded by the corresponding terms of CC BY 4.0.
Code is licensed separately: the reference code in qodiqa-gateway, qodiqa-verify and qodiqa-anchors is released under the Apache License 2.0. The text below is kept for the historical record of version 1.0.

PREAMBLE

The QODIQA specification corpus constitutes a body of technical standards governing deterministic runtime consent enforcement for AI systems. The publication of this corpus serves the public interest in establishing a coherent, interoperable,

and auditable framework for AI consent infrastructure. Publication is a deliberate act of standardization, not a grant of unrestricted use.

Public availability does not imply unrestricted modification rights. The integrity of a deterministic enforcement standard depends upon the fidelity of its normative content. A standard whose normative requirements may be silently altered, misattributed, or selectively presented cannot function as an infrastructure reference. The present License exists precisely to preserve that integrity.

This document, the QODIQA Public Specification License Version 1.0 (hereinafter "this License" or "QPSL v1.0"), governs the conditions under which the QODIQA specification corpus may be accessed, referenced, cited, implemented, and extended. **This License governs the specification corpus only.** Reference implementations, certification toolchains, and conformance testing infrastructure are governed by separate instruments unless expressly incorporated herein. Publication of the Specification Corpus does not constitute a patent licensing framework; no patent rights are granted by or inferred from this License.

The Governance Authority actively monitors compliance with this License and will pursue enforcement through all available legal and institutional mechanisms where violations are identified.

SECTION 1

Definitions

The following terms, when used in this License, carry the meanings assigned below. All definitions are deterministic and shall not be construed expansively or by analogy beyond their stated scope.

Specification Corpus

The complete set of documents, schemas, normative text, and associated non-normative materials published by the QODIQA Governance Authority under the QODIQA designation, as identified by official document identifiers of the form `QODIQA-*`.

Normative Content

Any provision, requirement, constraint, definition, invariant, or protocol specification within the Specification Corpus that establishes mandatory behavior. Normative Content is identified by the terms "shall," "shall not," "must," "must not," "required," "prohibited," and equivalent unambiguous imperative language.

Non-Normative Content

Explanatory text, rationale, examples, notes, appendices, and guidance materials that accompany Normative Content but do not independently establish mandatory behavior. Non-Normative Content is identified by designations such as "Note," "Example," "Informative Annex," or equivalent markers.

Derivative Specification

Any document, standard, schema, or protocol text that (a) reproduces a substantial portion of Normative Content from the Specification Corpus, (b) modifies, extends, restricts, or reinterprets Normative Content, or (c) presents itself as a profile, subset, superset, or adaptation of the Specification Corpus, whether or not it uses the QODIQA designation.

Implementation

Any software, hardware, system, service, or component that is designed, constructed, or represented as operating in conformance with any portion of the Specification Corpus, regardless of whether a Conformance Claim is made.

Conformance Claim

Any assertion, statement, label, badge, mark, or representation, whether explicit or implied, that an Implementation satisfies the requirements of any portion of the Specification Corpus.

Certification

A formal determination, issued by the Governance Authority or an Authorized Certification Body, that an Implementation has been evaluated against the Conformance Test Suite and found to satisfy applicable normative requirements. Certification is distinct from self-attestation and from implementation claims made without third-party evaluation.

Trademark

The designation "QODIQA," associated logos, certification marks, conformance badges, and any other distinctive identifiers controlled by the Governance Authority, whether or not formally registered in any jurisdiction.

Governance Authority

The body responsible for the QODIQA Specification Corpus, as constituted under the QODIQA Governance Charter. The Governance Authority holds exclusive authority to amend the Specification Corpus, issue Certifications, authorize Certification Bodies, and interpret this License in cases of dispute.

Authorized Certification Body

An entity that has been formally authorized by the Governance Authority, under procedures defined in the QODIQA Governance Charter, to conduct Conformance evaluations and issue Certification determinations on behalf of or in coordination with the Governance Authority.

SECTION 2

Grant of Public Access

Subject to the terms and conditions of this License, the Governance Authority grants to any person or entity ("Recipient") a worldwide, royalty-free, non-exclusive, non-transferable right to access the Specification Corpus for the purposes enumerated in Section 3 (Permitted Uses).

> **SCOPE OF GRANT**
>
> The rights granted in this section extend to reading, referencing, citing, and privately implementing the Specification Corpus. They do not extend to modification of Normative Content, publication of Derivative Specifications under the QODIQA designation, issuance of Conformance Claims beyond what is permitted under Section 6, or any use of the Trademark beyond what is permitted under Section 8.

Publication of the Specification Corpus in accessible form is a deliberate act of the Governance Authority to facilitate adoption, evaluation, and independent implementation. Such publication does not constitute abandonment of governance authority, relaxation of integrity requirements, or waiver of Trademark rights. The grant of public access is conditioned upon compliance with this License in its entirety.

Access obtained through any channel, including official publication repositories, third-party mirrors, academic archives, or derivative distributions, is governed by this License. Recipients who redistribute copies of the Specification Corpus must ensure that this License accompanies any such redistribution without modification.

SECTION 3

Permitted Uses

The following uses of the Specification Corpus are expressly permitted under this License without additional authorization from the Governance Authority, provided all other terms of this License are observed.

PERMITTED: UNRESTRICTED

Academic citation and scholarly analysis. Reproduction of limited excerpts of Normative or Non-Normative Content for purposes of academic citation, quotation, and scholarly commentary, in accordance with applicable fair use and fair dealing principles and subject to proper attribution.

Research and evaluation. Analysis, critique, comparison, and evaluation of the Specification Corpus for research purposes, including publication of findings in academic, technical, or regulatory contexts.

Internal evaluation. Reading and analysis of the Specification Corpus for the purpose of assessing its suitability for adoption, procurement, or regulatory compliance, including circulation within an organization for such purposes.

Private implementation. Development of Implementations for internal use, where no Conformance Claim is made to any external party.

Independent security review. Analysis of the Specification Corpus for purposes of identifying security vulnerabilities, adversarial risks, or implementation deficiencies, subject to responsible disclosure practices consistent with the QODIQA Governance Charter.

Regulatory mapping. Preparation of analyses mapping the Specification Corpus to applicable regulatory frameworks, standards, or legal requirements.

Commentary and educational use. Publication of explanatory commentary, tutorials, training materials, and educational content that describes or interprets the Specification Corpus, provided such materials clearly distinguish the author's views from Normative Content and do not misrepresent the content of the Specification Corpus.

3.1 Commercial Implementation

Commercial implementation of the Specification Corpus is permitted subject to the following conditions. An entity that develops, deploys, or distributes an Implementation for commercial purposes may do so without authorization from the Governance Authority, provided that: (a) no Conformance Claim beyond what is permitted under Section 6 is made; (b) no Trademark is used beyond what is permitted under Section 8; (c) the Implementation does not misrepresent the scope, status, or content of the Specification Corpus; and (d) the Implementation does not alter Normative Content and present the altered version as the official or canonical text of the Specification Corpus.

Commercial implementors who intend to make Certification-grade Conformance Claims must engage with the Governance Authority or an Authorized Certification Body as provided in Section 7.

SECTION 4

Restrictions

The following acts are prohibited under this License. Violation of any prohibition constitutes a material breach of this License and triggers the consequences set forth in Section 16 (Termination).

PROHIBITED: WITHOUT EXCEPTION

Misrepresentation as official release. Representing any modified, partial, or independently produced document as an official release of the QODIQA Specification Corpus, or presenting any such document as having been issued, approved, or endorsed by the Governance Authority when it has not.

Unauthorized certification claims. Claiming, representing, implying, or facilitating the belief that an Implementation has been Certified under the QODIQA Certification Framework when no Certification has been granted by the Governance Authority or an Authorized Certification Body.

Alteration and republication of Normative Content. Altering any Normative Content and publishing or distributing the altered version under the QODIQA designation, or in a manner that could reasonably be understood to represent the altered version as the canonical text.

Confusingly similar designation. Publishing any specification, standard, protocol, or document under a name, identifier, or designation that is confusingly similar to "QODIQA" or to any document identifier within the Specification Corpus, where such designation could lead a reasonable reader to conclude that the document is part of the official Specification Corpus.

Implied endorsement. Representing or implying that the Governance Authority has endorsed, validated, reviewed, or approved any product, service, Implementation, analysis, or commercial offering, unless such endorsement has been expressly granted in writing by the Governance Authority.

Confusingly similar certification marks. Creating, using, or distributing any mark, badge, label, or visual identifier that is confusingly similar to any official QODIQA certification mark, conformance badge, or Trademark, whether or not the QODIQA designation itself is used.

4.1 Anti-Confusion Clause

In all cases of ambiguity as to whether a particular act, designation, or representation falls within the prohibitions of this Section, the governing standard is whether a technically informed reader, exercising reasonable care, could be misled as to: (a) the official or authoritative status of a document or Implementation; (b) the certification status of an Implementation; or (c) the endorsement of any party by the Governance Authority. Where such misleading is reasonably possible, the act, designation, or representation is prohibited.

SECTION 5

Derivative Specifications

Any party that produces a Derivative Specification, as defined in Section 1, is subject to the requirements of this Section in addition to all other applicable provisions of this License.

5.1 Designation Restriction

A Derivative Specification shall not use the QODIQA designation, including the term "QODIQA," any document identifier of the form `QODIQA-*`, or any confusingly similar designation, unless the Governance Authority has granted express written authorization for such use. Authorization is granted on a case-by-case basis under procedures defined in the QODIQA Governance Charter and does not arise by implication from publication, citation, or implementation of the Specification Corpus.

5.2 Deviation Disclosure Obligation

Any Derivative Specification that modifies, restricts, or extends Normative Content must include a clearly prominent deviation disclosure statement that: (a) identifies the originating QODIQA specification document by its official identifier; (b) enumerates with specificity each normative requirement that has been added, removed, modified, or reinterpreted;

and (c) states that the Derivative Specification is not an official release of the QODIQA Specification Corpus and has not been approved by the Governance Authority.

5.3 Compatibility Claims

No Derivative Specification may represent itself as compatible with, conformant to, or interoperable with any portion of the Specification Corpus unless such compatibility has been validated through official conformance testing procedures administered by the Governance Authority or an Authorized Certification Body. Unvalidated compatibility claims constitute misrepresentation under Section 4.

5.4 Prohibition on Silent Semantic Modification

A Derivative Specification shall not alter the semantic content of any normative requirement, including through substitution of synonymous language, restructuring of logical dependencies, or modification of scope qualifications, without explicit disclosure pursuant to Section 5.2. Semantic modification without disclosure is prohibited regardless of whether the literal text of the Normative Content is reproduced verbatim.

SECTION 6

Conformance Claims

Conformance Claims are subject to a tiered regime that distinguishes between certified conformance, self-attested alignment, and informal description. The applicable tier governs both the language that may be used and the obligations that attach to the claimant.

TABLE 6.A: CONFORMANCE CLAIM CLASSIFICATION

TIER	PERMISSIBLE LANGUAGE	CONDITIONS
Certified	"Certified under QODIQA [document identifier]"	Requires Certification issued by Governance Authority or Authorized Certification Body. Subject to Certification mark usage rules in Section 8.
Self-Attested	"Built in alignment with the QODIQA specification" · "Independent implementation of QODIQA standard"	Permissible without Governance Authority authorization. Must be accompanied by prominent disclosure that the claim is self-attested and that no Certification has been granted.
Descriptive	"Designed to address requirements similar to those in QODIQA" · "Inspired by the QODIQA framework"	Permissible without restriction, provided no implication of formal conformance or Certification arises.

6.1 Self-Attestation Disclosure Requirement

Any Conformance Claim made at the Self-Attested tier must be accompanied, in the same document or communication, by a disclosure statement in substantially the following form: *"This conformance claim is a self-attestation. It has not been evaluated, verified, or certified by the QODIQA Governance Authority or any Authorized Certification Body. It does not constitute QODIQA Certification."* Omission of the required disclosure from a self-attested claim renders the claim a violation of Section 4.

6.2 False Claims

Any Conformance Claim that (a) uses Certified-tier language without a valid Certification, (b) omits the required Self-Attested tier disclosure, or (c) creates a false impression of conformance or Certification through the combination of language, imagery, or context, constitutes a material violation of this License and is subject to termination under Section 16.

NON-CONFORMANT REPRESENTATION

Any language that implies partial or selective compliance, or that exaggerates alignment with the Specification Corpus beyond the actual scope of a Recipient's implementation, is considered misleading and falls within the prohibited conduct defined in Section 4. Recipients shall not represent their implementations as "partially conformant," "substantially aligned," or any similar formulation that overstates verified compliance.

SECTION 7

Certification Boundary Protection

The QODIQA Certification Framework constitutes a protected institutional instrument. The integrity of Certification as a signal of conformance quality depends upon the exclusivity and control of Certification authority. Unauthorized replication or simulation of Certification processes undermines the informational value of Certification to the public, to regulators, and to the market.

Certification requires authorization under Governance Charter procedures. No act, agreement, or representation by any party other than the Governance Authority or its Authorized Certification Bodies creates, transfers, or implies Certification authority. The Governance Authority enforces this boundary as a continuous institutional obligation, not merely as a response to individual violations.

7.1 Protected Instruments

The following are protected instruments of the QODIQA Certification Framework. Their creation, issuance, use, or simulation outside of Governance Authority authorization is prohibited under this License.

- The QODIQA Certification Framework and all associated procedures and criteria documents
- Official conformance marks, certification badges, and associated visual identifiers
- Audit artifacts, evaluation reports, and attestation documents issued pursuant to official Certification procedures
- Certification determination letters and official conformance statements
- The QODIQA Conformance Test Suite and all associated test cases, scoring criteria, and evaluation instruments

7.2 Certification Engagement

Entities seeking Certification must engage with the Governance Authority or an Authorized Certification Body through procedures defined in the QODIQA Governance Charter. Certification engagements are subject to separate agreements that may impose additional obligations. This License does not constitute a Certification agreement and does not guarantee access to Certification processes.

SECTION 8

Trademark and Naming Policy

"QODIQA" is a protected designation of the Governance Authority. Its use is governed by this Section and, where applicable, by separate trademark licensing instruments issued by the Governance Authority. The provisions of this Section apply to all uses of the Trademark in connection with implementations, products, services, publications, and communications of any kind.

8.1 Permitted Descriptive Reference

The following phrases and usages are expressly permitted under this License without additional authorization, provided they are used accurately and not in a manner that implies Certification, official status, or endorsement.

Permitted

"Built in alignment with the QODIQA specification"

Permitted

"Independent implementation of the QODIQA standard"

Permitted

"Designed to address QODIQA-defined consent enforcement requirements"

Permitted

"Evaluated against QODIQA specification requirements [self-attested]"

Permitted

Accurate citation of QODIQA specification documents by their official identifiers in technical or regulatory documents

8.2 Prohibited Naming Patterns

The following usages are prohibited without express written authorization from the Governance Authority.

PROHIBITED NAMING PATTERNS

"Official." Use of the term "official" in combination with "QODIQA" in any context that implies the Governance Authority has authorized or endorsed a product, service, or implementation, unless such authorization has been expressly granted.

"Certified" (without Certification). Use of the term "certified" or "QODIQA-certified" in connection with any Implementation or service that has not received a Certification determination from the Governance Authority or an Authorized Certification Body.

Confusingly similar product names. Naming a product, service, tool, or framework with a designation that incorporates "QODIQA" or is confusingly similar thereto, unless authorized.

Domain names and organizational designations. Registering domain names, organizational names, or trade names that incorporate "QODIQA" in a manner that could imply official affiliation with the Governance Authority, unless authorized.

SECTION 9

Intellectual Property and Patent Boundary

This Section defines the intellectual property boundary applicable to the Specification Corpus. The provisions of this Section are intended to provide clarity to implementors and to the broader technical community without creating patent obligations of any kind.

9.1 No Implied Patent License

Publication of the Specification Corpus does not grant any patent license. No patent rights of any kind are conveyed by implication, estoppel, exhaustion, or otherwise by virtue of this License, the publication of the Specification Corpus, or any act or omission of the Governance Authority in connection with the Specification Corpus. Any party that implements

the Specification Corpus is solely responsible for identifying, evaluating, and securing any patent rights that may be necessary for such implementation.

9.2 Non-Infringement Disclaimer

The Governance Authority makes no representation, warranty, or assurance that implementation of the Specification Corpus will not infringe any patent, design right, or other intellectual property right held by any third party. The absence of a known patent claim against any portion of the Specification Corpus at the time of publication does not constitute a representation that no such claim exists or will arise. Responsibility for intellectual property clearance rests exclusively with each implementor.

9.3 Contributor Neutrality

Contribution of technical content to the Specification Corpus does not, by itself, constitute a grant of any patent license by the contributing party. Patent commitments by contributors, if any, must be made explicitly through a separate documented instrument that has been accepted by the Governance Authority and recorded in the QODIQA Governance Charter or an associated patent policy document. No patent commitment arises by inference from the act of contribution alone.

> **PATENT BOUNDARY CLARIFICATION**
>
> This Section does not create a patent pool, impose FRAND obligations, or require any party to grant patent rights as a condition of implementing or referencing the Specification Corpus. The provisions of this Section are intended solely to delineate the boundary of rights granted under this License and to provide predictability for implementors operating in diverse jurisdictions.

SECTION 10

No Warranty

The Specification Corpus is published on an "as is" basis. The Governance Authority makes no representations or warranties of any kind, express or implied, with respect to the Specification Corpus or any portion thereof.

- No warranty of fitness for a particular purpose. The Governance Authority does not warrant that the Specification Corpus is suitable for any specific application, deployment context, or regulatory purpose.
- No warranty of accuracy or completeness. The Governance Authority does not warrant that the Specification Corpus is free of errors, omissions, or ambiguities.
- No guarantee of regulatory compliance. Publication of the Specification Corpus does not constitute a representation that implementation of the Specification Corpus will satisfy any regulatory, legal, or contractual obligation in any jurisdiction.
- No operational guarantee. The Governance Authority does not guarantee that Implementations built according to the Specification Corpus will perform as intended, achieve any security objective, or satisfy any operational requirement.
- No warranty arising from course of dealing. No warranty arises from the Governance Authority's conduct in publishing, maintaining, or revising the Specification Corpus, or from any prior course of dealing between the Governance Authority and any Recipient.

Recipients are solely responsible for evaluating the suitability of the Specification Corpus for their intended purposes and for ensuring that their Implementations satisfy all applicable legal, regulatory, and operational requirements.

The Specification Corpus has not been designed, validated, or warranted for use in high-risk AI systems, critical infrastructure, safety-critical environments, or autonomous decision-making systems where failure or non-conformance could result in physical harm, regulatory liability, or systemic risk. Recipients deploying Implementations in such contexts

bear full and exclusive responsibility for all risk assessment, regulatory compliance, and safety validation applicable to their deployment environment.

Publication of the Specification Corpus does not constitute a representation that conformant implementations will satisfy the requirements of any safety certification, critical infrastructure standard, or regulatory regime, including without limitation those governing autonomous systems, medical devices, financial infrastructure, or public safety systems. No conformance determination under this corpus substitutes for applicable domain-specific certification obligations.

The Specification Corpus shall not be relied upon as a sufficient basis for deployment in safety-critical systems, autonomous decision-making environments, or regulated contexts where independent certification, domain-specific validation, or regulatory authorization is required.

SECTION 11

Limitation of Liability

To the maximum extent permitted by applicable law, the Governance Authority shall not be liable for any loss, damage, cost, or expense of any kind arising out of or in connection with: (a) the publication or availability of the Specification Corpus; (b) any Recipient's use of, reliance upon, or inability to use the Specification Corpus; (c) any error, omission, or ambiguity in the Specification Corpus; (d) any change to, withdrawal of, or failure to update the Specification Corpus; or (e) any act or omission of any Authorized Certification Body or other third party in connection with Certification activities.

This limitation applies to all categories of loss, including direct, indirect, incidental, consequential, special, exemplary, and punitive damages, and applies regardless of the theory of liability, whether in contract, tort, statute, or otherwise, and regardless of whether the Governance Authority has been advised of the possibility of such damages.

Where applicable law does not permit exclusion or limitation of certain categories of liability, the Governance Authority's liability shall be limited to the minimum extent required by such law.

Without limiting the generality of the foregoing, the Governance Authority shall not be liable for: (a) regulatory penalties, administrative fines, or compliance costs incurred by a Recipient in connection with any Implementation; (b) system failures, security incidents, data breaches, or operational disruptions arising from or related to any Implementation; (c) indirect, consequential, or downstream harms resulting from a Recipient's reliance on the Specification Corpus in safety-critical or high-risk deployment contexts; or (d) any loss or damage arising from the acts or omissions of Authorized Certification Bodies or third-party assessors. Each implementor bears full and sole responsibility for the security, reliability, and regulatory compliance of its own Implementation.

SECTION 12

Governance Authority

The QODIQA Governance Charter constitutes the supreme governance instrument of the Specification Corpus. This License is subordinate to the Governance Charter and shall be construed consistently with its provisions. In the event of any conflict between this License and the Governance Charter, the Governance Charter prevails.

12.1 Exclusive Authority

The Governance Authority holds exclusive authority over the following matters, which may not be delegated, transferred, or assumed by any other party except as expressly provided in the Governance Charter.

- Amendment, revision, and withdrawal of documents within the Specification Corpus

- Issuance of Certification determinations and authorization of Authorized Certification Bodies
- Specification evolution, versioning policy, and backward compatibility determinations
- Interpretation of this License and resolution of disputes arising under it
- Authorization of Trademark use beyond what is expressly permitted under Section 8
- Authorization of Derivative Specification naming under the QODIQA designation

12.2 Interpretive Authority

Within the scope of governance operations, including decisions concerning Certification, Trademark authorization, Derivative Specification approval, and amendment of the Specification Corpus, the Governance Authority's interpretation of this License shall be treated as authoritative and shall be the reference point for the resolution of internal disputes under the procedures defined in the Governance Charter.

Nothing in this Section shall be construed to limit, exclude, or override the jurisdiction of any court or competent adjudicative authority under applicable law. Courts retain authority to interpret and apply this License in accordance with applicable law, including the law of the governing jurisdiction identified in Section 15. The Governance Authority does not assert interpretive authority over matters that are properly subject to judicial determination.

Where ambiguity exists in the interpretation of any provision of this License, or where a dispute arises as to the application of License terms to a specific act or circumstance, the Governance Authority's written interpretive determination shall constitute the authoritative and binding interpretation for all governance proceedings, Certification

decisions, and Trademark authorizations conducted under this License, subject to any final judicial ruling on the same matter.

SECTION 13

Amendment Process

This License may be amended only by the Governance Authority acting through the procedures defined in the QODIQA Governance Charter. No community process, industry body, standards organization, or other entity outside the Governance Authority has authority to amend this License.

13.1 License Versioning

Each version of this License is identified by a version number in the form `QPSL v[Major].[Minor]`. The current version is `QPSL v1.0`. Amendments that alter the substantive rights or obligations of Recipients result in a new Major or Minor version. Recipients are bound by the version of this License that was in effect at the time they first accessed the Specification Corpus, unless they elect to be governed by a subsequent version.

13.2 Backward Compatibility

The Governance Authority will use reasonable efforts to ensure that amendments to this License do not impose materially more restrictive conditions on Permitted Uses that were expressly authorized under a prior version of this License, and that adequate notice of amendments is published prior to their effective date. These commitments are hortatory and do not create enforceable obligations upon the Governance Authority.

13.3 Prohibition on Unilateral Community Modification

No community process, pull request, public comment, or other collective action constitutes an amendment to this License. Purported amendments not issued by the Governance Authority are of no legal effect.

SECTION 14

Severability

If any provision of this License is held by a court or other competent authority to be invalid, unenforceable, or contrary to applicable law in any jurisdiction, such provision shall be severed from this License to the minimum extent necessary to render this License valid and enforceable in that jurisdiction. The remaining provisions of this License shall continue in full force and effect.

Where a severed provision reflects a material element of the bargain between the Governance Authority and Recipients (in particular, provisions relating to Certification Boundary Protection, Trademark and Naming, Derivative Specification restrictions, and Conformance Claim requirements) the Governance Authority reserves the right to treat the severance as grounds for declining to extend the Grant of Public Access under Section 2 in the relevant jurisdiction, pending replacement of the severed provision.

The unenforceability of any provision in one jurisdiction shall not affect its enforceability in any other jurisdiction.

SECTION 15

Governing Law and Jurisdiction

This License is governed by and construed exclusively in accordance with the laws of the jurisdiction in which the Governance Authority is legally domiciled, as recorded in the QODIQA Governance Charter. Conflict-of-laws principles

that would require application of the law of any other jurisdiction do not apply. Any court applying this License shall treat the law of the governing jurisdiction as the sole applicable substantive law.

15.1 Mandatory Local Law

Nothing in this License shall be construed to exclude, restrict, or override any right or obligation arising under mandatory provisions of applicable local law that cannot be waived by contract. Where any provision of this License conflicts with a mandatory provision of applicable local law in a given jurisdiction, the applicable local law provision prevails to the minimum extent necessary to establish compliance, and the remainder of this License continues in force.

15.2 Jurisdiction for Disputes

Any dispute arising under or in connection with this License that cannot be resolved through the Governance Authority's internal procedures under Section 12 shall be subject to the exclusive jurisdiction of the competent courts of the governing jurisdiction identified under this Section. Each party irrevocably submits to the personal jurisdiction of those courts and waives any objection to venue on grounds of inconvenience, improper venue, or lack of jurisdiction. Nothing in this License constitutes a mandatory arbitration clause, and no party is obligated to submit to arbitration without express agreement.

15.3 International Enforceability

The Governance Authority intends this License to be enforceable in all jurisdictions in which the Specification Corpus is accessed or implemented. Where enforcement requires adaptation to local legal requirements, such adaptation shall be made by the competent authorities of the relevant jurisdiction in a manner consistent with the purposes of this License, and shall not be construed as invalidating the License in other jurisdictions.

SECTION 16

Termination

The rights granted to a Recipient under this License terminate automatically upon the Recipient's material breach of any provision of this License. Material breach includes, without limitation, any act falling within the prohibitions of Section 4, any violation of Section 5, any false Conformance Claim under Section 6, any unauthorized Certification activity under Section 7, and any unauthorized Trademark use under Section 8.

16.1 Automatic Termination

Termination is automatic upon material breach and requires no notice, action, or communication from the Governance Authority. Upon termination, the Recipient must immediately cease all uses of the Specification Corpus beyond those that do not require a license, including cessation of any Conformance Claims and removal of any Trademark usage that was predicated upon the License.

16.2 Cure Period for Non-Willful Breach

Notwithstanding Section 16.1, where a breach is non-willful, that is, where the Recipient did not act with knowledge that the conduct in question constituted a violation of this License, the Recipient's rights are not automatically terminated if the Recipient cures the breach fully within thirty (30) calendar days of becoming aware of the breach. Cure must fully remedy all effects of the violation, including cessation of prohibited acts, correction of any published materials, and withdrawal of any unauthorized Conformance Claims or Trademark usage. The Governance Authority retains discretion to deny automatic cure effect in cases of repeated non-willful breach by the same Recipient.

The cure period established in this Section does not apply to: (a) false Certification claims under Section 6; (b) unauthorized use of official Certification marks or protected Certification instruments under Section 7; (c) fraudulent or

intentionally deceptive conduct; or (d) any breach that has caused material harm to the integrity of the Certification Framework. Such breaches remain immediately terminable under Section 16.1.

16.3 Reinstatement

A Recipient whose rights have terminated under this Section may seek reinstatement by: (a) promptly curing the breach to the satisfaction of the Governance Authority; (b) providing written notice to the Governance Authority of the cure; and (c) receiving written confirmation of reinstatement from the Governance Authority. Reinstatement is discretionary and is not guaranteed. Prior termination and reinstatement may be considered by the Governance Authority in evaluating future requests for Certification or Trademark authorization.

16.4 Effect on Derivative Works

Termination of a Recipient's rights under this License extends to all Derivative Specifications and materials produced by that Recipient under the authority of this License. Termination does not affect the rights of third parties who have independently received rights under this License, provided those third parties are not themselves in material breach.

SECTION 17

Enforcement and Remedies

The Governance Authority reserves all rights and remedies available under applicable law and in equity to enforce the terms of this License and to protect the integrity of the Specification Corpus, the Certification Framework, and the Trademark. The provisions of this Section are cumulative and do not limit any other right or remedy available to the Governance Authority.

Enforcement proceedings may be initiated by issuance of a written notice of violation identifying the specific breach and, where applicable under this License, specifying a reasonable period within which the violating party may cure the breach. If the breach is not fully cured within the specified period, or where no cure period applies under Section 17.5, the Governance Authority may escalate without further notice to legal remedies including injunctive relief, demands for correction, and any other available equitable or legal remedy. The Governance Authority's decision to issue or withhold a notice of violation does not limit its right to pursue immediate enforcement where the severity of the breach warrants it.

17.1 Right to Injunctive Relief

The Governance Authority shall be entitled to seek injunctive or other equitable relief in any court of competent jurisdiction to restrain any actual or threatened violation of this License, without the requirement to demonstrate actual damages, the inadequacy of monetary remedies, or any other condition beyond the existence or reasonable likelihood of a violation. The right to seek injunctive relief is in addition to, and not in substitution for, any other remedy available to the Governance Authority.

17.2 Right to Demand Cure and Correction

Upon identification of a violation of this License, the Governance Authority may require the violating party to: (a) immediately remove or retract any false Conformance Claims, misleading representations, or unauthorized Trademark uses; (b) issue a corrective statement to any audience that received or may have relied upon the violating materials; (c) permanently delete or destroy unauthorized reproductions of protected instruments; and (d) provide written confirmation of compliance with all corrective demands within a specified timeframe. Compliance with a demand for cure and correction does not, by itself, constitute reinstatement of License rights or waiver of any claims arising from the underlying violation.

17.3 Non-Waiver of Enforcement Rights

Failure by the Governance Authority to enforce any provision of this License on any occasion, or delay in initiating enforcement action, shall not constitute a waiver of the Governance Authority's right to enforce that provision or any other provision on any subsequent occasion. No course of dealing, pattern of conduct, or prior forbearance shall be construed as a modification of this License or as a waiver of any enforcement right. Each instance of non-enforcement is independent and does not prejudice any future enforcement action.

17.4 Cumulative Remedies

All rights and remedies provided in this License are cumulative and are not exclusive of any other rights or remedies that the Governance Authority may have at law, in equity, or under any other instrument or agreement. The exercise of any one remedy shall not preclude the exercise of any other remedy. The Governance Authority may pursue multiple remedies simultaneously or in any sequence it deems appropriate.

17.5 Immediate Enforcement for Severe Violations

Notwithstanding any cure period provisions set forth in Section 16, the Governance Authority reserves the right to initiate immediate enforcement proceedings, without prior notice or cure opportunity, in cases involving: (a) false or fraudulent Certification claims; (b) unauthorized use of official Certification marks, conformance badges, or other protected Certification instruments; (c) misrepresentation of Implementation status to a regulatory authority, a certification body, or the public; or (d) any conduct that the Governance Authority determines poses an immediate and material threat to the integrity of the Certification Framework or the informational value of Certification as a conformance signal. In such cases, the absence of a cure period is itself a condition of enforcement, not a procedural oversight.

SECTION 18

Institutional Continuity Clause

The QODIQA Specification Corpus constitutes a continuous institutional standard. Its governance structure, Certification infrastructure, and normative integrity are designed to persist across organizational changes, technological evolution, and external pressures. This Section establishes protections against forms of institutional fragmentation that would undermine the integrity and continuity of the Specification Corpus as a public technical reference.

> **Governance Continuity:** Challenges to the institutional integrity of the Specification Corpus through unauthorized forking, capture, or identity fragmentation do not divest the Governance Authority of its authority or alter the terms of this License.

18.1 Anti-Capture Provision

No party, including any past or present contributor to the Specification Corpus, any Authorized Certification Body, any commercial implementor, or any regulatory body, acquires governance authority over the Specification Corpus by virtue of contribution, adoption, endorsement, or commercial dependency. Governance authority is constituted exclusively under the QODIQA Governance Charter and may only be transferred or restructured through procedures defined therein.

18.2 Anti-Forking Under Identical Identity

The creation of a document corpus, certification program, or governance body that: (a) uses the QODIQA designation or any confusingly similar designation; (b) represents itself as a continuation, successor, or replacement of the official Specification Corpus; and (c) has not been authorized by or does not operate under the authority of the Governance Authority, constitutes a violation of this License and an infringement of the Governance Authority's Trademark rights, regardless of the motivations of the parties involved.

18.3 Continuity of Terms

This License, and the rights and obligations it establishes, runs with the Specification Corpus and binds all Recipients regardless of the channel through which they obtained access. Changes in the organizational structure, ownership, or operational status of the Governance Authority do not alter the terms of this License or extinguish any rights or obligations arising under it. In the event of a governance transition, the Governance Charter's succession provisions govern the identification of the successor authority.

18.4 Scope of Clause

Nothing in this Section shall be construed to prohibit legitimate academic critique, independent technical commentary, or the publication of alternative standards addressing similar technical problems under distinct designations that are not confusingly similar to QODIQA. The protections of this Section are directed at institutional identity integrity, not at the suppression of independent technical work.

DOCUMENT STATUS

Document Status and Classification

This document constitutes the normative governance license for the QODIQA specification corpus. It defines the conditions under which the corpus may be accessed, referenced, cited, implemented, and extended, and establishes the boundaries of permitted use, derivative specification rights, conformance claims, and certification authority.

The QODIQA Public Specification License is a governance instrument, not a permissive open-source license. Publication of the specification corpus is a deliberate act of standardization. The integrity of a deterministic enforcement standard depends upon the fidelity of its normative content, and this License exists to preserve that integrity.

This document is intended for:

- Organizations implementing or evaluating the QODIQA specification corpus
- Authors of derivative specifications or compatibility profiles
- Certification bodies and conformance test authors
- Legal and governance officers reviewing usage rights
- Regulatory and policy stakeholders engaged with AI governance standards

This document should be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0
- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Duțescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-PSLI-2026-001
TITLE	Public Specification License
SUBTITLE	Public Usage, Derivative Constraints, and Certification Boundary Framework
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Governance: Normative License
DOCUMENT STATUS	Historical record; superseded by CC BY 4.0 (see notice)
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 25 PART IV · GOVERNANCE AND LEGAL INSTRUMENTS

Positioning and Scope Limitation Statement

This document defines the institutional positioning of a deterministic runtime consent enforcement architecture, establishes precise technical and operational scope boundaries, and declares explicit non-claims regarding outcomes, guarantees, and authority beyond those boundaries.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-PSL-2026-001	1.0	2026	Positioning Statement	Institutional	NORMATIVE

PREAMBLE

This document defines the institutional positioning of QODIQA, establishes the precise boundaries of its technical and operational scope, and declares explicit non-claims with respect to outcomes, guarantees, and authority that lie outside those boundaries.

QODIQA is a deterministic runtime consent enforcement layer. It is neither a governance framework, a regulatory compliance solution, a safety certification system, nor a moral evaluation mechanism. The scope of QODIQA is bounded, defined, and immutable except through formal amendment under the applicable Governance Charter.

This document is intended to prevent scope misinterpretation, eliminate implied claims, reduce liability ambiguity, and provide a factual basis for evaluation by legal, regulatory, and technical review bodies. It must be read in conjunction with the full QODIQA Technical Corpus.

DOCUMENT STATUS

Intended Audience and Review Bodies

This document is addressed to technical, legal, regulatory, and policy review bodies evaluating the QODIQA standard. It is a non-marketing, strictly factual positioning instrument. It contains no aspirational language, no promotional framing, and no forward-looking claims.

Primary intended recipients include:

- Hyperscaler legal, safety, and policy review teams
- National and supranational AI regulatory authorities
- ISO/IEC and IEEE technical working groups
- Enterprise procurement and vendor risk assessment teams
- Formal certification and accreditation bodies
- Academic and independent technical reviewers
- Organizational AI governance and compliance functions

Document classification: **Public, Institutional Distribution.** Version 1.0 constitutes the initial formal release of this Positioning and Scope Limitation Statement. Future revisions require formal amendment proceedings under the QODIQA Governance Charter.

SECTION 00

Executive Positioning Summary

QODIQA introduces a deterministic enforcement boundary between AI system capability and legally authorized execution. It establishes a runtime control layer that ensures inference cannot occur outside explicitly defined consent and policy constraints.

QODIQA is a deterministic runtime consent enforcement layer that evaluates consent state, enforces declared authority boundaries, and gates AI inference execution through a fail-closed control mechanism. It does not modify model weights, does not perform moral evaluation, and does not constitute or replace a governance framework of any kind.

The rapid deployment of AI systems has created a structural gap between model capability and enforceable authority at runtime. Existing governance, compliance, and safety mechanisms operate outside the point of execution. This creates conditions in which technically possible actions may be executed without verifiable, real-time authorization. QODIQA addresses this gap by introducing deterministic enforcement at the inference boundary.

QODIQA operates at the inference boundary of AI system deployments. It intercepts execution requests, validates consent tokens against declared intent and policy, and permits or denies execution based on deterministic evaluation of those parameters. Enforcement is binary: execution proceeds only when all consent conditions are satisfied; otherwise, execution is denied. There are no partial approvals and no probabilistic outcomes.

BOUNDARY DEFINITION

QODIQA is a boundary control mechanism. Its scope begins at the point of an AI inference request and ends at the enforcement decision. All conditions prior to that point, including data preparation, model development, organizational policy design, and legal compliance, lie outside QODIQA's operational boundary. All conditions after that point, including downstream use, output interpretation, and impact on third parties, also lie outside QODIQA's operational boundary.

0.1 Definition in One Sentence

QODIQA is a deterministic runtime gate whose decision logic is stateless at the point of enforcement, meaning each enforcement decision is derived exclusively from the presented consent token, the registered policy snapshot, and the declared request parameters, without reference to adaptive memory, prior inference history, or probabilistic internal state, and which enforces a fail-closed execution policy at the inference boundary. Interaction with the consent registry and the generation of audit artifacts are external I/O operations and do not introduce stateful conditioning into the enforcement decision itself.

0.2 What QODIQA Does Not Attempt to Solve

The following problem domains are explicitly outside the scope of QODIQA and are not addressed by this standard in any form:

× Alignment of model behavior with human values or societal norms
× Safety of AI model outputs in any general, technical, or legal sense
× Fairness, bias detection, or mitigation in model predictions or outputs

×	Explainability of model internals, decision paths, or output rationale
×	Governance of the AI model development and training lifecycle
×	Legal compliance with data protection, AI regulation, or sector-specific obligations
×	Certification or assessment of AI system ethics under any framework
×	Prevention of misuse by operators deploying the system in bad faith
×	Elimination of residual risk inherent to AI system operation

1 The term "deterministic" throughout this document denotes behavior that is fully predictable, reproducible, and free from probabilistic inference, stochastic outcomes, or model-dependent variation.

2 "Fail-closed" denotes that QODIQA defaults to denial in all cases where consent state cannot be positively confirmed. Ambiguity results in denial, not in permissive fallback.

If adopted at scale, deterministic runtime enforcement establishes a verifiable execution boundary across AI systems, transforming consent and policy from declarative artifacts into enforceable conditions of operation. This introduces a new control layer within AI infrastructure, independent of model architecture and external governance frameworks.

Without deterministic enforcement at runtime, no AI system can establish a verifiable boundary between authorized and unauthorized execution.

SECTION 01

Core Functional Positioning

QODIQA performs a single, precisely defined function: it enforces runtime consent at the inference boundary. This section defines the five functional characterizations that constitute the complete scope of QODIQA's operational identity.

1.1 Five Functional Characterizations

CATEGORY	DESCRIPTION
A deterministic runtime consent enforcement layer	QODIQA evaluates whether a declared consent token satisfies all applicable policy conditions, including actor scope, purpose binding, temporal validity, and data classification, and enforces that evaluation as an absolute gate on inference execution.
A boundary control mechanism at inference-time	QODIQA's enforcement action is applied at the precise moment an inference request is submitted. It does not operate at rest, in training pipelines, or in post-execution contexts. Its jurisdiction is the inference boundary exclusively.
A fail-closed enforcement architecture	In the absence of a valid, positively confirmed consent token, QODIQA denies execution. No ambiguous, expired, malformed, or absent token results in permitted execution.
A control-plane artifact generation system	QODIQA produces structured audit artifacts, including enforcement records, denial events, and consent token validations, that are cryptographically bound, tamper-evident, and available for external audit and review.
A verification-constrained runtime gate	All enforcement decisions are made against a declared, immutable policy snapshot. QODIQA does not infer intent, does not exercise discretion, and does not apply probabilistic judgment to enforcement decisions.

1.2 Operational Clarifications

The following clarifications are provided to prevent mischaracterization of QODIQA's functional scope:

- × QODIQA does not alter model weights or any parameter of the underlying AI model.
- × QODIQA does not retrain, fine-tune, or modify any AI model before, during, or after enforcement.
- × QODIQA does not replace organizational governance structures, board-level oversight, or human accountability mechanisms.
- × QODIQA does not perform moral evaluation of AI outputs, use cases, or organizational intent.
- × QODIQA does not make recommendations or issue guidance on permissible AI uses beyond the formal policy constraints supplied by the deploying organization.

SECTION 02

Explicit Non-Claims

This section constitutes the formal non-claims declaration of the QODIQA standard. Each statement below is a precise, unconditional declaration of what QODIQA does not claim, does not guarantee, and does not provide. These declarations are not qualified by context, version, deployment configuration, or implementation detail. They are absolute.

> **NON-CLAIMS NOTICE**
>
> Conformance to the QODIQA standard, including full certification under its Conformance Test Suite, **does not imply, establish, or support any of the claims listed in this section.** Any representation to the contrary, whether by a deployer, integrator, or third party, is factually incorrect and not authorized by this standard.

2.1 Ethical and Safety Non-Claims

×	QODIQA does not guarantee ethical outcomes of AI system operation.
×	QODIQA does not guarantee the safety of AI model outputs in any technical, clinical, legal, or operational sense.
×	QODIQA does not evaluate, assess, or certify the alignment of any AI model with human values, societal norms, or ethical frameworks.
×	QODIQA does not provide fairness guarantees with respect to protected characteristics, demographic groups, or decision outcomes.
×	QODIQA does not detect, measure, or mitigate bias in model training data, inference outputs, or deployment patterns.
×	QODIQA does not provide explainability of model internals, attention mechanisms, feature attribution, or inference logic.

2.2 Legal and Regulatory Non-Claims

- × QODIQA neither guarantees nor certifies legal compliance with any law, regulation, or jurisdiction-specific requirement, including GDPR, the EU AI Act, CCPA, HIPAA, or any other regulatory instrument.
- × QODIQA does not constitute, substitute, or reduce the obligations of a legal compliance program. Certification under this standard confers no regulatory immunity, safe harbor status, or presumption of conformity under any applicable law.
- × Deploying organizations remain solely and fully legally responsible for the lawfulness of their AI system operations.

2.3 Governance and Oversight Non-Claims

- × QODIQA does not replace organizational governance structures, board oversight, executive accountability, or human oversight of AI systems at any operational level.
- × QODIQA does not reduce the obligations of organizational compliance programs.
- × QODIQA does not audit datasets, training corpora, or data supply chains.
- × QODIQA does not modify model cognition, inference behavior, or output distributions.

2.4 Security and Risk Non-Claims

- × QODIQA does not eliminate misuse risk from AI system deployments.
- × QODIQA does not prevent all attack vectors, bypass methods, or adversarial exploitation attempts.
- × QODIQA does not guarantee the cybersecurity posture of any system in which it is deployed.
- × QODIQA does not globally certify any AI system against any certification scheme, standard body, or national authority.

3 These non-claims apply regardless of the deployment configuration, version of the QODIQA standard employed, or the certification tier achieved by the deploying organization.

4 The enumeration of non-claims in this section is not exhaustive. Absence of a specific non-claim from this list does not imply that QODIQA provides the corresponding capability, guarantee, or assurance.

SECTION 03

Boundary of Technical Scope

The technical scope of QODIQA is defined by the operations it performs, the system boundaries within which it operates, and the artifact types it produces. This section establishes those boundaries with precision. All technical functions, system domains, and operational contexts not explicitly enumerated as within scope are to be treated as outside scope.

3.1 Defined Technical Scope

SCOPE DOMAIN	DEFINITION
Runtime-only scope	QODIQA operates exclusively during the execution lifecycle of an AI inference request. It has no function in pre-execution, training, or post-deployment phases.
Enforcement gateway boundary	QODIQA's enforcement action occurs at the single point of the inference gateway. Policy decisions are made at this boundary and are not propagated into model internals.
Artifact validation scope	QODIQA validates consent tokens and policy artifacts submitted as part of the inference request. Validation is deterministic and policy-based; it does not involve semantic interpretation of content.
Policy evaluation scope	QODIQA evaluates execution requests against an immutable, pre-registered policy snapshot. It does not dynamically modify policies, resolve policy conflicts, or exercise discretion in policy interpretation.
Consent registry interaction scope	QODIQA interacts with the consent registry solely for the purpose of resolving consent token validity at runtime. It does not manage, administer, or govern the consent registry itself.
Audit log generation scope	QODIQA generates cryptographically bound audit records for each enforcement decision. These records are production artifacts of the enforcement layer and do not constitute a general audit or compliance program.

3.2 Explicit Technical Exclusions

The following technical domains are explicitly excluded from the scope of QODIQA:

- × Training data pipelines, data labeling operations, and data quality assurance processes.
- × Model development lifecycle, including architecture design, training, evaluation, and deployment staging.
- × MLOps governance, continuous integration, model versioning, and production monitoring infrastructure.
- × Hardware-level safety controls, including trusted execution environments and secure enclaves not directly integrated with the QODIQA enforcement gateway.
- × Corporate compliance structures, risk management frameworks, and internal audit functions.
- × Application-layer security controls, network security, and infrastructure hardening outside the enforcement gateway perimeter.

TABLE 1 · TECHNICAL SCOPE SUMMARY

DOMAIN	WITHIN QODIQA SCOPE	OUTSIDE QODIQA SCOPE
Inference execution gate	Yes, enforcement decision	Not applicable
Consent token validation	Yes, deterministic evaluation	Not applicable
Audit artifact generation	Yes, tamper-evident record	Not applicable
Model training and weights	Not applicable	Not within scope
Output safety evaluation	Not applicable	Not within scope
Legal compliance program	Not applicable	Not within scope
Data labeling and curation	Not applicable	Not within scope

DOMAIN	WITHIN QODIQA SCOPE	OUTSIDE QODIQA SCOPE
General AI governance	Not applicable	Not within scope

SECTION 04

Regulatory Positioning Boundary

QODIQA maps to specific technical control requirements referenced by regulatory instruments, including the EU AI Act, GDPR, and NIST AI RMF. This mapping reflects the capacity of deterministic runtime consent enforcement to satisfy certain technical preconditions for regulatory compliance. It does not constitute regulatory compliance, legal certification, or authorization by any regulatory body.

Non-Reliance Clause. This document does not constitute legal advice. No party may rely on this document, or on any content within the QODIQA Technical Corpus, as a substitute for independent legal counsel. QODIQA makes no representation regarding the legal sufficiency of its standard, its certification process, or its regulatory alignment mappings for any specific jurisdiction or use case. Regulatory interpretation and the determination of legal obligations remain exclusively the responsibility of the deploying organization and its qualified legal advisors.

> REGULATORY LIMITATION
>
> **Conformance to the QODIQA standard does not imply legal compliance under any jurisdiction.** Deploying organizations remain solely responsible for ensuring that their AI system operations comply with all applicable laws, regulations, and sector-specific obligations in the jurisdictions in which they operate.

4.1 Nature of Regulatory Mapping

QODIQA's Regulatory Alignment Matrix identifies control domains in which deterministic runtime consent enforcement provides technical evidence relevant to regulatory obligation. This mapping is provided as an informational aid for compliance teams and does not:

- × Constitute a legal opinion on the applicability or interpretation of any regulatory instrument.
- × Guarantee that implementation of QODIQA satisfies any specific regulatory requirement.
- × Represent an official position of any regulatory authority.
- × Remove or reduce the obligation of organizations to conduct independent legal and compliance assessments.

4.2 Organizational Legal Responsibility

The following responsibilities remain with the deploying organization and are not delegated to or assumed by QODIQA under any circumstances:

- — Assessment of the legality of specific AI use cases under applicable law.
- — Selection, design, and implementation of a complete compliance program.
- — Data subject rights management, including access, rectification, erasure, and portability.
- — Appointment of Data Protection Officers, AI compliance officers, or other designated roles required by regulation.
- — Registration, notification, or authorization obligations with regulatory bodies.

- Impact assessments, including Data Protection Impact Assessments and Fundamental Rights Impact Assessments.

5 References to regulatory instruments in this section are provided for informational purposes. QODIQA makes no representation as to the current legal status, amendments, or enforcement interpretations of any referenced regulation.

SECTION 05

Security Scope Limitation

QODIQA enforces consent constraints deterministically within its defined enforcement boundary. It is not a cybersecurity product, a threat detection system, or a substitute for information security controls. This section defines the precise relationship between QODIQA's enforcement functions and the broader security posture of AI deployments.

5.1 What QODIQA Provides Within Its Security Boundary

- Deterministic enforcement of consent-based access controls at the inference gateway.
- Cryptographically bound audit artifacts that provide tamper-evident records of enforcement events.
- Fail-closed enforcement that defaults to denial under conditions of token invalidity, expiry, or absence.
- Policy-bound execution gates that prevent inference from proceeding without explicit, current consent authorization.

5.2 Security Capabilities Not Provided by QODIQA

- × QODIQA does not guarantee the cybersecurity posture of any system or infrastructure component in which it operates.
- × QODIQA does not replace ISO/IEC 27001 or any equivalent information security management standard.
- × QODIQA does not detect or mitigate insider risk, privilege abuse, or lateral movement within the deploying organization's infrastructure.
- × QODIQA does not prevent all bypass attacks against the enforcement gateway, including attacks that operate at the infrastructure level below the enforcement layer.
- × QODIQA does not provide cryptographic assurance of model integrity, supply chain security, or hardware root-of-trust verification.
- × QODIQA does not constitute a penetration-tested security control; its security properties require independent assessment in each deployment context.

5.3 Infrastructure Assumption

QODIQA assumes deployment within a secure, appropriately hardened infrastructure environment. The integrity of the enforcement gateway is predicated on the integrity of the infrastructure in which it operates. QODIQA's security properties are not maintained in environments where the infrastructure has been compromised. Deploying organizations are responsible for ensuring that the deployment environment meets the security prerequisites specified in the QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0.

Within its defined boundary, QODIQA reduces the risk of unauthorized inference execution, prevents execution under invalid or expired consent states, and eliminates ambiguity in enforcement decisions at runtime. It does not eliminate all attack vectors, but it constrains execution to explicitly authorized conditions.

SECTION 06

Organizational Responsibility Retention Clause

QODIQA is a technical constraint mechanism. It enforces consent boundaries that are defined by the deploying organization. It does not assume, reduce, or transfer the legal, operational, or ethical responsibilities of that organization. This section establishes the formal principle of organizational responsibility retention with respect to QODIQA deployment.

> *All legal, operational, ethical, and governance responsibility for the deployment and use of AI systems in which QODIQA is integrated remains, in its entirety, with the deploying organization. QODIQA provides constraint enforcement; it does not provide accountability.*

6.1 Non-Delegation of Accountability

The following principles govern the relationship between QODIQA enforcement and organizational accountability:

- Decision accountability for AI system outcomes is external to QODIQA. Enforcement of a consent constraint does not constitute an organizational decision; it reflects a decision already made by the organization in the form of a registered policy.
- Governance accountability cannot be delegated to runtime constraints. QODIQA's function is to enforce policy, not to substitute for the governance process that produces that policy.
- The legality of the policies registered within QODIQA is the sole responsibility of the deploying organization. QODIQA does not evaluate, validate, or certify the legal adequacy of any registered policy.

— Impact on data subjects, third parties, or society resulting from AI system operation under QODIQA enforcement is the responsibility of the deploying organization, not of the QODIQA standard or its implementing systems.

6.2 Scope of QODIQA's Provision

QODIQA provides deterministic constraint enforcement. It does not provide: organizational accountability, ethical judgment, legal authorization, regulatory approval, human oversight, risk management, or governance structure. The presence of QODIQA in a deployment does not reduce the obligation of the deploying organization to maintain all applicable governance and compliance mechanisms independently and in full.

6 This clause applies to all entities involved in the deployment of QODIQA-integrated systems, including primary deployers, system integrators, downstream operators, and any other party exercising control over the deployment environment.

SECTION 07

Misuse and Abuse Limitation

QODIQA cannot, by design, prevent all forms of misuse or abuse. This section defines the specific limitations of the standard with respect to intentional misuse, bad-faith deployment, and residual abuse vectors that remain outside the scope of its enforcement architecture.

7.1 Intentional Malicious Deployment

QODIQA cannot prevent an operator from deploying the system in a manner that is deliberately harmful, deceptive, or contrary to the intent of the policies registered within it. An operator with administrative access to the enforcement

gateway can disable, misconfigure, or remove QODIQA's enforcement mechanisms. QODIQA provides no technical control that is immune to operator-level override.

7.2 Good-Faith Assumption

QODIQA operates on the assumption that the deploying organization has registered policies that accurately reflect its legal and governance obligations, that the consent registry is maintained with integrity, and that the enforcement gateway is not deliberately compromised. QODIQA does not verify organizational intent, does not audit the accuracy of registered policies against external requirements, and does not detect deliberate bad-faith deployment.

7.3 Residual Misuse Risk

× QODIQA does not prevent the registration of policies that are themselves harmful, unlawful, or ethically deficient.
× QODIQA does not prevent an authorized operator from modifying consent tokens to circumvent intended restrictions.
× QODIQA does not prevent misuse that occurs within the scope of a validly authorized consent token.
× QODIQA does not prevent deployment configurations that technically satisfy enforcement requirements while producing harmful operational outcomes.

Detailed threat modeling assumptions, adversarial attack surface characterizations, and abuse case enumerations applicable to QODIQA deployments are defined in the QODIQA: Threat Model and Abuse Case Specification, Version 1.0. It does not constitute a complete adversarial analysis, and threat enumeration is externalized to that document by design.

RESIDUAL RISK NOTICE

Residual misuse risk is present in all QODIQA deployments. The standard reduces specific, well-defined categories of unauthorized AI execution. It does not eliminate the risk of AI misuse in all its forms. Organizations bear responsibility for managing residual misuse risk through complementary governance, oversight, and technical controls.

SECTION 08

Architectural Non-Expansion Clause

The scope of QODIQA is bounded and immutable as defined in this document. This section establishes the formal non-expansion clause, which constrains the interpretation, development, and extension of the QODIQA standard against scope creep, mission expansion, and overreach.

QODIQA is a runtime enforcement layer. It is not, and does not aim to become, a general AI governance framework, a regulatory authority, a global AI control mechanism, or a doctrine-setting body for AI ethics.

8.1 Formal Non-Expansion Declarations

× QODIQA does not aim to become a general AI governance framework encompassing model development, deployment strategy, organizational structure, or operational policy.
× QODIQA does not aim to replace, duplicate, or supplant the function of regulatory bodies at the national, supranational, or international level.
× QODIQA does not aim to centralize global AI control or to establish a universal enforcement authority over AI system operation.

× QODIQA does not define AI ethics doctrine, principles, or normative frameworks. It enforces policies defined externally by deploying organizations; it does not author those policies.
× QODIQA is not a policy authority. It does not issue guidance, recommendations, or normative positions on permissible AI use cases beyond the bounds of the technical standard.
× QODIQA does not seek to establish a market authority, certification monopoly, or single-source-of-truth for AI consent enforcement globally.

8.2 Scope Extension Procedure

Any extension of the functional scope of QODIQA beyond the boundaries defined in this document requires a formal amendment proceeding under the QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0. No extension is permitted through interpretive expansion, implied inclusion, or informal community agreement. The burden of proof for any proposed extension is that it falls within the bounded mission of deterministic runtime consent enforcement, as defined herein.

SECTION 09

Interaction with Existing Standards

QODIQA operates within an existing landscape of information security, AI governance, and regulatory compliance standards. This section defines the nature of the relationship between QODIQA and those standards. QODIQA complements selected existing standards by providing runtime enforcement evidence for specific control domains. It does not supersede, replace, or render redundant any existing standard.

> COMPLEMENTARY MECHANISM
>
> QODIQA is a complementary technical mechanism. **Its presence does not reduce the applicability or requirements of any external standard** to a deploying organization. All obligations arising from external standards remain in full force independently of QODIQA deployment.

STANDARD	COMPLEMENTARY RELATIONSHIP	NON-SUPERSESSION STATEMENT
ISO/IEC 27001	QODIQA provides deterministic access control evidence for AI inference boundaries, supporting selected controls in Annex A.	QODIQA does not replace ISO/IEC 27001 certification or the Information Security Management System it mandates.
ISO/IEC 42001	QODIQA provides technical enforcement artifacts relevant to AI management system control objectives at the inference layer.	QODIQA does not replace ISO/IEC 42001 or the AI Management System it establishes.
NIST AI RMF	QODIQA maps to selected GOVERN, MAP, and MEASURE functions within the AI Risk Management Framework, specifically for runtime enforcement evidence.	QODIQA does not replace NIST AI RMF or any profile derived from it.
EU AI Act	QODIQA supports technical control obligations in the EU AI Act relevant to runtime access control, logging, and human oversight facilitation.	QODIQA does not constitute compliance with the EU AI Act and does not replace a conformity assessment procedure.
GDPR	QODIQA provides purpose-bound, time-limited consent enforcement that supports lawful basis documentation for certain data processing operations.	QODIQA does not constitute a GDPR compliance program and does not replace legal obligations under Chapters II through V of the GDPR.

7 The standards referenced in this section are cited for informational purposes. The characterization of complementary relationships does not imply endorsement by or formal recognition from the publishing bodies of those standards.

SECTION 10

Certification Boundary Clarification

Certification under the QODIQA standard provides verifiable evidence that runtime consent enforcement is implemented as a deterministic, fail-closed control at the inference boundary. It confirms that execution cannot proceed in the absence of valid consent state as defined by policy constraints. Certification validates enforcement integrity, not outcome quality or regulatory compliance.

QODIQA certification, as defined by the QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0, validates a single, precisely bounded outcome: that the QODIQA enforcement gateway operates in conformance with the deterministic runtime consent enforcement specification. This section defines what certification does and does not signify.

10.1 What Certification Validates

- That the enforcement gateway correctly evaluates consent tokens against registered policy for the defined control points.
- That the fail-closed behavior operates as specified under conditions of invalid, expired, absent, or malformed tokens.
- That audit artifacts are generated, formatted, and bound to enforcement events in accordance with the QODIQA specification.
- That enforcement decisions are deterministic and reproducible across equivalent input conditions.
- That the implementation satisfies the technical conformance requirements defined in the QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0 at the applicable tier level.

10.2 What Certification Does Not Validate

- × Certification does not validate the ethics of the AI model being governed, its outputs, or its use cases.
- × Certification does not validate the integrity, provenance, or quality of training data.
- × Certification does not validate the safety of AI system outputs in any technical, clinical, or operational sense.
- × Certification does not validate regulatory compliance under any law or regulatory framework, and confers no regulatory immunity or presumption of conformity under any applicable law.

× Certification does not constitute an assurance that the deploying organization's policies are legally adequate, ethically appropriate, or operationally sufficient.
× Certification does not certify the security posture of the deployment environment, nor does it constitute or imply endorsement by any regulatory authority, national body, or standards organization.

> CERTIFICATION SCOPE
>
> **QODIQA certification is a technical conformance statement.** It confirms that the enforcement gateway behaves as specified. It does not evaluate, assess, or certify anything beyond the specified behavior of the enforcement layer itself.

SECTION 11

Residual Risk Acknowledgment

All AI systems contain residual risk. Residual risk refers to the risk that remains after all defined controls have been applied. QODIQA's deterministic runtime enforcement reduces specific, bounded categories of risk associated with unauthorized AI inference execution. It does not eliminate residual risk from AI system deployments, and it makes no representation to that effect.

No deterministic enforcement mechanism eliminates systemic AI risk. Runtime constraint layers cannot neutralize risks originating in model architecture, training data composition, or the aggregate effects of large-scale AI deployment. These risk categories are structural in nature and lie outside the reach of any enforcement boundary control.

> RESIDUAL RISK STATEMENT
>
> QODIQA reduces the risk of unauthorized, purpose-mismatched, or temporally invalid AI inference execution. **It does not reduce risks arising from model behavior, training data quality, output interpretation, organizational misuse, or systemic AI risk at the societal level.**

11.1 Risk Categories Addressed by QODIQA

- Unauthorized inference execution without valid consent token.
- Purpose-mismatched execution where declared intent does not correspond to registered policy.
- Temporally invalid execution where consent has expired, been revoked, or is not yet active.
- Scope-exceeded execution where the data classification or actor scope of the request exceeds authorized boundaries.
- Unaudited execution where enforcement events are not recorded in tamper-evident form.

11.2 Risk Categories Not Addressed by QODIQA

RISK CATEGORIES OUTSIDE QODIQA SCOPE · REQUIRE EXTERNAL GOVERNANCE

RISK CATEGORY	DESCRIPTION	REQUIRED MITIGATION SOURCE
Model output harm	Harmful, inaccurate, or biased outputs produced by an authorized inference execution.	Safety evaluation, red-teaming, output monitoring; external to QODIQA.
Training data risk	Privacy violations, bias, or legal exposure arising from training data composition.	Data governance, legal review, and ML lifecycle controls; external to QODIQA.
Policy design risk	Inadequately designed consent policies that permit harmful or unlawful use cases.	Legal and governance review of registered policies; external to QODIQA.
Systemic AI risk	Societal, economic, or geopolitical risks arising from the widespread deployment of AI systems.	Regulatory intervention, international governance frameworks; external to QODIQA.

Operator misuse	Deliberate deployment of QODIQA in bad faith by an operator seeking to circumvent its purpose.	Organizational accountability, legal enforcement; external to QODIQA.

11.3 External Management Requirement

Residual risk that falls outside QODIQA's enforcement scope must be managed through complementary controls established and maintained by the deploying organization. QODIQA does not specify, recommend, or mandate the nature of those complementary controls. The selection and implementation of all controls addressing residual risk beyond QODIQA's defined scope is the sole responsibility of the deploying organization and its governance structures.

SECTION 12

Formal Scope Freeze Declaration

This section constitutes the formal Scope Freeze Declaration for the QODIQA standard. It defines the conditions under which the scope of QODIQA may be modified, the process required for any such modification, and the prohibition on implicit or interpretive scope expansion.

> *The scope of QODIQA, as defined in this Positioning and Scope Limitation Statement, is frozen at the boundaries established herein. No activity, publication, certification, implementation, or interpretive practice may extend this scope absent formal amendment under the QODIQA, Governance Charter for the QODIQA Standard Corpus, Version 1.0.*

12.1 Scope Immutability Statement

The scope of QODIQA as defined in this document is the complete and exclusive scope of the standard. It is immutable through the following mechanisms:

- No publication by the QODIQA standards body, including technical specifications, implementation guides, or advisory documents, may expand scope without a formal amendment proceeding.
- No certification body operating under a QODIQA license may certify behaviors, properties, or capabilities outside the defined scope.
- No implementation of QODIQA may be represented as satisfying a function that falls outside the defined scope, regardless of technical capability.
- No marketing, communication, or positioning material associated with QODIQA may represent its scope as broader than defined herein.

12.2 Amendment Procedure

Any proposed expansion of QODIQA's scope must satisfy all of the following conditions prior to adoption:

- Formal submission of a scope amendment proposal under the QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0 amendment procedure.
- Technical review by the QODIQA Standards Body confirming that the proposed expansion falls within the mission of deterministic runtime consent enforcement.
- Review and approval by the QODIQA Governance Board under the defined supermajority threshold.

- Publication of a revised Positioning and Scope Limitation Statement, incorporating the amendment, with a new version number and formal effective date.
- A defined transition period during which prior-version implementations retain valid scope characterization.

12.3 Prohibition on Interpretive Elasticity

SCOPE FREEZE: CONSTITUTIONAL DECLARATION

The scope defined in this document admits of no interpretive elasticity. Where the scope of a proposed activity, representation, or certification is ambiguous with respect to these boundaries, the ambiguity is to be resolved by reference to the literal definitions contained herein. If the proposed activity cannot be clearly grounded in those literal definitions, it falls outside scope.

No implied inclusion is permitted. The fact that a capability, guarantee, or function would be desirable, beneficial, or complementary to QODIQA's defined function does not constitute grounds for treating it as within scope. Scope is defined by explicit inclusion; it is not expanded by proximity, analogy, or practical convenience.

No informal consensus, working group agreement, or community practice may constitute a de facto scope expansion. Only formal amendment under the QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0 produces a binding scope change. All informal understandings to the contrary are without legal or technical force.

QODIQA POSITIONING AND SCOPE LIMITATION STATEMENT: V1.0 SCOPE FREEZE IN EFFECT: AMENDMENT REQUIRED FOR MODIFICATION

8 This Scope Freeze Declaration supersedes all prior informal characterizations of QODIQA's scope. From the date of this document's formal issue, this declaration constitutes the authoritative and exclusive definition of QODIQA's institutional scope.

9 Any entity representing QODIQA's scope in terms inconsistent with this document bears sole responsibility for that misrepresentation. The QODIQA standards body assumes no liability for misrepresentations made by third parties, implementers, or licensees.

DOCUMENT STATUS

Document Status and Classification

This document defines the institutional positioning of QODIQA and establishes the precise boundaries of its technical and operational scope. It is a Positioning and Scope Limitation Statement, a formal instrument within the QODIQA Technical Corpus intended to prevent scope misinterpretation and eliminate implied claims. It is issued as an Institutional Positioning Document and is not a technical specification, a commercial instrument, or a legal opinion.

QODIQA is not presented as a theoretical framework, but as an implementable infrastructure standard designed to introduce enforceable authorization, deterministic policy execution, runtime verifiability, and auditable proof mechanisms into AI-native systems. All technical principles described in this document are designed to be production-oriented and architected for integration within large-scale AI ecosystems.

The material contained herein is intended for:

- Systems architects and infrastructure engineers
- Enterprise AI platform designers
- Security and governance officers
- Regulatory and policy stakeholders
- Organizations building or operating autonomous AI systems

This document should be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0
- QODIQA: Regulatory Alignment Matrix for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Threat Model and Abuse Case Specification, Version 1.0
- QODIQA: Residual Risk and Assumption Disclosure Annex, Version 1.0
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Duțescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-PSL-2026-001
TITLE	Positioning and Scope Limitation Statement
SUBTITLE	Institutional Positioning, Boundary Definition, and Explicit Non-Claims Framework
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Institutional Positioning Document
DOCUMENT STATUS	Normative, Institutional Positioning Document
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

PART V

V

Contextual, analytical and process support

7 documents

26	Executive Brief	953
27	Economic Impact Analysis	971
28	Regulatory Alignment Matrix	1022
29	Audit and Evidence Generation	1065
30	Audit Readiness and Evidence Pack	1102
31	Use Case Dossiers	1128
32	Corpus Change Log	1183

CHAPTER 26 PART V · CONTEXTUAL, ANALYTICAL AND PROCESS SUPPORT

Executive Brief for Deterministic Runtime Consent Enforcement

A concise institutional presentation of a framework for deterministic, machine-enforceable consent in artificial intelligence systems.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-EB-2026-001	1.0	2026	Executive Brief	Normative	NORMATIVE

ABSTRACT

This executive brief presents the institutional case for deterministic runtime consent enforcement as a necessary architectural property of artificial intelligence systems. Current consent mechanisms, whether contractual, policy-based, or procedural, operate outside the execution layer and cannot provide enforceable guarantees at runtime. QODIQA addresses this gap by positioning consent evaluation as a deterministic, machine-readable gate within the AI inference pipeline. This document characterizes the problem, describes the QODIQA framework at a strategic level, examines system-level implications, and articulates regulatory and operational consequences for deploying organizations.

SECTION 0

Executive Overview

Artificial intelligence systems now operate in consequential domains, including healthcare, financial services, legal analysis, and critical infrastructure, where the boundary between advisory output and operational action is increasingly narrow. In each of these contexts, consent is a foundational requirement: the operating premise that a human principal has authorized a specific class of action, in a specific context, under defined conditions.

The problem is structural. Consent, as currently implemented across AI deployments, exists exclusively as a pre-execution agreement: a contract, a terms-of-service clause, a policy document, or a governance control. None of these mechanisms operate within the runtime. Once an AI system begins processing, no technical layer evaluates whether the action being executed falls within the scope of what was consented to. Consent is asserted; it is never verified.

QODIQA, the Quantified Operational Declarative Infrastructure for Qualified Assent, is a framework for correcting this structural deficiency. It defines a deterministic consent enforcement layer that operates at the inference boundary, evaluating each AI action against a machine-readable consent record before execution proceeds. The result is a system in which consent is not a policy precondition but an execution property: a runtime invariant that either holds or terminates the operation.

> **CENTRAL PROPOSITION**
>
> Consent in AI systems must be enforced at the execution layer, not declared at the agreement layer. QODIQA provides the architectural specification for this enforcement.

This brief is addressed to institutional decision-makers evaluating the strategic implications of QODIQA adoption: executive leadership, legal and compliance officers, risk management functions, and technical architects responsible for AI governance. It does not assume prior familiarity with the full QODIQA technical corpus.

SECTION 1

Problem Definition

The deployment of AI systems in consequential operational contexts has outpaced the development of consent mechanisms adequate to those contexts. The gap is not primarily one of regulatory coverage or organizational intent. It is an architectural gap: the absence of any mechanism capable of enforcing consent at the moment an AI system acts.

Consent failures in AI deployments take several forms. An AI system may perform an action outside the scope of what a user authorized, because no mechanism exists to check scope at runtime. It may act on stale consent, a permission granted under conditions that have since changed, because no mechanism exists to verify temporal validity at execution time. It may aggregate data across multiple consented interactions in ways that exceed the sum of individual authorizations, because consent records are evaluated independently rather than cumulatively. It may proceed in the absence of any consent record, because the default behavior of most AI systems is permissive, not restrictive.

> STRUCTURAL DIAGNOSIS
>
> The fundamental problem is not that consent is absent from AI governance. It is that consent exists only as a pre-execution representation, a legal or policy artifact, rather than a runtime constraint. A representation of consent that cannot be verified at the moment of execution provides no operational guarantee.

This structural condition has direct legal, regulatory, and operational consequences. Legally, an organization cannot demonstrate that a specific AI action fell within the scope of a specific consent authorization if no record of runtime evaluation exists. Regulatorily, compliance obligations that require verifiable consent cannot be satisfied by systems that lack the capacity to enforce consent at runtime. Operationally, the inability to constrain AI behavior to consented scope creates exposure to outcomes that were neither authorized nor anticipated.

The problem is not addressable by improving existing consent documentation or governance procedures. Those approaches operate at the agreement layer. The deficiency is in the execution layer, and it requires an execution-layer

solution.

SECTION 2

Failure of Current Consent Models

Three dominant consent models are applied to AI systems: contractual consent, policy-based consent, and procedural consent. Each is structurally limited in its capacity to govern AI behavior at runtime.

Contractual Consent

Contractual consent establishes the legal basis for AI system operation. It specifies permitted use cases, data handling obligations, and authorized scope of action. However, contracts are interpreted and enforced ex post, not at the moment of execution. A contract cannot prevent an out-of-scope action; it can only provide legal recourse after one has occurred. In high-frequency, low-latency AI environments, this distinction is operationally significant: the harm of an out-of-scope action may be irreversible before legal mechanisms engage.

Policy-Based Consent

Policy-based consent translates contractual obligations into operational rules: data classification policies, access controls, usage restrictions. These controls are closer to the execution layer but remain external to it. They govern what data an AI system may access and in what contexts it may be invoked, but they do not evaluate whether a specific inference action falls within the scope of what was authorized. Policy controls operate at system boundaries; they do not operate within the inference pipeline.

Procedural Consent

Procedural consent relies on organizational processes, including training, review workflows, and audit procedures, to ensure AI system behavior remains within authorized scope. These controls are the most remote from execution. They depend on human judgment applied before or after the fact and provide no technical guarantee of runtime compliance. Audit procedures can identify violations after they occur; they cannot prevent them.

> **COMMON FAILURE MODE**
>
> All three models share a single structural deficiency: none evaluates consent at the moment of execution. Each operates outside the inference pipeline and therefore cannot provide deterministic enforcement of consent boundaries at runtime.

Supplementary technical controls, such as content filters, output classifiers, and model alignment techniques, address behavioral properties of AI systems but do not address the consent enforcement gap directly. A system can produce output that is technically compliant with content policies while acting outside the scope of what any user authorized. These are distinct failure modes, and the former does not mitigate the latter.

SECTION 3

Deterministic Runtime Consent Enforcement

Deterministic runtime consent enforcement is the architectural approach in which consent evaluation is performed as a mandatory gate within the AI execution pipeline, producing a binary outcome, permit or deny, before any inference action proceeds. The evaluation is deterministic: given identical inputs, it produces identical outputs. It is runtime: it executes at the moment of action, not before or after. It is consent-based: it evaluates the action against a verified record of what a principal has authorized.

QODIQA specifies this enforcement layer as a system component with defined inputs, a defined evaluation procedure, defined outputs, and defined failure behavior. The key architectural properties are as follows.

Machine-Readable Consent Records

Consent is represented as a structured, machine-readable artifact, not a natural-language document subject to interpretation, but a formal record specifying the authorizing principal, the authorized action classes, the context in which authorization applies, and the temporal boundaries of validity. The consent record is the input to the enforcement layer; it is not a reference document consulted by human reviewers.

Deterministic Evaluation

The enforcement layer evaluates each incoming action against the applicable consent record using a defined policy evaluation function. The function is deterministic: it produces the same output for the same input on every invocation. There is no probabilistic component, no discretionary judgment, no fallback to default permissiveness. Ambiguity in the consent record resolves to denial, not approval.

Fail-Closed Semantics

In the absence of a valid, applicable consent record, the enforcement layer denies the action. The system does not proceed under an assumption of implied consent. The default operational posture is restrictive. This is a non-negotiable property of any enforcement layer that provides meaningful guarantees: a system that proceeds without consent records in place does not enforce consent; it merely records its absence.

Cryptographic Auditability

Each enforcement decision, permit or deny, is recorded with a cryptographic integrity guarantee. The audit record cannot be modified without detection. The combination of deterministic evaluation and tamper-evident logging produces a

verifiable enforcement history: an institutional record demonstrating that every AI action was evaluated against a consent record and that the evaluation result was preserved.

> **ARCHITECTURAL POSITIONING**
>
> The QODIQA enforcement layer is positioned between the input processing stage and the inference execution stage of the AI pipeline. It is not a post-processing filter and not a pre-deployment governance control. It is an inline evaluation gate that no action can bypass.

SECTION 4

System-Level Implications

The introduction of a deterministic consent enforcement layer has implications that extend beyond the AI system itself. These implications affect how AI systems integrate with organizational infrastructure, how consent records are managed as a class of operational data, and how accountability is structured across the AI deployment stack.

Consent as Operational Data

In QODIQA-compliant systems, consent records are operational data in the same sense that configuration parameters and access credentials are operational data. They must be created, versioned, validated, stored, and retired as part of the operational lifecycle of the AI system. Organizations deploying QODIQA-compliant systems must treat consent record management as an infrastructure function, not an administrative one.

Separation of the Enforcement Layer

The QODIQA enforcement layer operates independently of the AI model it governs. The enforcement layer does not share code, state, or failure modes with the inference system. A failure in the inference system does not compromise the enforcement layer. A failure in the enforcement layer produces a fail-closed outcome: denial, not uncontrolled execution.

This separation provides the organizational assurance that consent enforcement cannot be inadvertently disabled by changes to the AI system itself.

Multi-Principal Environments

In enterprise AI deployments, multiple principals may hold concurrent consent relationships with the same AI system: individual users, organizational units, system administrators, and third-party integrators. QODIQA provides a framework for managing overlapping consent claims and resolving conflicts between them deterministically. The most restrictive applicable consent boundary governs execution; broader permissions held by one principal cannot override more restricted permissions held by another.

Temporal Validity Management

Consent records in the QODIQA framework carry explicit validity windows. An authorization valid at 09:00 may not be valid at 14:00, depending on how the consent record is structured. The enforcement layer evaluates temporal validity as part of every decision. This property requires that organizations actively manage consent record lifecycles, issuing new records when authorizations are renewed and revoking records when authorizations are withdrawn, rather than relying on implicit continuation of prior consent.

SECTION 5

Risk Containment Capabilities

From a risk management perspective, QODIQA provides three categories of capability: prevention, containment, and evidence.

Prevention

Deterministic enforcement prevents out-of-scope AI actions at the point of execution. An action that lacks a valid consent authorization does not execute; it is not logged as a violation and flagged for review. For organizations operating in regulated environments or processing sensitive data, this distinction is material. Prevention eliminates the class of harm associated with unauthorized AI action; it does not merely improve the speed with which such harm is detected.

Containment

When consent records are structured with appropriately scoped authorization boundaries, the enforcement layer contains AI system behavior within those boundaries by design. A system authorized to process medical records for diagnostic purposes cannot, in a QODIQA-compliant architecture, use the same consent authorization to perform billing analysis. The scope of each authorization is fixed in the consent record; the enforcement layer holds that scope deterministically. Scope creep, the gradual expansion of AI system behavior beyond its original authorization, becomes a consent record governance problem rather than an ongoing operational monitoring problem.

Evidence

The tamper-evident audit record produced by the enforcement layer constitutes verifiable evidence of consent evaluation for every AI action taken. This evidence supports incident investigation by providing a definitive record of what was authorized at the time of each action; it supports regulatory examination by demonstrating that consent evaluation occurred; and it supports litigation by providing a factual basis for claims about AI system behavior. Organizations cannot generate this category of evidence retroactively; it is produced only by systems that performed runtime enforcement at the time of action.

RISK CLASSIFICATION

- UNAUTHORIZED ACTION RISK.
  Eliminated by prevention at the enforcement layer.
- SCOPE CREEP RISK.
  Contained by fixed authorization boundaries in consent records.
- EVIDENTIARY RISK.
  Mitigated by cryptographically verifiable audit records.
- STALE CONSENT RISK.
  Addressed by temporal validity evaluation at runtime.

SECTION 6

Regulatory Alignment Positioning

Regulatory frameworks governing AI systems increasingly require demonstrable consent mechanisms, verifiable processing boundaries, and auditable records of AI decision-making. QODIQA is designed to provide the technical foundation for compliance with this class of requirement. It does not constitute legal or regulatory advice, and its applicability to specific regulatory obligations must be assessed in context.

Data Protection Frameworks

Data protection regulations impose requirements for specific, informed, and documented consent as a basis for processing personal data. QODIQA's machine-readable consent records are structurally aligned with these requirements: they specify the processing purpose, the data subject's authorization, and the temporal scope of that authorization in a form evaluable at runtime. The enforcement layer provides the technical mechanism by which processing is constrained to authorized purposes, a capability that policy controls alone cannot provide.

AI Governance Frameworks

Emerging AI governance frameworks impose obligations related to human oversight, risk classification, and technical accountability. QODIQA's deterministic enforcement model supports human oversight by ensuring that AI systems operate within boundaries explicitly defined by human principals. Its audit trail supports technical accountability by providing a verifiable record that those boundaries were evaluated and enforced for every action taken.

Sector-Specific Requirements

Regulated sectors, including healthcare, financial services, and legal services, impose consent obligations specific to their operational contexts. QODIQA's consent record schema is extensible to accommodate sector-specific authorization parameters. The enforcement framework is not domain-specific; the consent records it evaluates can be structured to reflect domain-specific requirements, allowing a single enforcement architecture to address obligations across multiple regulatory contexts.

Institutional Positioning

Organizations that deploy QODIQA-compliant AI systems are positioned to demonstrate, rather than assert, that their AI systems operate within authorized boundaries. The distinction between assertion and demonstration is increasingly significant in regulatory examinations: an organization that can produce a verifiable enforcement record for every AI action taken is in a materially different position from one that can produce only governance documentation describing its intent.

SECTION 7

Deployment Model

QODIQA is specified as an enforcement layer that is architecturally distinct from the AI systems it governs. The enforcement layer is not embedded in the AI model; it wraps the inference pipeline as an independent component with defined interfaces.

Integration Points

The QODIQA enforcement layer intercepts each AI action intent, the structured representation of what the AI system proposes to do, before execution. It resolves the applicable consent record, evaluates the proposed action against that record using the QODIQA policy evaluation function, and either permits execution or issues a denial response. The inference pipeline never receives an action intent that the enforcement layer has denied; the denial is final.

Consent Record Management

Consent records are created, stored, and managed outside both the AI system and the enforcement layer. They are provided to the enforcement layer as authoritative inputs at evaluation time. Organizations deploying QODIQA-compliant systems are responsible for the integrity and availability of their consent record stores. The QODIQA framework specifies the required schema and validation rules for consent records; it does not specify the storage infrastructure.

Operational Lifecycle

QODIQA-compliant deployments require operational processes for consent record issuance, renewal, scope modification, and revocation. Where consent is granted dynamically, for example in response to user interaction, the consent record creation process must be capable of generating valid records in real time, prior to the action for which authorization is sought.

Certification and Verification

The QODIQA corpus includes a Certification Framework for Deterministic Runtime Consent Enforcement, which specifies the conformance requirements that enforcement layer implementations must satisfy. Organizations deploying QODIQA-compliant systems should verify that their implementation satisfies the applicable certification requirements before relying on the enforcement layer for compliance purposes.

SECTION 8

Strategic Impact

The strategic significance of deterministic runtime consent enforcement extends beyond immediate risk management and regulatory compliance. It represents a foundational property for the class of AI deployments that will define institutional AI capability over the next decade.

As AI systems move from decision support toward operational agency, executing transactions, committing resources, and initiating communications on behalf of principals, the question of authorization becomes structurally equivalent to the question of identity in access control systems. An AI system that acts without verifiable authorization is not an authorized agent; it is an autonomous process whose actions cannot be attributed to any principal with legal or institutional confidence. Deterministic runtime consent enforcement is the mechanism by which AI agency is bounded to verified authorization.

Organizations that establish consent enforcement infrastructure now acquire a compound advantage. In the near term, they reduce their exposure to the categories of harm, including unauthorized action, scope creep, and evidentiary gaps, that currently characterize AI risk profiles. In the medium term, they establish the organizational capability to manage consent relationships at scale, enabling AI deployments that are both more capable and more controlled. In the long term,

they position themselves as institutions for which AI trustworthiness is a demonstrable property of their systems, not a claim about their intentions.

> **STRATEGIC PROPOSITION**
>
> The capacity to deploy AI systems with verifiable consent enforcement will become a differentiating institutional capability as AI governance requirements mature. Organizations that build this capacity early incur lower implementation costs and avoid the operational disruption of retroactive compliance.

The inverse is also true. Organizations that continue to rely on pre-execution consent mechanisms will face increasing difficulty demonstrating compliance with governance frameworks that require runtime verifiability. The evidentiary and liability exposure associated with non-enforcement grows as AI systems take more consequential actions in more regulated contexts.

SECTION 9

Limitations and Non-Coverage

QODIQA addresses a specific architectural gap: the absence of deterministic consent enforcement at the AI execution layer. It does not address the full scope of AI governance, and its adoption does not substitute for governance controls that operate at other layers of the AI system stack.

QODIQA does not govern the substantive content of consent. It enforces the boundaries that consent records define; it does not evaluate whether those boundaries are ethically appropriate, legally sufficient, or strategically advisable. The quality of consent enforcement is bounded by the quality of the consent records provided to the enforcement layer. Organizations that create poorly scoped, overly broad, or inadequately validated consent records will not benefit from enforcement of those records.

QODIQA does not address model alignment or output safety. A consent-authorized action may still produce harmful output if the underlying AI model is not appropriately aligned. Consent enforcement and output safety are complementary properties; neither substitutes for the other.

QODIQA does not address the validity of consent from a legal or ethical standpoint. A consent record that satisfies QODIQA's technical requirements may not satisfy the requirements of applicable law for valid consent. Organizations must ensure that their consent record governance processes produce authorizations that are legally valid in their operational jurisdictions.

QODIQA does not provide guarantees about the behavior of AI systems not integrated with a compliant enforcement layer. Its properties apply only to actions evaluated by the enforcement layer. Portions of an AI system's operational scope not routed through the enforcement layer are not subject to its protections.

> NON-COVERAGE SUMMARY
>
> - Substantive content of consent records
> - Legal validity of consent under applicable law
> - AI model alignment and output safety
> - Governance controls external to the enforcement layer
> - Actions not routed through a QODIQA-compliant enforcement layer

SECTION 10

Conclusion

Consent, as currently implemented in AI deployments, is a representation rather than an enforcement. It exists as a legal artifact, a policy document, or a governance control, each of which operates outside the execution layer and therefore

cannot provide deterministic guarantees at the moment an AI system acts. The structural consequence is a systematic gap between what principals authorize and what AI systems do.

QODIQA closes this gap by specifying a deterministic consent enforcement layer that operates within the AI execution pipeline. The enforcement layer evaluates every action against a verified consent record, applies fail-closed semantics in the absence of valid authorization, and produces a tamper-evident audit trail of every enforcement decision. The result is a system in which consent is not a pre-execution claim but a runtime invariant.

For institutional decision-makers, the implications are direct. Organizations that deploy QODIQA-compliant systems acquire the capacity to prevent unauthorized AI action, contain AI behavior within authorized boundaries, and demonstrate compliance through verifiable enforcement records. Organizations that do not face growing evidentiary and liability exposure as AI systems take increasingly consequential actions in increasingly regulated contexts.

Execution-layer consent enforcement is not a regulatory response to current requirements. It is the architectural precondition for AI deployments that are both capable and accountable. QODIQA provides the specification for that architecture.

This executive brief is a component of the QODIQA standard corpus. Readers seeking technical depth on the enforcement framework, system architecture, security profile, or certification requirements should consult the QODIQA Technical Whitepaper and associated corpus documents.

DOCUMENT STATUS

Document Status and Corpus Alignment

This document is an executive brief within the QODIQA specification corpus. It provides a strategic and institutional overview of the QODIQA framework for deterministic runtime consent enforcement in artificial intelligence systems, addressed to organizational decision-makers who require a concise and authoritative characterization of the framework's scope, rationale, and implications.

This brief is intended to be read in conjunction with the following corpus documents, which provide technical depth on the subjects introduced herein:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement
- QODIQA: Terminology and Normative Definitions
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Governance Charter for the QODIQA Standard Corpus

Version 1.0 represents the initial formal release of this executive brief as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dițescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-EB-2026-001
TITLE	Executive Brief for Deterministic Runtime Consent Enforcement
SUBTITLE	Strategic Overview for Institutional Decision-Makers
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Executive Brief
DOCUMENT STATUS	Normative
CORPUS ALIGNMENT	QODIQA Core v1.0 · 68-Point Framework v3.0 · Whitepaper v1.0
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 27 PART V · CONTEXTUAL, ANALYTICAL AND PROCESS SUPPORT

Economic Impact Analysis of Runtime Consent Enforcement

Structural economic and risk-surface analysis of deterministic runtime consent enforcement architectures, an analytical companion to the standard corpus.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-EIP-2026-001	1.0	2026	Analytical White Paper	Informative	INFORMATIVE

ABSTRACT

This paper presents a structured economic and risk-surface analysis of a specific class of AI infrastructure: deterministic runtime consent enforcement architectures, namely systems in which authorization state is evaluated and recorded as an immutable, machine-verifiable artifact at the moment of AI action execution, rather than inferred retroactively from logs. It does not constitute investment guidance, revenue projection, or regulatory legal advice.

The analysis examines how runtime determinism addresses identifiable cost surfaces across ambiguity-driven liability exposure, audit reconstruction overhead, incident containment latency, and multi-agent propagation risk. QODIQA is treated throughout as a reference implementation of this architectural class, not as the source of the model's validity. Modeling is conservative, scenario-based, and assumption-bound throughout. All figures represent estimated structural cost bands derived from first-principles reasoning about the operational mechanics of enforced and non-enforced governance environments, not actuarial measurements or empirical benchmarks. This paper is an independent technical

and economic analysis, prepared for institutional review by risk, compliance, legal, and infrastructure leadership audiences.

METHODOLOGICAL NOTE

Methodological Positioning and Limitations

REQUIRED READING: ANALYTICAL SCOPE AND CONSTRAINTS

This document is a first-principles analytical model. It reasons forward from the known structural properties of enforced and non-enforced AI governance architectures to estimate the cost surfaces those properties generate. It does not rely on empirical breach datasets, actuarial claims tables, or published industry benchmarks for AI governance incidents, because no such datasets exist in sufficient specificity or public accessibility for this domain at the time of writing.

Accordingly, this model makes no predictive claims. It does not forecast incident frequency, regulatory fine probability, or litigation likelihood. It does not purport to measure what will happen to any specific organization. Its purpose is structural cost analysis: to reason about the categories of cost that the presence or absence of deterministic enforcement architecture creates, and to establish order-of-magnitude bounds on those categories based on stated operational assumptions.

Every numeric figure in this paper is a structural estimate. The assumptions from which it is derived are stated explicitly wherever the figure appears. Numbers should be interpreted as directional cost bands within their stated assumption constraints, not as point estimates, actuarial projections, or empirically validated benchmarks. Actual costs will vary materially based on organizational scale, incident frequency, regulatory jurisdiction, and implementation fidelity.

No component of this model derives from cited external research, benchmark surveys, or industry studies. The model is self-consistent within its stated assumptions and is intended to be evaluated on the basis of its structural reasoning. External validation (through empirical deployment data, actuarial analysis, or academic study) would materially strengthen the model and is acknowledged as absent.

On the Absence of Empirical Data

The most foreseeable critique of this model is the absence of empirical data. That absence is real and acknowledged. It is also, at this stage of the field, structurally unavoidable. Empirical datasets for AI governance incident costs (specifically, for the labor and legal costs attributable to authorization ambiguity in AI-deployed systems) do not currently exist in any publicly accessible, domain-specific form. This is not because such incidents are rare; it is because the governance layer at which these costs arise is not yet routinely instrumented, categorized, or disclosed at the granularity required for actuarial modeling. Incident costs are absorbed into general legal, compliance, and engineering line items. Authorization failures, where they are detected at all, are rarely classified as such in disclosed reporting. The domain lacks the incident taxonomy, disclosure standards, and historical depth that actuarial modeling requires. Building empirical models for costs that are not yet measured is not methodologically feasible; waiting for such datasets before conducting structural analysis would mean deferring the analysis indefinitely, during a period when AI deployment is accelerating and governance infrastructure decisions are being made now. First-principles structural modeling is not a substitute for empirical validation; it is the appropriate analytical instrument for the present state of evidence, and it is offered as such. Structural reasoning from known architectural properties toward cost surface estimates is a defensible methodology precisely when empirical alternatives do not yet exist. The validity of the arguments in this paper therefore does not depend on empirical confirmation; it depends on whether the causal chains (ambiguity to investigation cost, absence of artifacts to audit overhead, propagation without gating to risk amplification) are structurally sound. Readers are invited to evaluate them on that basis.

What This Analysis Does Not Claim

- × It does not claim that any specific organization will experience the cost bands described
- × It does not claim that deterministic enforcement architectures have been empirically validated to produce the modeled reductions
- × It does not claim that QODIQA, as a reference implementation, has been deployed and measured at production scale

× It does not claim actuarial equivalence with insurance, financial, or legal modeling disciplines

× It does not constitute a regulatory compliance determination or legal position on any applicable framework

> The analytical credibility of this paper rests on the coherence of its structural reasoning, not on external validation it does not possess. Readers are encouraged to evaluate the causal chains (ambiguity toward investigation cost; absence of artifacts toward audit overhead; propagation without gating toward risk amplification) on their merits and to apply independent judgment about the plausibility of the stated cost bands within their organizational context.

GOVERNANCE CONTEXT

Relationship to Existing Governance Frameworks

Deterministic runtime consent enforcement architectures operate at the execution layer of AI governance, the point at which a specific AI action is permitted or denied based on evaluated authorization state. This position is distinct from, and complementary to, the policy and governance layers addressed by existing regulatory and standards frameworks. This section clarifies the relationship between execution-layer enforcement and those frameworks without claiming compliance mappings or regulatory equivalence.

General Data Protection Regulation (GDPR)

GDPR Article 7 establishes conditions for the validity of consent, and Article 5 imposes data minimization and purpose limitation principles. Execution-layer enforcement directly supports the operationalization of these principles: by evaluating consent validity and purpose declaration at the moment of AI action execution, it generates a timestamped, immutable record of whether the conditions for lawful processing were evaluated and what the outcome was. This does not constitute GDPR compliance (compliance determinations require qualified legal review of the full processing context)

but it does produce the evidentiary substrate that compliance demonstration requires. The cost implications of this distinction are addressed in Section 5.

EU AI Act

The EU AI Act imposes transparency, logging, and human oversight obligations on high-risk AI systems under Articles 12 and 13. Execution-layer enforcement, through its mandatory production of decision artifacts at runtime, is structurally aligned with the logging and auditability requirements of these provisions. However, alignment is not compliance: the EU AI Act's requirements extend to model documentation, conformity assessments, and risk management systems that are outside the scope of runtime consent enforcement. This model positions execution-layer enforcement as one component of a compliant AI governance architecture, not as a sufficient condition for it.

NIST AI Risk Management Framework (AI RMF)

The NIST AI RMF GOVERN function addresses organizational practices for managing AI risk, including documentation, accountability, and oversight mechanisms. The MEASURE function addresses evaluation of AI system performance against defined criteria. Execution-layer enforcement contributes to the GOVERN.1 (policies and procedures) and MANAGE.2 (incident response) functions by providing machine-verifiable evidence of authorization state at execution time. It does not address the full breadth of the AI RMF's risk management scope, which includes model accuracy, societal impact, and organizational governance structures.

ISO/IEC 42001

ISO/IEC 42001 establishes a management system standard for AI, requiring organizations to demonstrate systematic approaches to AI risk identification, treatment, and monitoring. Execution-layer enforcement is relevant to Clause 8 (Operation) and Clause 9 (Performance Evaluation) in the degree to which it provides objective evidence of control

operation. As with other frameworks, execution-layer conformance is a component of, not a substitute for, a complete management system implementation.

POSITIONING SUMMARY

Execution-layer enforcement architectures address a specific gap: the absence of machine-verifiable authorization state at the moment of AI action. They do not replace policy-layer governance, ethical design practices, model risk management, or organizational accountability structures. They operate below and in support of those layers, providing the evidentiary infrastructure that higher-layer governance requires. The economic analysis in this paper is scoped to this execution layer; it does not model the full cost of GDPR, EU AI Act, NIST AI RMF, or ISO/IEC 42001 compliance.

KEY ECONOMIC FINDINGS

- Deterministic runtime enforcement is expected to reduce ambiguity-driven investigation cost, under conditions of complete enforcement coverage, by making authorization state queryable rather than requiring forensic reconstruction
- Audit reconstruction overhead is expected to compress from a modeled 40–180 analyst-hours (log-based forensics) to a modeled 2–12 hours (artifact retrieval and verification) per material incident, under full-coverage assumptions; both bands are structural estimates, not empirical measurements
- Incident containment time is expected to decrease from day-to-week bands toward hour-to-day bands due to deterministic authorization evidence at each execution point, subject to artifact store integrity and complete enforcement deployment
- Multi-agent propagation risk is expected to become bounded through deterministic gating at agent boundaries, constraining upstream failures rather than allowing cascade multiplication, under conditions of complete gate deployment across all agent interfaces
- Governance cost structure is expected to shift from variable, incident-driven investigation labor toward fixed, predictable infrastructure operational cost under enforced architectures
- Annual operational cost of conformant enforcement infrastructure represents approximately 7–35% of modeled mid-band annual baseline direct cost exposure, under full-conformance and mid-band incident frequency assumptions, a structural cost-ratio observation, not a validated return-on-investment figure

VISUAL ANALYSIS

Governance Architecture Comparison

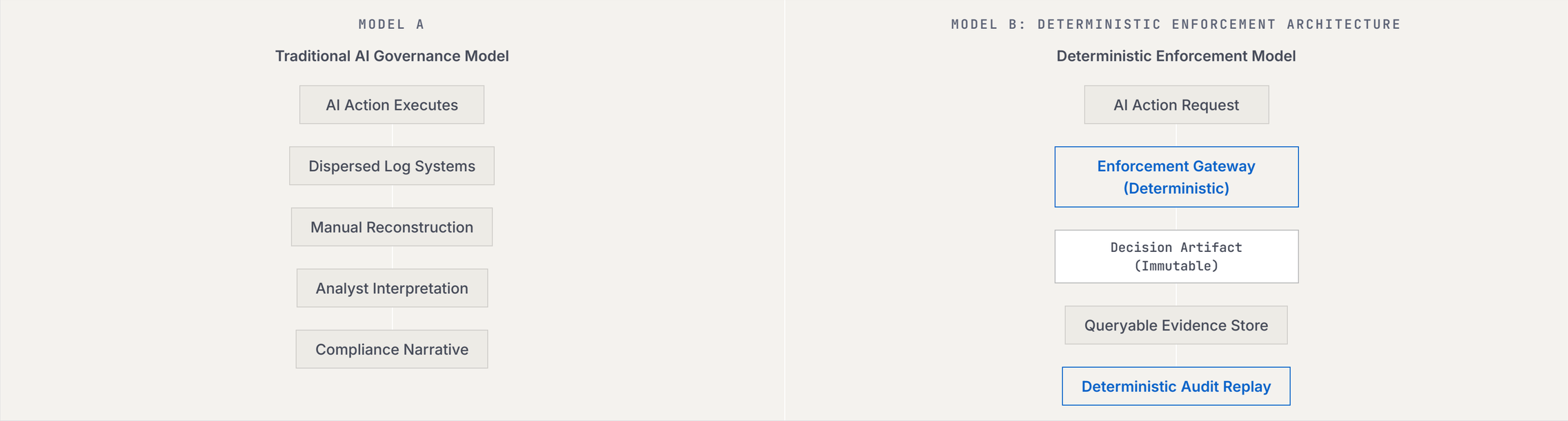


FIGURE 1: MODEL A REQUIRES FORENSIC RECONSTRUCTION AT INQUIRY TIME; MODEL B PRODUCES MACHINE-VERIFIABLE ARTIFACTS AT EXECUTION TIME, COMPRESSING AUDIT OVERHEAD AND ELIMINATING AUTHORIZATION AMBIGUITY.

SECTION 1

Executive Economic Overview

The economics of AI governance are rarely analyzed with structural precision. Most discussions remain at the policy level, addressing compliance obligations, ethical requirements, or reputational considerations in qualitative terms. This paper takes a different approach: it attempts to quantify the economic implications of a specific infrastructure gap in AI-deployed organizational environments: the absence of deterministic runtime consent enforcement: using conservative modeling, clearly stated assumptions, and scenario-bounded analysis.

The core economic premise is straightforward. **Ambiguity is expensive.** When an organization cannot deterministically establish whether a given AI action was authorized (for the purpose declared, using the data involved, at the time it occurred) every downstream consequence becomes harder to contain, defend, and close. Audit reconstruction takes longer. Legal review is more involved. Incident response lacks precision. Regulatory inquiry requires more documentation.

These costs are not hypothetical; they are the structural consequence of enforcement architectures that are unable to produce machine-verifiable decision artifacts at execution time.

> Deterministic runtime enforcement architectures address this gap by converting runtime authorization from a probabilistic inference into a deterministic artifact. The economic effect of that conversion is the subject of this analysis: **not the elimination of risk, but the reduction of its most expensive structural properties.** QODIQA is examined as a reference implementation of this architectural class throughout this paper.

1.1 Scope of Analysis

This analysis models cost surfaces across the following domains: unauthorized AI action exposure, purpose drift liability, revocation non-propagation, audit reconstruction cost, regulatory penalty exposure, incident response amplification, and multi-agent propagation risk. It does not project revenue, market sizing, competitive positioning, or investment return. It does not constitute legal advice with respect to regulatory penalties or litigation exposure.

1.2 Modeling Constraints

All estimates in this paper are band-based, reflecting low, mid, and high exposure scenarios across defined organizational profiles. They are not point estimates and must not be interpreted as guaranteed outcomes or empirically validated benchmarks. The modeling is structural in nature: it reasons about cost dynamics from first principles, drawing on the mechanistic properties of deterministic enforcement architectures and non-enforced governance environments. No component of this model is derived from actuarial claims data, published breach cost surveys, or empirical incident measurements. The assumptions underlying each figure are stated explicitly, and the figures should be interpreted as directional cost bands within those assumption constraints, not as predictions of organizational cost experience.

1.3 Relationship to the QODIQA Reference Architecture

This paper uses enforcement architectures conformant with the QODIQA Core Standard v1.0 and the 68-Point Enforcement Framework v3.0 as its reference implementation of the broader class of deterministic runtime consent enforcement systems. The economic properties modeled here (immutable decision artifacts, deterministic policy evaluation, fail-closed semantics, cryptographic integrity) are properties of the architectural class, not exclusive properties of QODIQA. QODIQA is referenced as a concrete specification of this class, providing definitional precision to the architectural properties under analysis.

References to enforcement properties assume full conformance with the QODIQA specifications where QODIQA is cited. Partial deployments, non-conformant implementations, or alternative architectures that approximate but do not fully satisfy these properties may produce narrower or different cost profiles than those modeled here. The economic claims in this paper are contingent on the full set of properties being present; they should not be interpreted as applicable to partial or approximate implementations.

SECTION 2

Risk Surface Categories

The following categories define the primary economic risk surfaces created by operating AI systems without deterministic runtime consent enforcement. Each represents a distinct class of cost exposure that enforcement infrastructure is designed to address.

2.1 Unauthorized Action Exposure

When AI systems execute actions without deterministically confirmed consent, the organization faces potential liability for the consequences of those actions. The economic exposure is not limited to the immediate harm caused; it includes the cost of establishing after the fact what the authorization state was, whether consent existed, and what obligations were triggered. Without a deterministic enforcement record, this determination requires forensic reconstruction of system state from logs, policy snapshots, and execution traces, a process that is both expensive and imprecise.

2.2 Purpose Drift Liability

AI systems deployed for declared purposes may, over time, be applied to adjacent purposes for which consent was not obtained. Purpose drift is economically costly because it is often invisible in real-time; the drift accumulates through incremental decisions, each individually arguable, until an aggregate pattern emerges that creates material liability.

Without deterministic enforcement that records and validates the purpose declaration at execution time, establishing the drift pattern and its associated liability requires exhaustive forensic analysis.

2.3 Revocation Non-Propagation

Consent revocation in distributed AI systems requires that the revocation signal propagate to all components that might act on the revoked consent. Without deterministic enforcement, this propagation is neither guaranteed nor verifiable, and in architectures where components cache consent state or evaluate it asynchronously, the revocation gap may persist for operationally significant durations before detection. The economic consequence is continued AI action on revoked consent, creating liability exposure that may not be detected until audit or incident. The cost of retroactive remediation, which must account for every action taken after revocation, significantly exceeds the cost of deterministic enforcement at the point of execution, where the revocation state would be evaluated before any action proceeds.

2.4 Audit Reconstruction Overhead

Regulatory audits, internal compliance reviews, and litigation discovery in AI-deployed organizations currently require labor-intensive reconstruction of authorization states that were never recorded in queryable form. The cost of this reconstruction is paid on every audit cycle and in every incident investigation. It is not a tail-risk cost; it is an operational cost that recurs predictably and scales with AI deployment volume.

2.5 Regulatory Penalty Exposure

GDPR, the EU AI Act, and equivalent regulatory frameworks impose penalties for consent violations, inadequate transparency, and insufficient control mechanisms. The economic exposure from regulatory penalty is not primarily determined by whether violations occurred; it is substantially determined by whether the organization can demonstrate that appropriate controls existed and functioned. Without enforcement infrastructure, this demonstration requires arguing

from process documentation and policy attestation; with enforcement infrastructure, it is supported by machine-verifiable artifact evidence.

2.6 Incident Response Amplification

When an AI-related incident occurs in a non-enforced environment, the response effort is amplified by the inability to quickly establish the authorization state at the time of the incident. Incident responders must expand the investigation boundary to include all actions that might be related, because the system cannot deterministically exclude actions that were actually authorized. This over-investigation is the primary source of incident response cost amplification in AI-deployed organizations.

2.7 Multi-Agent Propagation Risk

In architectures where AI agents call other AI agents, consent violations do not remain isolated. An agent acting beyond its authorized scope may trigger downstream actions by other agents, each of which inherits the upstream consent failure. The economic consequence is a multiplier effect: the cost of a single upstream authorization failure scales with the number of downstream actions it initiated. Without deterministic enforcement at each gate, the propagation scope cannot be established without exhaustive trace analysis.

SECTION 3

Baseline Economic Exposure Model

The following model establishes a baseline cost envelope for organizations operating AI systems without deterministic runtime consent enforcement. It uses assumed operational parameters across three organizational profiles and three incident frequency bands. All figures are structural estimates, not actuarial measurements.

3.1 Modeling Assumptions

STATED ASSUMPTIONS: REQUIRED READING

AI usage volume: 500,000 to 50,000,000 AI-executed actions per year, varying by organizational profile.

Consent ambiguity rate: Estimated 2–8% of AI actions where consent validity cannot be deterministically confirmed at time of execution, based on the structural properties of non-enforced systems.

Incident frequency: Low (1–3 material consent-related incidents per year), Medium (4–12), High (13+).

Audit overhead estimate: 40–180 analyst-hours per material incident for log-based reconstruction in non-enforced environments.

Legal review cost: $15,000–$250,000 per material incident requiring external legal counsel, depending on regulatory context and data sensitivity.

3.2 Baseline Cost Band Table

TABLE 3.1: ANNUAL BASELINE EXPOSURE BY INCIDENT BAND (WITHOUT DETERMINISTIC ENFORCEMENT)

COST CATEGORY	LOW BAND	MID BAND	HIGH BAND
Audit Reconstruction (Internal Labor)	$40k – $120k	$180k – $540k	$540k – $1.8M
Incident Response Scoping	$20k – $80k	$90k – $360k	$300k – $1.2M
Legal Review and Counsel	$15k – $75k	$75k – $500k	$250k – $2.5M
Compliance Documentation	$30k – $90k	$90k – $270k	$270k – $900k
Regulatory Inquiry Response	$0 – $150k	$50k – $750k	$200k – $5M+
Total Annual Exposure Estimate	$105k – $515k	$485k – $2.4M	$1.56M – $11.4M+

These bands represent the direct operational cost of operating without enforcement infrastructure, exclusive of any regulatory fines, class-action exposure, or reputational remediation. They reflect the cost of institutional effort applied to

resolving ambiguity that deterministic enforcement would have removed at the point of execution.

3.3 Structural Cost Driver

The dominant cost driver across all bands is not the frequency of actual violations; it is the **inability to distinguish authorized from unauthorized actions without significant forensic effort**. In a system without deterministic enforcement artifacts, every investigated incident must treat all ambiguous actions as potentially unresolved, regardless of whether they were actually problematic. This means that even a small number of genuine violations can generate investigation effort that spans a much larger population of ambiguous but legitimate actions, a structural over-investigation that is the principal source of audit cost inflation in AI-deployed organizations.

The cost bands in this model reflect direct operational costs only. They exclude tail-risk events such as class-action proceedings, maximum-scale regulatory fines, or systemic reputational remediation. The absence of tail-risk figures in this model does not imply the absence of tail-risk exposure; it reflects a deliberate methodological boundary intended to keep the model within the scope of structural reasoning rather than speculative estimation. Organizations should assess tail-risk exposure separately through qualified legal and actuarial review.

3.4 Modeling Methodology Clarification

The cost bands presented in this section are derived from structural reasoning about the operational mechanics of non-enforced AI governance environments, not from actuarial datasets, breach insurance claims, or empirical incident measurements. The methodology is explicitly first-principles in character: it reasons forward from the known properties of log-based reconstruction, cross-system correlation, and legal review cycles to estimate the labor and external cost surfaces those processes generate.

Analyst-hour bands for audit reconstruction are derived from the composition of work required in log-based governance environments: cross-system log correlation to establish event sequences, schema normalization across heterogeneous

storage systems, timestamp reconciliation where execution events and consent records are logged at different granularities, and interpretive narrative construction for auditor or legal audiences. Each of these phases contributes independently to the total reconstruction burden.

No component of this model derives from cited external research, benchmark surveys, or industry studies. The model is self-consistent within its stated assumptions and is intended to be evaluated on the basis of its structural reasoning, not its conformance to external data sources that do not yet exist for this specific domain.

SECTION 4

Economic Impact of Deterministic Runtime Enforcement

Deterministic runtime enforcement does not eliminate risk. It does, however, structurally alter several of the cost dynamics described in the baseline model by removing the condition (authorization ambiguity) from which the largest cost categories derive. The following analysis addresses each affected cost category and traces the reduction mechanism to its structural cause, rather than asserting a reduction percentage without causal grounding.

4.1 Reduction in Ambiguity Cost

When every executed AI action produces an immutable decision artifact (containing the resolved consent state, the evaluated policy version, the declared purpose, and the outcome), the set of actions where authorization state is indeterminate at query time becomes structurally empty. As specified in the 68-Point Enforcement Framework v3.0, conformant enforcement points are required to evaluate consent state and policy version at the moment of execution and to record the deterministic outcome as an immutable artifact before permitting downstream action. This structural requirement, if fully satisfied, is expected to eliminate the authorization ambiguity surface under conditions of complete and uninterrupted enforcement coverage.

Under this architecture, organizations are positioned (to the extent that artifact integrity is preserved and enforcement coverage is complete) to establish at query time whether a given action was authorized, for what purpose, and under which policy, without forensic reconstruction. This is expected to reduce, under defined enforcement conditions, the class of investigation that currently consumes the largest share of audit labor: the determination of what the authorization state was at time of execution.

4.2 Reduction in Audit Reconstruction Time

Current audit processes for AI-related incidents rely on log aggregation, cross-system correlation, and analyst interpretation. In enforced environments, a queried decision artifact is expected to provide directly: the request context hash, the consent state at evaluation, the applicable policy identifier and version, and the deterministic outcome, assuming the artifact store is intact, queryable, and the enforcement architecture has been operating without interruption. Under these conditions, the reconstruction step is expected to be replaced by artifact retrieval. The 40–180 hour analyst-time band for log-based reconstruction represents an estimate derived from structural decomposition of the investigation workflow phases described in Section 3.4, not an empirical measurement. The 2–12 hour band for artifact retrieval and verification is similarly derived from the structural simplification that querying a keyed artifact store represents relative to multi-system log correlation. Both figures should be interpreted as directional estimates within their stated assumption constraints, not as benchmarks applicable to any specific organization or deployment.

4.3 Reduction in Incident Containment Time

Mean time to scope determination (the interval between incident detection and a defensible boundary on what was affected) is a primary driver of incident response cost. In non-enforced systems, scope is constructed by expanding the investigation boundary until the team can argue with reasonable confidence that no additional material actions fall outside it, a process that tends to run longer than anticipated because each expanded boundary reveals additional ambiguous actions that must themselves be evaluated. In enforced systems operating under full artifact coverage, scope

becomes determinable by query: all actions where the decision artifact indicates a specific consent scope, purpose, or data classification within a defined time window. Where deterministic evaluation has been consistently applied across all execution points, this query-based scoping approach is expected to compress containment timelines from day-to-week bands toward hour-to-day bands. Organizations with partial enforcement coverage should expect proportionally reduced compression of this timeline, as ungated execution paths restore the reconstruction overhead that full coverage would otherwise remove.

4.4 Cost Comparison Table

TABLE 4.1: MODELED COST REDUCTION BY CATEGORY (MID-BAND SCENARIO, FULL-CONFORMANCE ASSUMPTION)

COST CATEGORY	BASELINE EXPOSURE (MODELED)	WITH DETERMINISTIC ENFORCEMENT (MODELED)	REDUCTION MECHANISM
Audit Reconstruction	$180k – $540k	$20k – $80k	*Reconstruction overhead compresses to artifact retrieval under full coverage*
Incident Response Scoping	$90k – $360k	$25k – $90k	*Scope determined by artifact query rather than expanding investigation*
Legal Review Overhead	$75k – $500k	$30k – $150k	*Evidentiary basis is artifact-supplied; reduces interpretive legal work*
Compliance Documentation	$90k – $270k	$20k – $60k	*Documentation generated at execution time, not assembled after the fact*
Regulatory Inquiry Response	$50k – $750k	$20k – $200k	*Artifact-based response reduces reconstruction and narrative burden*

All figures in Table 4.1 are structural estimates derived from first-principles reasoning about enforcement mechanics. They represent modeled cost bands under full-conformance conditions and mid-band incident frequency, as defined in Section 3.1. They are not derived from empirical deployment data, actuarial measurements, or industry benchmarks. The "With Deterministic Enforcement" column represents expected costs under the assumption that the enforcement

architecture is fully conformant, has complete coverage of all AI action execution points, and maintains uninterrupted artifact integrity. Partial or non-conformant implementations will produce narrower reductions than those indicated.

> The cost reduction mechanism is structurally consistent across all categories: **deterministic artifacts compress the investigation burden** by replacing the need to infer authorization state with the ability to retrieve it. The economic benefit is not a function of deterministic enforcement reducing the frequency of violations; it is a function of making the authorization record queryable rather than requiring forensic reconstruction. This distinction matters: the model does not depend on enforcement preventing more incidents; it depends on enforcement making existing incidents cheaper to investigate and close.

4.5 Defensibility Posture

Beyond direct cost reduction, deterministic enforcement is expected to alter the organization's posture in regulatory and legal proceedings under conditions where the artifact corpus is intact and legally admissible. Where a non-enforced organization must reconstruct and argue what its authorization state was (relying on narrative construction from logs, policy snapshots, and analyst interpretation), an organization with complete artifact coverage is expected to be able to present machine-verifiable authorization records directly. This shift from narrative explanation to artifact-based demonstration is structurally significant in proceedings where the burden of proof is on the organization. The degree to which this posture improvement translates into reduced proceeding length or cost will depend on the legal jurisdiction, the regulatory framework, and the specific conduct under review; no generalizable litigation cost reduction is claimed.

SECTION 5

Audit and Compliance Cost Model

Audit preparation and compliance documentation represent a persistent operational overhead in AI-deployed regulated organizations. This section models the cost structure of audit activity in both non-enforced and artifact-based enforcement environments.

5.1 Audit Cost Components

TABLE 5.1: AUDIT COST COMPONENT COMPARISON

COMPONENT	WITHOUT ENFORCEMENT	WITH ARTIFACT-BASED REPLAY
Audit Preparation	60–200 hrs/audit cycle	10–40 hrs/audit cycle
Log Reconstruction	High: cross-system correlation	Minimal: artifact retrieval
Evidence Packaging	Manual curation, interpretive	Structured export, verifiable
Internal Compliance Review	8–24 hrs per incident	1–4 hrs per incident
External Auditor Query Response	Days to weeks per query	Hours per query
Audit Trail Defensibility	Inferred, arguable	Deterministic, replayable

5.2 The Artifact-Based Replay Model

The central compliance cost innovation introduced by deterministic enforcement is the artifact-based replay model. Rather than asking "what happened and why?" (which requires forensic reconstruction, typically involving multiple log systems with inconsistent granularity and schema), an auditor operating against a complete artifact corpus can request retrieval of a specific decision artifact and receive the exact inputs, policy version, consent state, and outcome that

existed at execution time. Conditional on complete artifact coverage and the operational availability of replay infrastructure, this model is analytically expected to transform audit from a narrative construction exercise into a verification exercise, compressing the time and interpretive labor required by the highest-cost audit activities. Incomplete artifact coverage, arising from enforcement gaps, system interruptions, or partial deployment, would reduce the proportion of audit work amenable to this compression, restoring reconstruction overhead for the affected execution period.

5.3 Regulatory Framework Alignment Cost

Demonstrating compliance with GDPR Article 7 (consent conditions), EU AI Act Article 13 (transparency requirements), and NIST AI RMF GOVERN.1 (governance documentation) typically requires documentation of decision processes, data handling justifications, and consent records. In non-enforced environments, this documentation is assembled after the fact, retrospectively constructed from logs, policy records, and process attestations, each of which introduces interpretive uncertainty. In artifact-enforced environments, the compliance evidence substrate is produced as a structural byproduct of the enforcement mechanism at execution time, specifically because the artifact contains the consent state, policy version, and purpose declaration that compliance documentation would otherwise need to reconstruct. The marginal cost of compliance documentation is analytically expected to decrease materially under this architecture, within the stated modeling assumptions. Whether the artifact corpus meets the evidentiary standards required by the applicable regulatory framework is a legal determination, not a technical one, and requires qualified legal review in the relevant jurisdiction.

SECTION 6

Incident Response Containment Modeling

Incident response in AI-deployed environments presents distinct challenges from conventional cybersecurity incident response. The question is not only "what system was breached" but "what was authorized, what was not, and what was the consent state at each relevant execution point." Deterministic enforcement alters the cost structure of this inquiry fundamentally.

6.1 Mean Time to Attribution

In non-enforced environments, attributing an AI action to a specific authorization context requires correlation across application logs, consent storage systems, policy records, and execution traces. This correlation is often imprecise: consent records may be timestamped at a coarser granularity than execution events, or stored in separate systems with inconsistent schemas.

Estimated mean time to attribution in non-enforced environments ranges from 6 hours to 14 days depending on system architecture and logging completeness, a range that reflects both the variability of log infrastructure maturity and the interpretive labor required when logs were designed for operational monitoring rather than forensic attribution. In artifact-enforced environments, where deterministic evaluation has been consistently applied, attribution reduces to artifact identifier lookup, a process measurable in minutes to hours rather than days.

6.2 Mean Time to Scope Determination

Scope determination (establishing which actions, data subjects, and systems were involved in an incident) is the most expensive phase of AI incident response. In non-enforced environments, scope is constructed by expanding the investigation boundary until the team can argue with reasonable confidence that no additional material actions fall within the incident scope.

In artifact-enforced environments operating under complete coverage, scope is expected to be addressable by query: all decision artifacts matching specified consent scope, data classification, or agent identity within a defined time window. This is expected to reduce scope determination from a multi-day analytical effort to a structured query operation, under the conditions that enforcement has been continuously applied across all relevant execution points, the artifact store is intact and queryable, and the query infrastructure is available during the incident response period. Gaps in any of these conditions will partially restore the reconstruction overhead that enforcement otherwise removes.

TABLE 6.1: MODELED INCIDENT RESPONSE TIMELINE COMPARISON (FULL-COVERAGE ASSUMPTION)

METRIC	WITHOUT ENFORCEMENT (MODELED)	WITH DETERMINISTIC ENFORCEMENT (MODELED, FULL COVERAGE)
Mean Time to Attribution	6 hrs – 14 days	15 min – 4 hrs
Mean Time to Scope Determination	2 days – 6 weeks	2 hrs – 2 days
Log Integrity Validation	Uncertain: mutable logs	Cryptographic verification available where signing is implemented
Cross-System Trace Effort	High: manual correlation	Reduced: artifact-anchored tracing where coverage is complete
Regulatory Notification Confidence	Low: scope uncertain	Higher: scope queryable under complete artifact coverage

All timeline bands in Table 6.1 are structural estimates. The "Without Enforcement" column is derived from the decomposition of investigation workflow phases described in Section 3.4. The "With Deterministic Enforcement" column assumes full and continuous enforcement coverage across all relevant AI execution points. Neither column represents empirically measured incident response data.

6.3 Forensic Ambiguity Cost

Forensic ambiguity (the condition where the authorization state at execution time cannot be established with certainty at investigation time) is the primary structural cost amplifier in AI incident response. It drives over-notification (informing more data subjects than necessary because scope cannot be bounded), over-containment (suspending more systems

than affected because the boundary of impact cannot be determined), and extended legal review (because counsel cannot advise with confidence on exposure scope). Where deterministic evaluation has been consistently applied, enforcement architecture addresses the structural cause of forensic ambiguity by recording authorization state at execution time rather than leaving it to be inferred at investigation time. The degree to which forensic ambiguity is reduced in practice depends on the completeness of enforcement deployment, the integrity of the artifact corpus, and the operational availability of artifact query infrastructure during response activities, conditions that are matters of implementation quality rather than architectural design alone.

SECTION 7

Multi-Agent System Risk Amplification Economics

As AI deployments mature, tool-calling and multi-agent architectures are becoming standard rather than exceptional. In these environments, an AI agent may invoke other agents, consume outputs from automated pipelines, and chain tool calls across multiple systems. The economic consequence of this architecture is a risk amplification effect: a single upstream authorization failure propagates through the system, generating a cascade of actions each of which may independently constitute a consent violation.

7.1 The Propagation Multiplier

Consider an orchestrating agent authorized for purpose A that inadvertently invokes a downstream agent with a request parameterized for purpose B. Without enforcement at the downstream gate, purpose B actions execute under the upstream agent's authorization context, a context that does not cover purpose B. The exposure is not one unauthorized action but a sequence of them.

If the downstream agent itself calls further tools, the propagation continues. The propagation multiplier (the number of derivative actions generated by a single upstream authorization failure) can reach an order of magnitude or more in

complex agentic workflows.

> The economic significance of the propagation multiplier is that **incident scope, remediation cost, and notification obligations scale with it**. An incident that would be contained to a single action in a simple pipeline becomes a multi-action consent event in a multi-agent architecture, with corresponding increases in audit scope, legal review, and potential regulatory exposure.

7.2 Deterministic Gating as Cost Containment

A deterministic enforcement gate at each agent boundary is structured to constrain the propagation effect, assuming no degradation of enforcement guarantees across agent interfaces. If every agent invocation is subject to independent consent and policy evaluation at its own enforcement gate, an upstream authorization failure is analytically expected to be blocked at the first downstream gate it encounters rather than propagating through the system. The economic benefit is not merely a reduction in the likelihood of additional violations; it is the expected bounding of incident scope to a queryable set of artifacts, rather than an unbounded cascade requiring exhaustive trace reconstruction. This benefit is contingent on enforcement gates being present at every agent boundary; architectures with selectively deployed gates will contain propagation only at gated boundaries, leaving ungated paths as residual propagation channels that restore the unbounded cost structure at those interfaces.

7.3 Risk Surface Growth Function

In a system with N agent nodes and an average fan-out of F (downstream calls per node), an upstream authorization failure in a non-enforced system can affect up to F^(N-1) actions before reaching leaf nodes. This is a structural property of the architecture, not a worst-case anomaly. With deterministic gating at each node, the affected action set is bounded at 1 (the originating action) plus a deterministic query identifying which downstream gates rejected. The cost

curve is therefore exponential in non-enforced architectures and constant-bounded in enforced ones, relative to the propagation structure of the system.

SECTION 8

Implementation Cost Envelope

A balanced analysis requires accounting for the costs introduced by deterministic enforcement infrastructure alongside the costs it mitigates. The following estimates reflect the deployment and operational overhead of a conformant reference implementation of this architectural class. These are structural estimates derived from the known components of enforcement architecture deployment; actual costs will depend on organizational scale, existing infrastructure, and implementation approach and may vary materially from these bands.

TABLE 8.1: IMPLEMENTATION AND OPERATIONAL COST BANDS

COST COMPONENT	INITIAL DEPLOYMENT	ANNUAL OPERATIONAL
Enforcement Gateway Deployment	$40k – $200k	$10k – $50k
Policy Registry Configuration	$15k – $60k	$8k – $30k
Cryptographic Infrastructure	$5k – $25k	$3k – $12k
Integration Engineering	$30k – $180k	$10k – $40k
Monitoring and Alerting	$8k – $30k	$5k – $20k
Training and Operational Readiness	$10k – $40k	$5k – $15k
Total Implementation Band	**$108k – $535k**	**$41k – $167k**

8.1 Cryptographic Overhead

Deterministic enforcement introduces cryptographic signing of decision artifacts, which imposes computational overhead at the enforcement gateway. In well-architected implementations, this overhead is expected to be sub-millisecond per decision for standard hardware signing operations, an estimate derived from the operational characteristics of common asymmetric signing algorithms at the key lengths appropriate for artifact integrity, not from measured deployment data. For high-throughput environments (millions of decisions per day), dedicated cryptographic hardware may be warranted, representing an additional infrastructure cost in the $10k–$60k range for initial procurement. This range is a structural estimate based on current hardware security module procurement patterns and should be verified against current market pricing before use in capital planning.

8.2 Cost Introduced vs. Cost Expected to Be Mitigated

Comparing the modeled implementation cost envelope to the baseline exposure model: for mid-band incident frequency as defined in Section 3.1, the baseline annual direct operational exposure is modeled at $485k to $2.4M. The annual operational cost of maintaining a conformant enforcement infrastructure is modeled at $41k–$167k. Within the stated modeling assumptions, the operational cost of enforcement infrastructure represents approximately 7–35% of the annual baseline direct cost exposure it is designed to address, at mid-band incident frequency, before accounting for any incident reduction effect. This ratio is presented as a cost-structure comparison (specifically, a comparison between the fixed cost of maintaining the infrastructure and the variable cost of operating without it), not a return-on-investment claim. Both figures are structural estimates with the assumption dependencies described in Sections 3.1 and 8; neither has been empirically validated against deployment experience.

> The implementation cost envelope does not decrease as incident frequency increases; it is relatively fixed. The baseline exposure, however, scales directly with incident frequency. This asymmetry means that the cost differential between enforced and non-enforced environments **widens as incident frequency rises.**

SECTION 9

Scenario Analysis

The following three scenarios apply the cost model to defined organizational profiles. Each scenario presents a baseline risk envelope, an estimated post-enforcement adjustment, and structural observations. No return-on-investment projections or valuation claims are made.

As noted in Section 3, all scenario cost figures represent direct operational costs. They do not model catastrophic tail-risk events, including class-action proceedings, maximum-band regulatory penalties under GDPR or EU AI Act, or systemic remediation following high-profile incidents. The scenarios below should be read as direct-cost envelopes; organizations must assess tail-risk exposure through separate actuarial and legal processes.

9.1 Scenario A: Mid-Size Enterprise

PROFILE	BASELINE RISK ENVELOPE	POST-ENFORCEMENT ESTIMATE
Mid-Size Enterprise	**Without Enforcement**	**With Enforcement**
SCALE 250–2,500 employees; 1–5 AI-deployed services	ANNUAL $105k – $515k direct cost	ANNUAL $30k – $130k direct cost
VOLUME 500k – 5M AI actions/year	INCIDENTS Low-to-medium band (1–8/yr)	IMPL. COST $108k – $535k initial
REGULATION GDPR, sector-specific	AUDIT 60–180 hrs per cycle	OPERATIONAL $41k – $100k/yr

Structural Observation: At this scale, enforcement infrastructure represents a meaningful proportion of the first-year total cost. The economic case strengthens from year two onward, when implementation costs are amortized and the operational cost-to-exposure differential becomes the dominant factor. Regulatory audit efficiency is the primary near-term benefit at this profile.

9.2 Scenario B: Large Regulated Institution

PROFILE	BASELINE RISK ENVELOPE	POST-ENFORCEMENT ESTIMATE
Large Regulated Institution	**Without Enforcement**	**With Enforcement**
SCALE 5,000+ employees; 10–50 AI-deployed services	ANNUAL $1.5M – $11.4M+ direct cost	ANNUAL $300k – $2.5M direct cost
VOLUME 10M – 200M AI actions/year	INCIDENTS Medium-to-high band (8–20+/yr)	IMPL. COST $250k – $1.2M initial (multi-service)
REGULATION GDPR, EU AI Act, PSD2, DORA, or equivalent	AUDIT 180–600 hrs per cycle across services	OPERATIONAL $100k – $350k/yr

Structural Observation: At regulated institution scale, enforcement infrastructure cost is a significantly smaller proportion of baseline exposure. The primary benefit at this profile is regulatory defensibility: the ability to respond to supervisory inquiries and regulatory examinations with artifact-based evidence rather than reconstructed narratives. Multi-service

deployments benefit from shared policy registry and artifact storage infrastructure, reducing per-service implementation cost at scale.

9.3 Scenario C: AI-Native Platform at Scale

PROFILE	BASELINE RISK ENVELOPE	POST-ENFORCEMENT ESTIMATE
AI-Native Platform	**Without Enforcement**	**With Enforcement**
SCALE Millions of end users; AI as core product function	ANNUAL $5M – $50M+ direct cost band	ANNUAL $1M – $10M direct cost band
VOLUME 100M – 10B+ AI actions/year	INCIDENTS High band; regulatory inquiry near-certain	IMPL. COST $500k – $3M initial (platform-grade)
REGULATION GDPR, EU AI Act high-risk obligations, CCPA/CPRA	MULTI-AGENT Propagation multiplier structurally elevated	OPERATIONAL $200k – $800k/yr

Structural Observation: At AI-native platform scale, enforcement infrastructure becomes a product compliance requirement rather than an optional risk mitigation layer. The EU AI Act's high-risk system classification requirements, the operational scale of consent management, and the structural propagation risk in multi-agent architectures combine to make non-enforced operation structurally untenable above certain incident thresholds.

Infrastructure Density Note: At 100M–10B+ AI actions per year, artifact storage and cryptographic verification introduce scaling considerations distinct from lower-volume profiles. Cryptographic verification at scale benefits from shared registry infrastructure: because policy registry lookups and signing operations are performed against a single authoritative registry shared across all platform services, enforcement scaling cost increases sub-linearly relative to action volume. The marginal infrastructure cost per additional service or agent node is substantially lower than the initial deployment cost, as shared cryptographic infrastructure, policy registries, and artifact storage layers are amortized across the platform.

SECTION 10

Residual Risk and Economic Non-Coverage

Intellectual integrity requires explicit acknowledgment of what deterministic runtime consent enforcement does not address and what economic exposure it does not reduce. The following categories fall outside the scope of execution-layer enforcement architectures of this class.

10.1 Model Performance Failures

Deterministic enforcement governs whether an AI action was authorized. It does not govern whether the AI performed that action correctly, accurately, or safely. Model hallucinations, factual errors, biased outputs, and performance degradation are outside the scope of consent enforcement and represent distinct economic exposures that enforcement infrastructure does not address. Organizations should not interpret consent enforcement conformance as a statement about model quality, reliability, or safety.

10.2 Business Strategy and Product Decisions

Poor product strategy, incorrect market assumptions, or flawed business decisions made using AI-generated outputs are not addressable by consent enforcement. Deterministic enforcement architectures verify that AI actions are authorized at the execution layer; they do not govern whether authorized AI actions lead to sound business outcomes. The economic risk of bad strategy is unaffected by enforcement architecture.

10.3 Ethical Design Costs

Consent enforcement is not equivalent to ethical AI design. An AI system may be fully conformant with consent enforcement requirements while producing outputs or taking actions that raise ethical concerns. Ethical design (addressing fairness, bias, societal impact, and value alignment) is a separate discipline requiring separate investment.

Runtime consent enforcement architectures do not reduce ethical design cost and should not be treated as a proxy for ethical compliance or responsible AI certification.

10.4 Training Data and Intellectual Property Exposure

Consent enforcement operates at the point of AI action execution. It does not address upstream questions about training data licensing, intellectual property in model outputs, or data provenance obligations. These represent distinct legal and economic exposures that require separate governance mechanisms.

10.5 Product Liability Unrelated to Consent

Where AI outputs cause physical harm, financial loss, or other product liability, the enforcement of consent at the time of action does not affect the product liability exposure itself. Consent documentation may be relevant in some liability contexts, but it does not constitute a defense against harm caused by AI system malfunction, incorrect output, or unsafe design.

- Model accuracy, reliability, and safety failures: not addressed by enforcement
- Business and strategic decision quality: outside enforcement scope
- Ethical design obligations: distinct discipline, separate cost
- Training data licensing and IP provenance: upstream of enforcement layer
- Product liability for AI-caused harm: not mitigated by consent record
- Reputational damage from public-facing AI behavior: indirect effect only

SECTION 11

Long-Term Infrastructure Stabilization Model

Beyond direct cost reduction in specific incident categories, deterministic enforcement introduces a longer-term economic property: cost stabilization. This section examines the mechanisms by which enforcement infrastructure reduces the volatility of compliance, audit, and legal costs over time.

11.1 Reduced Compliance Cost Volatility

In non-enforced environments, compliance cost is volatile: it scales with incident frequency, regulatory inquiry volume, and the labor intensity of any given audit, variables that are structurally difficult to forecast because they depend on both internal incident patterns and external regulatory attention cycles that organizations do not control. Compliance budgeting in AI-deployed organizations without enforcement infrastructure routinely requires contingency reserves that may or may not be consumed, depending on incident patterns and regulatory attention cycles.

To the extent that enforcement infrastructure is fully and continuously deployed, it converts a significant portion of variable compliance cost into fixed operational cost (the infrastructure budget), which is foreseeable and stable regardless of the incident frequency that materializes in any given year.

11.2 Reduced Litigation Unpredictability

Legal disputes arising from AI actions involve significant discovery and evidentiary preparation cost. In non-enforced environments, this cost is amplified by the need to reconstruct authorization states that were not recorded, a process that is adversarial in character, since opposing parties have different incentives regarding what the reconstruction reveals. In enforced environments, assuming no degradation of enforcement guarantees, the evidentiary basis, the artifact corpus, is already available and machine-verifiable, specifically by eliminating the reconstruction phases that would otherwise require manual correlation of incomplete log data across heterogeneous systems.

This structural reduction in litigation unpredictability has compounding effects: it reduces legal budget variance, decreases settlement pressure created by evidentiary uncertainty, and shortens resolution timelines in proceedings where the evidentiary dispute is the primary cost driver.

11.3 Standardization Effect on Insurance Pricing

> **FORWARD-LOOKING STRUCTURAL OBSERVATION: NON-QUANTIFIED, NO ACTUARIAL BASIS**
>
> This subsection identifies a structural dynamic in the emerging AI liability insurance market. No actuarial basis exists for quantifying premium effects at this time, and no pricing projections are advanced here.

As AI-specific insurance products develop, underwriters are likely to differentiate organizations on the basis of demonstrable governance controls. The structural distinction between an organization that can produce machine-verifiable, artifact-based consent enforcement records and one that relies on policy documentation and process attestation is qualitatively significant from an underwriting perspective. The former reduces claims investigation cost and scope dispute, while the latter requires interpretive reconstruction under adversarial conditions. This analysis makes no claim about the direction or magnitude of any such differential; it notes only that the structural incentive exists for underwriters to recognize enforcement conformance as a relevant risk variable.

11.4 Cost Predictability Benefits

The compound effect of the stabilization properties described above is a shift in the economic character of AI compliance from a variable, incident-driven cost to a predictable, infrastructure-driven cost. This predictability has intrinsic institutional value: it enables reliable budget planning, reduces the reserve requirements that compliance uncertainty creates, and removes a class of organizational risk that currently occupies disproportionate leadership attention in AI-deployed organizations.

Infrastructure that converts uncertain liability into foreseeable operational cost is economically valuable independent of any particular cost reduction it produces. Predictability is itself a stabilization property, and one that non-enforced environments structurally cannot provide.

ECONOMIC INTERPRETATION

Economic Interpretation

The primary economic effect of deterministic enforcement is not the elimination of risk. It is the structural reduction of authorization ambiguity, the condition that makes risk expensive to investigate, scope, and close. This distinction matters because ambiguity and risk, while related, have different cost structures and different addressability.

CORE ECONOMIC MECHANISM

Risk is a function of probability and magnitude. Some portion of AI-related risk is irreducible regardless of governance architecture. Ambiguity, however, is an architectural property; it arises from the absence of machine-readable authorization state at execution time, and it is addressable through infrastructure design. The cost of ambiguity is not the cost of incidents that occurred; it is the cost of investigating all actions that might have been unauthorized, because the architecture cannot distinguish them from those that were not.

In non-enforced environments, every material incident requires forensic reconstruction: log correlation, schema normalization, analyst interpretation, and legal review of a population of ambiguous actions that includes both the problematic and the merely uninvestigated. This investigation overhead is not proportional to the actual violation; it is proportional to the ambiguity surface, the total set of actions where authorization cannot be immediately established.

Deterministic enforcement is expected to reduce the ambiguity surface to near-zero at the point of execution, under conditions of complete and uninterrupted enforcement coverage. Every action produces an immutable decision artifact; every artifact is queryable; every query is expected to

return a deterministic result. The investigation labor that dominates audit cost in non-enforced environments is expected to become artifact retrieval under this architecture, a structurally faster and lower-cost operation, subject to the artifact store remaining intact and queryable.

This mechanism explains why the class of enforcement infrastructure specified in documents such as the QODIQA Core Standard v1.0 and operationalized through structured enforcement point frameworks is analytically expected to produce cost reduction across heterogeneous incident categories. The reduction does not depend on incident type. It depends on the universal structural property shared by all AI governance cost: the need to establish what the authorization state was at the time an action was taken. That property is structural and consistent across audit, incident response, legal review, and regulatory inquiry, and it is addressable, at the execution layer, in a way that no policy-layer control can replicate.

A secondary economic effect is the shift in governance cost structure from reactive to proactive. Non-enforced governance spends against incidents as they materialize, with cost scaling unpredictably against incident frequency and regulatory attention cycles that are outside the organization's control. Enforced governance, to the extent that deterministic evaluation is consistently applied, converts a significant portion of that variable cost into fixed infrastructure operational expense: foreseeable, budgetable, and stable regardless of incident patterns in a given year.

ECONOMIC SUMMARY

Ambiguity drives investigation cost. Deterministic artifacts are expected to reduce authorization ambiguity at execution time, under conditions of complete coverage. Governance cost is expected to shift from reactive investigation labor toward proactive infrastructure cost. The modeled magnitude of this shift, at mid-band incident frequency and under full-conformance assumptions, materially exceeds the annual operational cost of maintaining conformant enforcement infrastructure. Neither claim has been empirically validated; both are advanced as structural propositions subject to the methodological constraints stated in the Methodological Positioning section of this paper.

SECTION 12

Institutional Closing Statement

This analysis has sought to demonstrate, through structured modeling and explicit assumption-bound reasoning, the economic consequence of a specific infrastructure gap: the absence of deterministic runtime consent enforcement in AI-deployed organizational environments.

The argument is not that deterministic enforcement architectures eliminate cost, reduce risk to zero, or guarantee outcomes. The argument is structural and narrower: **systems that lack execution-time authorization artifacts structurally incur non-zero ambiguity cost. This cost cannot be eliminated through policy-layer controls alone, because policy-layer controls do not produce queryable execution-time records. The absence of deterministic enforcement implies persistent reconstruction overhead.** The cost dynamics described throughout this paper (audit reconstruction overhead, incident scoping uncertainty, multi-agent propagation exposure, regulatory inquiry latency) share a common structural cause. In each case, the cost is amplified by the organization's inability to establish, at query time, what its authorization state was at execution time. This is not a contingent property of any particular organization's practices; it is a structural consequence of the architecture.

> Deterministic runtime enforcement does not alter the behavior of AI systems; it records and enforces the authorization context within which they operate. The economic benefit is not behavioral. It is evidentiary. **What cannot be proven cannot be defended. What cannot be defended must be resolved through the most expensive available mechanism: litigation, regulatory proceeding, or exhaustive reconstruction.**

12.1 What This Analysis Reasons Toward

This paper has modeled, through first-principles structural reasoning, the following propositions: that the baseline economic exposure of operating without deterministic enforcement is structured and directionally estimable from stated assumptions, specifically because the cost categories are mechanistically traceable to the absence of queryable execution-time records; that the expected cost reduction properties of enforcement infrastructure are traceable to specific causal mechanisms rather than asserted without grounding; that the modeled implementation cost of enforcement infrastructure is substantially lower than the annual baseline direct cost exposure it is designed to address at medium-to-high incident frequencies, within the stated modeling assumptions; and that the long-term cost stabilization effect of enforcement infrastructure is expected to reduce the volatility properties of AI compliance cost that currently create significant planning challenges for risk and finance leadership. None of these propositions has been empirically validated; all are advanced as structural arguments subject to the methodological constraints stated in the Methodological Positioning section of this paper.

12.2 The Infrastructure Framing

Deterministic runtime consent enforcement is not a revenue model, a product feature, or a compliance checkbox. It is an infrastructure stabilization layer, analogous to database transaction logging, cryptographic certificate infrastructure, or network access control in the degree to which it provides a foundational evidentiary property that higher-level governance functions depend upon. Formal specifications of this layer provide normative definition to the architectural properties that produce the economic effects described in this paper, but the economic logic is a property of the architectural class and holds for any implementation that fully satisfies those properties.

Organizations that invest in this class of infrastructure are not acquiring a competitive advantage; they are removing a structural vulnerability (the absence of execution-time authorization records) that will otherwise become more expensive as AI deployment scales, regulatory scrutiny intensifies, and multi-agent architectures mature. That vulnerability is not

discretionary; it is a property of any AI governance architecture that does not produce queryable, immutable authorization records at execution time.

> Systems without execution-time authorization artifacts incur reconstruction cost at every audit and every incident, as a structural property of their architecture. That cost is not discretionary; it is the price of operating without a queryable authorization record. The question this analysis poses is not whether such infrastructure should be built. It is whether the cost of building it is smaller than the cost of not having it. The modeling in this paper reasons toward a single, bounded answer: within the stated assumptions, and at medium-to-high incident frequency, it is.

This analysis has been prepared for institutional review. It does not constitute investment advice, legal counsel, or a guarantee of any specific outcome. All modeling is assumption-bound, conservative, and subject to the limitations stated herein.

DOCUMENT STATUS

Document Status and Classification

This document is an independent technical and economic analysis. It is authored by an individual researcher and issued in association with the QODIQA documentation corpus as an analytical companion to the QODIQA Core Standard v1.0. It is version-controlled and subject to update as regulatory frameworks, incident datasets, and infrastructure cost benchmarks evolve. This version carries identifier QODIQA-EIP-2026-001 and reflects conditions current as of the date of publication.

The economic modeling contained herein is structural in character, derived from first-principles reasoning about the operational properties of enforced and non-enforced AI governance environments. It does not represent actuarial measurement, empirical benchmarking, investment guidance, or legal advice. All figures are directional cost bands whose derivation assumptions are stated explicitly throughout the document. This analysis has not been externally validated through empirical deployment data and should be evaluated on the basis of its structural reasoning.

This paper is intended for review by:

- Risk officers and chief information security officers
- Legal and compliance leadership
- AI governance committees
- Infrastructure architects and platform engineers
- Institutional procurement review bodies

- Regulatory and policy stakeholders

It is not intended for general public distribution without institutional context.

This document should be read together with the following related specifications:

- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Audit and Evidence Generation Model, Version 1.0
- QODIQA: Audit Readiness and Evidence Pack, Version 1.0
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Conformance Verification Specification, Version 1.0
- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Corpus Change Log and Version History, Version 1.0
- QODIQA: Economic Impact Analysis of Deterministic Runtime Consent Enforcement, Version 1.0

- QODIQA: Executive Brief for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Extended Adversarial Threat Model, Version 1.0
- QODIQA: Failure Handling and Recovery Specification, Version 1.0
- QODIQA: Global Enforcement Invariants for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0
- QODIQA: Implementation Conformance Checklist, Version 1.0
- QODIQA: Implementation Playbook for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Interoperability and Deployment Constraints, Version 1.0
- QODIQA: Master Index and Readers Guide, Version 1.0
- QODIQA: Non-Compliance Conditions and Failure Modes, Version 1.0
- QODIQA: Positioning and Scope Limitation Statement, Version 1.0
- QODIQA: Public Specification License, Version 1.0
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Reference Implementation Specification - Minimal Deterministic Enforcement Stack, Version 1.0

- QODIQA: Regulatory Alignment Matrix for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Residual Risk and Assumption Disclosure Annex, Version 1.0
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0
- QODIQA: System Boundary and Trust Model Specification, Version 1.0
- QODIQA: Terminology and Normative Definitions, Version 1.0
- QODIQA: Threat Model and Abuse Case Specification, Version 1.0
- QODIQA: Use Case Dossiers for Runtime Consent Enforcement Deployments, Version 1.0
- QODIQA: Worked Example End to End Scenario, Version 1.0

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Duțescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-EIP-2026-001
TITLE	QODIQA: Economic Impact Analysis of Deterministic Runtime Consent Enforcement, Version 1.0
AUTHOR	Bogdan Duțescu: Independent Technical and Economic Analysis
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Independent Analytical White Paper
NORMATIVE STATUS	Informational Analysis: Non-Normative
RELATION TO STANDARD	Non-normative analytical companion; QODIQA used as reference implementation
GOVERNING AUTHORITY	QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

INSTITUTIONAL

Scope

This document provides an independent economic and risk-surface analysis of the general class of deterministic runtime consent enforcement architectures. It uses the QODIQA Core Standard v1.0 as a reference specification for this architectural class, drawing on its definitional precision regarding decision artifacts, fail-closed enforcement semantics, and policy registry requirements. It models the structural cost implications of operating AI systems with and without enforcement infrastructure meeting these properties, across organizational profiles and incident frequency bands.

It does not define normative enforcement requirements and should be interpreted as an independent analytical companion to the QODIQA technical specifications. Organizations seeking normative implementation guidance should refer to the 68-Point Enforcement Framework v3.0 and the QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0. All modeling in this paper is structural, assumption-bound, first-principles in character, and explicitly non-actuarial. It has not been validated against empirical deployment data and should be treated as a structured analytical framework for institutional reasoning, not as an empirically grounded cost model.

INSTITUTIONAL

Terminology

The following definitions apply to terms as used in this paper. For normative terminology applicable to the full QODIQA standard corpus, refer to the QODIQA: Terminology and Normative Definitions, Version 1.0.

Deterministic Enforcement	A runtime authorization mechanism that evaluates consent state and applicable policy at the moment of AI action execution and produces a verifiable, immutable decision artifact recording the outcome. Distinguished from probabilistic or policy-level governance by its machine-verifiable output and fail-closed behavior.
Decision Artifact	An immutable record produced at execution time containing the resolved consent state, applicable policy version, declared purpose, request context hash, and authorization outcome. Decision artifacts are the primary evidentiary unit in artifact-based audit and incident response workflows. In the QODIQA reference architecture, decision artifacts are cryptographically signed and stored in an append-only record that supports deterministic replay.

Ambiguity Surface
The set of AI actions in a non-enforced environment where authorization state cannot be deterministically established at query time without forensic reconstruction. The ambiguity surface is the primary driver of audit reconstruction and incident scoping cost in the baseline exposure model.

Propagation Multiplier
In multi-agent architectures, the number of downstream AI actions generated from a single upstream authorization failure when no deterministic enforcement gate is present at agent boundaries. The propagation multiplier scales incident scope, remediation cost, and notification obligations.

Fail-Closed Semantics
An enforcement behavior in which the absence of a valid, resolvable consent record or policy version causes the action to be denied rather than permitted. Fail-closed enforcement is designed to ensure that ambiguous or unresolvable authorization states do not result in unauthorized AI action execution, by defaulting to denial rather than permitting execution under conditions of uncertainty.

Artifact-Based Replay
An audit methodology in which historical AI actions are verified by retrieving and replaying their corresponding decision artifacts rather than reconstructing authorization state from logs. Artifact-based replay is the primary compliance efficiency mechanism introduced by deterministic enforcement infrastructure.

INSTITUTIONAL

Conformance Language

The key words **MUST**, **MUST NOT**, **SHOULD**, **SHOULD NOT**, and **MAY** in the QODIQA specification corpus indicate requirement levels for conformant implementations of the QODIQA enforcement architecture.

These terms describe normative enforcement expectations for systems claiming conformance with the QODIQA standard. In this document, such terms appear only in cross-references to normative specifications and do not

themselves impose normative obligations. For normative requirement definitions, refer to the QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0.

INSTITUTIONAL

Non-Coverage

This paper explicitly does not model the following categories of economic exposure. These are noted here to clarify the analytical boundary of the cost modeling contained in the document and to prevent misinterpretation of the scope of benefit claimed.

- × Catastrophic regulatory penalty exposure: maximum-band GDPR, EU AI Act, or sector-specific fines are excluded from all cost tables
- × Class-action litigation dynamics: aggregate user or data subject claims are not modeled
- × Reputational recovery costs: public-facing brand remediation, communications counsel, and media management are outside the model
- × Sector-specific enforcement obligations: financial services, healthcare, and critical infrastructure obligations may differ materially from the general model
- × Model accuracy and AI safety failures: enforcement architecture does not address model performance, hallucination, or output quality
- × Training data provenance and intellectual property exposure: upstream legal obligations are not within the enforcement scope modeled
- × Insurance premium effects: no actuarial basis exists for quantifying AI liability insurance pricing at this time

> **ANALYTICAL BOUNDARY**
>
> The cost model covers direct operational costs of governance ambiguity. Tail-risk events, sector-specific obligations, and non-consent economic exposures require separate actuarial and legal analysis. The absence of tail-risk figures does not imply the absence of tail-risk exposure.

INSTITUTIONAL

QODIQA Standard Corpus

This document is an analytical companion to the QODIQA standard corpus: a structured set of technical specifications, analytical papers, governance instruments, and implementation resources. The corpus provides the normative architectural definitions against which the economic properties analyzed in this paper are assessed. Documents in the corpus are cross-referenced and versioned under the QODIQA Change Log and Version Control Protocol.

Scope Note

This paper draws on QODIQA corpus specifications as a reference architecture for the class of deterministic runtime consent enforcement systems under analysis. Where properties of QODIQA-conformant implementations are referenced, those properties are drawn from the QODIQA Core Standard and associated specifications. This document does not itself specify conformance obligations; it analyzes the economic implications of conformance as defined elsewhere in the corpus.

- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Audit and Evidence Generation Model, Version 1.0
- QODIQA: Audit Readiness and Evidence Pack, Version 1.0
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0

- QODIQA: Conformance Verification Specification, Version 1.0
- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper, Version 1.0
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Corpus Change Log and Version History, Version 1.0
- QODIQA: Economic Impact Analysis of Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Executive Brief for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Extended Adversarial Threat Model, Version 1.0
- QODIQA: Failure Handling and Recovery Specification, Version 1.0
- QODIQA: Global Enforcement Invariants for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0
- QODIQA: Implementation Conformance Checklist, Version 1.0
- QODIQA: Implementation Playbook for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Interoperability and Deployment Constraints, Version 1.0
- QODIQA: Master Index and Readers Guide, Version 1.0

- QODIQA: Non-Compliance Conditions and Failure Modes, Version 1.0
- QODIQA: Positioning and Scope Limitation Statement, Version 1.0
- QODIQA: Public Specification License, Version 1.0
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Reference Implementation Specification - Minimal Deterministic Enforcement Stack, Version 1.0
- QODIQA: Regulatory Alignment Matrix for Deterministic Runtime Consent Enforcement, Version 1.0
- QODIQA: Residual Risk and Assumption Disclosure Annex, Version 1.0
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0
- QODIQA: System Boundary and Trust Model Specification, Version 1.0
- QODIQA: Terminology and Normative Definitions, Version 1.0
- QODIQA: Threat Model and Abuse Case Specification, Version 1.0
- QODIQA: Use Case Dossiers for Runtime Consent Enforcement Deployments, Version 1.0
- QODIQA: Worked Example End to End Scenario, Version 1.0

INSTITUTIONAL

Institutional Disclaimer

> This document is an independent technical and economic analysis of deterministic runtime consent enforcement architectures and their structural implications for organizational cost and risk posture. It does not constitute regulatory guidance, legal advice, financial recommendation, or investment analysis of any kind. It is authored by an individual researcher; the conclusions and structural reasoning expressed herein represent the author's analytical positions, not those of any standards body, employer, or institutional entity.

All cost bands and modeling figures presented in this paper are structural estimates derived from first-principles reasoning about the operational mechanics of enforced and non-enforced AI governance environments. They are not actuarial measurements, empirical benchmarks, or guaranteed outcomes. No figures in this paper have been validated against empirical deployment data. Actual costs will vary materially based on organizational scale, incident frequency, regulatory jurisdiction, and implementation approach. The figures should not be used as a basis for investment decisions, capital planning, or regulatory positioning without independent actuarial, legal, and technical review.

This paper makes no claims regarding the regulatory compliance status of any organization implementing deterministic enforcement infrastructure conformant with the QODIQA specification or any other specification. Regulatory compliance determinations require qualified legal counsel familiar with the applicable jurisdiction, sector, and specific organizational context.

The author and QODIQA Standards and Research do not accept liability for decisions made on the basis of the economic modeling contained in this paper. The structural cost arguments presented are intended to support informed institutional reasoning, not to substitute for it. Organizations should conduct independent actuarial, legal, and technical review appropriate to their circumstances before making infrastructure investment decisions of any kind.

INSTITUTIONAL

Recommended Citation

CITE THIS DOCUMENT AS

Duțescu, Bogdan.
Economic Impact Analysis of Deterministic Runtime Consent Enforcement.
QODIQA: Economic Impact Analysis of Deterministic Runtime Consent Enforcement, Version 1.0.
QODIQA Standard Corpus, 2026.
Document Identifier: QODIQA-EIP-2026-001.

Persistent Reference: QODIQA-EIP-2026-001

CHAPTER 28 PART V · CONTEXTUAL, ANALYTICAL AND PROCESS SUPPORT

Regulatory Alignment Matrix for AI Governance Frameworks

Technical alignment matrix for deterministic runtime enforcement across major AI and data governance frameworks.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-RAM-2026-001	1.0	2026	Mapping Specification	Normative	NORMATIVE

ABSTRACT

This document presents a **technical control alignment matrix** mapping QODIQA enforcement constructs to major AI and data governance frameworks: GDPR, EU AI Act, NIST AI Risk Management Framework, ISO/IEC 27001, ISO/IEC 42001, and the OECD AI Principles.

The matrix is structured around a deterministic enforcement anchor model. For each regulatory requirement, it identifies which QODIQA controls provide structural support, the evidence artifacts generated, and the residual scope not addressed by runtime enforcement. **This document does not constitute legal advice. It does not certify regulatory compliance. It is a technical cross-reference instrument.**

The matrix is intended for enterprise risk teams, compliance architects, regulatory affairs specialists, and AI governance practitioners evaluating the control coverage characteristics of QODIQA within their broader compliance posture.

HOW TO READ THIS MATRIX (NON-NORMATIVE)

Direct: The control enforces the requirement deterministically at runtime.
Structural: The system produces verifiable evidence supporting the requirement, but does not enforce it directly.
Out of Scope: The requirement lies outside the runtime enforcement boundary and is not addressed.

SECTION 1

Purpose and Scope

1.1 Purpose of This Matrix

This Regulatory Alignment Matrix provides a structured, technical mapping of QODIQA enforcement constructs to the requirements of major AI and data governance frameworks. Its purpose is to enable organizations deploying QODIQA to understand, in precise structural terms, which regulatory control objectives are addressed by deterministic runtime consent enforcement, which are partially supported through evidence generation, and which fall outside the scope of the QODIQA enforcement boundary.

The matrix answers four operational questions: Which QODIQA controls correspond to which regulatory requirements? What evidence artifacts does QODIQA generate that may be relevant to regulatory demonstrability? Where does QODIQA not claim coverage? How does deterministic runtime enforcement reduce the ambiguity surface at execution time?

MANDATORY DISCLAIMER

This document does not constitute legal advice. It does not represent regulatory guidance from any supervisory authority. It does not guarantee, certify, or warrant compliance with any regulation, standard, or law referenced herein. All regulatory interpretation, gap analysis, legal assessment, and compliance determination must be performed by qualified legal counsel and certified compliance professionals in the relevant jurisdiction. QODIQA's role is technical control enforcement, not regulatory adjudication.

1.2 Scope of Interpretation

Mappings in this document are limited to **structural and functional alignment** between QODIQA enforcement constructs and published regulatory text or guidance. The matrix does not interpret regulatory intent, predict supervisory authority interpretation, or assess the legal sufficiency of any particular implementation. Regulatory frameworks are cited in their published form as of Q1 2026.

1.3 Relationship to QODIQA Corpus

TABLE 1.3 - RELATIONSHIP TO CORPUS DOCUMENTS

DOCUMENT	DOCUMENT ID	RELATIONSHIP TO THIS MATRIX
Core Standard v1.0	QODIQA-CORE-1.0	Normative source for enforcement constructs cited in mappings
68-Point Enforcement Framework	QODIQA-68P-1.0	Source of control identifiers used in alignment tables
Certification Framework v1.0	QODIQA-CERT-1.0	Defines audit artifacts referenced as evidence outputs
Implementation Playbook v1.0	QODIQA-PLAY-1.0	Operational context for deployment-level control activation
Security and Cryptographic Profile v1.0	QODIQA-SEC-1.0	Cryptographic evidence generation supporting audit trails
Reference Architecture v1.0	QODIQA-ARCH-1.0	Structural enforcement model underlying all control mappings

1.4 Frameworks Covered

TABLE 1.4 - REGULATORY AND STANDARDS FRAMEWORKS IN SCOPE

FRAMEWORK	JURISDICTION / BODY	VERSION / EDITION	MAPPING TYPE
General Data Protection Regulation (GDPR)	European Union	Regulation 2016/679	Article-level control mapping
EU Artificial Intelligence Act	European Union	Regulation 2024/1689	Obligation-level control mapping
NIST AI Risk Management Framework	NIST (USA)	AI RMF 1.0 (2023)	Function/category alignment
ISO/IEC 27001	ISO / IEC JTC 1	ISO/IEC 27001:2022	Control family alignment
ISO/IEC 42001	ISO / IEC JTC 1/SC 42	ISO/IEC 42001:2023	Clause-level structural mapping
OECD AI Principles	OECD	OECD/LEGAL/0449 (2024 rev.)	Principle-level structural support

1.5 Scope Limitation

This document provides a technical crosswalk between the QODIQA enforcement model and selected regulatory frameworks. The mappings presented are analytical in nature and are intended to support architectural alignment discussions and technical interpretation. They do not constitute legal advice, regulatory interpretation, or formal compliance certification.

The Regulatory Alignment Matrix identifies where QODIQA technical controls structurally correspond to regulatory control objectives. It does not assess the sufficiency of those controls for any specific deployment context, nor does it substitute for jurisdiction-specific legal analysis. Organizations must conduct independent compliance assessments with qualified legal and regulatory counsel.

1.6 Regulatory Interpretation Disclaimer

The mappings presented in this matrix represent an analytical interpretation of how deterministic runtime consent enforcement may interact with existing governance frameworks. The alignment claims are structural and technical - they reflect the logical correspondence between QODIQA enforcement constructs and the stated control objectives of each framework.

Final regulatory interpretation remains the responsibility of competent authorities and qualified legal advisors. QODIQA does not claim regulatory authority, does not certify compliance outcomes, and does not warrant that any deployment of QODIQA controls satisfies the requirements of any referenced regulation or standard.

SECTION 2

Mapping Methodology

2.1 Control-Based Mapping Model

All mappings in this document are structured around QODIQA enforcement controls as the primary anchor. Each regulatory requirement is evaluated against the QODIQA control set to determine whether a direct structural correspondence exists, whether QODIQA provides supporting evidence without direct enforcement, or whether the requirement falls outside the enforcement boundary. This approach prevents overclaiming while identifying genuine structural contributions.

2.2 Mapping Logic Chain

Each alignment entry follows a four-stage traceability chain from QODIQA control through to residual gap disclosure:

QODIQA CONTROL → REGULATORY REQUIREMENT → EVIDENCE ARTIFACT → RESIDUAL GAP

2.3 Coverage Classification Scheme

TABLE 2.3 - COVERAGE CLASSIFICATION DEFINITIONS

CLASSIFICATION	DEFINITION	INDICATOR
Direct	QODIQA control directly enforces the requirement at runtime through deterministic gating, token validation, or cryptographic verification.	DIRECT
Structural	QODIQA generates structured, replayable evidence that supports demonstrability of the requirement but does not constitute enforcement of the requirement itself.	STRUCTURAL
Partial	QODIQA addresses a subset of the requirement within the runtime enforcement boundary. Other dimensions of the requirement require complementary controls outside QODIQA.	PARTIAL
Out of Scope	The requirement addresses areas outside the QODIQA enforcement boundary (e.g., model training, fairness, business process governance). No coverage claimed.	OUT OF SCOPE

2.4 Evidence Artifact Classes

TABLE 2.4 - QODIQA EVIDENCE ARTIFACT REFERENCE

ARTIFACT CLASS	DESCRIPTION	SOURCE
Consent Token (CT)	Cryptographically signed consent record with subject identity binding, scope, expiry, and purpose encoding	Core Standard 4; SEC Profile 4
Audit Ledger Entry (ALE)	Append-only, hash-chained record of every enforcement decision at the SUP boundary	Core Standard 7; SEC Profile 8
Enforcement Attestation (EA)	Signed summary of enforcement posture for a defined period or system scope	Certification Framework 6
Revocation Record (RR)	Timestamped, cryptographically bound record of consent withdrawal with propagation confirmation	Core Standard 6; SEC Profile 6
Purpose Binding Record (PBR)	Deterministic record of the declared purpose scope at time of consent grant and enforcement evaluation	Core Standard 3; 68-Point Framework C
Lifecycle Event Record (LER)	Structured record of AI system lifecycle state transitions from training through retirement	Core Standard 9; Reference Architecture 5

2.5 Interpretation Constraints

Mappings are **structural, not legal**. The presence of a QODIQA control in an alignment table does not imply that deploying QODIQA satisfies the associated regulatory requirement. Regulatory compliance requires legal analysis, operational controls, organizational governance, and in many cases external audit - none of which QODIQA substitutes for. QODIQA provides a technical control plane that may support compliance demonstration; determination of actual compliance status is outside its scope.

2.6 Deterministic Execution Boundary Definition

EXEC-BOUNDARY-DEF **Deterministic Execution Boundary - Formal Definition** NORMATIVE

Boundary	The **System Usage Plane (SUP)** is the deterministic enforcement boundary interposed between all external requestors and all AI system invocation targets. No invocation path may bypass the SUP boundary. The SUP boundary is architecturally invariant across all QODIQA deployment profiles.
Enforcement Invariant	**For all invocation i:** execute(i) if and only if there exists token T such that: valid(T) and not-expired(T) and not-revoked(T) and purpose_match(T, declared_scope(i)). Where any conjunct evaluates to false, the SUP MUST emit a deterministic BLOCK decision and generate an Audit Ledger Entry recording the failure condition, timestamp, and requestor identity binding.
Evaluation Time	Token validation is performed at invocation time, not at consent grant time. A token valid at grant may be non-valid at invocation if it has expired or been revoked in the intervening period. The enforcement evaluation is synchronous with respect to invocation; no deferred or asynchronous evaluation model is permitted.
Cryptographic Verifiability	Token validity is established through cryptographic signature verification against the Trust Anchor public key. Validation does not depend on system state beyond the revocation registry and the Trust Anchor. The verification function is deterministic and stateless with respect to application-layer state.

Default Posture	The SUP boundary operates in default-deny posture. In the absence of a valid, non-expired, non-revoked, purpose-matched token, processing MUST NOT proceed. An ambiguous, malformed, or unparseable token state is treated as a denial condition, not as a pass-through. No fail-open behavior is permitted under any QODIQA compliance profile.
Mapping Assumption	All regulatory alignment mappings in Sections 3 through 10 of this document assume the enforcement invariant above is operative. Coverage classifications (Direct, Structural, Partial) are valid only for deployments in which the SUP boundary invariant is enforced without modification. Deployments that bypass, weaken, or partially implement the SUP boundary invalidate all Direct coverage classifications for their deployment context.

SECTION 2.X

QODIQA Control Index and Traceability Registry

This section provides the formal control registry for all QODIQA enforcement constructs referenced in the regulatory alignment tables of Sections 3 through 10. Each control is assigned a stable identifier, domain classification, enforcement layer, evidence output, control type, and maturity classification. Regulatory reviewers, auditors, and risk analysts should use this registry as the primary reference for control traceability across all framework mappings.

2.X.1 Consent Domain Controls

TABLE 2.X.1 - CONSENT DOMAIN CONTROL REGISTRY

CONTROL ID	CONTROL DOMAIN	CORE REFERENCE	ENFORCEMENT LAYER	EVIDENCE ARTIFACT	CONTROL TYPE	MATURITY
CT-01	Consent	Core Standard 4.1	Token	CT, ALE	Deterministic Enforcement	Foundational
CT-02	Consent	Core Standard 4.2	Token	CT, PBR	Deterministic Enforcement	Foundational
CT-03	Consent	Core Standard 4.3	Token	CT	Deterministic Enforcement	Foundational

CONTROL ID	CONTROL DOMAIN	CORE REFERENCE	ENFORCEMENT LAYER	EVIDENCE ARTIFACT	CONTROL TYPE	MATURITY
CT-EXPIRY	Consent	Core Standard 4.4	Token	CT (expiry field), ALE	Deterministic Enforcement	Foundational
CT-SCOPE	Consent	Core Standard 4.5	Token	CT, PBR	Deterministic Enforcement	Advanced

2.X.2 Purpose Limitation Controls

TABLE 2.X.2 - PURPOSE DOMAIN CONTROL REGISTRY

CONTROL ID	CONTROL DOMAIN	CORE REFERENCE	ENFORCEMENT LAYER	EVIDENCE ARTIFACT	CONTROL TYPE	MATURITY
PL-01	Purpose	Core Standard 3.1; 68-Point C.01	SUP Boundary	PBR, ALE	Deterministic Enforcement	Foundational
PL-02	Purpose	Core Standard 3.2; 68-Point C.02	SUP Boundary	PBR, ALE	Deterministic Enforcement	Advanced
PL-03	Purpose	Core Standard 3.3; 68-Point C.03	SUP Boundary	PBR, CT	Deterministic Enforcement	Advanced
PBR-01	Purpose	Core Standard 3.4	Token	PBR	Structural Evidence	Foundational

2.X.3 Enforcement Domain Controls

TABLE 2.X.3 - ENFORCEMENT DOMAIN CONTROL REGISTRY

CONTROL ID	CONTROL DOMAIN	CORE REFERENCE	ENFORCEMENT LAYER	EVIDENCE ARTIFACT	CONTROL TYPE	MATURITY
ENF-01	Enforcement	68-Point E.01	SUP Boundary	ALE	Deterministic Enforcement	Foundational

CONTROL ID	CONTROL DOMAIN	CORE REFERENCE	ENFORCEMENT LAYER	EVIDENCE ARTIFACT	CONTROL TYPE	MATURITY
ENF-07	Enforcement	68-Point E.07	SUP Boundary	ALE, PBR	Deterministic Enforcement	Advanced
ENF-DEFAULT-DENY	Enforcement	Core Standard 2.1; 68-Point E.10	SUP Boundary	ALE (BLOCK events)	Deterministic Enforcement	Foundational
ENF-HALT	Enforcement	Core Standard 2.3	SUP Boundary	ALE (HALT events)	Deterministic Enforcement	Advanced
ENF-RETIRE	Enforcement	Core Standard 9.4	SUP Boundary	LER	Deterministic Enforcement	Advanced
ENF-SESSION	Enforcement	Core Standard 5.2	SUP Boundary	ALE (session records)	Deterministic Enforcement	Advanced
ENF-TEST	Enforcement	Certification Framework 4	SUP Boundary	Test records	Governance Support	Advanced
ENF-THIRD-PARTY	Enforcement	Core Standard 8.1	SUP Boundary	ALE, CT	Deterministic Enforcement	High-Assurance
ENF-REVOKE	Enforcement	Core Standard 6.3	Registry	RR, ALE	Deterministic Enforcement	Foundational

2.X.4 Revocation Domain Controls

TABLE 2.X.4 - REVOCATION DOMAIN CONTROL REGISTRY

CONTROL ID	CONTROL DOMAIN	CORE REFERENCE	ENFORCEMENT LAYER	EVIDENCE ARTIFACT	CONTROL TYPE	MATURITY
REV-01	Consent	Core Standard 6.1; SEC Profile 6.1	Registry	RR, ALE	Deterministic Enforcement	Foundational
REV-02	Consent	Core Standard 6.2; SEC Profile 6.2	Registry	RR	Deterministic Enforcement	Advanced

CONTROL ID	CONTROL DOMAIN	CORE REFERENCE	ENFORCEMENT LAYER	EVIDENCE ARTIFACT	CONTROL TYPE	MATURITY
REV-03	Consent	Core Standard 6.3; SEC Profile 6.3	Registry	RR, ALE	Deterministic Enforcement	High-Assurance

2.X.5 Audit Domain Controls

TABLE 2.X.5 - AUDIT DOMAIN CONTROL REGISTRY

CONTROL ID	CONTROL DOMAIN	CORE REFERENCE	ENFORCEMENT LAYER	EVIDENCE ARTIFACT	CONTROL TYPE	MATURITY
AUD-01	Audit	Core Standard 7.1; SEC Profile 8.1	Ledger	ALE	Structural Evidence	Foundational
AUD-02	Audit	Core Standard 7.2; SEC Profile 8.2	Ledger	ALE	Structural Evidence	Foundational
AUD-03	Audit	Core Standard 7.3	Ledger	ALE	Structural Evidence	Advanced
AUD-04	Audit	Core Standard 7.4	Ledger	ALE	Structural Evidence	Advanced
AUD-05	Audit	Core Standard 7.5; Certification Framework 5	Ledger	ALE, EA	Structural Evidence	Advanced
AUD-RETAIN	Audit	Core Standard 7.6	Ledger	ALE	Governance Support	Foundational

2.X.6 Certification and Cryptographic Controls

TABLE 2.X.6 - CERTIFICATION AND CRYPTO DOMAIN CONTROL REGISTRY

CONTROL ID	CONTROL DOMAIN	CORE REFERENCE	ENFORCEMENT LAYER	EVIDENCE ARTIFACT	CONTROL TYPE	MATURITY
CERT-01	Audit	Certification Framework 1	Ledger	EA	Governance Support	Foundational
CERT-04	Audit	Certification Framework 4	Ledger	EA, Test records	Governance Support	Advanced
SEC-ALG-01	Crypto	SEC Profile 3.1 (AES-256-GCM)	Key Management	CT (encrypted), ALE	Deterministic Enforcement	Foundational
SEC-ALG-02	Crypto	SEC Profile 3.2 (SHA-3/SHA-256)	Key Management	ALE (hash chain)	Deterministic Enforcement	Foundational
SEC-ALG-03	Crypto	SEC Profile 3.3 (Ed25519)	Key Management	CT (signature)	Deterministic Enforcement	Foundational
SEC-ALG-AGILITY-01	Crypto	SEC Profile 3.4	Key Management	SEC Profile 3.4	Governance Support	High-Assurance
SEC-KEY-01	Crypto	SEC Profile 5.1 (Key storage)	Key Management	SEC Profile 5	Deterministic Enforcement	Foundational
SEC-KEY-02	Crypto	SEC Profile 5.2 (Prohibited patterns)	Key Management	SEC Profile 5	Deterministic Enforcement	Advanced
SEC-KEY-03	Crypto	SEC Profile 5.3 (FIPS 140-2 L3, dual-control)	Key Management	SEC Profile 5	Deterministic Enforcement	High-Assurance
SEC-NET-01	Crypto	SEC Profile 10.1 (TLS 1.3)	Key Management	SEC Profile 10	Deterministic Enforcement	Foundational
SEC-NET-PFS	Crypto	SEC Profile 10.2 (ECDHE)	Key Management	SEC Profile 10	Deterministic Enforcement	Advanced

2.X.7 Lifecycle Domain Controls

TABLE 2.X.7 - LIFECYCLE DOMAIN CONTROL REGISTRY

CONTROL ID	CONTROL DOMAIN	CORE REFERENCE	ENFORCEMENT LAYER	EVIDENCE ARTIFACT	CONTROL TYPE	MATURITY
LER	Lifecycle	Core Standard 9; Reference Architecture 5	Ledger	LER	Structural Evidence	Foundational
CERT-REVIEW	Lifecycle	Certification Framework 7	Ledger	EA	Governance Support	Advanced

SECTION 3

GDPR Technical Alignment

The General Data Protection Regulation (Regulation 2016/679) establishes the foundational legal framework for personal data processing in the European Union. QODIQA's deterministic consent enforcement constructs correspond structurally to several GDPR obligations, particularly those requiring demonstrable consent, purpose limitation, and records of processing. The following tables map QODIQA controls to GDPR Articles with explicit coverage classification and evidence artifact identification.

> SCOPE LIMITATION
>
> GDPR encompasses legal basis determination, data subject rights administration, Data Protection Officer obligations, international transfer mechanisms, and supervisory authority interactions - all of which are outside QODIQA's technical scope. This section addresses only those GDPR obligations where QODIQA's runtime enforcement constructs provide structural technical support.

3.1 Article 5 - Principles Relating to Processing

TABLE 3.1 - GDPR ARTICLE 5 ALIGNMENT

ARTICLE 5 PRINCIPLE	REQUIREMENT SUMMARY	QODIQA CONTROL(S)	ENFORCEMENT MECHANISM	EVIDENCE ARTIFACT	COVERAGE
Art. 5(1)(a) - Lawfulness, Fairness, Transparency	Processing must be lawful, fair, and transparent to the data subject	CT-01, CT-02, CT-03	Consent token records declared legal basis; SUP boundary enforces processing only when valid token present	Consent Token (CT), Audit Ledger Entry (ALE)	PARTIAL
Art. 5(1)(b) - Purpose Limitation	Data collected for specified, explicit, and legitimate purposes; not processed further in incompatible manner	PL-01, PL-02, PL-03, ENF-07	Purpose hash encoded in consent token; SUP evaluates purpose binding at every invocation; processing blocked on mismatch	Purpose Binding Record (PBR), ALE	DIRECT
Art. 5(1)(c) - Data Minimisation	Data adequate, relevant, and limited to what is necessary	Outside enforcement boundary	QODIQA does not govern data collection volume or field selection	N/A	OUT OF SCOPE
Art. 5(1)(d) - Accuracy	Data accurate and kept up to date	Outside enforcement boundary	QODIQA does not govern data quality	N/A	OUT OF SCOPE
Art. 5(1)(e) - Storage Limitation	Data retained no longer than necessary	CT-EXPIRY, ENF-RETIRE	Consent token expiry enforced at SUP; processing blocked on expired token; LER records retirement transitions	CT (expiry field), LER	PARTIAL
Art. 5(1)(f) - Integrity and Confidentiality	Appropriate security of personal data	SEC-ALG-01 through SEC-ALG-03, SEC-KEY-01 through SEC-KEY-03	Cryptographic integrity enforcement on all consent tokens and audit entries; Ed25519 / AES-256-GCM enforcement	CT (signature), ALE (hash chain)	STRUCTURAL
Art. 5(2) - Accountability	Controller must demonstrate compliance with principles	AUD-01 through AUD-05, CERT-01	Append-only audit ledger with cryptographic tamper evidence; enforcement attestation generation	ALE, Enforcement Attestation (EA)	STRUCTURAL

3.2 Article 6 - Lawfulness of Processing

TABLE 3.2 - GDPR ARTICLE 6 ALIGNMENT

ARTICLE 6 BASIS	REQUIREMENT SUMMARY	QODIQA CONTROL(S)	COVERAGE	RESIDUAL SCOPE
Art. 6(1)(a) - Consent	Data subject has given consent for one or more specific purposes	CT-01, CT-02, CT-03, PL-01	DIRECT	Consent validity under Art. 7 (freely given, specific, informed, unambiguous) is a legal determination; QODIQA enforces the token, not consent quality
Art. 6(1)(b) - Contract	Processing necessary for contract performance	CT legal-basis field (informational)	STRUCTURAL	QODIQA can encode basis type in token; enforcement of contractual necessity is outside scope
Art. 6(1)(c) - Legal Obligation	Processing necessary for legal obligation compliance	CT legal-basis field (informational)	STRUCTURAL	Legal obligation determination is entirely outside enforcement boundary
Art. 6(1)(e-f) - Public / Legitimate Interest	Processing for public task or legitimate interests	Outside enforcement boundary	OUT OF SCOPE	Balancing test required; cannot be technically enforced at runtime

3.3 Article 7 - Conditions for Consent

TABLE 3.3 - GDPR ARTICLE 7 ALIGNMENT

ART. 7 REQUIREMENT	QODIQA CONTROL(S)	ENFORCEMENT MECHANISM	EVIDENCE ARTIFACT	COVERAGE
Art. 7(1) - Demonstrability of consent	CT-01, AUD-02, CERT-01	Cryptographically signed consent token with subject binding; append-only ledger entry at consent capture time; enforcement attestation available for audit	CT, ALE, EA	DIRECT
Art. 7(2) - Intelligibility of consent request	Outside enforcement boundary	UX and presentation layer - not addressed by QODIQA	N/A	OUT OF SCOPE
Art. 7(3) - Right to withdraw consent	REV-01, REV-02, REV-03, ENF-REVOKE	Revocation propagates to registry within defined SLA; SUP consults registry at each invocation; processing blocked on revoked token; revocation timestamp recorded	RR, ALE	DIRECT

ART. 7 REQUIREMENT	QODIQA CONTROL(S)	ENFORCEMENT MECHANISM	EVIDENCE ARTIFACT	COVERAGE
Art. 7(3) - Withdrawal as easy as giving consent	Outside enforcement boundary	UX parity is a product design requirement; QODIQA enforces downstream effect of withdrawal	N/A	OUT OF SCOPE
Art. 7(4) - Conditioning of consent	Outside enforcement boundary	Legal assessment of conditionality; not deterministic at runtime	N/A	OUT OF SCOPE

3.4 Articles 13–14 - Transparency Information

TABLE 3.4 - GDPR ARTICLES 13-14 ALIGNMENT

REQUIREMENT	QODIQA CONTROL(S)	COVERAGE	RESIDUAL SCOPE
Provision of processing information to data subject	CT (purpose scope field, informational)	OUT OF SCOPE	Privacy notice delivery is outside enforcement boundary; purpose field in CT supports record of declared purpose only
Documentation of purposes at point of collection	PBR-01, CT-02	STRUCTURAL	Purpose Binding Record documents purpose scope at consent time; legal sufficiency of purpose description requires separate determination

3.5 Article 25 - Data Protection by Design and Default

TABLE 3.5 - GDPR ARTICLE 25 ALIGNMENT

ART. 25 REQUIREMENT	QODIQA CONTROL(S)	ENFORCEMENT MECHANISM	EVIDENCE ARTIFACT	COVERAGE
Art. 25(1) - Data protection by design	ENF-01 through ENF-10, CT-01 through CT-03, SEC-ALG-01	QODIQA enforces consent as a structural prerequisite to AI execution; default posture is deny; cryptographic integrity built into architecture	ALE (deny events), CT (design artifact)	DIRECT
Art. 25(2) - Data protection by default	ENF-DEFAULT-DENY, ENF-01	SUP boundary operates default-deny; processing requires valid, non-expired, non-revoked consent token; no processing on ambiguous or absent consent	ALE (blocked invocations)	DIRECT

ART. 25 REQUIREMENT	QODIQA CONTROL(S)	ENFORCEMENT MECHANISM	EVIDENCE ARTIFACT	COVERAGE
Technical and organisational measures for processing principles	All QODIQA enforcement controls	QODIQA constitutes a technical measure; organisational measures are outside scope	EA	PARTIAL

3.6 Article 30 - Records of Processing Activities

TABLE 3.6 - GDPR ARTICLE 30 ALIGNMENT

ART. 30 REQUIREMENT	QODIQA CONTROL(S)	EVIDENCE ARTIFACT	COVERAGE	RESIDUAL SCOPE
Name and contact details of controller	Outside enforcement boundary	N/A	OUT OF SCOPE	Organisational record maintenance
Purposes of processing	PBR-01, CT-02	PBR, CT	STRUCTURAL	QODIQA purpose records support ROPA population; full ROPA maintenance is organisational
Categories of data subjects and personal data	Outside enforcement boundary	N/A	OUT OF SCOPE	Data inventory and classification outside QODIQA scope
Recipients and transfers	Outside enforcement boundary	N/A	OUT OF SCOPE	Transfer governance outside scope
Time limits for erasure	CT-EXPIRY, LER	CT (expiry), LER	STRUCTURAL	Expiry records support ROPA; erasure execution is outside enforcement boundary
Security measures description	All SEC controls	EA	STRUCTURAL	QODIQA security controls documented in SEC Profile; compliance assessment separate

3.7 Article 32 - Security of Processing

TABLE 3.7 - GDPR ARTICLE 32 ALIGNMENT

ART. 32 REQUIREMENT	QODIQA CONTROL(S)	ENFORCEMENT MECHANISM	EVIDENCE ARTIFACT	COVERAGE
Pseudonymisation and encryption	SEC-ALG-01 (AES-256-GCM), SEC-ALG-03 (Ed25519)	Cryptographic enforcement on token issuance and verification; all consent artifacts encrypted in transit per SEC-NET-01	CT (encrypted fields), ALE	STRUCTURAL
Confidentiality, integrity, availability, resilience	SEC-KEY-01 through SEC-KEY-03, SEC-NET-01, AUD-01	HSM-backed key storage; TLS 1.3 enforcement; append-only audit structure; revocation SLA enforcement	EA, ALE	STRUCTURAL
Process for regular testing and evaluation	CERT-04, AUD-05	Certification framework mandates periodic enforcement attestation; audit ledger integrity verification	EA	STRUCTURAL

SECTION 4

EU AI Act Technical Alignment

The EU Artificial Intelligence Act (Regulation 2024/1689) establishes a risk-based regulatory framework for AI systems deployed in the European Union. QODIQA's runtime enforcement architecture aligns structurally with the Act's obligations relating to logging, record-keeping, human oversight support, and operational transparency for high-risk AI systems. The following tables address those obligations specifically.

SCOPE LIMITATION

The EU AI Act encompasses risk classification, conformity assessment, notified body certification, post-market monitoring, market surveillance, and AI system registration - all substantially outside QODIQA's technical scope. Mappings below address only the operational and logging obligations where QODIQA provides structural technical support. QODIQA does not constitute a conformity assessment instrument and does not substitute for EU AI Act certification processes.

ARTICLE 10 STRUCTURAL BRIDGE

QODIQA does not govern training data quality, dataset representativeness, or data governance obligations under EU AI Act Article 10. These requirements address the upstream model development phase, which precedes the SUP enforcement boundary. No coverage is claimed for Article 10. However, enforcement records generated at inference time - specifically Audit Ledger Entries documenting each invocation context, purpose scope, and consent state - may provide traceability artifacts relevant to post-deployment risk analysis under Articles 72 and 73. The utility of such artifacts in post-deployment contexts is informational; it does not extend QODIQA's coverage classification for Article 10 and does not reduce the obligation for independent compliance with Article 10 requirements.

4.1 Article 9 - Risk Management System

TABLE 4.1 - EU AI ACT ARTICLE 9 ALIGNMENT

ART. 9 OBLIGATION	QODIQA CONTROL(S)	ENFORCEMENT MECHANISM	EVIDENCE ARTIFACT	COVERAGE
Risk management system for high-risk AI - establishment and maintenance	ENF-01 through ENF-10; CERT-01	QODIQA provides a runtime enforcement control that reduces consent-related risk surface; enforcement attestation documents control operation	EA	PARTIAL
Risk identification and analysis	Outside enforcement boundary	Risk assessment is an organisational and analytical function; QODIQA provides enforcement, not risk analysis	N/A	OUT OF SCOPE
Residual risk evaluation and mitigation measures	SEC Profile - 14 Non-Coverage Disclosure	QODIQA's explicit non-coverage disclosure supports residual risk documentation	SEC Profile 14	STRUCTURAL

4.2 Article 12 - Record-Keeping and Logging

TABLE 4.2 - EU AI ACT ARTICLE 12 ALIGNMENT

ART. 12 REQUIREMENT	QODIQA CONTROL(S)	ENFORCEMENT MECHANISM	EVIDENCE ARTIFACT	COVERAGE
Art. 12(1) - Automatic logging capability	AUD-01, AUD-02, AUD-03	Every invocation at SUP boundary generates an append-only audit ledger entry; logging is non-bypassable at enforcement layer	ALE (all enforcement events)	DIRECT

ART. 12 REQUIREMENT	QODIQA CONTROL(S)	ENFORCEMENT MECHANISM	EVIDENCE ARTIFACT	COVERAGE
Art. 12(2) - Traceability across lifetime	AUD-01 through AUD-05, LER	Hash-chained audit ledger provides tamper-evident traceability; lifecycle event records cover training through retirement transitions	ALE, LER	DIRECT
Art. 12(3) - Log period retention	AUD-RETAIN	Audit ledger retention policy configurable per deployment profile; tamper evidence maintained throughout retention period	ALE	DIRECT
Art. 12(4) - High-risk: logging of each use period	AUD-02, ENF-SESSION	Session-level enforcement records generated per invocation sequence; consent token validity verified and recorded at session initiation	ALE (session records)	DIRECT

4.3 Article 13 - Transparency and Provision of Information

TABLE 4.3 - EU AI ACT ARTICLE 13 ALIGNMENT

ART. 13 REQUIREMENT	QODIQA CONTROL(S)	COVERAGE	RESIDUAL SCOPE
System designed to be sufficiently transparent	CT (purpose encoding), ALE (decision records)	STRUCTURAL	Transparency to deployers and users via QODIQA audit logs; model transparency is outside scope
Instructions for use - capabilities, limitations, human oversight	Outside enforcement boundary	OUT OF SCOPE	Documentation obligation; QODIQA does not produce AI system instructions for use
Disclosure when interacting with AI system	Outside enforcement boundary	OUT OF SCOPE	User notification is a product/UX layer obligation

4.4 Article 14 - Human Oversight

TABLE 4.4 - EU AI ACT ARTICLE 14 ALIGNMENT

ART. 14 REQUIREMENT	QODIQA CONTROL(S)	COVERAGE	RESIDUAL SCOPE
Oversight measures enabling human intervention	ENF-HALT, REV-01	PARTIAL	QODIQA supports consent revocation as an enforcement-level halt mechanism; broader human oversight interfaces are outside scope

ART. 14 REQUIREMENT	QODIQA CONTROL(S)	COVERAGE	RESIDUAL SCOPE
Ability to override AI system output	Outside enforcement boundary	OUT OF SCOPE	Output override is an AI system interface obligation; QODIQA operates at consent enforcement, not output layer
Monitoring of operation for anomalies and malfunctions	AUD-04, AUD-05	STRUCTURAL	QODIQA audit ledger enables anomaly detection at enforcement layer; broader system monitoring is outside scope

4.5 Article 17 - Quality Management System

TABLE 4.5 - EU AI ACT ARTICLE 17 ALIGNMENT

ART. 17 REQUIREMENT	QODIQA CONTROL(S)	COVERAGE	RESIDUAL SCOPE
QMS - strategies, techniques for achieving compliance	CERT-01 through CERT-04	STRUCTURAL	Certification framework provides structured compliance assessment for QODIQA controls; broader QMS is organisational
QMS - procedures for design, development, and monitoring	LER, Implementation Playbook	STRUCTURAL	QODIQA lifecycle controls provide evidence for QMS documentation; process governance outside scope
Record-keeping for QMS	AUD-01 through AUD-05, EA	STRUCTURAL	QODIQA's enforcement records support QMS documentation requirements

4.6 Article 26 - Deployers' Obligations

TABLE 4.6 - EU AI ACT ARTICLE 26 ALIGNMENT

ART. 26 OBLIGATION	QODIQA CONTROL(S)	COVERAGE	RESIDUAL SCOPE
Appropriate use within intended purpose	PL-01 through PL-03	DIRECT	Purpose-bound enforcement at runtime; contextual hash verification prevents out-of-scope use
Human oversight assignment	Outside enforcement boundary	OUT OF SCOPE	Personnel and governance assignment is organisational

ART. 26 OBLIGATION	QODIQA CONTROL(S)	COVERAGE	RESIDUAL SCOPE
Monitoring for risks to health, safety, or fundamental rights	AUD-04	STRUCTURAL	QODIQA enforcement anomaly logs support monitoring; risk content evaluation outside scope
Keeping logs as per Art. 12 for the required period	AUD-01 through AUD-05, AUD-RETAIN	DIRECT	QODIQA audit ledger satisfies technical logging obligation; organisational retention policy configuration required

SECTION 5

NIST AI Risk Management Framework Alignment

The NIST AI Risk Management Framework (AI RMF 1.0, January 2023) provides a structured approach to managing risks associated with AI systems across four core functions: **Govern, Map, Measure, and Manage**. QODIQA's deterministic runtime enforcement contributes to risk traceability, control effectiveness evidence, and operational accountability within AI RMF-aligned programs.

> **SCOPE LIMITATION**
>
> The NIST AI RMF is a voluntary framework intended for organisational use. Alignment with the AI RMF does not constitute NIST certification or endorsement. QODIQA's contribution is technical - providing enforcement controls and evidence artifacts that support the Govern, Measure, and Manage functions. Organisational AI governance programs, risk culture, and stakeholder engagement (which constitute the majority of Govern and Map function requirements) are outside QODIQA's technical scope.

5.1 GOVERN Function

TABLE 5.1 - NIST AI RMF GOVERN ALIGNMENT

GOVERN CATEGORY	GOVERN SUBCATEGORY	QODIQA CONTROL(S)	CONTRIBUTION	COVERAGE
GV-1: Policies, processes, procedures	GV-1.1 - AI risk management policies	CERT-01, Implementation Playbook 1	QODIQA certification framework provides policy-level enforcement documentation	STRUCTURAL
GV-1: Policies	GV-1.2 - Accountability for AI risk	AUD-01 through AUD-05, EA	Append-only audit ledger with enforcement attestation provides technical accountability evidence	STRUCTURAL
GV-4: Organisational teams	GV-4.1 - Roles and responsibilities	Outside enforcement boundary	Organisational role assignment	OUT OF SCOPE
GV-6: Policies for third-party risk	GV-6.1 - Third-party AI risk management	ENF-THIRD-PARTY, CT-SCOPE	QODIQA enforces consent boundary at third-party AI invocation; purpose binding prevents scope creep across integrations	PARTIAL

5.2 MAP Function

TABLE 5.2 - NIST AI RMF MAP ALIGNMENT

MAP CATEGORY	MAP SUBCATEGORY	QODIQA CONTROL(S)	CONTRIBUTION	COVERAGE
MP-2: Scientific and established methods	MP-2.3 - AI system context and scope	CT (purpose scope), PBR	Purpose Binding Record documents processing context at consent grant; supports AI system context mapping	STRUCTURAL
MP-3: AI risks and benefits	MP-3.4 - Risks to data subjects	PL-01, ENF-DEFAULT-DENY, REV-01	Default-deny posture and revocation enforcement reduce runtime risk to data subjects	PARTIAL
MP-4: Risk identification	MP-4.1 - Categorisation of risks	Outside enforcement boundary	Risk categorisation is analytical; QODIQA provides enforcement, not risk analysis	OUT OF SCOPE
MP-5: Impacts on individuals	MP-5.2 - Privacy risks	CT-01, PL-01, REV-01	Consent enforcement and withdrawal mechanisms directly address runtime privacy risk to individuals	DIRECT

5.3 MEASURE Function

TABLE 5.3 - NIST AI RMF MEASURE ALIGNMENT

MEASURE CATEGORY	SUBCATEGORY	QODIQA CONTROL(S)	CONTRIBUTION	COVERAGE
MS-1: AI risk measurement methods	MS-1.1 - Identified and documented AI risks	SEC Profile 14 (Non-Coverage Disclosure)	Explicit non-coverage disclosure provides documented residual risk boundary	STRUCTURAL
MS-2: AI risk evaluation	MS-2.5 - Testing conducted in realistic conditions	CERT-04, ENF-TEST	Certification framework mandates enforcement testing under operational conditions; test records generated	STRUCTURAL
MS-3: AI system performance	MS-3.3 - Effectiveness of risk controls	AUD-05, EA	Enforcement attestation provides measurable evidence of control operation; blocked-invocation rates quantifiable from ALE	DIRECT
MS-4: Feedback and incident capture	MS-4.1 - Collection of incident feedback	AUD-03, AUD-04	Enforcement anomaly records support incident identification; consent failure events captured in ALE	STRUCTURAL

5.4 MANAGE Function

TABLE 5.4 - NIST AI RMF MANAGE ALIGNMENT

MANAGE CATEGORY	SUBCATEGORY	QODIQA CONTROL(S)	CONTRIBUTION	COVERAGE
MG-1: Risk treatment	MG-1.3 - Responses to identified risks	REV-01 through REV-03, ENF-HALT	Revocation and enforcement halt mechanisms provide technical risk response at runtime	DIRECT
MG-2: Risk treatment monitoring	MG-2.2 - Effectiveness of risk treatments	AUD-05, EA	Enforcement attestation and audit ledger analysis support ongoing treatment effectiveness evaluation	STRUCTURAL
MG-3: Risk management improvement	MG-3.1 - Identified risk improvements	SEC-ALG-AGILITY-01, CERT-REVIEW	Algorithm agility framework and certification review cycle support continuous improvement	STRUCTURAL
MG-4: Residual risk	MG-4.1 - Residual risks documented	Section 9 Non-Coverage Disclosure (this document)	Explicit residual risk and non-coverage disclosure provides documented residual risk register input	DIRECT

SECTION 6

ISO/IEC 27001 Control Family Alignment

ISO/IEC 27001:2022 (Information Security Management Systems) provides a framework for establishing, implementing, maintaining, and continually improving an information security management system (ISMS). QODIQA's security and enforcement controls operate as a runtime enforcement layer supplementing, but not replacing, an organisation's ISMS. The following tables map QODIQA to relevant Annex A control families.

> **SCOPE LIMITATION**
>
> QODIQA does not implement an ISMS. It provides a runtime enforcement control plane that contributes technical controls relevant to several ISO 27001 control families. ISO 27001 certification requires an organisation-wide management system, risk assessment process, statement of applicability, and independent audit - none of which QODIQA substitutes for. Controls below are informative mappings only.

6.1 A.5 - Organisational Controls (Selected)

TABLE 6.1 - ISO 27001:2022 ANNEX A.5 ALIGNMENT

ISO 27001 CONTROL	CONTROL TITLE	QODIQA CONTROL(S)	COVERAGE	QODIQA CONTRIBUTION
A.5.18	Access rights	CT-01, CT-SCOPE, ENF-01	PARTIAL	Consent token scopes function as access rights in AI processing context; SUP enforces scope boundaries deterministically
A.5.20	Addressing security within supplier agreements	ENF-THIRD-PARTY	STRUCTURAL	QODIQA's enforcement at third-party AI invocation boundary supports supplier security control documentation
A.5.33	Protection of records	AUD-01, AUD-02, AUD-RETAIN	DIRECT	Append-only, cryptographically tamper-evident audit ledger provides protected records with configurable retention
A.5.36	Compliance with information security policies	CERT-01, EA	STRUCTURAL	Enforcement attestation provides structured evidence of consent enforcement policy compliance

6.2 A.8 - Technological Controls

TABLE 6.2 - ISO 27001:2022 ANNEX A.8 ALIGNMENT

ISO 27001 CONTROL	CONTROL TITLE	QODIQA CONTROL(S)	COVERAGE	QODIQA CONTRIBUTION
A.8.2	Privileged access rights	SEC-KEY-03 (dual-control)	PARTIAL	Dual-control key activation for High-Assurance profile enforces separation of duties at cryptographic layer
A.8.15	Logging	AUD-01 through AUD-05	DIRECT	Non-bypassable audit logging of all enforcement events with hash-chain integrity; tamper-evident retention
A.8.16	Monitoring activities	AUD-04, AUD-05	STRUCTURAL	Audit ledger provides monitoring data for enforcement anomaly detection; alerting on consent failures
A.8.24	Use of cryptography	SEC-ALG-01 through SEC-ALG-03, SEC-ALG-AGILITY-01	DIRECT	Normative cryptographic algorithm requirements; agility framework; FIPS-aligned algorithm selection; HSM key storage
A.8.25	Secure development life cycle	LER, Implementation Playbook 4	STRUCTURAL	QODIQA lifecycle event records and playbook integration guidance support SDL documentation
A.8.28	Secure coding	Outside enforcement boundary	OUT OF SCOPE	Application code security is outside QODIQA scope; QODIQA enforces consent boundary, not code quality
A.8.33	Test information	CERT-04, ENF-TEST	STRUCTURAL	Certification framework test procedures provide structured test evidence documentation

> QODIQA supplements an ISMS by adding a deterministic runtime enforcement layer at the AI execution boundary. It does not replace ISMS processes for asset management, physical security, supplier management, business continuity, or human resource security - all of which remain organisational responsibilities outside QODIQA's technical scope.

SECTION 7

ISO/IEC 42001 AI Management System Alignment

ISO/IEC 42001:2023 (AI Management Systems) provides requirements for establishing, implementing, maintaining, and continually improving an artificial intelligence management system (AIMS). QODIQA operates as a technical control component within the operational layer of an ISO 42001-aligned AIMS, providing runtime enforcement, consent governance, and audit evidence generation.

> **SCOPE LIMITATION**
>
> ISO/IEC 42001 encompasses organisational context, leadership commitment, planning, support, operation, performance evaluation, and improvement - representing a full management system scope. QODIQA's contribution is specifically at the operational control layer (Clause 8) and performance evaluation layer (Clause 9). All other AIMS requirements remain outside QODIQA's technical scope and must be addressed through organisational governance structures.

7.1 Clause 6 - Planning

TABLE 7.1 - ISO 42001 CLAUSE 6 ALIGNMENT

ISO 42001 CLAUSE	REQUIREMENT	QODIQA CONTROL(S)	COVERAGE	QODIQA CONTRIBUTION
6.1.1	Actions to address AI risks and opportunities	ENF-01 through ENF-10; SEC Profile 14	PARTIAL	QODIQA defines technical enforcement actions for consent risk; non-coverage disclosure identifies residual risk requiring organisational treatment
6.1.2	AI risk assessment	Outside enforcement boundary	OUT OF SCOPE	Risk assessment is an analytical and organisational function

7.2 Clause 8 - Operation

TABLE 7.2 - ISO 42001 CLAUSE 8 ALIGNMENT

ISO 42001 CLAUSE	REQUIREMENT	QODIQA CONTROL(S)	COVERAGE	QODIQA CONTRIBUTION
8.1	Operational planning and control	ENF-01 through ENF-10; Implementation Playbook 3	DIRECT	QODIQA provides the runtime operational control plane for consent enforcement; playbook defines integration procedures
8.3	AI system lifecycle management	LER, ENF-RETIRE	PARTIAL	Lifecycle event records cover training, deployment, and retirement transitions; training-phase governance outside QODIQA scope
8.4	Data governance	CT-01, PBR-01, REV-01	PARTIAL	Consent token and purpose binding records contribute to data governance documentation; data quality and inventory are outside scope
8.5	Responsible development	Outside enforcement boundary	OUT OF SCOPE	Development practices, impact assessments, and responsible AI considerations are organisational
8.6	Responsible deployment and operation	ENF-01 through ENF-10; PL-01 through PL-03	DIRECT	QODIQA's deterministic enforcement constitutes a core technical component of responsible deployment; purpose gating prevents out-of-scope operation

7.3 Clause 9 - Performance Evaluation

TABLE 7.3 - ISO 42001 CLAUSE 9 ALIGNMENT

ISO 42001 CLAUSE	REQUIREMENT	QODIQA CONTROL(S)	COVERAGE	QODIQA CONTRIBUTION
9.1	Monitoring, measurement, analysis, and evaluation	AUD-01 through AUD-05, EA	DIRECT	Audit ledger provides quantitative enforcement performance data; enforcement attestation supports periodic measurement
9.2	Internal audit	CERT-01 through CERT-04	STRUCTURAL	QODIQA certification framework provides structured audit evidence and assessment procedures for internal audit use

ISO 42001 CLAUSE	REQUIREMENT	QODIQA CONTROL(S)	COVERAGE	QODIQA CONTRIBUTION
9.3	Management review	EA (periodic reporting)	STRUCTURAL	Enforcement attestation reports provide structured input to AIMS management review; broader review governance is organisational

7.4 Annex A (ISO 42001) - AI-Specific Controls

TABLE 7.4 - ISO 42001 ANNEX A SELECTED CONTROLS

ANNEX A CONTROL	TITLE	QODIQA CONTROL(S)	COVERAGE
A.2.2	AI system processes - data for AI	CT-01, PBR-01	STRUCTURAL
A.2.4	Documentation	ALE, EA, LER	STRUCTURAL
A.3.3	Governance of AI - objectives	CERT-01	STRUCTURAL
A.4.2	Human oversight of AI systems	REV-01, ENF-HALT	PARTIAL
A.6.1	AI system logging	AUD-01 through AUD-05	DIRECT
A.7.4	Security of AI system	SEC Profile - all sections	DIRECT

SECTION 8

OECD AI Principles Structural Support

The OECD AI Principles (OECD/LEGAL/0449, updated 2024) provide non-binding guidance on trustworthy AI across five values-based principles. Mappings below identify where QODIQA's technical enforcement constructs provide structural support for the technical dimensions of these principles. Philosophical, policy, and societal dimensions of the principles are outside QODIQA's technical scope and are not addressed here.

8.1 Principle-Level Alignment

TABLE 8.1 - OECD AI PRINCIPLES TECHNICAL SUPPORT

OECD PRINCIPLE	TECHNICAL DIMENSION	QODIQA CONTROL(S)	TECHNICAL CONTRIBUTION	COVERAGE
1.1 - Inclusive Growth, Sustainable Development, Well-being	AI benefits distributed with respect to affected persons' interests	Outside enforcement boundary	N/A	OUT OF SCOPE
1.2 - Human-centred Values and Fairness	Respect for rule of law; privacy; freedom from manipulation; human dignity	CT-01, PL-01, REV-01, ENF-DEFAULT-DENY	Consent enforcement and revocation support privacy protection at runtime; default-deny posture prevents processing without valid consent; purpose binding prevents manipulation through scope creep	PARTIAL
1.3 - Transparency and Explainability	Actors should be transparent about AI systems; provide meaningful information	ALE, CT, EA	Audit ledger provides verifiable, replayable record of enforcement decisions; consent token encodes declared purpose; enforcement attestation provides structured transparency artifact	STRUCTURAL
1.4 - Robustness, Security, and Safety	AI systems should be robust, secure, safe throughout lifetime	SEC Profile (all sections); LER; ENF-DEFAULT-DENY	Cryptographic integrity enforcement; HSM-backed key management; hash-chained audit integrity; default-deny posture; lifecycle event tracking through retirement	DIRECT
1.5 - Accountability	Actors accountable for proper functioning of AI systems; mechanisms for redress	AUD-01 through AUD-05; EA; CERT-01 through CERT-04	Append-only tamper-evident audit ledger provides accountability evidence; enforcement attestation documents control operation; certification framework supports structured accountability review	DIRECT

> The OECD AI Principles are **policy-level instruments** requiring organisational commitment, governance structures, and societal engagement that substantially exceed the scope of any technical control. QODIQA's structural contributions are confined to the technical dimensions identified above. No claim is made that deploying QODIQA constitutes adherence to or implementation of the OECD AI Principles as a whole.

SECTION 9

Residual Risk and Non-Coverage Disclosure

This section provides a definitive, structured disclosure of areas outside QODIQA's enforcement boundary. These disclosures are intended to prevent overclaiming in compliance contexts and to ensure that organizations understand where complementary controls are required. Items listed here are not gaps or deficiencies - they reflect the deliberate and appropriate boundary of a runtime consent enforcement system.

NON-COV-DISCLOSURE **Definitive Non-Coverage Declaration** ● NORMATIVE

QODIQA provides deterministic runtime consent enforcement. It generates structured, replayable evidence. It does not provide, and does not claim to provide, coverage for any item listed in Section 9.1. Organizations MUST NOT represent QODIQA deployment as satisfying the non-covered areas listed herein without independent complementary controls and qualified legal or compliance assessment.

9.1 Explicit Non-Coverage Table

TABLE 9.1 - DEFINITIVE NON-COVERAGE DISCLOSURE

AREA	DESCRIPTION	WHY OUTSIDE SCOPE	WHERE ADDRESSED
Model Training Governance	Data provenance, training data consent, training-phase data handling, model card generation	QODIQA enforces at inference/use-time SUP boundary; training phase precedes enforcement layer	ML governance frameworks; data management systems; EU AI Act Art. 10 (training data)
Algorithmic Fairness and Bias	Evaluation of model outputs for demographic bias, disparate impact, or fairness metrics	QODIQA enforces consent boundary, not output content or fairness characteristics	Fairness evaluation frameworks; AI impact assessments; EU AI Act Art. 10(2)(f)

AREA	DESCRIPTION	WHY OUTSIDE SCOPE	WHERE ADDRESSED
Content Moderation	Evaluation of AI-generated output for harmful, illegal, or policy-violating content	QODIQA operates at enforcement layer upstream of output generation; does not inspect output content	Output moderation systems; trust and safety controls; DSA compliance infrastructure
Legal Validity of Consent	Determination of whether a given consent satisfies GDPR Art. 7 requirements (freely given, specific, informed, unambiguous)	Legal validity is a jurisprudential determination; QODIQA enforces the presence and integrity of a consent token, not its legal quality	Legal counsel; DPO assessment; supervisory authority guidance
Data Subject Rights Administration	Subject access requests, right to erasure (Art. 17), portability (Art. 20), objection (Art. 21)	QODIQA enforces processing boundaries; data subject rights administration requires organisational processes, not only enforcement controls	DSR management platforms; legal operations; DPO office
Privacy Notice and UX Delivery	Provision of privacy information to data subjects (Art. 13–14); intelligibility of consent interface	User interface and notice delivery are product/design layer obligations outside enforcement boundary	Product design; privacy engineering; UX governance
International Data Transfers	GDPR Chapter V - SCCs, adequacy decisions, BCRs	Transfer mechanisms are legal instruments; QODIQA does not govern data movement or transfer basis	Legal counsel; EDPB guidance; transfer impact assessments
Business Process Governance	Organisational AI governance structures, ethics boards, decision authority, AI policy	Governance is an organisational and managerial function; QODIQA provides a technical control, not governance structures	AI governance frameworks; ISO 42001 (organisational clauses); board-level AI governance
ISMS Completeness	Full ISO 27001 ISMS scope including asset management, physical security, HR security, BCP	QODIQA supplements selected control families; it does not implement or certify an ISMS	Certified ISMS programme; ISO 27001 audit
Conformity Assessment and Certification	EU AI Act conformity assessment; CE marking; notified body engagement	Conformity assessment is a legal and regulatory process conducted by qualified bodies; QODIQA provides technical controls, not certification	Notified bodies; EU AI Act Chapters V–VI
Incident Notification to Authorities	GDPR Art. 33–34 breach notification; EU AI Act Art. 73 serious incident reporting	Notification obligations require organisational incident response processes beyond enforcement layer	Incident response plans; supervisory authority notification procedures
AI System Impact Assessment	DPIA (GDPR Art. 35); AI system fundamental rights impact assessment (EU AI Act)	Impact assessments require interdisciplinary analysis of systemic effects; QODIQA provides enforcement records as input, not assessment capability	DPIA process; DPO; fundamental rights teams

SECTION 10

Cross-Framework Comparative Matrix

This section consolidates the preceding section-by-section alignments into a single cross-framework matrix, presenting QODIQA enforcement controls against all six frameworks simultaneously. The matrix is intended for rapid cross-reference in enterprise risk and compliance review contexts. Coverage indicators follow the classification scheme defined in Section 2.3.

This section provides a consolidated cross-framework view of control alignment. It highlights where deterministic runtime enforcement provides direct coverage, where it supports evidentiary traceability, and where residual gaps remain outside the enforcement boundary.

EXECUTIVE SUMMARY - CROSS-FRAMEWORK COVERAGE

- → **Strong Coverage Domains:** runtime enforcement (deterministic gating), consent validation and revocation handling, audit logging and cryptographic traceability
- → **Partial Coverage Domains:** transparency and explainability support (evidence-level), risk management integration (indirect)
- → **Out of Scope Domains:** model training processes, fairness and bias mitigation, organizational governance structures

STRONG COVERAGE
RUNTIME ENFORCEMENT, CONSENT VALIDATION, AUDIT TRACEABILITY

→

PARTIAL COVERAGE
TRANSPARENCY SUPPORT, RISK MAPPING

→

OUT OF SCOPE
TRAINING, FAIRNESS, GOVERNANCE

10.1 Consolidated Control-to-Framework Matrix

TABLE 10.1 - QODIQA CROSS-FRAMEWORK ALIGNMENT SUMMARY

QODIQA CONTROL DOMAIN	KEY CONTROLS	GDPR	EU AI ACT	NIST AI RMF	ISO 27001	ISO 42001	OECD AI	EVIDENCE TYPE	PRIMARY RISK DOMAIN
Consent Token Enforcement	CT-01, CT-02, CT-03	DIRECT	PARTIAL	PARTIAL	PARTIAL	DIRECT	PARTIAL	CT, ALE	Consent Enforcement
Purpose Limitation Enforcement	PL-01, PL-02, PL-03	DIRECT	DIRECT	PARTIAL	PARTIAL	DIRECT	PARTIAL	PBR, CT, ALE	Privacy
Default-Deny Enforcement	ENF-DEFAULT-DENY, ENF-01	DIRECT	STRUCTURAL	PARTIAL	STRUCTURAL	DIRECT	PARTIAL	ALE (denied invocations)	Consent Enforcement
Revocation Enforcement	REV-01, REV-02, REV-03	DIRECT	PARTIAL	DIRECT	STRUCTURAL	PARTIAL	PARTIAL	RR, ALE	Consent Enforcement
Audit Ledger Integrity	AUD-01 through AUD-05	STRUCTURAL	DIRECT	DIRECT	DIRECT	DIRECT	DIRECT	ALE, EA	Traceability
Enforcement Attestation	CERT-01 through CERT-04	STRUCTURAL	STRUCTURAL	STRUCTURAL	STRUCTURAL	STRUCTURAL	STRUCTURAL	EA	Accountability
Cryptographic Algorithm Enforcement	SEC-ALG-01, SEC-ALG-02, SEC-ALG-03	STRUCTURAL	STRUCTURAL	STRUCTURAL	DIRECT	DIRECT	STRUCTURAL	CT (signature), ALE	Security
Cryptographic Agility	SEC-ALG-AGILITY-01	STRUCTURAL	STRUCTURAL	STRUCTURAL	DIRECT	STRUCTURAL	STRUCTURAL	SEC Profile 3.4	Security
Key Management	SEC-KEY-01, SEC-KEY-02, SEC-KEY-03	STRUCTURAL	STRUCTURAL	STRUCTURAL	DIRECT	DIRECT	STRUCTURAL	SEC Profile 5	Security

QODIQA CONTROL DOMAIN	KEY CONTROLS	GDPR	EU AI ACT	NIST AI RMF	ISO 27001	ISO 42001	OECD AI	EVIDENCE TYPE	PRIMARY RISK DOMAIN
Lifecycle Event Records	LER, ENF-RETIRE	STRUCTURAL	DIRECT	STRUCTURAL	STRUCTURAL	PARTIAL	STRUCTURAL	LER	Lifecycle Integrity
Purpose Binding Records	PBR-01, CT-02	STRUCTURAL	STRUCTURAL	STRUCTURAL	STRUCTURAL	STRUCTURAL	STRUCTURAL	PBR	Traceability
Transport Security	SEC-NET-01, SEC-NET-PFS	STRUCTURAL	STRUCTURAL	STRUCTURAL	DIRECT	STRUCTURAL	STRUCTURAL	SEC Profile 10	Security
Third-Party Enforcement	ENF-THIRD-PARTY	PARTIAL	PARTIAL	PARTIAL	STRUCTURAL	PARTIAL	STRUCTURAL	ALE, CT	Governance Support
Model Training / Fairness / Bias	N/A	OUT OF SCOPE	OUT OF SCOPE	OUT OF SCOPE	OUT OF SCOPE	OUT OF SCOPE	OUT OF SCOPE	See Section 9.1	N/A
Legal Consent Validity	N/A	OUT OF SCOPE	OUT OF SCOPE	OUT OF SCOPE	OUT OF SCOPE	OUT OF SCOPE	OUT OF SCOPE	See Section 9.1	N/A
Data Subject Rights Admin	N/A	OUT OF SCOPE	OUT OF SCOPE	OUT OF SCOPE	OUT OF SCOPE	OUT OF SCOPE	OUT OF SCOPE	See Section 9.1	N/A

10.2 Coverage Density Summary

TABLE 10.2 - COVERAGE DISTRIBUTION BY FRAMEWORK

FRAMEWORK	DIRECT COVERAGE AREAS	STRUCTURAL COVERAGE AREAS	PARTIAL COVERAGE AREAS	OUT OF SCOPE AREAS
GDPR	Art. 5(1)(b) purpose limitation; Art. 7(1) demonstrability; Art. 7(3) revocation; Art. 25(1)(2) privacy by design/default	Art. 5(1)(f) security; Art. 5(2) accountability; Art. 30 records; Art. 32 security measures	Art. 5(1)(a) lawfulness; Art. 5(1)(e) storage; Art. 6(1)(a) consent basis; Art. 25 organisational	Data minimisation; accuracy; rights admin; transfer mechanisms; notice delivery; DPIA

FRAMEWORK	DIRECT COVERAGE AREAS	STRUCTURAL COVERAGE AREAS	PARTIAL COVERAGE AREAS	OUT OF SCOPE AREAS
EU AI Act	Art. 12 logging (all sub-requirements); Art. 26(a) intended purpose; Art. 26 log retention	Art. 9 risk management (residual); Art. 13 transparency; Art. 17 QMS; Art. 14 monitoring	Art. 9 risk management; Art. 14 oversight	Conformity assessment; CE marking; model transparency; output oversight; bias evaluation
NIST AI RMF	MS-3.3 control effectiveness; MG-1.3 risk treatment; MG-4.1 residual risk; MP-5.2 privacy risks	GV-1.1 policies; GV-1.2 accountability; MS-1.1 risk measurement; MS-2.5 testing; MS-4.1 incidents; MG-2.2 monitoring	GV-6.1 third-party; MP-3.4 data subject risk; MP-2.3 system context	Organisational AI governance; risk categorisation; stakeholder engagement; explainability
ISO/IEC 27001	A.5.33 records protection; A.8.15 logging; A.8.24 cryptography	A.5.20 supplier; A.5.36 policy compliance; A.8.16 monitoring; A.8.25 SDL; A.8.33 test info	A.5.18 access rights; A.8.2 privileged access	ISMS scope (asset, physical, HR, BCP); all organisational ISMS clauses
ISO/IEC 42001	8.1 operational control; 8.6 responsible deployment; 9.1 monitoring; A.6.1 logging; A.7.4 security	6.1.1 risk actions; 9.2 internal audit; 9.3 management review; A.2.2 data; A.2.4 documentation; A.3.3 governance	8.3 lifecycle; 8.4 data governance; A.4.2 human oversight	Risk assessment; leadership; responsible development; organisational AIMS clauses
OECD AI Principles	1.4 Robustness/Security/Safety; 1.5 Accountability	1.3 Transparency/Explainability	1.2 Human-centred values	1.1 Inclusive growth; societal and policy dimensions of all principles

Deterministic runtime enforcement provides the highest level of control assurance at the point of system invocation. Coverage is strongest in areas where enforcement can be applied synchronously and cryptographically verified. Domains such as model training, fairness, and organizational governance remain outside this boundary and require complementary control frameworks.

SECTION 10.X

Regulatory Ambiguity Surface Reduction Analysis

10.X.1 Definition: Regulatory Ambiguity Surface

For the purposes of this analysis, **regulatory ambiguity surface** is defined as: *the set of execution-time states in which legal obligations depend on non-deterministic enforcement* - that is, states where processing may occur without verifiable consent token validation, purpose binding evaluation, or revocation registry consultation. In non-deterministic enforcement environments, the ambiguity surface is unbounded at runtime: any invocation may or may not have occurred with valid consent, and no cryptographically verifiable record distinguishes compliant from non-compliant invocations after the fact.

Deterministic enforcement - as implemented through the QODIQA SUP boundary invariant defined in Section 2.6 - reduces this ambiguity surface by converting probabilistic compliance postures into binary enforcement decisions with cryptographically bound, replayable evidence. The following table quantifies this reduction across identified risk vectors.

10.X.2 Risk Vector Analysis

TABLE 10.X.2 - AMBIGUITY SURFACE REDUCTION BY RISK VECTOR

RISK VECTOR	DEFINITION	NON-DETERMINISTIC ENVIRONMENT	DETERMINISTIC ENFORCEMENT IMPACT	RESIDUAL RISK
Silent Consent Drift	Processing continues under a consent scope that has materially narrowed since grant, without enforcement-layer detection	No runtime gate; scope evaluation performed at audit or complaint time, if at all; processing and non-compliance may be separated by months or years of invocation history	Purpose hash in CT evaluated against declared scope at every invocation; scope mismatch results in deterministic BLOCK; ALE entry generated for each blocked invocation with timestamp and failure reason; drift detectable in real time from ALE	Consent scope definition quality is upstream of enforcement boundary; QODIQA enforces the declared scope, not its legal adequacy; legal sufficiency determination remains outside scope

RISK VECTOR	DEFINITION	NON-DETERMINISTIC ENVIRONMENT	DETERMINISTIC ENFORCEMENT IMPACT	RESIDUAL RISK
Scope Creep	AI system invoked for purposes beyond declared consent scope, either deliberately or through feature expansion	Without enforcement gating, purpose scope expansion is governed only by internal policy controls; violations may be undetected until DPA investigation or data subject complaint	PL-01 through PL-03 enforce purpose binding at invocation time; new purpose requires new consent token with updated scope; invocations under undeclared purposes blocked at SUP boundary; every attempt generates ALE entry	Determination of whether a new use case constitutes a compatible purpose under GDPR Art. 5(1)(b) requires legal analysis outside QODIQA scope; QODIQA enforces the technical boundary, not the legal determination of compatibility
Revocation Lag	Processing continues after consent withdrawal due to propagation delays, registry synchronisation failures, or system unawareness of revocation event	In poll-based or batch-updated systems, revocation may take hours or days to propagate; processing during lag period constitutes processing without valid legal basis; no cryptographic evidence of when processing actually stopped	REV-01 through REV-03 enforce revocation propagation SLA; SUP consults registry at each invocation; revocation takes effect within defined SLA window; RR provides cryptographic timestamp of revocation and propagation confirmation; post-revocation invocations blocked and ALE-recorded	SLA window creates a bounded residual processing risk during propagation period; SLA duration is configurable by profile (Foundational / Advanced / High-Assurance) and must be assessed against applicable regulatory requirements in each jurisdiction
Undocumented Invocation	AI system invoked without any record of the invocation event, rendering post-hoc compliance review impossible	Logging dependent on application-layer implementation; logs may be mutable, deletable, incomplete, or absent; no cryptographic guarantee of completeness or tamper-evidence; logging gaps create irresolvable ambiguity in regulatory proceedings	AUD-01 through AUD-05 enforce non-bypassable, append-only, hash-chained logging of every enforcement decision at the SUP boundary; logging is architecturally pre-application-layer; cryptographic tamper evidence maintained throughout retention period; completeness verifiable through hash chain integrity check	QODIQA logs enforcement decisions, not application-layer invocation content; logs do not capture AI output, user interaction detail, or data processed; content-level audit remains outside QODIQA scope and requires complementary application-layer logging
Consent State Ambiguity at Audit Time	Inability to reconstruct the consent state operative at the time of any specific historical AI invocation	Without cryptographic binding, reconstructing historical consent state requires assembling evidence from multiple mutable systems; reconstruction may be contested, incomplete, or impossible; creates significant regulatory exposure in investigations and DPA proceedings	Each ALE entry records consent token state, purpose scope, expiry, revocation status, and cryptographic token identifier at invocation time; historical consent state for any recorded invocation is fully reconstructable from the ALE and the consent token archive; evidence is cryptographically verifiable and non-repudiable	Reconstruction capability depends on retention policy compliance (AUD-RETAIN); organisations must configure retention periods to meet applicable regulatory requirements; QODIQA does not determine applicable retention periods, which are a legal and regulatory analysis function

10.X.3 Ambiguity Surface Reduction Summary

> Deterministic enforcement does not eliminate regulatory risk. It **converts non-deterministic ambiguity into bounded, documented, cryptographically verifiable states**. The ambiguity surface is reduced to: (a) the SLA window during revocation propagation; (b) the legal sufficiency of consent scope definitions upstream of the enforcement boundary; and (c) content-layer audit gaps outside QODIQA's enforcement scope. All other vectors listed above are addressed within the SUP boundary invariant. No claim of regulatory compliance follows from this reduction analysis.

SECTION 11

Institutional Closing Statement

This matrix has mapped QODIQA's deterministic runtime enforcement controls to the requirements of six major AI and data governance frameworks. The mappings are structural and technical. They establish where QODIQA's enforcement architecture provides genuine, verifiable support for regulatory control objectives - and where it does not.

The matrix demonstrates that QODIQA's contribution is concentrated in three areas: **deterministic consent enforcement** at the AI execution boundary, **structured evidence generation** through tamper-evident audit ledgers and cryptographically bound artifacts, and **residual risk reduction** through default-deny posture and purpose-bound processing gates. These contributions are technically meaningful and can reduce the ambiguity surface at the execution boundary where compliance failures most frequently originate.

At the same time, the matrix confirms that QODIQA does not address - and does not claim to address - the substantial majority of requirements in each framework that require organisational governance, legal interpretation, human judgment,

or external certification. The non-coverage disclosures in Section 9 are as important as the coverage claims in Sections 3 through 8.

INSTITUTIONAL POSITION

QODIQA provides deterministic runtime enforcement at the AI execution boundary. It generates structured, replayable evidence through cryptographically bound audit artifacts. It reduces ambiguity at the execution boundary where consent obligations are most frequently violated. It does not replace legal compliance frameworks, organisational governance structures, or regulatory certification processes. It is a technical control plane that can structurally support them - with precisely defined scope and explicitly bounded non-coverage.

FINAL MANDATORY DISCLAIMER

- → Nothing in this document constitutes legal advice, regulatory guidance, compliance certification, or a warranty of any kind.
- → This document is a technical cross-reference instrument. Regulatory compliance determinations must be made by qualified legal counsel and certified compliance professionals with jurisdiction-specific expertise.
- → QODIQA makes no representations regarding the legal sufficiency of any deployment of QODIQA controls for any regulatory purpose.
- → All framework citations are for structural cross-reference only and do not imply endorsement, affiliation, or certification by any standards body, regulatory authority, or government agency.

DOCUMENT STATUS

Document Status and Classification

This document presents a technical control alignment matrix mapping QODIQA enforcement constructs to major AI and data governance frameworks. It is issued as a Technical Mapping Specification and is not a product specification, a commercial certification program, or a legal instrument.

QODIQA is not presented as a theoretical framework, but as an implementable infrastructure standard designed to introduce enforceable authorization, deterministic policy execution, runtime verifiability, and auditable proof mechanisms into AI-native systems.

The material contained herein is intended for:

- Enterprise risk teams and compliance architects
- Regulatory affairs specialists
- AI governance practitioners
- Security and governance officers
- Organizations building or operating autonomous AI systems

This matrix serves as a canonical reference for organizations evaluating the control coverage characteristics of QODIQA within their broader compliance posture. All regulatory framework citations are for structural cross-reference only.

This document should be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Implementation Playbook for Deterministic Runtime Consent Enforcement
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement
- QODIQA: Governance Charter for the QODIQA Standard Corpus

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dușescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-RAM-2026-001
TITLE	Regulatory Alignment Matrix for AI Governance Frameworks
SUBTITLE	Technical Control Alignment of QODIQA Enforcement Constructs to Major AI and Data Governance Frameworks
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Technical Mapping Specification
DOCUMENT STATUS	Normative - Technical Mapping Specification
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 29 PART V · CONTEXTUAL, ANALYTICAL AND PROCESS SUPPORT

Audit and Evidence Generation Model for Runtime Consent Enforcement

Deterministic audit principles, evidence generation architecture, and cryptographic integrity guarantees for verifiable AI consent enforcement.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-AUDIT-2026-001	1.0	2026	Technical Specification	Normative	NORMATIVE

ABSTRACT

Deterministic runtime consent enforcement depends, in equal measure, on the integrity of its execution path and on the completeness of the record that execution path produces. This document specifies the Audit and Evidence Generation Model for QODIQA: the architectural principles, event taxonomy, evidence structure, cryptographic integrity mechanisms, and operational constraints that together constitute a verifiable, tamper-evident record of every consent enforcement decision made by the system. The model defines what constitutes evidence within the QODIQA framework, how evidence is generated and serialized in a deterministic and replay-consistent manner, and how it is preserved in an append-only log architecture that supports independent verification. The document further addresses failure modes, privacy constraints, data minimization obligations, and the relationship between audit artifacts and the QODIQA certification framework. The audit and evidence model is a normative component of the QODIQA standard corpus and is required for any implementation claiming conformance.

SECTION 1

System Role of Audit and Evidence

Audit within the QODIQA architecture is not an ancillary reporting function. It is a first-class system responsibility that operates in parallel with enforcement. Every enforcement decision produces a corresponding evidence record. The two are inseparable: an enforcement decision without a corresponding evidence record is incomplete and does not satisfy the QODIQA determinism guarantee.

The audit subsystem transforms the internal state of each enforcement event into a durable, verifiable, and externally inspectable artifact. It converts ephemeral computation into permanent record. This transformation is the mechanism by which QODIQA transcends run-time enforcement and enters the domain of institutional accountability.

1.1 Architectural Position

The audit subsystem occupies a co-terminal position within the QODIQA enforcement pipeline. It does not precede enforcement and does not follow enforcement as a downstream side effect. It operates as a co-process that captures the full computational context of each enforcement event before that context is released. This co-terminal positioning ensures that the evidence record and the enforcement decision share an identical view of system state at the moment of evaluation.

The audit subsystem reads from the same frozen context that the Enforcement Gateway uses to evaluate policy. It does not derive evidence from the enforcement output alone, which would produce an incomplete record. It captures the full input contract, the resolved consent state, the applicable policy version, the evaluation result, and the enforcement outcome as a single composite artifact.

ARCHITECTURAL INVARIANT

Evidence generation is not optional and cannot be deferred. An enforcement decision that cannot be simultaneously recorded as a complete evidence artifact must be treated as a system failure. The fail-closed semantics of the Enforcement Gateway extend to the audit subsystem without exception.

1.2 Relationship with the Enforcement Gateway

The Enforcement Gateway is the component within QODIQA that applies policy rules to a frozen evaluation context and produces a deterministic enforcement decision. The audit subsystem is its mandatory partner. The Gateway produces the decision; the audit subsystem produces the proof that the decision was made correctly, from a specific set of inputs, under a specific policy version, at a specific time.

The relationship is not hierarchical. The audit subsystem does not validate the Gateway's decision. It records it. Validation of the decision is the responsibility of downstream verification processes, which operate against the evidence record. The audit subsystem is the source of truth for those verification processes. Its integrity is therefore a prerequisite for any external audit or compliance claim.

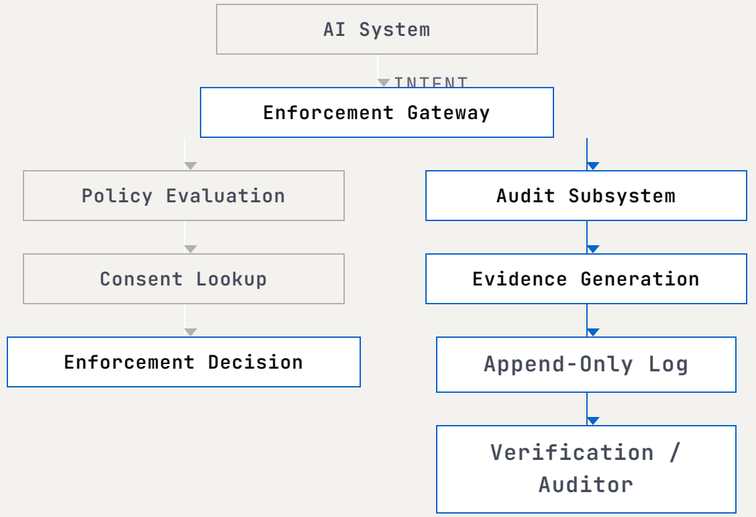


FIGURE 1.1: QODIQA AUDIT FLOW ARCHITECTURE. THE ENFORCEMENT GATEWAY AND THE AUDIT SUBSYSTEM EXECUTE CO-TERMINALLY: THEY SHARE AN IDENTICAL FROZEN CONTEXT AND PRODUCE THEIR RESPECTIVE OUTPUTS WITHIN THE SAME ATOMIC OPERATIONAL UNIT. EVIDENCE GENERATION IS NOT A DOWNSTREAM SIDE EFFECT; IT IS A PARALLEL, CO-EQUAL PROCESS. THE APPEND-ONLY LOG IS THE DURABLE SUBSTRATE THAT ENABLES INDEPENDENT VERIFICATION WITHOUT OPERATIONAL ACCESS TO THE LIVE SYSTEM.

1.3 Trust Boundary Implications

The audit subsystem operates within the QODIQA trust boundary. Its outputs, once committed to the append-only log, are considered authoritative records of enforcement history. External systems, auditors, regulatory bodies, and certification authorities that interact with QODIQA evidence records must treat those records as the canonical representation of system behavior.

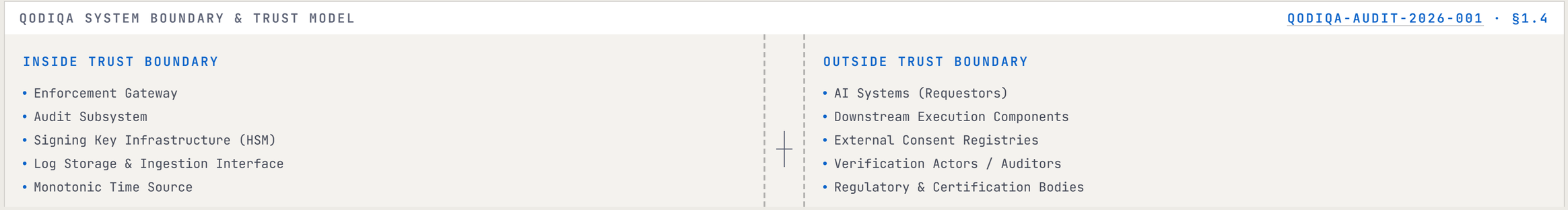
The trust boundary requires that the audit subsystem be isolated from the components it observes. The subsystem must not be reachable from the AI inference path in a way that would permit the inference path to alter, suppress, or selectively produce evidence records. Isolation is a structural requirement, not a configuration option.

1.4 System Boundary and Trust Isolation

The QODIQA trust boundary defines the set of components whose operation is governed by and accountable to the QODIQA specification. Components inside the boundary are subject to the full set of determinism, immutability, and integrity requirements defined in this corpus. Components outside the boundary are treated as external principals that interact with QODIQA through defined interfaces and whose behavior QODIQA does not assume or control.

The boundary is not a network perimeter; it is an accountability perimeter. Components inside the boundary must be operated, monitored, and auditable by the deploying organization. Any component that can influence an enforcement decision or an evidence record must be inside the boundary or its influence must be mediated through a defined, auditable interface.

The fundamental isolation rule is absolute: an AI system must not be able to influence the generation of the evidence record that corresponds to its own enforcement event. This prohibition extends to indirect influence through shared resources, side channels, timing manipulation, or administrative access to the audit subsystem.



• Policy & Consent Version Store • Integrity Monitoring Function	• External Anchor Services • Organizational Administration

Isolation Rule: No component outside the trust boundary may alter, suppress, or selectively trigger evidence generation for any enforcement event. Any system that violates this condition is non-conformant.

SECTION 2

Deterministic Audit Principles

The QODIQA audit model is governed by five foundational principles. These principles are not implementation preferences; they are normative requirements that every conforming implementation must satisfy. Deviation from any principle constitutes a non-conformance that invalidates the auditability guarantee.

2.1 Immutability

Evidence records, once committed to the audit log, must not be altered, updated, or deleted under any operational condition. Immutability is absolute. It applies to the record content, the record metadata, and the record's position within the log sequence. Systems that expose mutation interfaces to the audit log, even for the purpose of error correction, do not conform to this specification.

> **PRINCIPLE 1: IMMUTABILITY**
>
> A committed evidence record is permanent. No post-commit mutation is permitted under any condition, including administrative action, error correction, or legal obligation. Corrections are achieved through supplementary records, not modification of existing ones.

2.2 Verifiability

Every evidence record must be independently verifiable. Verifiability means that a party in possession of the record, its associated cryptographic proof, and the applicable policy and consent state versions can confirm that the record accurately represents the enforcement decision that occurred. Verifiability does not require access to the live QODIQA system. It requires only the record itself and the referenced immutable artifacts.

> **PRINCIPLE 2: VERIFIABILITY**
>
> Each evidence record must contain or reference all inputs necessary for an independent party to reproduce the enforcement decision and confirm it matches the recorded outcome. Verification must be achievable without operational access to the producing system.

2.3 Non-Repudiation

Evidence records must be cryptographically signed such that the producing system cannot later deny having made the recorded decision. Non-repudiation requires that the signing key be controlled by the QODIQA enforcement subsystem and that the signature be verifiable against a published public key or certificate chain. Key management practices that permit retroactive alteration of signing authority are incompatible with this principle.

> **PRINCIPLE 3: NON-REPUDIATION**
>
> A cryptographic signature over each evidence record, produced by a key held within the QODIQA trust boundary, provides non-repudiation. The signing authority cannot deny the decision, and the decision cannot be attributed to any other system or time.

2.4 Completeness

The audit log must contain a complete record of every enforcement event processed by the system. Gaps are not acceptable. A missing record is not an absence of evidence; it is evidence of a gap in audit coverage, which is itself an audit finding. Implementations must detect and report gaps automatically. Manual gap identification is insufficient.

PRINCIPLE 4: COMPLETENESS

Every enforcement event produces exactly one evidence record. No event is excluded, suppressed, or deferred. Completeness is a structural guarantee, enforced by the co-terminal architecture of the audit subsystem, not a statistical property achievable through sampling.

2.5 Time Consistency

Evidence records must include timestamps derived from a monotonic, tamper-evident time source. Time consistency means that the ordering of records in the audit log is consistent with the actual temporal ordering of enforcement events. Systems that rely on client-supplied timestamps, mutable system clocks, or network time sources subject to manipulation do not satisfy this principle.

PRINCIPLE 5: TIME CONSISTENCY

Timestamps in evidence records are derived from a monotonic time source controlled within the QODIQA trust boundary. The temporal ordering of records in the audit log is authoritative and must match the actual sequence of enforcement decisions.

INVARIANTS

Deterministic Audit Invariants

The following invariants are derived from the five deterministic audit principles and constitute the hard operational law of the QODIQA audit subsystem. They are not guidelines, targets, or best practices. They are non-negotiable system properties. Any implementation that cannot demonstrate adherence to every invariant fails to conform to this specification. No business, operational, or regulatory justification overrides these invariants.

DETERMINISTIC AUDIT INVARIANTS		QODIQA-AUDIT-2026-001 · §2a
INV-001	No Enforcement Without Evidence	The Enforcement Gateway must not return an enforcement decision to any calling system unless the corresponding evidence record has been successfully committed to the append-only log. A decision that is returned without a committed evidence record does not constitute a valid QODIQA enforcement decision and must be treated as a system failure.
INV-002	Exactly-One Record per Event	Each enforcement event produces exactly one evidence record of each applicable event class. Duplicate records, partial records, and missing records for events that reached the Enforcement Gateway are prohibited. Exactly-one semantics are enforced at the ingestion interface and at the audit completeness monitoring layer.
INV-003	Immutable Record Guarantee	No committed evidence record may be altered, updated, overwritten, or deleted under any operational condition, by any system actor, through any interface, including administrative, emergency, or maintenance interfaces. The immutability guarantee applies to all fields of the record including its position within the log sequence.
INV-004	Hash Chain Continuity	Every evidence record within a log partition, at position i greater than zero, must carry in its `prev_event_hash` field the SHA-256 digest of the canonical serialization of the record at position i minus one. A gap in the hash chain is a chain integrity failure and invalidates all records from the point of failure forward for verification purposes.
INV-005	Signature Validity Requirement	Every evidence record must carry a valid ECDSA-P256 signature produced by the Enforcement Gateway's signing key at the time of record production. The signature must be verifiable using the published public key associated with the producer identity at the record's timestamp. A record with an absent, invalid, or unverifiable signature is not valid evidence.
INV-006	Monotonic Ordering Requirement	Log sequence numbers within a partition must be strictly monotonically increasing. Timestamps within a partition must be non-decreasing. The ingestion interface must reject any record whose log sequence number is not exactly one greater than the preceding record's sequence number. Out-of-order or duplicate sequence numbers are fatal ingestion errors.
INV-007	Audit Availability = Enforcement Availability	The availability of the audit subsystem is equivalent to the availability of the enforcement system itself. When the audit subsystem is unable to accept and commit evidence records, the Enforcement Gateway must enter fail-closed state and deny all pending enforcement requests. No enforcement decision may be issued during a period of audit unavailability, regardless of its urgency or operational impact.
INV-008	No Post-Commit Mutation	Post-commit mutation of any evidence record is prohibited under any condition without exception. This prohibition extends to redaction, anonymization, field nullification, compression that alters content, format migration that changes field values, and any other operation that produces a different byte sequence for the record content. Corrections are achieved exclusively through supplementary integrity events appended to the log.

NORMATIVE STATUS

These invariants are co-normative with the five deterministic audit principles of Section 2. A conforming implementation must satisfy all eight invariants without exception. Any implementation that violates any invariant is non-conformant with this specification, regardless of its compliance with other QODIQA corpus documents. Certification cannot be granted to an implementation with a known invariant violation.

SECTION 3

Evidence Generation Model

Evidence, within the QODIQA framework, is a precisely defined construct. It is not a log entry in the colloquial sense, nor is it a debug trace or an operational metric. Evidence is a structured, cryptographically bound record that captures the complete deterministic state of an enforcement event in a form suitable for independent verification and legal-grade accountability.

3.1 Definition of Evidence

An evidence record within QODIQA is a composite artifact that contains: the identity of the enforcement event that produced it; the complete input contract presented to the Enforcement Gateway; the resolved consent state at the moment of evaluation; the version-pinned policy that was applied; the intermediate evaluation trace; the final enforcement decision; and the cryptographic linkage that binds this record to the preceding record in the append-only log.

Evidence is generated at enforcement time, not retrospectively. It captures the state as it existed during evaluation. Post-hoc reconstruction of evidence is not permitted and would not constitute evidence under this specification. The evidence record is produced within the same atomic operational unit as the enforcement decision itself.

3.2 Evidence Types

The QODIQA evidence model recognizes four canonical evidence types. Each type addresses a distinct dimension of the enforcement event and together they constitute the complete evidentiary record of a QODIQA enforcement lifecycle.

EVIDENCE TYPE 1: DECISION EVIDENCE

The primary evidence artifact produced by each enforcement event. Decision evidence records the final deterministic outcome: allow, deny, or allow-with-restriction. It includes the complete input contract, the resolved consent state reference, the policy version applied, and the resulting decision with its rationale expressed in terms of the applicable policy clauses.

EVIDENCE TYPE 2: POLICY EVALUATION TRACE

A structured record of the policy evaluation process for a given enforcement event. The trace records each policy clause that was evaluated, the intermediate truth value assigned to each clause, the logical structure of the evaluation, and the clause or clauses that were determinative in the final decision. Policy evaluation traces enable independent verification of the evaluation logic and support detection of policy misconfiguration.

EVIDENCE TYPE 3: CONSENT VERIFICATION PROOF

A cryptographically verifiable record demonstrating that consent was resolved against a specific, version-pinned consent state at a specific time. The consent verification proof includes the consent record identifier, the version hash of the consent state, the consent scope, purpose bindings, temporal validity window, and the resolved consent value. This type enables independent parties to verify that consent was valid at the moment of enforcement without requiring access to the live consent registry.

EVIDENCE TYPE 4: REJECTION AND ALLOW LOGS

Structured outcome records that provide a compact, searchable representation of enforcement results across high-volume operational contexts. Rejection logs capture the complete identity of denied requests, the specific policy clause or consent condition responsible for the denial, and the enforcement timestamp. Allow logs capture authorized requests, including any restrictions applied as conditions of authorization. These records support operational monitoring, anomaly detection, and aggregate compliance reporting.

SECTION 4

Audit Event Taxonomy

The QODIQA audit subsystem recognizes a standardized set of event classes that collectively describe the lifecycle of a single enforcement transaction. Each event class has a canonical identifier, a required field set, and a deterministic structure. Implementations must emit events conforming to this taxonomy. Non-standard event formats are not recognized as audit evidence under this specification.

4.1 Event Class: Inference Request Received

This event is emitted when a conforming AI system presents an intent declaration to the QODIQA Enforcement Gateway. It represents the initiation of an enforcement transaction and marks the beginning of the audit record chain for that transaction.

EVT-001 · INFERENCE REQUEST RECEIVED: REQUIRED FIELDS

FIELD	TYPE	DESCRIPTION	REQUIRED
event_id	UUID v4	Globally unique identifier for this event instance	Mandatory
event_class	string	Canonical value: `EVT-001-INFERENCE-REQUEST-RECEIVED`	Mandatory
transaction_id	UUID v4	Shared identifier binding all events within one enforcement transaction	Mandatory
timestamp_utc	ISO 8601	Monotonic UTC timestamp of event generation	Mandatory
requestor_id	string	Stable identifier of the AI system or agent submitting the request	Mandatory
declared_intent	object	Complete intent declaration as submitted, without modification	Mandatory
declared_purpose	string	Machine-readable purpose code from the approved purpose taxonomy	Mandatory

FIELD	TYPE	DESCRIPTION	REQUIRED
declared_scope	object	Scope declaration: data categories, subject identifiers, processing context	Mandatory
declared_duration	interval	Requested processing duration or expiry, expressed as ISO 8601 interval	Mandatory
prev_event_hash	SHA-256	Hash of the preceding event in the log chain; null for chain genesis	Mandatory

4.2 Event Class: Policy Evaluation

This event is emitted upon completion of the policy evaluation phase. It records the policy version applied, the evaluation trace, and the intermediate result prior to the final enforcement decision. Policy evaluation events provide the traceability necessary to verify that the correct policy was applied to the correct input.

EVT-002 · POLICY EVALUATION: REQUIRED FIELDS

FIELD	TYPE	DESCRIPTION	REQUIRED
event_id	UUID v4	Globally unique identifier for this event instance	Mandatory
event_class	string	Canonical value: `EVT-002-POLICY-EVALUATION`	Mandatory
transaction_id	UUID v4	Binding identifier shared with the originating EVT-001 event	Mandatory
timestamp_utc	ISO 8601	Monotonic UTC timestamp of evaluation completion	Mandatory
policy_id	string	Stable identifier of the policy document applied	Mandatory
policy_version_hash	SHA-256	Content-hash of the exact policy version used in evaluation	Mandatory
evaluation_trace	array	Ordered list of clause evaluations, each with clause_id, input_values, and truth_value	Mandatory
determinative_clauses	array	Identifiers of the clause(s) responsible for the evaluation outcome	Mandatory

FIELD	TYPE	DESCRIPTION	REQUIRED
intermediate_result	enum	Pre-decision evaluation result: PERMIT, DENY, RESTRICT, INDETERMINATE	Mandatory
prev_event_hash	SHA-256	Hash of the preceding event in the log chain	Mandatory

4.3 Event Class: Consent Lookup

This event is emitted when the Enforcement Gateway queries the consent registry. It records the exact consent state retrieved, its version, its temporal validity, and the resolution result. Consent lookup events are the evidentiary basis for consent verification proofs and are required for any claim that a consent condition was satisfied at the time of enforcement.

EVT-003 · CONSENT LOOKUP: REQUIRED FIELDS

FIELD	TYPE	DESCRIPTION	REQUIRED
event_id	UUID v4	Globally unique identifier for this event instance	Mandatory
event_class	string	Canonical value: EVT-003-CONSENT-LOOKUP	Mandatory
transaction_id	UUID v4	Binding identifier shared with the originating EVT-001 event	Mandatory
timestamp_utc	ISO 8601	Monotonic UTC timestamp of consent registry query	Mandatory
consent_subject_ref	string	Opaque, non-PII reference to the consent subject	Mandatory
consent_record_id	string	Identifier of the consent record retrieved from the registry	Mandatory
consent_version_hash	SHA-256	Content-hash of the retrieved consent record version	Mandatory
consent_scope_match	boolean	Whether the retrieved consent scope covers the declared request scope	Mandatory
consent_purpose_match	boolean	Whether the retrieved consent purpose covers the declared purpose	Mandatory

FIELD	TYPE	DESCRIPTION	REQUIRED
consent_temporal_valid	boolean	Whether the consent record was temporally valid at the lookup timestamp	Mandatory
consent_resolution	enum	Resolution result: VALID, REVOKED, EXPIRED, NOT_FOUND, SCOPE_MISMATCH	Mandatory
prev_event_hash	SHA-256	Hash of the preceding event in the log chain	Mandatory

4.4 Event Class: Enforcement Decision

This event records the final deterministic enforcement decision produced by the Enforcement Gateway. It is the primary audit artifact and constitutes the authoritative record of system authorization behavior. Every enforcement transaction that reaches a terminal state must produce exactly one Enforcement Decision event.

EVT-004 · ENFORCEMENT DECISION: REQUIRED FIELDS

FIELD	TYPE	DESCRIPTION	REQUIRED
event_id	UUID v4	Globally unique identifier for this event instance	Mandatory
event_class	string	Canonical value: EVT-004-ENFORCEMENT-DECISION	Mandatory
transaction_id	UUID v4	Binding identifier shared with all events in this enforcement transaction	Mandatory
timestamp_utc	ISO 8601	Monotonic UTC timestamp of decision finalization	Mandatory
decision	enum	Final enforcement decision: ALLOW, DENY, ALLOW_WITH_RESTRICTION	Mandatory
decision_basis	object	Structured rationale referencing policy clauses and consent conditions that produced the decision	Mandatory
restrictions	array	If decision is ALLOW_WITH_RESTRICTION: ordered list of applicable restrictions, each with restriction_id and description	Conditional
input_contract_hash	SHA-256	Content-hash of the complete input contract evaluated	Mandatory

FIELD	TYPE	DESCRIPTION	REQUIRED
policy_version_hash	SHA-256	Content-hash of the policy version applied, as referenced in EVT-002	Mandatory
consent_version_hash	SHA-256	Content-hash of the consent state resolved, as referenced in EVT-003	Mandatory
gateway_signature	bytes	ECDSA-P256 signature over the canonical serialization of this record, produced by the Gateway signing key	Mandatory
prev_event_hash	SHA-256	Hash of the preceding event in the log chain	Mandatory

4.5 Event Class: Block / Allow Outcome

This event is emitted by the system component that enforces the Enforcement Gateway's decision at the execution layer. It confirms that the decision was communicated to, and acted upon by, the downstream system. Block/Allow Outcome events close the enforcement transaction record and provide evidence that the decision was not merely recorded but actually applied.

EVT-005 · BLOCK / ALLOW OUTCOME: REQUIRED FIELDS

FIELD	TYPE	DESCRIPTION	REQUIRED
event_id	UUID v4	Globally unique identifier for this event instance	Mandatory
event_class	string	Canonical value: `EVT-005-BLOCK-ALLOW-OUTCOME`	Mandatory
transaction_id	UUID v4	Binding identifier shared with all events in this enforcement transaction	Mandatory
timestamp_utc	ISO 8601	Monotonic UTC timestamp of outcome application	Mandatory
outcome	enum	Applied outcome: `BLOCKED`, `ALLOWED`, `ALLOWED_WITH_RESTRICTION`	Mandatory
enforcing_component_id	string	Identifier of the system component that applied the decision	Mandatory
enforcement_decision_ref	UUID v4	event_id of the corresponding EVT-004 Enforcement Decision event	Mandatory

FIELD	TYPE	DESCRIPTION	REQUIRED
prev_event_hash	SHA-256	Hash of the preceding event in the log chain	Mandatory

SECTION 5

Evidence Structure Specification

The evidence structure specification defines the canonical form in which QODIQA evidence records are serialized, stored, and transmitted. Canonical form is not a recommendation; it is a normative requirement. Evidence records that deviate from canonical serialization are not recognized as conforming QODIQA evidence and cannot be used to satisfy verification or compliance obligations.

5.1 Canonical Schema

Each evidence record is represented as a JSON object conforming to the QODIQA Evidence Schema v1.0. The schema is published as part of the QODIQA standard corpus and is referenced by SHA-256 content hash in each evidence record's metadata block. This self-referential schema pinning ensures that the schema version used to produce a record is unambiguously identifiable, regardless of subsequent schema revisions.

The canonical schema defines a two-part structure: a mandatory metadata envelope and a mandatory payload body. The metadata envelope contains fields common to all event classes. The payload body is event-class-specific and contains the fields defined in the Audit Event Taxonomy.

CANONICAL EVIDENCE RECORD STRUCTURE

```
{
  "qodiqa_schema_version": "1.0",
  "qodiqa_schema_hash":    "<SHA-256 hash of QODIQA-EVIDENCE-SCHEMA-v1.0>",
```

```
  "metadata": {
    "event_id":           "<UUID v4>",
    "event_class":        "<canonical event class identifier>",
    "transaction_id":     "<UUID v4>",
    "timestamp_utc":      "<ISO 8601 UTC timestamp, nanosecond precision>",
    "log_sequence_num":   <monotonic integer, unique within log partition>,
    "prev_event_hash":    "<SHA-256 of canonical serialization of preceding record>",
    "producer_id":        "<stable identifier of the QODIQA subsystem instance>",
    "producer_signature": "<ECDSA-P256 signature over this record's canonical form>"
  },
  "payload": {
    // event-class-specific fields as defined in Section 4
  }
}
```

5.2 Field-Level Definitions

All timestamp fields use ISO 8601 extended format in the UTC timezone with nanosecond precision. The format is: `YYYY-MM-DDTHH:MM:SS.nnnnnnnnnZ`. Timestamps that do not conform to nanosecond precision are not accepted by the QODIQA audit log ingestion interface.

All hash fields contain the hexadecimal encoding of a SHA-256 digest, lowercase, without prefix, exactly 64 characters in length. Hash fields that contain uppercase characters, prefix notation such as `sha256:`, or incorrect length are rejected.

All signature fields contain the Base64url encoding of an ECDSA-P256 signature over the canonical serialization of the record, as defined in Section 5.3. Signatures are computed after all other fields in the record have been finalized. The producer_signature field is excluded from the data over which the signature is computed; it contains the result of that computation.

All UUID fields conform to RFC 4122 version 4 format. UUIDs are represented as lowercase hyphenated strings. UUIDs generated by the QODIQA system must exhibit cryptographically adequate randomness and must not be predictable or sequential in a way that would permit pre-commitment attacks.

5.3 Deterministic Serialization

The canonical serialization of a QODIQA evidence record is a deterministic JSON encoding produced according to the following rules, in order: all object keys are sorted lexicographically in ascending byte order; all string values are encoded as UTF-8; no insignificant whitespace is included; all floating-point values are expressed in their shortest representation that round-trips; boolean values are lowercase `true` or `false`; null values are expressed as `null`; array element order is preserved as specified in the schema.

Deterministic serialization is the prerequisite for reproducible hash computation and signature verification. Any deviation from these rules produces a different serialization, a different hash, and an invalid signature. Implementations must validate that their serialization produces byte-identical output for a given record structure before being considered conforming.

The canonical serialization specification is not subject to interpretation. It is a precise algorithmic procedure. Implementations must reference the QODIQA Canonical Serialization Test Vector Suite, distributed with the standard corpus, to validate their serialization behavior before deployment.

SECTION 6

Cryptographic Integrity Model

The cryptographic integrity model specifies the mechanisms by which QODIQA evidence records are bound together into a tamper-evident chain and individually signed to provide non-repudiation. Cryptographic integrity is not an optional

hardening measure. It is a structural property of the audit subsystem and a prerequisite for any verification or certification claim.

6.1 Hash Chaining

Each evidence record includes the SHA-256 hash of the immediately preceding record in the audit log partition as the value of the `prev_event_hash` field. This hash chain binds the sequence of records such that any modification, insertion, or deletion of a record invalidates the hashes of all subsequent records. The chain is broken at the log partition boundary; the genesis record of each partition contains `null` in the `prev_event_hash` field.

Hash chaining provides tamper detection at the log level. It does not, by itself, prevent tampering; it ensures that tampering is detectable. The detection capability depends on the verifier having access to at least one authentic hash value, which is provided by the external anchoring mechanism described in Section 6.4.

HASH CHAIN VERIFICATION PROCEDURE

```
for each record R[i] in partition where i > 0:
  expected_prev_hash = SHA-256(canonical_serialize(R[i-1]))
  assert R[i].metadata.prev_event_hash == expected_prev_hash,
    "Chain integrity failure at position i: expected hash does not match"
```

6.2 Tamper-Evidence Guarantees

The hash chain provides a strong tamper-evidence guarantee: any modification to any record in a partition makes the chain verifiably inconsistent from the point of modification forward. This guarantee holds under the assumption that the attacker cannot recompute valid SHA-256 hashes faster than the system appends new records.

The tamper-evidence guarantee does not extend to: truncation of the log from the end; suppression of records prior to their commitment; or replacement of the entire partition with a freshly constructed chain. These attack vectors are

addressed by external anchoring and integrity monitoring, described below.

6.3 Signing Requirements

Each evidence record is individually signed by the QODIQA Enforcement Gateway's signing key using ECDSA with the P-256 curve and SHA-256 as the hash function. The signature is computed over the canonical serialization of the complete record, excluding the `producer_signature` field itself.

The signing key must be an asymmetric key held within a hardware security module or equivalent trusted execution environment. Software-only key storage does not satisfy this requirement for production deployments. The corresponding public key must be published through the QODIQA Certification Registry under the deploying organization's certified entry. Key rotation must follow the key management procedures specified in the QODIQA Security and Cryptographic Profile.

Signatures must be verified before any evidence record is accepted by the audit log ingestion interface. Records that fail signature verification are rejected and the rejection is itself recorded as a system integrity event.

> **ARCHITECTURAL CONSTRAINT: SIGNING KEY ISOLATION**
>
> The ECDSA-P256 signing key used to produce evidence record signatures must be held within a hardware security module or equivalent trusted execution environment that is physically and logically isolated from the AI inference path. Any implementation that stores the signing key in software-accessible memory reachable from the AI inference path is non-conformant. Key exfiltration from this environment must be architecturally impossible, not merely operationally prevented.

6.4 Optional External Anchoring

Implementations may optionally anchor the audit log to an external tamper-evident structure, such as a Merkle tree whose root is periodically published to an immutable external record, a transparency log service, or a distributed ledger.

External anchoring provides tamper-evidence guarantees that extend beyond the organizational boundary and are verifiable by parties who do not have access to the QODIQA system itself.

When external anchoring is employed, the anchor reference must be included in the log partition metadata and associated with the range of records covered by the anchor. Anchoring does not replace the hash chain or individual record signatures; it supplements them with an additional, externally verifiable integrity proof.

The use of external anchoring is strongly recommended for deployments subject to regulatory audit, cross-organizational enforcement, or third-party certification. Anchoring frequency should be calibrated to the acceptable window of undetected tampering for the deployment context.

SECTION 7

Append-Only Audit Log Architecture

The audit log is the durable storage substrate for QODIQA evidence records. Its architectural properties are as important as the records it contains. An audit log that permits modification, reordering, or deletion of records provides no stronger guarantees than a mutable database, regardless of the cryptographic protections applied to individual records.

7.1 Strict Append-Only Behavior

The audit log must expose only append operations to the components that write evidence records. Read and append operations may be available to authorized consumers. Update, delete, and truncate operations must not be exposed through any interface, including administrative interfaces, backup and restore procedures, and emergency access paths.

Implementations must document how append-only behavior is enforced at the storage layer and must demonstrate that no code path within the QODIQA system or its dependencies can produce a mutation of committed records. This

documentation is required as part of the QODIQA Certification Framework conformance evidence package.

> **ARCHITECTURAL CONSTRAINT**
>
> The audit log storage implementation must enforce append-only semantics at the hardware or operating-system level, or through a trusted intermediary service that provides equivalent guarantees. Application-level enforcement of append-only behavior is insufficient and does not satisfy this requirement.

7.2 Ordering Guarantees

Records within a log partition must be ordered by their monotonic timestamp and log sequence number. The log sequence number is a monotonically increasing integer, unique within a partition, assigned at the time of record commitment. The combination of partition identifier and log sequence number provides a globally unique, stable position reference for each record.

The ordering guarantee requires that the audit log ingestion interface reject records with timestamps or sequence numbers that violate monotonicity. Out-of-order submissions, whether due to clock skew, network reordering, or operational errors, must be rejected and the rejection recorded as a system integrity event. Any implementation that accepts out-of-order records without rejection is non-conformant.

7.3 Replay Consistency

The audit log must support deterministic replay: given the same sequence of evidence records, any compliant QODIQA verifier must produce the same verification result. Replay consistency requires that all state referenced by evidence records, including policy versions and consent state versions, remain available and immutable for as long as the records that reference them are retained.

Implementations must maintain version-pinned copies of all policy and consent state artifacts referenced by audit records. These copies must be stored in an immutable object store and must not be subject to garbage collection, compaction, or expiration policies that would render historical records unverifiable.

SECTION 8

Evidence Retrieval and Verification

The QODIQA audit subsystem must provide a retrieval interface that enables authorized parties to access evidence records and perform verification operations. The retrieval interface is read-only. It must not expose any operation that could modify, reorder, or delete records.

8.1 Evidence Access Model

Access to evidence records is governed by a role-based authorization model that is separate from, and independent of, the QODIQA Enforcement Gateway's authorization model. The audit subsystem must not process its own access requests through the Enforcement Gateway; doing so would create a circular dependency that could compromise audit completeness.

Three access roles are defined. The Audit Reader role provides read access to individual records by event_id or transaction_id. The Partition Verifier role provides read access to complete partitions and the metadata necessary for hash chain verification. The Certification Auditor role provides read access to all evidence and referenced artifacts necessary to perform a complete conformance audit.

Access to evidence records containing consent subject references must be controlled such that only parties with a legitimate audit purpose can retrieve records associated with a specific subject. General-purpose search and retrieval interfaces must not permit enumeration of consent subject references.

8.2 Verification Flows

Evidence verification consists of two distinct operations that may be performed independently or in combination. Record verification confirms that a specific evidence record is authentic, unmodified, and was produced by the QODIQA system. Chain verification confirms that a sequence of evidence records forms an unbroken, tamper-evident chain.

Record verification proceeds as follows: retrieve the evidence record by event_id; retrieve the signing public key associated with the producer_id at the record's timestamp; compute the canonical serialization of the record excluding the producer_signature field; verify the ECDSA-P256 signature against the serialization using the retrieved public key; confirm that the timestamp falls within the validity period of the signing key certificate.

Chain verification proceeds as follows: retrieve all records in the partition ordered by log_sequence_num; for each record at position i greater than zero, compute SHA-256 of the canonical serialization of the record at position i minus one; confirm that the computed hash matches the prev_event_hash field of the record at position i; report any mismatch as a chain integrity failure with the position and the two records involved.

8.3 Third-Party Validation Capability

A conforming QODIQA implementation must support third-party validation without requiring the validator to have operational access to the QODIQA system. Third-party validation is enabled by the combination of: the complete evidence record, including all referenced artifact hashes; the published signing public keys of the QODIQA deployment; the immutable copies of all policy and consent state versions referenced by the record; and, if external anchoring is employed, the anchor proof for the relevant partition range.

Third-party validators must be able to perform both record verification and chain verification using only these artifacts. A QODIQA deployment that requires validator access to live system APIs to complete verification does not satisfy the third-party validation requirement.

SECTION 9

Failure Modes and Audit Gaps

The QODIQA audit subsystem must be designed for failure. System components fail. Networks partition. Storage substrates become temporarily unavailable. The design question is not whether failures will occur but what the system's behavior must be when they do. This section specifies the required failure behavior for the audit subsystem and defines how gaps in audit coverage must be detected, recorded, and reported.

Missing Logs

A missing log is defined as the absence of one or more expected evidence records for enforcement events that are known to have occurred. Missing logs may result from audit subsystem failures, storage failures, record ingestion rejections, or software defects. In all cases, the absence of an expected record is itself an audit-significant event that must be recorded and reported.

The QODIQA audit subsystem must implement continuous completeness monitoring that detects gaps in the log sequence number series within each partition. Detected gaps trigger an automatic integrity alert and, where the gap is attributable to a known failure, a gap record is appended to the log documenting the nature and extent of the missing coverage.

> REQUIRED BEHAVIOR: MISSING LOG RESPONSE
>
> - Detect gaps in log sequence numbers within 60 seconds of commitment of the first record following the gap.
> - Emit a Gap Notification event documenting the partition identifier, the sequence number range of the gap, the timestamp of detection, and the best available explanation for the gap.
> - Escalate unresolved gaps to the designated integrity monitoring function within the implementation's defined escalation window.
> - Mark all enforcement decisions within the gap range as having unverifiable audit coverage in all subsequent audit and certification reports.

Corruption Scenarios

Corruption of an evidence record is detected through hash chain verification and individual record signature verification. When corruption is detected, the corrupted record and all subsequent records in the partition must be flagged. The original, pre-corruption content of records cannot be recovered from the audit log itself; recovery requires resort to external backup or anchor proofs.

Implementations must maintain isolated, cryptographically protected backup copies of evidence records that are not reachable through the same administrative paths as the primary audit log. Backup copies must be verified on a scheduled basis to ensure they remain uncorrupted and complete.

Partial Evidence

Partial evidence occurs when an enforcement transaction produces some but not all of its expected events. For example, an EVT-001 record may exist without a corresponding EVT-004 or EVT-005 record, indicating that the transaction did not reach a terminal state. Partial evidence records are retained in the audit log and flagged as incomplete transactions.

Incomplete transactions are ambiguous from an audit perspective: they demonstrate that an enforcement attempt was made but do not establish the outcome. Implementations must report incomplete transactions in audit summaries and must not treat them as authorization evidence for either direction.

Required Fail-Closed Behavior

When the audit subsystem is unavailable or unable to commit a new evidence record, the Enforcement Gateway must deny all pending enforcement requests. The Gateway must not issue authorization decisions that cannot be simultaneously recorded as evidence. This fail-closed behavior is absolute and cannot be overridden by operational policy, service level agreements, or administrative instruction.

The fail-closed requirement means that audit subsystem availability is on the critical path for AI system operation. Implementations must design the audit subsystem for high availability commensurate with the availability requirements of the AI systems it serves. Single-point-of-failure architectures in the audit path are incompatible with the QODIQA reliability standard.

SECTION 10

Privacy and Data Minimization

The QODIQA audit subsystem operates at the intersection of accountability and privacy. Comprehensive audit records are necessary for verifiable enforcement; comprehensive records of data processing activities can themselves constitute privacy risks if they contain personal information. The privacy and data minimization requirements in this section resolve this tension through a set of structural constraints on what the audit subsystem records and how those records are handled.

What Is Logged

Evidence records must contain sufficient information to enable independent verification of the enforcement decision. This includes: the intent declaration structure, expressed in terms of purpose, scope, and duration categories, without raw data content; the consent subject reference, expressed as an opaque, non-reversible identifier that does not encode personal information; the consent resolution result and the attributes that determined it, without including the raw consent record content; and the enforcement decision and its basis, without including the content of the AI inference request or response.

The audit record documents the authorization event, not the data processing that the authorization event permits or denies. The content of the AI inference, the personal data processed, and the model outputs are outside the scope of the audit log and must not be included in evidence records.

What Is Not Logged

The following categories of information must never appear in QODIQA evidence records under any circumstances: raw personal data of any kind; AI model inputs and outputs; consent subject identifiers that can be linked to natural persons without access to a separately held mapping; data category labels that, in combination, constitute a re-identification risk; and system credentials, API keys, or internal network addresses.

Redaction Constraints

Redaction of evidence records after commitment is prohibited by the immutability principle. Implementations must ensure that records are produced in compliant form at the time of generation. There is no post-commitment redaction mechanism, and no post-commitment redaction mechanism may be implemented.

If an evidence record is inadvertently produced containing non-compliant content, the appropriate response is to flag the record as containing a data quality defect, record the nature of the defect in a supplementary integrity event, and investigate the root cause to prevent recurrence. The record itself must not be modified, deleted, or withheld from authorized audit access.

Note: Data minimization requirements in this specification apply to the QODIQA audit subsystem specifically. Broader data protection obligations applicable to the deploying organization and the AI systems operating under QODIQA governance are outside the scope of this document and must be addressed through applicable regulatory frameworks.

SECTION 11

Integration with the Certification Framework

The QODIQA Certification Framework defines the conditions under which a QODIQA deployment may be certified as conformant with the standard corpus. The audit and evidence model is directly integrated with the certification

framework: evidence records are the primary compliance artifacts, and the ability of a deployment to produce, preserve, and make verifiable its evidence records is a prerequisite for certification.

Audit as Compliance Artifact

Certification assessors evaluating a QODIQA deployment for conformance will rely on the audit log as the authoritative record of system behavior during the assessment period. The assessor must be able to independently verify a representative sample of enforcement decisions by performing record verification and chain verification as defined in Section 8. A deployment that cannot support this verification process cannot be certified.

The completeness of the audit log is itself a conformance criterion. Deployments that exhibit unexplained gaps in their enforcement record, incomplete transactions without corresponding explanatory integrity events, or chain integrity failures that cannot be accounted for by documented system events will receive a qualified or conditional certification status.

Evidence Retention Requirements

Evidence records must be retained for a minimum retention period defined by the QODIQA Certification Framework and, where applicable, by the regulatory requirements of the jurisdiction in which the deployment operates. The minimum QODIQA retention period is seven years from the date of record commitment. Retention does not permit archival in a form that would degrade verification capability; all referenced artifacts must remain available for the full retention period.

Audit-Driven Conformance Reporting

Deploying organizations may generate automated conformance reports from the audit log that summarize enforcement activity, consent resolution rates, policy application patterns, and gap metrics over a specified period. These reports,

when accompanied by the underlying evidence records and verification proofs, constitute the primary documentary basis for certification renewal and regulatory reporting.

SECTION 12

Implementation Constraints

The audit and evidence model imposes real constraints on the systems that implement it. These constraints are not theoretical; they have direct implications for system latency, storage capacity, and operational throughput. This section provides implementation guidance intended to help deploying organizations plan conformant implementations without introducing unacceptable operational overhead.

Latency Impact

Evidence generation occurs on the critical path of the enforcement decision. The audit subsystem must produce, sign, and commit a complete evidence record before the Enforcement Gateway returns its decision to the calling system. The latency budget for evidence generation is therefore bounded by the latency requirements of the overall enforcement transaction.

QODIQA specifies a maximum evidence generation latency target of 5 milliseconds at the 99th percentile under sustained production load. This target assumes a co-located audit subsystem with local HSM access. Deployments that require network-remote audit log infrastructure must account for network latency in their latency budget and may need to design the audit path accordingly, for example through batched signing with individual pre-commitment record locking.

Implementations must monitor evidence generation latency continuously and alert when the 99th percentile exceeds 5 milliseconds. Sustained latency exceedance is a performance non-conformance that must be addressed through

architectural remediation, not latency budget adjustment.

Storage Constraints

Evidence records are structured JSON objects. A typical enforcement transaction produces five event records with a combined uncompressed size in the range of 4 to 12 kilobytes, depending on policy complexity and evaluation trace depth. At high enforcement volumes, storage accumulation can be substantial.

Storage planning must account for the seven-year minimum retention requirement. At an enforcement rate of 10,000 transactions per second, the uncompressed evidence record volume is approximately 360 gigabytes per day, or approximately 920 terabytes over seven years before compression. Compression ratios of 4:1 to 8:1 are achievable on structured JSON evidence records, reducing the practical storage requirement to approximately 115 to 230 terabytes for this throughput profile. Storage estimates must be revised for each deployment based on actual enforcement rates and policy complexity.

Throughput Considerations

The audit log ingestion interface must sustain throughput commensurate with the peak enforcement rate of the deployment, with sufficient headroom to absorb burst traffic without evidence loss. Log ingestion throughput must be tested under peak load conditions as part of the pre-production conformance validation process.

Hash chain computation and signature verification at high throughput are computationally intensive operations. Implementations must benchmark their cryptographic processing capacity against their expected enforcement rates and provision sufficient compute resources to avoid cryptographic operations becoming the throughput bottleneck. Parallel verification pipelines are permitted for read operations, but write operations must maintain strict sequential ordering within each partition to preserve hash chain integrity.

SECTION 13

Residual Risk and Non-Coverage

This specification defines a rigorous audit and evidence model that provides strong verifiability guarantees within its defined scope. It does not provide, and does not claim to provide, universal guarantees against all possible audit failures or all possible modes of system compromise. This section explicitly characterizes the residual risks and the limitations of the QODIQA audit model.

What Audit Does Not Guarantee

The QODIQA audit model guarantees that enforcement decisions, once made and recorded, are preserved with verifiable integrity and cannot be silently altered or suppressed. It does not guarantee the correctness of the enforcement decisions themselves. If the policy or consent state inputs to the Enforcement Gateway are corrupted, incomplete, or fraudulently constructed, the audit record will faithfully document an incorrect decision without being able to detect the incorrectness of the inputs.

The audit model does not provide coverage for enforcement events that are entirely suppressed before reaching the audit subsystem. A compromised Enforcement Gateway that issues authorization decisions without triggering the audit path would produce no evidence records and would therefore produce no detectable audit gap until the absence of records was noticed through external means. Mitigating this risk requires architectural controls at the integration boundary, not audit system controls.

The audit model does not provide coverage for the behavior of AI systems operating downstream of an authorization decision. Once the Enforcement Gateway issues an ALLOW decision, the audit subsystem's jurisdiction ends. If the authorized AI system acts outside the scope of the authorization, that deviation occurs in a space that is not monitored by the QODIQA audit subsystem.

Explicit Limitations

The cryptographic integrity model provides tamper-evidence guarantees that depend on the security of the ECDSA-P256 signature scheme and SHA-256 hash function. Both are considered secure against current and near-term classical computing adversaries. The model does not provide quantum-resistant security guarantees. Organizations operating in threat environments where quantum adversaries are a credible concern should consult the QODIQA Security and Cryptographic Profile for guidance on migration to post-quantum cryptographic primitives.

The audit model provides a record of what the system decided; it does not provide a record of why human operators made the policy and consent configuration choices that led to those decisions. The institutional accountability for those choices rests with the humans and organizations that govern the QODIQA deployment, not with the audit subsystem.

Finally, the usefulness of the audit record is contingent on the operational practices of the deploying organization. An audit log that is never examined, whose integrity is never verified, and whose evidence is never presented to accountable parties provides no practical accountability benefit, regardless of its technical conformance. The QODIQA audit model provides the infrastructure for accountability; whether that infrastructure is used effectively is an organizational governance question outside the scope of this specification.

This document specifies the QODIQA Audit and Evidence Generation Model as a normative component of the QODIQA standard corpus. It defines required behaviors, structural properties, and implementation constraints. It does not constitute legal advice, regulatory guidance, or a guarantee of compliance with any specific legal framework. Deploying organizations remain responsible for ensuring that their implementations satisfy applicable legal and regulatory obligations in their respective jurisdictions.

DOCUMENT STATUS

Document Status and Corpus Alignment

This document is a core normative specification of the QODIQA standard corpus. It specifies the Audit and Evidence Generation Model for deterministic runtime consent enforcement: the principles, event taxonomy, evidence structure, cryptographic integrity mechanisms, log architecture, retrieval and verification procedures, failure behavior, privacy constraints, certification integration, implementation constraints, and residual risk characterization that together define the QODIQA audit subsystem.

This document is co-normative with the QODIQA Core Standard and the QODIQA Reference Architecture. An implementation that does not conform to this document does not conform to the QODIQA standard, regardless of its conformance to other corpus documents.

This document should be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement
- QODIQA: Terminology and Normative Definitions

- QODIQA: Threat Model and Abuse Case Specification
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Governance Charter for the QODIQA Standard Corpus

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dușescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-AUDIT-2026-001
TITLE	Audit and Evidence Generation Model for Runtime Consent Enforcement
SUBTITLE	Deterministic Audit Principles, Evidence Structure, and Cryptographic Integrity for QODIQA
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Technical Specification: Normative
DOCUMENT STATUS	Normative
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 30 PART V · CONTEXTUAL, ANALYTICAL AND PROCESS SUPPORT

Audit Readiness and Evidence Pack

Structured audit evidence, compliance mapping, and verification artifacts for the consent enforcement standard corpus.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-ARP-2026-001	1.0	2026	Evidence Pack	Normative	NORMATIVE

ABSTRACT

This document constitutes the Audit Readiness and Evidence Pack for the QODIQA deterministic runtime consent enforcement framework. It defines the evidence architecture, compliance mapping, audit trail requirements, and verification procedures that enable organizations deploying QODIQA to demonstrate lawful and technically defensible AI operation to regulators, auditors, and institutional counterparties. Unlike documentation-centric compliance models, QODIQA generates audit evidence as a structural property of execution: each decision produces a stable, reproducible artifact that can be independently verified without reliance on narrative reconstruction or interpretive logs. This pack formalizes how that evidence is organized, preserved, and presented in the context of regulatory obligations arising from data protection law, AI regulation, and sector-specific compliance frameworks. The document further specifies maturity levels for audit readiness, pre-audit verification protocols, and the canonical structure of a QODIQA evidence submission.

SECTION 1

Purpose and Scope

The QODIQA Audit Readiness and Evidence Pack is a normative component of the QODIQA standard corpus. Its purpose is to translate the determinism, auditability, and proof guarantees established in the QODIQA Technical Whitepaper into a concrete operational framework for audit preparation, evidence collection, and compliance demonstration.

Regulatory environments governing AI systems increasingly require that organizations be able to demonstrate not merely that governance policies exist, but that they are enforced at execution time in a verifiable, reproducible manner. This requirement cannot be satisfied through documentation alone. It demands a form of evidence that is structural: evidence that emerges from the system itself rather than from post hoc reconstruction.

QODIQA is designed so that audit evidence is a byproduct of correct operation. Every decision the system produces is a stable artifact with a deterministic identifier, a frozen evaluation context, and a traceable policy version. This document specifies how those artifacts are organized, retained, and presented in a format suitable for regulatory, contractual, and internal audit purposes.

The scope of this document covers: the dimensions of audit readiness applicable to QODIQA deployments; the structure and classification of evidence generated by the framework; the mapping of QODIQA system properties to obligations arising under applicable regulatory instruments; the procedural requirements for audit trail maintenance and integrity verification; the pre-audit verification checklist and decision replay protocol; the canonical structure of an evidence pack submission; and the operational practices that sustain continuous audit readiness.

This document does not address legal interpretation of regulatory obligations, implementation-specific deployment configurations, or upstream consent capture systems. Those domains are addressed in separate corpus documents as referenced in the Document Status section.

SECTION 2

Audit Readiness Framework

Audit readiness is not a state achieved immediately before an audit. It is a continuous operational property, maintained through the same mechanisms that ensure correct system behavior during normal operation. For QODIQA deployments, audit readiness and operational correctness are structurally equivalent: a system that enforces consent deterministically and produces stable decision artifacts is, by design, audit-ready.

The following sections define the dimensions of audit readiness, the categories of evidence that QODIQA generates, and the maturity levels against which organizations may assess their current posture.

2.1 Readiness Dimensions

Audit readiness for a QODIQA deployment is assessed across four dimensions. Each dimension corresponds to a structural property of the system rather than a procedural requirement.

Decision traceability is the ability to locate, identify, and retrieve any past decision artifact given a reference identifier, time range, actor identifier, or action type. This dimension is satisfied when the decision artifact store is queryable, retained according to policy, and structurally complete for each decision.

Consent state verifiability is the ability to demonstrate, for any past decision, that the consent state resolved at evaluation time was valid, scoped, and current. This dimension is satisfied when consent references captured in frozen evaluation contexts can be matched against consent state records with confirmed validity at the relevant timestamp.

Policy immutability is the ability to demonstrate that the policy version applied in any past decision has not been altered since that decision was recorded. This dimension is satisfied when policies are versioned with cryptographic integrity protection and version history is retained without modification.

Replay correctness is the ability to re-evaluate any past decision using its frozen context and obtain an identical outcome. This dimension is the strongest evidence of determinism and is satisfied when the replay protocol consistently produces results matching the recorded decision identity.

2.2 Evidence Categories

Evidence produced by a QODIQA deployment falls into four categories, each corresponding to a readiness dimension.

TABLE 1. QODIQA EVIDENCE CATEGORIES AND CORRESPONDING READINESS DIMENSIONS

EVIDENCE CATEGORY	PRIMARY SOURCE	READINESS DIMENSION
Decision artifacts	Decision Artifact Generator	Decision traceability
Consent state records	Consent Resolver	Consent state verifiability
Policy version snapshots	Policy Engine	Policy immutability
Replay verification outputs	Verification protocol execution	Replay correctness

All four categories must be present and complete for an evidence pack to be considered audit-ready. Partial evidence packs, in which one or more categories are absent or incomplete, do not satisfy the structural auditability requirements of QODIQA and must not be presented to regulators or auditors as complete submissions.

2.3 Readiness Maturity Levels

The following maturity levels provide a structured assessment model for organizations evaluating their current audit readiness posture. Each level is defined by the capabilities present, not by the volume of evidence or time in operation.

LEVEL 1: PARTIAL

Decision artifacts are produced and retained. Consent state records exist but are not structurally linked to decision artifacts. Policy versioning is present but cryptographic integrity is not enforced. Replay has not been validated.

LEVEL 2: STRUCTURED

All four evidence categories are produced. Frozen evaluation contexts reference consent records and policy versions. Replay protocol has been executed and validated for a representative sample of decisions. Integrity checks are performed periodically.

LEVEL 3: CONTINUOUS

All evidence categories are produced and retained with automated integrity verification. Replay correctness is continuously monitored. Evidence packs can be assembled on demand for any time period within retention scope. Denial records are preserved with the same fidelity as allow decisions.

Level 3 represents the target state for organizations operating QODIQA in regulated environments. Levels 1 and 2 are recognized as transitional states during deployment and maturation phases.

1 Maturity level assessment is self-reported and subject to validation during formal audit procedures. Organizations should not represent a maturity level to external parties without having performed the verification protocols described in Section 6.

SECTION 3

Evidence Architecture

The evidence architecture of QODIQA is not an overlay applied to the system after the fact. It is a direct consequence of the architectural constraints that make the system deterministic. Because every decision is derived from an immutable, canonical set of inputs, the evidence required to verify that decision is produced at evaluation time and requires no subsequent reconstruction.

The following sections describe the structure and content of each evidence category in technical detail.

3.1 Decision Artifact Evidence

A decision artifact is the primary evidence unit of a QODIQA deployment. It is produced by the Decision Artifact Generator at the conclusion of every evaluation, regardless of outcome. Each artifact contains the following elements.

DECISION ARTIFACT STRUCTURE

```
DecisionArtifact = {
    "decision_id":        str,      # SHA-256 derived, 24-char hex
    "timestamp_utc":      datetime, # Evaluation time, frozen
    "outcome":            str,      # ALLOW | DENY | ALLOW_WITH_RESTRICTIONS
    "restrictions":       list,     # If ALLOW_WITH_RESTRICTIONS
    "context_ref":        str,      # Reference to frozen EvaluationContext
    "consent_ref":        str,      # Reference to resolved ConsentState
    "policy_version":     str,      # Exact policy version applied
    "ruleset_version":    str       # Exact ruleset version applied
}
```

The `decision_id` field is deterministically derived from the canonical representation of the evaluation context and outcome. Given the same inputs, the identifier is always identical. This property enables independent verification: any auditor with access to the frozen context and outcome can recompute the identifier and confirm its authenticity without trust in the originating system.

EVIDENCE COMPLETENESS REQUIREMENT

- All fields must be present and non-null
- Artifacts must be retained for the full retention period
- Artifacts must not be modified after creation
- DENY outcomes must be retained with the same completeness as ALLOW outcomes

3.2 Consent State Evidence

Consent state evidence demonstrates that the consent resolved at evaluation time was valid, current, and appropriately scoped. The frozen evaluation context captures a reference to the resolved consent record, not the record itself, ensuring that the decision artifact remains stable even if the consent state subsequently changes.

For audit purposes, organizations must be able to produce the consent state record identified by the reference in the frozen context and demonstrate that it was in a valid, non-revoked state at the evaluation timestamp. This requires that consent stores retain historical consent state, including timestamps of revocation, expiry, and scope modifications, for the full audit retention period.

CONSENT STATE RECORD (AUDIT-RELEVANT FIELDS)

```
ConsentStateRecord = {
    "consent_id":     str,
    "subject_id":     str,
    "purpose":        str,
    "scopes":         list[str],
    "valid_from":     datetime,
    "valid_until":    datetime,
    "revoked":        bool,
    "revoked_at":     datetime | None,
    "created_at":     datetime,
```

```
    "record_version": int
}
```

The audit claim that must be substantiable from this evidence is: at the time of evaluation, the referenced consent record existed, was not revoked, had not expired, and covered the purpose and scopes declared in the execution request.

3.3 Policy Version Evidence

Policy version evidence demonstrates that the rule set applied during evaluation has not been altered since the decision was recorded. Because QODIQA policies are immutable and versioned, this evidence takes the form of a policy version snapshot retained alongside its cryptographic digest.

POLICY VERSION SNAPSHOT STRUCTURE

```
PolicyVersionSnapshot = {
    "version":          str,
    "effective_from":   datetime,
    "effective_until":  datetime | None,
    "rules_digest":     str,     # SHA-256 of canonical rule representation
    "rules_ref":        str,     # Content-addressed reference
    "authored_by":      str,
    "approved_by":      str,
    "approved_at":      datetime
}
```

Policy version snapshots must be retained without modification. Any change to a policy ruleset produces a new version with a new digest. The digest of a version applied in a past decision can be recomputed from the retained rules content and compared to the recorded value, providing cryptographic proof of immutability.

3.4 Replay Verification Evidence

Replay verification evidence is produced by executing the decision replay protocol against a decision artifact and its associated frozen context. It demonstrates that the system's evaluation function is deterministic: the same inputs produce the same outcome, independent of when or where the evaluation is performed.

REPLAY VERIFICATION RECORD

```
ReplayVerificationRecord = {
    "original_decision_id":  str,
    "replay_decision_id":    str,
    "match":                 bool,
    "replay_timestamp_utc":  datetime,
    "replay_environment":    str,
    "performed_by":          str
}
```

A replay verification record where `match == true` and both decision identifiers are equal constitutes positive proof of determinism for that decision. Systematic replay verification across a representative sample of decisions provides confidence in the correctness of the evaluation function across time and environment.

1 Replay verification records themselves become part of the evidence pack and should be retained alongside the decision artifacts they reference.

2 A mismatch in replay verification indicates either a system integrity issue or an evidence corruption event. Both must be treated as critical findings and investigated prior to any audit submission.

SECTION 4

Regulatory Compliance Mapping

The QODIQA framework is designed to be regulation-agnostic at the enforcement layer. It does not interpret law or encode regulatory obligations directly. However, its structural properties map onto obligations arising under data

protection law, AI regulation, and sector-specific compliance frameworks in ways that can be demonstrated through the evidence architecture described in Section 3.

The following mappings are illustrative rather than exhaustive. Organizations operating in regulated environments must conduct their own legal and compliance analysis to confirm applicability. Translations from QODIQA system properties to regulatory obligations are presented as structural correspondences, not legal opinions.

4.1 General Data Protection Regulation (GDPR) Mapping

TABLE 2. QODIQA PROPERTIES MAPPED TO GDPR OBLIGATIONS

GDPR OBLIGATION	RELEVANT ARTICLE	QODIQA SYSTEM PROPERTY
Lawfulness of processing	Art. 6	Explicit consent resolution prior to execution
Purpose limitation	Art. 5(1)(b)	Purpose binding in intent declaration and consent scope
Data minimisation	Art. 5(1)(c)	Data type declaration in closed intent contract
Storage limitation	Art. 5(1)(e)	Time-bounded consent validity, retention restrictions in ALLOW_WITH_RESTRICTIONS
Accountability	Art. 5(2)	Decision artifacts as verifiable proof of authorized processing
Records of processing	Art. 30	Decision artifact store with actor, purpose, and outcome
Right of access	Art. 15	Decision traceability by subject identifier
Right to erasure	Art. 17	Consent revocation reflected immediately; future decisions denied

4.2 EU AI Act Mapping

The EU AI Act establishes obligations for providers and deployers of high-risk AI systems, including requirements for transparency, human oversight, technical documentation, and record-keeping. QODIQA's enforcement architecture maps

onto several of these obligations.

TABLE 3. QODIQA PROPERTIES MAPPED TO EU AI ACT OBLIGATIONS

EU AI ACT REQUIREMENT	QODIQA SYSTEM PROPERTY
Technical documentation (Art. 11)	Policy version snapshots with authorship and approval chain
Record-keeping (Art. 12)	Decision artifact store with frozen evaluation context
Transparency to deployers (Art. 13)	Deterministic outcomes with explicit restriction declarations
Human oversight measures (Art. 14)	Separation of human responsibility and mechanical enforcement
Accuracy and robustness (Art. 15)	Fail-closed semantics; no partial evaluation or best-effort decisions

4.3 Sectoral Obligations

Sector-specific regulatory frameworks in financial services, healthcare, and critical infrastructure impose additional obligations that intersect with the QODIQA evidence architecture. While comprehensive mapping across all sectoral instruments is outside the scope of this document, three structural properties are broadly applicable.

Immutable audit trails are required by financial services regulators and healthcare data protection authorities in most jurisdictions. QODIQA's decision artifacts satisfy this requirement structurally: they are produced at evaluation time, are not modifiable after creation, and carry cryptographic identifiers derived from their content.

Automated decision records are required under algorithmic accountability frameworks applicable in financial credit, insurance, and employment contexts. QODIQA's decision artifacts contain the information required to explain the factual basis of an automated decision, namely the declared intent, resolved consent state, and policy version applied, without exposing raw data or model internals.

Incident evidence preservation is required under breach notification and incident reporting frameworks. QODIQA denial records, produced when execution is refused due to consent failure or policy mismatch, constitute pre-existing evidence of enforcement activity that may be relevant to incident investigation and regulatory notification.

Organizations deploying QODIQA in regulated sectors should conduct a gap analysis between the evidence categories described in this document and the specific evidence requirements of applicable sectoral instruments, retaining the results as part of their compliance documentation.

SECTION 5

Audit Trail Requirements

An audit trail in the context of QODIQA is not a passive log. It is a structured, integrity-protected sequence of decision artifacts, consent state records, and policy version snapshots that collectively constitute verifiable proof of authorized AI execution over a defined period. The following sections specify the structural, retention, and integrity requirements that an audit trail must satisfy.

5.1 Structural Requirements

Every decision produced by a QODIQA deployment must generate a corresponding entry in the audit trail. There are no excluded decision types. Denials, approvals, and approvals with restrictions must all be recorded with equal completeness. Selective retention of only allow decisions is a structural failure that invalidates the audit trail.

Each audit trail entry must contain, at minimum: the complete decision artifact, the reference to the frozen evaluation context, a reference to the consent state record valid at evaluation time, and the policy version identifier with its associated digest.

Audit trail entries must be written atomically with the decision. A decision that is communicated to a downstream system without a corresponding audit trail entry is an enforcement failure. Systems must be designed so that audit trail write failures result in decision denial, not silent continuation.

5.2 Retention Policy

Audit trail retention periods must be defined prior to deployment and reviewed against applicable regulatory requirements. Where regulatory retention periods differ across applicable instruments, the longest applicable period governs.

> **MINIMUM RETENTION REQUIREMENTS**
>
> Decision artifacts and associated frozen contexts must be retained for a minimum period consistent with the longest applicable regulatory obligation. Policy version snapshots must be retained for the lifetime of the deployment plus the maximum applicable audit trail retention period. Consent state records, including modification history, must be retained for the same period as decision artifacts that reference them.

Retention policy must be documented, version-controlled, and included in the evidence pack submitted for audit. Changes to retention policy must be recorded with effective dates and must not result in early deletion of evidence that falls within the retention period of the superseded policy.

5.3 Integrity Verification

The integrity of the audit trail must be verifiable through cryptographic means. Reliance on access controls or procedural safeguards alone is insufficient: evidence that cannot be independently verified for integrity cannot be relied upon to prove that it has not been altered.

Integrity verification is performed at two levels. At the artifact level, the `decision_id` field of each artifact is a deterministic function of its content. Recomputing this identifier and comparing it to the recorded value confirms that the

artifact has not been modified. At the trail level, a hash chain or equivalent mechanism links audit trail entries in sequence, enabling detection of deletions or insertions that would not be detectable through individual artifact verification alone.

Integrity verification must be performed at regular intervals, not solely in anticipation of an audit. Organizations operating at maturity level 3 perform continuous automated integrity verification and produce periodic integrity attestation records as part of their evidence pack.

1 Integrity verification failures must be treated as critical findings, logged, and investigated. The investigation record and its outcome must be retained as part of the audit trail for the period in which the failure occurred.

SECTION 6

Verification Procedures

The verification procedures defined in this section are executed as part of audit preparation and ongoing operational assurance. They are not substitutes for external audit; they are the internal controls that make external audit defensible.

6.1 Pre-Audit Checklist

The following checklist must be completed and documented prior to any audit submission. Each item corresponds to a structural requirement described in this document.

EVIDENCE COMPLETENESS

- All decision artifacts for the audit period are present and complete
- Denial records are present with equal completeness to allow records
- Frozen evaluation contexts are retrievable for all decision artifacts
- Consent state records exist for all consent references in decision artifacts
- Policy version snapshots exist for all policy versions referenced in decision artifacts
- Replay verification records exist for at least the required sample coverage

INTEGRITY VERIFICATION

- Decision artifact identifiers have been recomputed and match recorded values
- Policy version digests have been recomputed and match recorded values
- Audit trail sequence integrity has been verified for the audit period
- No integrity failures are outstanding without resolved investigation records

RETENTION COMPLIANCE

- Retention policy is documented and version-controlled
- No evidence within the retention window has been deleted or modified
- Retention periods have been verified against current regulatory requirements

6.2 Decision Replay Protocol

The decision replay protocol is the primary verification procedure for demonstrating determinism. It is executed by re-evaluating a selection of past decisions using their frozen evaluation contexts and confirming that the computed decision identifiers match the recorded values.

REPLAY PROTOCOL EXECUTION

```
def replay_decision(artifact: DecisionArtifact) → ReplayVerificationRecord:
    context = load_frozen_context(artifact.context_ref)
    policy  = load_policy_version(artifact.policy_version)

    replayed_outcome = evaluate(context, policy)
    replayed_id = decision_id(context, replayed_outcome)

    return ReplayVerificationRecord(
        original_decision_id = artifact.decision_id,
        replay_decision_id   = replayed_id,
        match                = replayed_id == artifact.decision_id,
        replay_timestamp_utc = utcnow(),
        replay_environment   = current_environment(),
        performed_by         = current_operator()
    )
```

The protocol must be executed against a sample that is representative of the decision population for the audit period. The sample must include decisions with each outcome type (ALLOW, DENY, ALLOW_WITH_RESTRICTIONS), decisions from different policy versions, and decisions involving different actors and purposes. All replay verification records must be retained and included in the evidence pack.

6.3 Policy Immutability Check

The policy immutability check verifies that no policy version applied during the audit period has been modified after its effective date. The check is performed by recomputing the rules digest for each policy version and comparing it to the digest recorded in the policy version snapshot.

IMMUTABILITY VERIFICATION

```
def verify_policy_version(snapshot: PolicyVersionSnapshot) → bool:
    rules    = load_rules_content(snapshot.rules_ref)
```

```
    computed = sha256(canonicalize(rules))
    return computed == snapshot.rules_digest
```

Any version for which the computed digest does not match the recorded value must be flagged as a potential integrity failure and investigated before the audit submission proceeds.

6.4 Consent Coverage Analysis

Consent coverage analysis verifies that every allow decision in the audit period is traceable to a valid, non-revoked consent state that was in effect at the evaluation timestamp. The analysis is performed by joining decision artifacts with consent state records on the consent reference and checking temporal validity.

The output of consent coverage analysis is a coverage report indicating, for each allow decision, whether a valid consent state reference exists, whether the consent was in scope for the declared purpose, and whether the consent was unrevoked at the evaluation timestamp. Decisions for which coverage cannot be confirmed must be investigated and documented in the audit trail.

> Consent coverage analysis is a verification procedure, not an enforcement mechanism. Its purpose is to surface gaps in the evidence record that may indicate upstream consent capture or consent store issues. It does not retroactively alter decision outcomes.

SECTION 7

Evidence Pack Structure

A QODIQA evidence pack is a structured, self-contained collection of evidence materials organized for presentation to a regulator, auditor, or institutional counterparty. The pack must be complete, internally consistent, and accompanied by an index that maps each item to the regulatory obligation or audit objective it satisfies.

7.1 Required Documents

Every evidence pack must include the following mandatory documents. Additional documents may be included at the discretion of the submitting organization, but the mandatory set is the minimum required for a submission to be considered complete.

MANDATORY EVIDENCE PACK DOCUMENTS

- Evidence pack index with regulatory or audit objective mapping
- Decision artifact export for the audit period (complete, all outcome types)
- Consent state record export for all referenced consent identifiers
- Policy version snapshot set for all versions referenced in the audit period
- Replay verification records for the required sample
- Pre-audit checklist with completion status and signatory
- Audit trail integrity attestation with verification methodology
- Retention policy documentation current as of the audit period

7.2 Artifact Inventory

The evidence pack must include a machine-readable artifact inventory that lists every decision artifact in the export with its identifier, timestamp, outcome, and policy version. The inventory enables auditors to confirm that the export is complete and to locate specific decisions for verification.

ARTIFACT INVENTORY ENTRY

```
ArtifactInventoryEntry = {
    "decision_id":    str,
    "timestamp_utc":  str,   # ISO 8601
    "outcome":        str,
```

```
    "actor_id":       str,
    "purpose":        str,
    "policy_version": str
}
```

The inventory must be signed with a digest computed over the complete set of decision identifiers in the export, enabling verification that no entries have been added, removed, or modified after the inventory was generated.

7.3 Submission Format

Evidence packs must be submitted in a format that preserves the integrity of all included artifacts and enables independent verification without proprietary tooling. The canonical submission format is a structured archive containing JSON-encoded artifacts, human-readable index documents, and a top-level manifest that identifies the archive contents and their cryptographic digests.

Where a regulator or auditor specifies an alternative submission format, the organization must confirm that the alternative format preserves the structural completeness and integrity properties required by this document before adopting it. Conversion to an alternative format must not result in loss of any mandatory fields or integrity references.

1 Evidence packs should be transmitted over secure channels with end-to-end integrity protection. The transmission method should be documented in the evidence pack index.

2 Organizations retaining copies of submitted evidence packs should ensure that copies are subject to the same integrity protection and retention requirements as the originals.

SECTION 8

Operational Readiness

Operational readiness for audit is a continuous state maintained through normal system operation, not a preparation effort triggered by an impending audit. The following sections address the operational practices that sustain continuous

audit readiness and the specific handling requirements for incidents and denial records.

8.1 Continuous Readiness Practices

Continuous audit readiness requires that the evidence generation, retention, and integrity verification procedures described in this document are embedded in operational workflows rather than treated as exceptional audit-time activities.

Organizations should maintain a running audit readiness dashboard that tracks the completeness of the four evidence categories, the currency of integrity verification, the sample coverage of replay verification, and the remaining time before the earliest evidence retention deadline. Any shortfall against these metrics should trigger an operational response before it becomes an audit finding.

Periodic internal audit exercises, in which a simulated evidence pack is assembled for a historical period and subjected to the verification procedures in Section 6, provide confidence that the full end-to-end audit process functions correctly before it is required in a regulatory context.

8.2 Incident Evidence Handling

When an incident occurs that may be subject to regulatory investigation, the evidence materials relevant to the incident period must be preserved with additional protections to prevent alteration or deletion. This may require creating a preserved snapshot of the audit trail segment corresponding to the incident period before any operational changes are made to systems that generated the relevant decisions.

The incident evidence record should include the preserved snapshot, a chronological summary of decisions made during the incident period, denial records if the incident involved enforcement failures or attempted bypasses, and any integrity verification records covering the incident period.

8.3 Denial Record Preservation

Denial records are among the most operationally significant evidence materials in a QODIQA deployment. They demonstrate that the enforcement layer was active and functioning correctly, and they provide evidence that consent violations or policy mismatches were detected and blocked rather than silently permitted.

Denial records must be preserved with the same completeness and integrity as allow records. They must include the full decision artifact, including the frozen evaluation context that shows which consent or policy condition caused the denial. In regulated contexts, denial records may be requested by regulators as evidence of effective consent enforcement, independent of whether any incident occurred.

> The preservation of denial records is not merely a compliance obligation; it is a demonstration of the value of the enforcement layer. An organization that cannot produce denial records cannot demonstrate that its consent enforcement was active during any given period.

SECTION 9

Limitations and Scope

The evidence architecture and procedures described in this document are subject to constraints that arise from the deliberate design boundaries of the QODIQA framework. These constraints are not temporary gaps; they are structural choices that preserve the determinism and defensibility of the system.

9.1 Evidence Does Not Substitute for Legal Analysis

The regulatory compliance mappings in Section 4 are structural correspondences, not legal opinions. Organizations must conduct their own legal analysis to confirm that QODIQA evidence satisfies the specific requirements of applicable

instruments in their jurisdiction and sector. The evidence pack structure defined in this document is a starting point, not a substitute for legal advice.

9.2 Upstream Consent Capture Is Out of Scope

QODIQA enforces consent at the execution layer; it does not capture consent from data subjects. The validity of the consent captured upstream and stored in the consent store is a separate concern that falls outside the scope of this document. Evidence of upstream consent capture mechanisms must be maintained separately and cross-referenced in the evidence pack where relevant.

9.3 Content and Semantic Evidence Is Not Produced

QODIQA does not inspect data content or produce evidence relating to the substance of data processed by AI systems. Evidence of data minimization, content appropriateness, or model behavior must be produced by other components of the AI governance stack and included in the evidence pack as supplementary materials where regulatory requirements demand it.

9.4 Evidence Covers Enforcement, Not Outcome

The evidence produced by QODIQA demonstrates that enforcement was correctly applied at the execution layer. It does not constitute evidence that the AI system's output was appropriate, accurate, or free from harm. Outcome-level governance evidence remains the responsibility of model governance and operational monitoring systems operating downstream of QODIQA.

SECTION 10

Conclusion

The shift from documentation-centric compliance to execution-layer enforcement changes the nature of audit evidence. In conventional governance models, an audit reconstructs what happened from logs, records, and testimony. In a QODIQA deployment, an audit verifies what happened against stable, cryptographically identified artifacts that were produced at the moment of execution and have not changed since.

This distinction is not cosmetic. It changes the burden of proof from plausibility to reproducibility. An organization that can replay any past decision and obtain an identical result, trace that result to an immutable policy version and a time-bounded consent state, and present this evidence in a structured, integrity-protected pack is in a fundamentally different position from one that can only offer logs and assertions.

The audit readiness and evidence architecture described in this document is a consequence of the QODIQA design rather than an addition to it. Organizations that deploy QODIQA correctly, and maintain the operational practices described here, are continuously audit-ready. Evidence packs are assembled from materials that already exist, verified by procedures that are already embedded in operational workflows, and submitted in formats that preserve the structural properties that make them verifiable.

As regulatory scrutiny of AI systems intensifies, the ability to demonstrate authorized, purposeful, and temporally bounded AI execution will become a baseline expectation. QODIQA provides the infrastructure through which that demonstration moves from aspiration to proof.

DOCUMENT STATUS

Document Status and Corpus Alignment

This document is a normative component of the QODIQA specification corpus. It defines the audit readiness framework, evidence architecture, compliance mapping, and verification procedures applicable to organizations deploying QODIQA as a runtime consent enforcement layer for artificial intelligence systems.

This document derives its structural authority from the properties established in the QODIQA Technical Whitepaper and should be read in conjunction with that document. The evidence categories, verification procedures, and evidence pack structure defined here are direct translations of the determinism, auditability, and proof guarantees described in the whitepaper into operational and procedural form.

This document should be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement
- QODIQA: Terminology and Normative Definitions

- QODIQA: Threat Model and Abuse Case Specification
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Governance Charter for the QODIQA Standard Corpus

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dușescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-ARP-2026-001
TITLE	Audit Readiness and Evidence Pack
SUBTITLE	Deterministic Runtime Consent Enforcement for Artificial Intelligence Systems
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Evidence Pack
DOCUMENT STATUS	Normative
GOVERNING AUTHORITY	QODIQA Governance Charter
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 31 PART V · CONTEXTUAL, ANALYTICAL AND PROCESS SUPPORT

Use Case Dossiers for Runtime Consent Enforcement Deployments

Sector-specific deployment analyses and risk-containment frameworks for deterministic AI consent enforcement across four industry sectors.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-UCD-2026-001	1.0	2026	Use Case Dossiers	Public Release	NOT STATED

SCOPE AND PURPOSE

This document presents four sector-specific use case dossiers for deployments of QODIQA deterministic runtime consent enforcement. Each dossier defines the runtime consent failure surface in a target sector, analyzes the execution path with and without enforcement, provides a structured risk surface reduction analysis, specifies the evidentiary artifact model, documents operational constraints and bypass vector considerations, and includes a deployment maturity model.

The sectors addressed are: Healthcare and Clinical AI Systems; Financial Services and Algorithmic Decisioning; Media and Content Generation Platforms; and Enterprise AI Assistants and Knowledge Systems. These dossiers are a technical operational annex to the QODIQA Core Standard and do not constitute legal advice or sector-specific compliance assessments. Sector-specific regulatory compliance requires qualified legal counsel in the applicable jurisdiction.

QODIQA defines enforcement strictly at the execution boundary. No execution outside this boundary is considered valid within the system model.

DOSSIER I OF IV

QODIQA Use Case Dossier - Healthcare and Clinical AI Systems

Deterministic Runtime Consent Enforcement in Clinical and Health Information Environments

VERSION	DOCUMENT ID	DATE	STATUS
1.0	QODIQA-UCD-2026-001-D1	2026	Active - Public Release

SECTOR-SPECIFIC ENFORCEMENT CRITICALITY

Life-critical decision pathways where inference errors are irreversible require enforcement at the execution boundary, not at the policy layer. Multi-layer consent regimes governing PHI, genetic data, and sensitive health categories create a complex consent surface that cannot be managed through static authorization alone. The irreversible harm potential of unauthorized clinical AI execution establishes healthcare as the highest-criticality sector for deterministic runtime consent enforcement.

This establishes this sector as a high-criticality domain requiring deterministic enforcement at execution time.

D1 - SECTION 1

Sector Context Overview

Healthcare AI systems encompass clinical decision support systems (CDSS), diagnostic imaging classifiers, predictive risk stratification models, autonomous medication management pipelines, patient communication agents, and multi-institution data exchange brokers. Deployment patterns involve AI components embedded within EHR platforms, PACS infrastructure, clinical workflow orchestrators, and interoperability layers such as HL7 FHIR APIs.

Multi-agent exposure is inherent: a single clinical AI pipeline may traverse patient identity resolution, diagnostic inference, prescription recommendation, and billing code generation - each step potentially governed by distinct consent instruments. Data sensitivity is highest-class: PHI, genetic, biometric, mental health, and reproductive health records each carry specialized consent regime requirements.

1.1 Regulatory Environment (Reference Only)

Reference frameworks include HIPAA (US), GDPR and national health data laws (EU), EU AI Act High-Risk AI classification for medical device software, FDA SaMD guidance, and emerging national health AI governance frameworks. This dossier does not assess compliance with any specific regulatory instrument.

D1 - SECTION 2

Current Runtime Consent Failure Surface

The following taxonomy classifies failure modes by enforcement breakdown type at runtime.

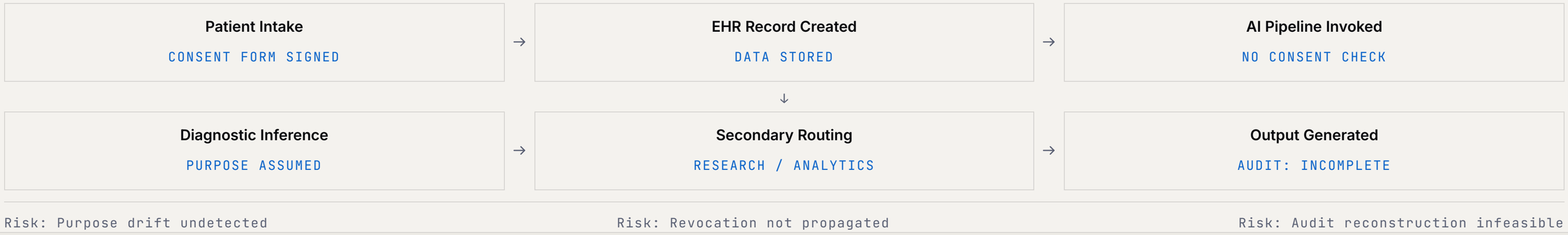
AUTHORIZATION MISINTERPRETATION	ASSUMED CONSENT	Broad intake consent treated as perpetual authorization across all downstream AI functions including diagnostic models, research pipelines, and billing agents - regardless of original purpose scope.
CONSENT DRIFT	PURPOSE DRIFT	Patient data collected under treatment consent silently routed to AI training pipelines, population health analytics, or commercial research brokers. The purpose boundary is defined in policy but not enforced at the execution boundary.
EXECUTION GAP	REVOCATION GAP	When a patient revokes consent, revocation is recorded in a consent management system. AI agents operating in parallel execution contexts continue processing previously authorized data batches until revocation propagates - which may take hours or never reach distributed cache layers.
AUDIT FAILURE	AUDIT GAPS	AI inference events logged at application layer; not linked to the specific consent token that authorized underlying data access. Audit reconstruction requires manual correlation across EHR logs, AI event logs, and consent registry exports - operationally infeasible at scale.
AUTHORIZATION MISINTERPRETATION	AUTHORIZATION VS. CONSENT	Role-based access controls authorizing clinical staff are conflated with patient consent to AI processing. A physician's authorization to view a record does not constitute patient consent for machine learning inference over that record.

MULTI-AGENT RISK	MULTI-AGENT OPACITY	In federated deployments, downstream nodes execute inference on data received from upstream orchestrators without verifying the consent token that authorized the original release. Processing occurs without local consent verification.

D1 - SECTION 3

Execution Path Without Deterministic Enforcement

FIG D1-A - UNENFORCED CLINICAL AI EXECUTION CHAIN

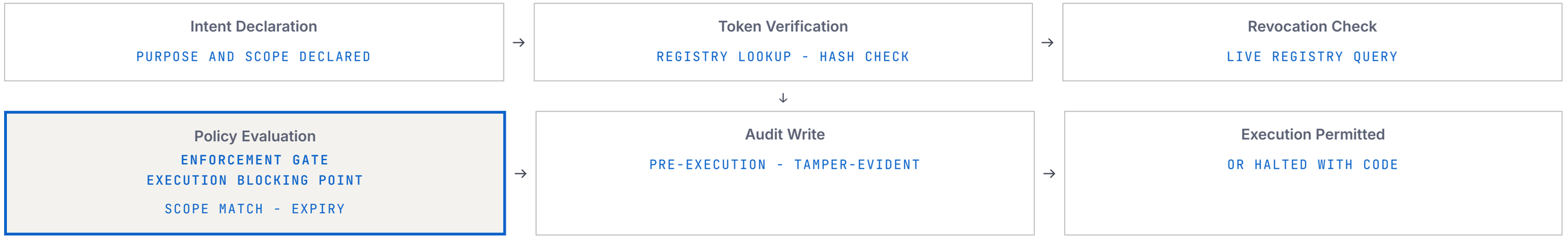


D1 - SECTION 4

Execution Path With QODIQA Runtime Enforcement

Under QODIQA deployment, each AI execution step is preceded by a synchronous consent gate. The gate verifies a structured consent token against declared intent, purpose scope, and revocation state before permitting data access or action execution.

FIG D1-B - QODIQA-ENFORCED CLINICAL AI EXECUTION CHAIN



Gate: Synchronous - Deterministic Artifact: Immutable per-event Halt: Purpose mismatch - BLOCKED **SINGLE DETERMINISTIC ENFORCEMENT BOUNDARY (NON-BYPASSABLE)**

WITHOUT ENFORCEMENT	WITH QODIQA ENFORCEMENT
× Purpose assumed from intake consent	→ Purpose declared and matched per execution step
× Revocation propagation: eventual, unreliable	→ Revocation: synchronous gate check before each action
× Secondary routing: unchecked	→ Secondary routing: blocked if scope mismatch
× Audit: post-hoc reconstruction only	→ Audit: written before execution, replayable
× Multi-site nodes: no local verification	→ Federated nodes: each verifies locally against registry
Consent: documentation artifact	Consent: executable control plane

D1 - SECTION 5

Risk Surface Reduction Analysis

TABLE D1-1 - RISK REDUCTION BY CATEGORY, HEALTHCARE SECTOR

RISK CATEGORY	WITHOUT ENFORCEMENT	WITH QODIQA	RESIDUAL RISK
Purpose Drift	Undetected; downstream AI functions consume data under broad intake consent	Purpose scope matched at each gate; mismatched requests blocked with audit record	Scope definition errors in token authoring; policy misconfiguration
Revocation Propagation	Revocation recorded in consent system; AI agents continue until cache expiry	Live revocation check at each gate; revoked tokens return DENY, execution halted	Registry availability; race condition within millisecond window at gate query time
Audit Completeness	Inference events logged at application layer; consent instrument not linked	Pre-execution audit record includes token ID, declared intent, evaluation result, timestamp	Log storage integrity; audit system availability; log export fidelity
Cross-System Exposure	Federated nodes receive data without local consent verification	Each node performs independent gate check; token propagated with data payload	Token forgery risk if cryptographic profile not implemented; registry synchronization latency
Authorization / Consent Conflation	RBAC authorization treated as patient consent for AI processing	Consent gate is distinct from access control layer; each evaluated independently	Integration design must correctly separate RBAC from consent gate

TABLE D1-3 - OPERATIONAL RISK DELTA (INDICATIVE)

RISK CATEGORY	WITHOUT ENFORCEMENT	WITH QODIQA
Unauthorized Inference	High	Eliminated at gate boundary
Purpose Drift	High	Near-zero (bounded by token scope quality)
Revocation Latency	Hours to days	Bounded by defined TTL or live query
Audit Reconstruction	Manual - operationally infeasible	Deterministic replay from immutable records
Multi-Agent Propagation	Unverified across nodes	Independent gate check per node

Interpretation: QODIQA collapses runtime consent risk from probabilistic enforcement to deterministic execution control.

D1 - SECTION 6

Evidence and Artifact Model

Each QODIQA gate evaluation produces a structured, tamper-evident artifact. These artifacts constitute the evidentiary record for regulatory inquiry, internal audit, or patient access requests.

The following minimal artifact set defines the irreducible enforcement surface required for deterministic validation.

MINIMAL ARTIFACT SET (REQUIRED)

Every QODIQA-conformant gate evaluation MUST produce at minimum the following artifact types. Additional artifacts defined in this section extend the minimum set.

- Consent Token (hash-bound)
- Intent Declaration Object
- Enforcement Decision Record
- Audit Log Entry (append-only)

ART-D1-01 **Consent Gate Evaluation Record** Per-execution record: token ID, declared intent, purpose scope evaluated, evaluation result (PERMIT / DENY / BLOCK), timestamp, executing agent identifier, data subject reference.	**ART-D1-02** **Revocation Check Log** Timestamped log of each revocation registry query: query time, token state at query, response latency. Demonstrates live check was performed, not cached result applied.

ART-D1-03

Purpose Scope Mismatch Report

Generated on BLOCK events: declared purpose vs. token-authorized scope, blocking rule reference, agent identity, data subject reference.

ART-D1-04

Federated Node Verification Receipt

Per-node record confirming independent gate check was performed, token verified, registry queried. Enables audit reconstruction across distributed execution environments.

ART-D1-05

Consent Token Lineage Record

Maps data subject consent tokens to all execution events authorized under those tokens. Supports patient right-of-access requests and organizational audit of AI data use scope.

ART-D1-06

Replay Package

Structured export combining gate evaluation records, token state snapshots, and policy version in effect at time of execution. Enables deterministic replay without requiring live system access.

Core Requirement: All artifacts are written before execution completes. An audit record exists for every gate evaluation regardless of whether execution was ultimately permitted or blocked. This property is non-negotiable under QODIQA Core Standard Section 7.

D1 - SECTION 7

Operational Constraints

OPERATIONAL CONSTRAINTS - HEALTHCARE DEPLOYMENT	
LATENCY	Each gate evaluation adds a synchronous network round-trip to the consent registry. In latency-critical workflows - real-time monitoring, emergency decision support - this overhead must be measured and accepted or mitigated via local token caching with bounded TTL. Caching introduces a revocation window that must be explicitly defined and accepted by the deploying organization.
REGISTRY AVAILABILITY	Registry unavailability must result in a defined default posture - typically DENY with logging - not silent PERMIT. Organizations must provision registry infrastructure accordingly and validate failover behavior under load.

TOKEN AUTHORING	Token authoring errors - overly broad purpose scope, incorrect subject identifiers, absent expiry - propagate into enforcement behavior. Token authoring discipline is an organizational and clinical informatics requirement, not a system property.
INTEGRATION DEPTH	Embedding QODIQA gates into existing EHR-integrated AI pipelines requires modification of each AI invocation point. Organizations must plan for phased rollout and maintain a registry of gated vs. ungated AI execution paths.
KEY MANAGEMENT	Cryptographic token verification requires PKI infrastructure or equivalent key management. Key rotation, revocation, and escrow must be planned as prerequisites, not post-deployment additions.
ORGANIZATIONAL DISCIPLINE	QODIQA enforces what tokens specify. If clinical consent processes are operationally inconsistent - consent obtained at incorrect scope, revocations not recorded promptly - enforcement will reflect those upstream gaps.

7.A Quantified Operational Characteristics (Illustrative, Non-Normative)

The following characteristics are illustrative only and subject to infrastructure architecture, registry topology, network conditions, and token payload size. They are provided to support architectural planning, not to constitute performance guarantees.

OPERATIONAL OVERHEAD REFERENCE - HEALTHCARE DEPLOYMENT · ILLUSTRATIVE - NON-NORMATIVE - NOT A PERFORMANCE GUARANTEE	
GATE LATENCY	Typical deployment observations indicate synchronous gate round-trip of `2 - 15 ms` under local registry with in-network token resolution. Latency increases to `20 - 80 ms` range under cross-region registry lookup. Figures vary substantially by infrastructure. No production guarantee is implied.
BATCH AMPLIFICATION	Batch processing pipelines issuing one registry query per patient record may generate `N × gate queries` for a batch of N records. At clinical-scale batches (10,000 - 500,000 records), registry infrastructure must be sized for sustained query throughput proportional to batch volume and frequency. Local caching with bounded TTL reduces amplification at cost of a defined revocation window.
AUDIT LOG GROWTH	Audit record volume is proportional to gate evaluation frequency. An AI pipeline issuing 5 gate checks per patient encounter at 1,000 encounters/day produces approximately `5,000 audit records/day` at that site alone. Multi-site and multi-pipeline deployments compound this proportionally. Retention, indexing, and access control for audit records require dedicated infrastructure planning.
STORAGE OVERHEAD	Structured audit records with token ID, timestamps, declared intent, and evaluation result typically occupy `1 - 4 KB per record` in normalized form. Long-term retention for regulatory purposes (commonly 6 - 10 years in health sectors) requires durable, immutable storage provisioning scaled to projected record volume.

COLD-START RECOVERY	Following a registry outage, the default posture must be DENY-on-unknown. Recovery time objective (RTO) for the registry directly determines the duration of AI pipeline interruption. Organizations must define and test registry RTO as part of business continuity planning before activating enforcement in production.

D1 - SECTION 7A

Enforcement Bypass and Adversarial Risk Considerations

QODIQA enforces only execution paths that are architecturally routed through the consent gate. Prevention of architectural bypass is a deployment responsibility. The following vectors represent conditions under which enforcement may be circumvented, absent deliberate architectural controls.

TABLE D1-2 - BYPASS VECTORS, HEALTHCARE SECTOR

BYPASS VECTOR	DESCRIPTION	ENFORCEMENT DEPENDENCY	RESIDUAL RISK
Direct Model Invocation	An AI agent or pipeline calls the inference model directly via internal API, bypassing the QODIQA gate layer entirely. Common in systems where the gate is implemented as an optional middleware rather than a mandatory ingress control.	Requires architectural enforcement: gate must be the exclusive path to model invocation. Network policy or API gateway controls must prevent direct model endpoint access.	Partial gate coverage deployments are directly vulnerable. Deployment must enumerate and close all direct invocation paths before activating enforcement claims.
Cached Token Replay	A previously valid token, cached locally after a successful gate evaluation, is replayed for a subsequent request after the underlying consent has been revoked. The gate is invoked but evaluates a stale cached result rather than querying the live registry.	Token cache TTL must be explicitly bounded and documented. Cache invalidation on revocation events requires an event-push mechanism from the registry to cache layers.	TTL-bounded caching introduces a defined revocation window. The window duration is a deployment decision and must be accepted and documented by the organization.
Registry Spoofing	An attacker or misconfigured system substitutes a fraudulent registry endpoint that returns PERMIT responses for all queries, without performing actual token validation.	Registry endpoint authentication must be cryptographically verified. TLS with certificate pinning or mutual TLS between gate and registry prevents endpoint substitution.	Residual risk in environments without strong registry authentication. Cryptographic profile implementation (QODIQA Security and Cryptographic Profile) is required to mitigate this vector.

BYPASS VECTOR	DESCRIPTION	ENFORCEMENT DEPENDENCY	RESIDUAL RISK
Token Forgery	A forged consent token, constructed to match expected format but not issued by an authorized token authority, is presented to the gate. Without cryptographic signature verification, the gate may accept the forged token.	All tokens must be cryptographically signed by an authorized issuer. Gate must verify signature chain before evaluating token contents. Unsigned tokens must be rejected.	In deployments where token signing is not implemented, this vector is fully exploitable. Signing implementation is a prerequisite for security-grade enforcement.
Shadow Pipeline	An alternate data extraction or processing path - a legacy ETL job, a direct database query, an unmanaged batch script - accesses PHI and feeds it to AI processes without passing through the QODIQA enforcement layer.	Requires comprehensive pipeline inventory. All data access paths to AI-consumed data must be identified and either gated or explicitly prohibited. This is an organizational and architectural governance requirement.	Shadow pipelines are commonly discovered during deployment audits of complex health IT environments. A pipeline inventory audit is recommended prior to enforcement activation.
Mis-Scoped Token Exploitation	A token with an overly broad purpose scope - issued in error during token authoring - is used to authorize AI processing well beyond the patient's actual consent. The gate permits the request because the token technically covers the declared purpose, even if the token scope was incorrectly defined.	Enforcement fidelity is bounded by token authoring quality. Token scope review and approval workflows are organizational controls that must complement technical enforcement.	Technical enforcement cannot detect a correctly-formatted token that was mis-scoped during issuance. Process controls at token issuance are the primary mitigation.
Stale Replication Window	In distributed registry deployments with replication lag, a node operating on a stale replica may return PERMIT for a token that has already been revoked on the primary registry node. The gate is invoked but against outdated state.	Replication lag bounds must be documented and published as part of the enforcement SLA. Critical use cases may require primary-node-only queries, accepting higher latency.	Replication-based deployment introduces a time-bounded window during which revocation is not fully effective across all nodes. This window must be accepted and documented.

Architectural Responsibility: QODIQA gate enforcement is effective only for execution paths architecturally routed through the gate. No enforcement mechanism can govern execution paths that do not pass through it. Deployment organizations bear responsibility for ensuring gate coverage is comprehensive, verified, and audited on a defined schedule.

Non-Bypass Guarantee: Any execution path that does not pass through the QODIQA enforcement gate is considered non-conformant and outside the system's guarantees. Enforcement is defined strictly at the execution boundary.

D1 - SECTION 8

Residual and Out-of-Scope Risks

OUTSIDE QODIQA ENFORCEMENT SCOPE - HEALTHCARE SECTOR
Ø Clinical AI model accuracy, bias, and diagnostic error rates. QODIQA verifies that execution was authorized; it does not evaluate whether the model produces clinically safe or accurate outputs.
Ø Training data compliance. Data used to train AI models prior to deployment is not governed by runtime consent enforcement. Historical data use and training data provenance are separate governance requirements.
Ø Content and quality of patient-facing consent disclosures. QODIQA enforces what tokens specify. Whether the underlying consent process adequately informs patients of AI processing is a clinical ethics and legal compliance matter.
Ø Physician liability and clinical decision accountability. Enforcement of consent boundaries at the AI execution layer does not alter the liability framework for clinical decisions made with AI assistance.
Ø Equity and fairness in AI-assisted clinical decisions. Disparate impact, demographic bias, and health equity considerations are model governance concerns outside the runtime consent enforcement layer.

D1 - SECTION 9

Institutional Closing - Healthcare Dossier

Clinical AI systems process data of the highest sensitivity under consent frameworks designed for human-to-human care relationships. Deterministic enforcement does not resolve the complexity of clinical consent law, nor does it substitute for the organizational discipline required to structure consent instruments correctly. It ensures that **execution cannot proceed without a verifiable, non-revoked, scope-matched consent token** - converting consent from a documentation artifact into an operational control boundary.

In healthcare AI deployments, the gap between consent documentation and consent enforcement is not a compliance nuance. It is a structural vulnerability in the authorization architecture of AI systems operating on irreplaceable personal health information. Deterministic enforcement closes that gap at the execution boundary.

APPENDIX A1 - DOSSIER I

Enforcement Deployment Maturity Levels (Illustrative)

TIER 1

Advisory Logging Only

GATE COVERAGE No enforcement gates active. Consent events logged passively from existing EHR audit trails. No execution blocking.

REGISTRY STATE Consent registry not integrated with AI pipelines. Tokens may be issued but are not verified at runtime.

AUDIT COMPLETENESS Partial. Application-layer logs only. No consent token linkage.

REVOCATION Not enforced at runtime. Revocation recorded in consent system only.

CERTIFICATION Does not meet deterministic enforcement criteria.

DEPLOYMENT SIGNAL Experimental

TIER 2

Partial Gate Coverage

GATE COVERAGE Gates active on selected AI pipelines (e.g., diagnostic inference only). Other pipelines - research routing, billing - remain ungated.

REGISTRY STATE Registry integrated for gated pipelines. Ungated pipelines operate outside enforcement perimeter.

AUDIT COMPLETENESS Partial. Gated pipeline events linked to tokens. Ungated events unlinked.

REVOCATION Enforced within gated scope only. Ungated pipelines retain revocation gap.

CERTIFICATION Does not meet deterministic enforcement criteria. Coverage gap must be documented.

DEPLOYMENT SIGNAL Controlled Deployment

TIER 3 - MINIMUM FOR DETERMINISTIC ENFORCEMENT

Full Deterministic Enforcement

GATE COVERAGE All AI execution paths gated. No ungated AI data access paths permitted. Coverage verified by deployment audit.

REGISTRY STATE Live registry integration with defined failover posture (DENY-on-unavailable). Revocation propagation SLA defined and tested.

TIER 4

Certified Conformance Deployment

GATE COVERAGE Full coverage as Tier 3, verified by external conformance assessor against QODIQA Certification Framework.

REGISTRY STATE Cryptographic token signing implemented per QODIQA Security and Cryptographic Profile. Key management governance in place.

AUDIT COMPLETENESS	Complete. Pre-execution audit record for every gate evaluation. Token ID linked to each record.
REVOCATION	Enforced at all gates. Revocation window bounded by documented TTL or primary-node query policy.
CERTIFICATION	Meets QODIQA Core deterministic enforcement standard. Eligible for Tier 3 conformance assessment.
DEPLOYMENT SIGNAL	Production-Critical

AUDIT COMPLETENESS	Complete with tamper-evident audit chain. Replay packages verified by conformance assessor. Export format certified.
REVOCATION	Revocation SLA documented, tested, and certified. Maximum revocation window formally declared in conformance statement.
CERTIFICATION	QODIQA Tier 4 Conformance Certificate issued. Renewal cadence defined per Certification Framework.
DEPLOYMENT SIGNAL	Regulatory-Grade

Note: Only Tier 3 and above meet the deterministic runtime enforcement criteria defined in QODIQA Core Standard. Tier 1 and Tier 2 deployments may not assert deterministic enforcement properties. Organizations operating at Tier 1 or 2 must not represent their deployments as QODIQA-conformant in regulatory disclosures or audit responses without explicit qualification of partial coverage scope.

DOSSIER II OF IV

QODIQA Use Case Dossier - Financial Services and Algorithmic Decisioning

Deterministic Runtime Consent Enforcement in Credit, Risk, and Automated Financial Decision Environments

VERSION	DOCUMENT ID	DATE	STATUS
1.0	QODIQA-UCD-2026-001-D2	2026	Active - Public Release

SECTOR-SPECIFIC ENFORCEMENT CRITICALITY

Systemic financial risk can propagate rapidly through automated decisioning pipelines before enforcement failures are detected. Regulatory exposure under automated decisioning frameworks requires a verifiable, replayable consent basis at each decision point. High-frequency automated execution compresses the window between an enforcement failure and its downstream consequences to milliseconds, making pre-execution gate enforcement the only operationally viable control mechanism.

This establishes this sector as a high-criticality domain requiring deterministic enforcement at execution time.

D2 - SECTION 1

Sector Context Overview

Financial services AI systems span credit scoring and underwriting models, fraud detection engines, algorithmic trading systems, AML pattern recognition, personalized product recommendation, and automated customer interaction agents. Deployment includes embedded AI modules within core banking platforms, standalone risk decisioning APIs, third-party data enrichment pipelines, and real-time transaction monitoring infrastructure.

Multi-agent exposure is significant: a single loan application may be evaluated by identity verification agents, credit bureau data consumers, behavioral scoring models, and fraud detection classifiers - each operating under potentially distinct data use authorizations. Consumer protection frameworks impose explicit data use constraints that create meaningful runtime consent obligations.

1.1 Regulatory Environment (Reference Only)

Reference frameworks include GDPR (Art. 22 automated decision rights), EU AI Act High-Risk AI classification for creditworthiness assessment, FCRA (US), PSD2 open banking data use constraints, and CCPA. This dossier does not constitute legal advice or compliance assessment.

D2 - SECTION 2

Current Runtime Consent Failure Surface

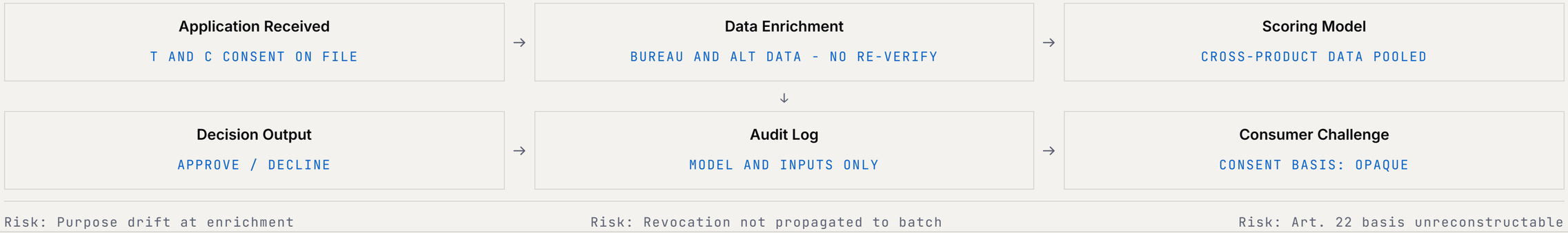
The following taxonomy classifies failure modes by enforcement breakdown type at runtime.

CONSENT DRIFT	ASSUMED CONSENT	Terms-and-conditions consent obtained at account opening authorizes all subsequent AI-driven data processing. The specific AI models, data categories, and decision types that will consume this consent are not enumerated or verified at each decisioning step.
CONSENT DRIFT	PURPOSE DRIFT	Data collected under fraud prevention authorization consumed by marketing propensity models or credit limit decisions. The purpose boundary exists in data governance policy but is not enforced at the point of data consumption by each AI pipeline.
CONSENT DRIFT	THIRD-PARTY ENRICHMENT	Bureau data, alternative data, and open banking feeds arrive with consent tokens or attestations from source systems. These tokens are not re-verified at the point of decisioning model consumption. The decisioning layer assumes validity without confirmation.
CONSENT DRIFT	CROSS-PRODUCT POOLING	Data from savings account behavior pooled into credit risk models under a unified consent framework. The consumer's original consent was for savings management, not creditworthiness inference. Purpose expansion occurs without a consent boundary check.
CONSENT DRIFT	REVOCATION GAP	Consumer consent withdrawal processed by compliance teams and propagated to data systems. Running AI batch jobs may process withdrawn-consent records during the propagation window, which can extend to multiple business days.
CONSENT DRIFT	AUDIT INCOMPLETENESS	Automated credit decision audit trails record model version and input variables but do not identify the specific consent instrument that authorized each input data category. Regulatory examination of automated decision legitimacy cannot be reconstructed with precision.

D2 - SECTION 3

Execution Path Without Deterministic Enforcement

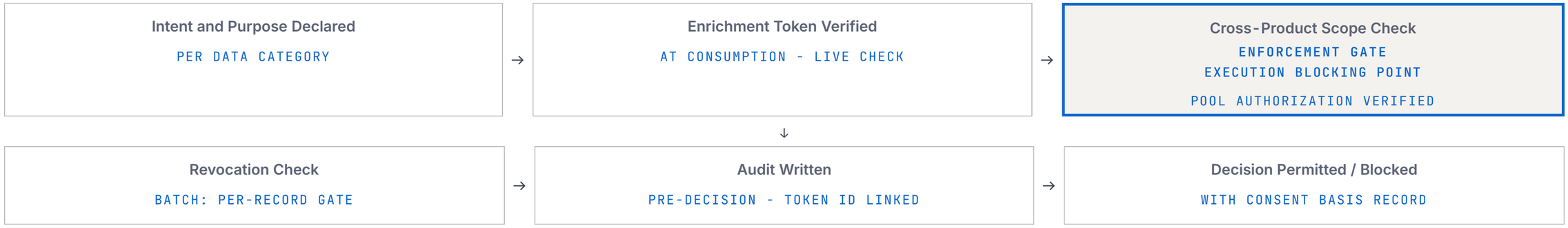
FIG D2-A - UNENFORCED ALGORITHMIC CREDIT DECISION CHAIN



D2 - SECTION 4

Execution Path With QODIQA Runtime Enforcement

FIG D2-B - QODIQA-ENFORCED ALGORITHMIC DECISION CHAIN



Gate: Per data category, per model input Artifact: Decision linked to token ID Challenge: Consent basis replayable **SINGLE DETERMINISTIC ENFORCEMENT BOUNDARY (NON-BYPASSABLE)**

WITHOUT ENFORCEMENT	WITH QODIQA ENFORCEMENT
× Enrichment tokens assumed valid	→ Enrichment tokens verified at consumption
× Cross-product pooling: policy-governed only	→ Cross-product pooling: scope-checked at gate
× Batch jobs: revocation lag unaddressed	→ Batch: per-record consent gate before scoring
× Audit: model inputs logged, consent basis absent	→ Audit: token ID linked to each decision record
× Art. 22 challenge: consent basis unreconstructable	→ Challenge: consent basis replayable from audit package
Decision legitimacy: asserted, not evidenced	Decision legitimacy: evidenced, replayable

D2 - SECTION 5

Risk Surface Reduction Analysis

TABLE D2-1 - RISK REDUCTION BY CATEGORY, FINANCIAL SERVICES SECTOR

RISK CATEGORY	WITHOUT ENFORCEMENT	WITH QODIQA	RESIDUAL RISK
Enrichment Data Consent	Third-party tokens assumed valid; no re-verification at consumption	Tokens verified at each consumption event; expired or revoked tokens blocked	Third-party token issuance quality; source consent process fidelity
Cross-Product Purpose Drift	Data pooled across products without per-pool consent scope check	Each pooling operation requires scope-matched token; mismatches blocked with record	Token scope definition must accurately reflect consumer consent language
Batch Revocation Lag	Revoked consent records processed in overnight batch until propagation completes	Per-record gate check in batch pipeline; revoked tokens return DENY, record skipped	Registry query latency at batch scale; infrastructure sizing requirements
Regulatory Explanation	Consent basis for automated decisions not captured at decision time	Pre-decision audit record links token ID to each decision; consent basis replayable	Audit record retention policy; export fidelity for regulatory examination
Consumer Challenge Rights	Art. 22 / FCRA challenge cannot be satisfied with current audit records	Replay package provides timestamped consent basis for any challenged decision	Challenge request processing still requires human review of replay output

TABLE D2-3 - OPERATIONAL RISK DELTA (INDICATIVE)

RISK CATEGORY	WITHOUT ENFORCEMENT	WITH QODIQA
Unauthorized Inference	High	Eliminated at gate boundary
Purpose Drift	High (cross-product pooling)	Near-zero (per-pool scope check)
Revocation Latency	Overnight batch cycle	Per-record gate at batch execution
Audit Reconstruction	Manual correlation - infeasible at scale	Deterministic - token ID linked per decision
Multi-Agent Propagation	Enrichment tokens not re-verified	Verified at each consumption event

Interpretation: QODIQA collapses runtime consent risk from probabilistic enforcement to deterministic execution control.

D2 - SECTION 6

Evidence and Artifact Model

The following minimal artifact set defines the irreducible enforcement surface required for deterministic validation.

MINIMAL ARTIFACT SET (REQUIRED)

Every QODIQA-conformant gate evaluation MUST produce at minimum the following artifact types. Additional artifacts defined in this section extend the minimum set.

- Consent Token (hash-bound)
- Intent Declaration Object
- Enforcement Decision Record
- Audit Log Entry (append-only)

ART-D2-01

Decision Gate Record

Per-decision record: token IDs for all data categories consumed, declared purpose, scope evaluation result, decision output reference, timestamp, model version identifier.

ART-D2-02

Enrichment Verification Receipt

Per-enrichment event: source token ID, verification timestamp, token state at verification, registry response time. Demonstrates live check at consumption.

ART-D2-03

Batch Consent Gate Log

Per-record log of batch scoring gate evaluations: subject identifier, token state, gate result. DENY events include revocation timestamp for gap analysis.

ART-D2-04

Automated Decision Explanation Package

Structured export for consumer challenge or regulatory examination: decision ID, timestamp, token IDs for all inputs, scope evaluation records, policy version. Enables deterministic replay of consent basis.

D2 - SECTION 7

Operational Constraints

OPERATIONAL CONSTRAINTS - FINANCIAL SERVICES DEPLOYMENT	
BATCH THROUGHPUT	Per-record consent gate checks in overnight batch scoring pipelines impose registry query volume that may be orders of magnitude higher than real-time deployment. Infrastructure must be sized for peak batch throughput. Local caching with bounded TTL may be operationally necessary; the revocation window introduced must be documented and accepted.
TOKEN GRANULARITY	Effective enforcement requires consent tokens at sufficient granularity to distinguish data category and processing purpose. Coarse-grained tokens reduce enforcement precision. Token architecture must align with the scope boundaries that regulations and consumer agreements establish.
THIRD-PARTY INTEROPERABILITY	Verification of third-party enrichment tokens requires source systems to issue tokens in formats compatible with the QODIQA verification layer. Where third-party providers do not support token issuance, proxy verification or organizational attestation mechanisms must be designed and their limitations documented.
MODEL INTEGRATION	Embedding consent gates at the data input layer of each scoring model requires integration with model serving infrastructure. Shadow-mode testing prior to enforcement activation is advisable to validate gate behavior without operational impact.

7.A Quantified Operational Characteristics (Illustrative, Non-Normative)

OPERATIONAL OVERHEAD REFERENCE - FINANCIAL SERVICES DEPLOYMENT · ILLUSTRATIVE - NON-NORMATIVE - NOT A PERFORMANCE GUARANTEE	
GATE LATENCY (REAL-TIME)	Typical deployment observations indicate gate round-trip of `2 - 15 ms` under local registry topology for real-time decisioning. Sub-5 ms is achievable with co-located registry. Latency characteristics must be measured under production load, not only synthetic benchmarks.
BATCH QUERY VOLUME	A batch scoring run of 1,000,000 consumer records with one gate check per record and three data category tokens per record generates approximately `3,000,000 registry queries per batch run`. Registry infrastructure must be capacity-planned against realistic batch schedules and concurrent run scenarios.
THIRD-PARTY ENRICHMENT	Enrichment token verification adds one registry round-trip per enrichment source per application. Applications using five enrichment sources incur approximately `five additional gate evaluations per application`, each subject to source registry latency rather than internal registry latency.
AUDIT RECORD VOLUME	At one audit record per gate evaluation, a financial institution processing 500,000 applications monthly with an average of 8 gate evaluations per application generates approximately `4,000,000 audit records monthly`. Regulatory retention requirements (commonly 5 - 7 years in financial sectors) determine long-term storage provisioning.

COLD-START RECOVERY	Registry downtime during a batch window results in deferred processing of all records in that window under a DENY-on-unavailable posture. Recovery time objective for the registry directly determines batch pipeline interruption duration. SLA commitments must account for this dependency.

D2 - SECTION 7A

Enforcement Bypass and Adversarial Risk Considerations

QODIQA enforces only execution paths architecturally routed through the consent gate. The following vectors represent conditions under which enforcement may be circumvented in financial services deployments, absent deliberate architectural and operational controls.

TABLE D2-2 - BYPASS VECTORS, FINANCIAL SERVICES SECTOR

BYPASS VECTOR	DESCRIPTION	ENFORCEMENT DEPENDENCY	RESIDUAL RISK
Direct Scoring Model Invocation	Scoring model accessed directly via internal API or batch script, bypassing QODIQA gate. Common in legacy batch frameworks where gate integration has not been completed.	Gate must be architecturally mandatory - not optional middleware. Internal model endpoints must be access-controlled to prohibit direct invocation from ungated callers.	Any ungated direct invocation path renders enforcement claims for that pipeline inaccurate. Deployment audit must enumerate and remediate all direct model access paths.
Enrichment Cache Replay	Third-party enrichment token verified once and cached. Subsequent applications consume the cached verification result even after the enrichment provider has revoked the authorization.	Enrichment verification cache TTL must be bounded and aligned with maximum acceptable revocation lag. Cache invalidation events from enrichment providers require a defined notification mechanism.	Without enrichment provider revocation push events, cache invalidation relies entirely on TTL expiry. The revocation window is bounded only by TTL duration.
Shadow Batch Pipeline	Legacy batch processes - overnight credit refresh, portfolio risk recalculation - access consumer data stores directly without passing through the QODIQA enforcement layer. These pipelines may predate gate integration and continue to operate in parallel.	Complete pipeline inventory required. Legacy batch jobs must be brought within the enforcement perimeter or formally prohibited. Inventory must be maintained as pipelines are added or modified.	Shadow batch pipelines are a high-probability finding in complex financial institutions. A formal pipeline audit is a prerequisite for enforcement integrity claims.

BYPASS VECTOR	DESCRIPTION	ENFORCEMENT DEPENDENCY	RESIDUAL RISK
Token Scope Exploitation	A consumer consent token issued with a broad scope - covering "all financial services processing" - is used to authorize cross-product data pooling that the consumer did not intend to authorize. The gate permits because the token scope matches, but the scope was defined at an insufficiently granular level.	Token scope granularity must be aligned with regulatory and consumer agreement language. Legal review of token scope definitions against applicable consumer protection law is required before deployment.	Enforcement is bounded by the quality of token scope design. A technically valid but legally inadequate token scope produces technically permitted but potentially non-compliant decisions.
Stale Registry Replication	Distributed registry nodes serving regional decisioning centers may operate on replicated state with defined lag. A consumer revocation event processed on the primary registry node may not be reflected on regional nodes within the replication window, resulting in PERMIT responses for revoked tokens.	Replication topology and lag bounds must be documented. For high-stakes decisions (credit decline, account closure), primary-node query may be required regardless of latency cost.	Replication lag is an inherent property of distributed systems. The window duration must be formally accepted by the organization and disclosed in relevant governance documentation.
Token Forgery via Third Party	A fraudulent enrichment provider or data broker issues tokens that are structurally valid but not backed by genuine consumer consent. Without issuer signature verification, the gate cannot distinguish a legitimately issued token from a forged one.	All tokens must be cryptographically signed by verified issuers. Third-party token issuers must be enrolled in a managed token authority registry. Unsigned or unknown-issuer tokens must be rejected.	Issuer verification infrastructure requires coordination with third-party data providers. Where providers cannot issue signed tokens, proxy attestation with explicit risk acknowledgement is the alternative.

Architectural Responsibility: QODIQA gate enforcement is effective only for execution paths architecturally routed through the gate. Prevention of direct model invocation, shadow pipeline bypass, and cache replay requires deployment-level architectural controls that are the responsibility of the implementing organization, not properties of the enforcement layer itself.

Non-Bypass Guarantee: Any execution path that does not pass through the QODIQA enforcement gate is considered non-conformant and outside the system's guarantees. Enforcement is defined strictly at the execution boundary.

D2 - SECTION 8

Residual and Out-of-Scope Risks

OUTSIDE QODIQA ENFORCEMENT SCOPE - FINANCIAL SERVICES SECTOR
Ø Model fairness and disparate impact. Consent enforcement does not evaluate whether scoring models produce discriminatory outcomes. Fair lending compliance requires separate model governance frameworks.
Ø Business logic correctness. QODIQA gates authorize execution; they do not validate that the decision logic applied is accurate, appropriate, or free of errors.
Ø Consumer disclosure adequacy. Whether the underlying consent disclosure accurately represents AI processing in consumer-comprehensible language is a regulatory compliance and legal matter outside the enforcement layer.
Ø Training data consent and historical data use. Data used to develop models deployed in production is outside the scope of runtime enforcement. Historical data governance is a separate requirement.
Ø Third-party data source compliance. QODIQA verifies tokens issued by third parties; it does not audit the consent processes of data brokers or bureau providers.

D2 - SECTION 9

Institutional Closing - Financial Services Dossier

Algorithmic decisioning in financial services operates at a scale where individual consent verification was historically impractical. Runtime enforcement infrastructure makes per-record, per-decision consent gate checks operationally feasible. The consequence is that **automated decisions become traceable to specific, verifiable consent instruments** - a capability that consumer protection frameworks increasingly require but that current financial AI infrastructure does not provide by default.

Consent in financial AI is not a disclosure exercise. It is an authorization architecture problem. Deterministic enforcement converts the authorization model from one based on assumed consent at account opening to one based on verified, purpose-scoped authorization at each point of automated data consumption.

APPENDIX A2 - DOSSIER II

Enforcement Deployment Maturity Levels (Illustrative)

TIER 1

Advisory Logging Only

GATE COVERAGE	No enforcement gates active. Consent events observed from existing decisioning audit logs. No execution blocking or token verification.
REGISTRY STATE	Not integrated with AI decisioning pipelines. Token inventory may exist but is not queried at runtime.
AUDIT COMPLETENESS	Partial. Application-layer decision logs only. Consent basis not linked.
REVOCATION	Not enforced at runtime. Revocation processed via compliance workflows only.
CERTIFICATION	Does not meet deterministic enforcement criteria.
DEPLOYMENT SIGNAL	Experimental

TIER 2

Partial Gate Coverage

GATE COVERAGE	Gates active on primary decisioning pipelines. Enrichment verification, batch scoring, and legacy pipelines may remain ungated.
REGISTRY STATE	Integrated for gated pipelines. Enrichment provider tokens not uniformly re-verified. Batch pipelines may operate outside gate perimeter.
AUDIT COMPLETENESS	Partial. Primary pipeline decisions linked to tokens. Batch and enrichment events may be unlinked.
REVOCATION	Enforced within gated scope. Batch revocation lag and enrichment revocation gap remain.
CERTIFICATION	Does not meet deterministic enforcement criteria. Coverage gaps must be documented and disclosed.
DEPLOYMENT SIGNAL	Controlled Deployment

TIER 3 - MINIMUM FOR DETERMINISTIC ENFORCEMENT

Full Deterministic Enforcement

GATE COVERAGE	All decisioning pipelines gated, including batch, enrichment, and cross-product pooling. No ungated data access paths to AI-consumed data. Coverage verified by deployment

TIER 4

Certified Conformance Deployment

GATE COVERAGE	Full coverage as Tier 3, verified by external conformance assessor. Third-party token issuer registry maintained and audited.

	audit.
REGISTRY STATE	Live registry with DENY-on-unavailable posture. Enrichment provider tokens re-verified at consumption. Batch per-record gate active.
AUDIT COMPLETENESS	Complete. Pre-execution audit record per gate evaluation across all pipelines. Token ID linked to each decision.
REVOCATION	Enforced at all gates. Revocation window bounded by TTL and documented in organizational governance.
CERTIFICATION	Meets QODIQA Core deterministic enforcement standard. Eligible for Tier 3 conformance assessment.
DEPLOYMENT SIGNAL	Production-Critical

REGISTRY STATE	Cryptographic token signing implemented. Third-party enrichment token signatures verified against enrolled issuer certificates.
AUDIT COMPLETENESS	Tamper-evident audit chain certified. Art. 22 explanation packages validated for completeness against regulatory requirements.
REVOCATION	Maximum revocation window formally declared in conformance statement. Replication lag bounds certified.
CERTIFICATION	QODIQA Tier 4 Conformance Certificate issued. Renewal cadence defined per Certification Framework.
DEPLOYMENT SIGNAL	Regulatory-Grade

Note: Only Tier 3 and above meet the deterministic runtime enforcement criteria defined in QODIQA Core Standard. Tier 1 and Tier 2 deployments may not assert deterministic enforcement properties in regulatory disclosures, consumer rights responses, or audit submissions without explicit qualification of coverage scope and identified gaps.

DOSSIER III OF IV

QODIQA Use Case Dossier - Media and Content Generation Platforms

Deterministic Runtime Consent Enforcement in AI-Assisted Content Creation and Distribution Environments

VERSION 1.0 DOCUMENT ID QODIQA-UCD-2026-001-D3 DATE 2026 STATUS Active - Public Release

SECTOR-SPECIFIC ENFORCEMENT CRITICALITY

The massive scale of AI-assisted content generation creates a consent surface that cannot be managed through manual review or post-generation auditing. Reputational and misinformation cascade risk from unauthorized likeness or voice use can propagate irreversibly across distribution networks before a violation is identified. Lack of provenance traceability in generated outputs means that enforcement must occur at the generation boundary, where a consent record can be attached to each output before distribution.

This establishes this sector as a high-criticality domain requiring deterministic enforcement at execution time.

D3 - SECTION 1

Sector Context Overview

Media and content generation platforms encompass AI systems used for text generation, image synthesis, voice cloning, video production, personalized content recommendation, automated journalism, and synthetic media distribution. The consent surface is multi-party: consent is required from data subjects whose content trained the model, individuals whose likeness or voice is used in generation, and consumers whose behavioral data drives personalization.

Deployment patterns span consumer-facing generative tools, B2B content production pipelines, publishing automation, advertising personalization, and synthetic media distribution infrastructure. The distinctive characteristic of this sector is the structural separation between the training layer - where consent for data inclusion governs - and the inference layer - where consent for output generation governs. These layers have distinct consent architectures and distinct enforcement boundaries.

1.1 Regulatory Environment (Reference Only)

Reference frameworks include GDPR (biometric data, Art. 22 profiling), EU AI Act general-purpose AI and synthetic media provisions, state-level biometric privacy statutes (Illinois BIPA, Texas CUBI), and copyright law developments affecting AI training data. This dossier does not assess compliance with any specific regulatory instrument.

D3 - SECTION 2

Current Runtime Consent Failure Surface

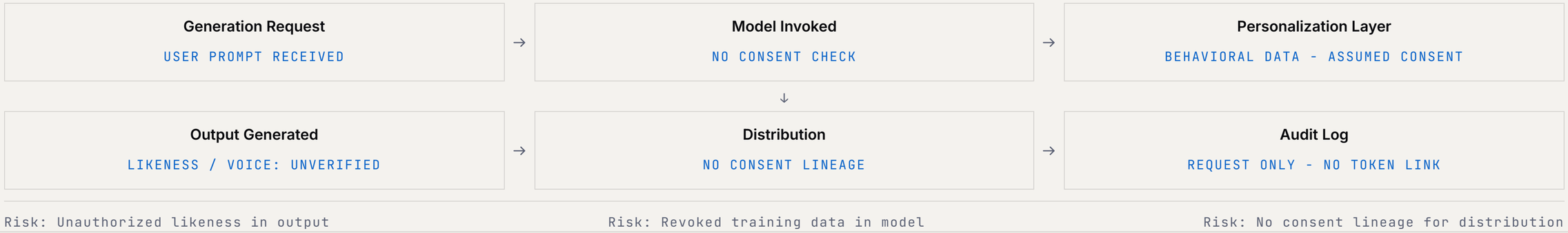
The following taxonomy classifies failure modes by enforcement breakdown type at runtime.

AUTHORIZATION MISINTERPRETATION	TRAINING CONSENT	Content used to train generative models consumed under broad platform terms or web scraping without per-creator consent verification at ingestion. No machine-verifiable record of authorized training data inclusion exists per content element.
CONSENT DRIFT	LIKENESS AND VOICE USE	Voice cloning and image synthesis models generate outputs incorporating identifiable characteristics of specific individuals without a per-generation consent gate verifying that the relevant individual has authorized their characteristics to be used in the requested generation context and purpose.
CONSENT DRIFT	PURPOSE DRIFT	Behavioral engagement data collected under content personalization consent silently consumed by advertising targeting models, recommendation systems, and model fine-tuning pipelines. Each consuming system assumes authorization from the original engagement consent without scope verification.
EXECUTION GAP	REVOCATION AT SCALE	Creators requesting removal from training datasets or revoking voice/likeness licensing have no mechanism to propagate revocation to active generation models. Models continue producing derivative outputs without any runtime check against current revocation state.
CONSENT DRIFT	CROSS-PLATFORM POOLING	Consent obtained on one platform applied to model training and personalization on affiliated platforms without consumer awareness. No per-platform consent scope verification at cross-platform data consumption.
CONSENT DRIFT	AUDIT ABSENCE	Generative AI systems log output requests but do not maintain records linking each generation event to the consent instruments that authorized use of constituent training data, personalization inputs, or likeness/voice elements incorporated in the output.

D3 - SECTION 3

Execution Path Without Deterministic Enforcement

FIG D3-A - UNENFORCED CONTENT GENERATION EXECUTION CHAIN

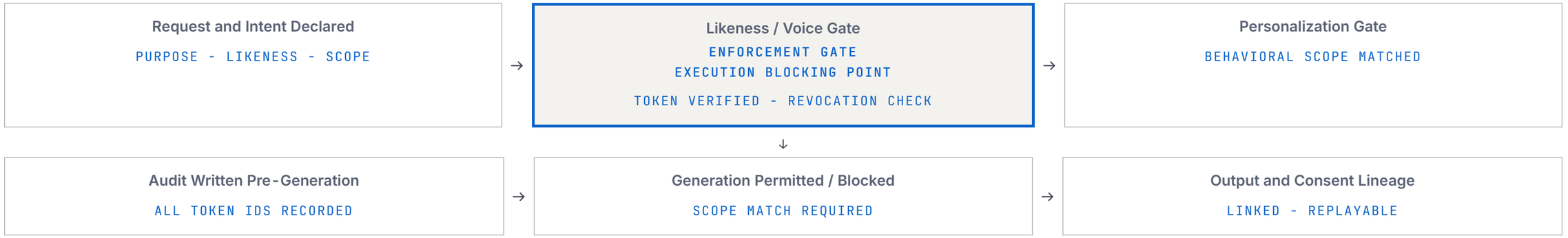


D3 - SECTION 4

Execution Path With QODIQA Runtime Enforcement

QODIQA enforcement in content generation applies at distinct layers: at training data ingestion (consent for each data element's inclusion), at generation invocation (consent for likeness/voice use in the specific generation context), and at personalization data consumption (scope-matched verification of behavioral data use authority). Each layer is governed by distinct consent tokens and distinct gate configurations.

FIG D3-B - QODIQA-ENFORCED CONTENT GENERATION CHAIN



Gate: Per generation event Artifact: Output linked to consent chain Revocation: Blocks generation if active **SINGLE DETERMINISTIC ENFORCEMENT BOUNDARY (NON-BYPASSABLE)**

WITHOUT ENFORCEMENT	WITH QODIQA ENFORCEMENT
× Likeness/voice use: unchecked at generation	→ Likeness/voice gate: verified per generation event
× Personalization data: scope assumed	→ Personalization: scope matched before data consumption
× Revocation: not propagated to generation layer	→ Revocation: active check blocks generation
× Output: no consent lineage attached	→ Output: consent lineage record attached
× Audit: request logged, consent basis absent	→ Audit: token chain linked to each output
Consent: inferred from platform terms	Consent: verified, replayable per output

D3 - SECTION 5

Risk Surface Reduction Analysis

TABLE D3-1 - RISK REDUCTION BY CATEGORY, MEDIA AND CONTENT GENERATION SECTOR

RISK CATEGORY	WITHOUT ENFORCEMENT	WITH QODIQA	RESIDUAL RISK
Unauthorized Likeness/Voice	No gate at generation step; model uses incorporated characteristics without per-event consent check	Gate verifies active, non-revoked consent token for likeness/voice use in declared generation context	Model internalization of characteristics not addressable at runtime; requires training governance upstream
Purpose Drift (Personalization)	Behavioral data consumed across purposes under broad engagement consent	Per-consumption scope check; data blocked if declared purpose exceeds token-authorized scope	Consent token granularity; token authoring accuracy relative to consumer agreement language
Revocation at Generation Layer	Revocation events not propagated to active generation infrastructure	Live revocation check at each generation gate; revoked tokens return DENY	Race condition between revocation and in-flight generation requests; registry availability
Consent Lineage for Outputs	Generated outputs have no attached consent provenance record	Each output linked to audit record containing all evaluated token IDs	Downstream distribution of outputs outside platform control; lineage record not propagated with content
Cross-Platform Data Pooling	Consent obtained on source platform assumed valid for pooled model use on partner platforms	Cross-platform consumption requires scope-matched token verification at each consuming system	Partner platform integration requirements; token interoperability between platform systems

TABLE D3-3 - OPERATIONAL RISK DELTA (INDICATIVE)

RISK CATEGORY	WITHOUT ENFORCEMENT	WITH QODIQA
Unauthorized Inference	High (likeness/voice unverified)	Eliminated at generation gate
Purpose Drift	High (behavioral data cross-purpose)	Near-zero (per-consumption scope match)
Revocation Latency	Revocation not propagated to generation	Live check at each generation event
Audit Reconstruction	No consent lineage for outputs	Each output linked to full gate record chain
Multi-Agent Propagation	Cross-platform distribution unverified	Token verified at each platform consumption

Interpretation: QODIQA collapses runtime consent risk from probabilistic enforcement to deterministic execution control.

D3 - SECTION 6

Evidence and Artifact Model

The following minimal artifact set defines the irreducible enforcement surface required for deterministic validation.

MINIMAL ARTIFACT SET (REQUIRED)

Every QODIQA-conformant gate evaluation MUST produce at minimum the following artifact types. Additional artifacts defined in this section extend the minimum set.

- Consent Token (hash-bound)
- Intent Declaration Object
- Enforcement Decision Record
- Audit Log Entry (append-only)

ART-D3-01

Generation Gate Record

Per-generation-event record: request ID, declared intent and purpose, all token IDs evaluated (likeness, voice, personalization data), gate result, timestamp, output reference.

ART-D3-02

Likeness/Voice Authorization Receipt

Confirms token verification for identifiable individual characteristics in each generation event. Revocation state at time of generation included. Provides evidence basis for content provenance claims.

ART-D3-03

Content Consent Lineage Record

Links each generated output to the full chain of consent gate records. Supports platform liability defense, creator rights verification, and regulatory inquiry response.

ART-D3-04

Revocation Block Event Record

Generated when a generation request is blocked due to active revocation: token ID, revocation timestamp, generation request reference, requesting user category.

D3 - SECTION 7

Operational Constraints

OPERATIONAL CONSTRAINTS - MEDIA AND CONTENT GENERATION DEPLOYMENT	
GENERATION LATENCY	Generation gate checks add synchronous overhead to real-time generation pipelines. High-throughput consumer generation platforms require gate infrastructure scaled to handle concurrent generation volume without perceptible latency increase. Performance testing under load is required before production activation.
LIKENESS SCOPE DEFINITION	Defining the scope of a "likeness" or "voice" consent token requires careful legal and technical alignment. Overly narrow token scope may block legitimate generation; overly broad scope may fail to honor revocation at the required granularity. Token architecture design requires legal and product alignment.
TOKEN ECOSYSTEM MATURITY	Consent token issuance for creators, voice talent, and data subjects requires an operational token issuance infrastructure that may not exist on many platforms. Establishing token enrollment, revocation, and verification workflows is a prerequisite for enforcement deployment.

7.A Training / Inference Boundary Limitation

This subsection defines the structural boundary between QODIQA runtime enforcement scope and the training-time consent domain. This boundary is architectural, not a limitation of implementation quality.

TRAINING / INFERENCE BOUNDARY - ENFORCEMENT SCOPE DEFINITION
• Runtime enforcement governs the inference execution boundary - the point at which a deployed model processes a specific request under a specific declared intent. It does not govern processes that occurred prior to model deployment, including the selection, ingestion, and preprocessing of training data.
• When a generative model is trained on data incorporating specific characteristics - voice patterns, facial geometry, stylistic attributes - those characteristics are encoded into the model's weight parameters. This encoding occurs at training time. The relationship between training data consent state and model weight contents is established before deployment and cannot be altered by runtime mechanisms applied after deployment.

- A consent revocation event issued after model training cannot retroactively modify model weights to remove representation of the characteristics whose consent was revoked. The encoding in the weight space is not addressable by post-training runtime controls. QODIQA runtime enforcement can block generation requests that reference a revoked consent token, but it cannot prevent the model from having internalized those characteristics at training time.
- Runtime enforcement at the inference layer provides the following protections against revocation after training: blocking new generation requests that explicitly invoke the revoked characteristic (likeness gate, voice gate); preventing new distribution of platform-generated outputs under a revoked authorization; and creating an audit record of all generation events that occurred before and after revocation. It does not provide: removal of the characteristic from model weights; blocking of generation that incidentally draws on internalized patterns without explicit invocation; or retroactive invalidation of outputs generated before revocation.
- Training dataset governance - including per-data-element consent verification at ingestion, data provenance tracking, and training data revocation workflow design - is structurally distinct from runtime enforcement and must be addressed as a separate model lifecycle governance requirement. Organizations cannot substitute runtime enforcement for training data governance and must implement both independently.

7.B Quantified Operational Characteristics (Illustrative, Non-Normative)

OPERATIONAL OVERHEAD REFERENCE - MEDIA AND CONTENT GENERATION DEPLOYMENT · ILLUSTRATIVE - NON-NORMATIVE - NOT A PERFORMANCE GUARANTEE	
GATE LATENCY	Typical deployment observations indicate gate round-trip of `2 - 20 ms` under local registry for per-generation-event checks. Consumer-facing platforms with sub-second generation expectations must budget gate latency as a fixed overhead component and validate total response time under concurrent load.
HIGH-VOLUME GENERATION	Platforms serving high concurrent generation volume - e.g., 10,000 simultaneous generation requests - generate proportional registry query load. Registry infrastructure must be horizontally scalable. At `10,000 concurrent requests × 3 gate checks per request`, registry must sustain approximately `30,000 queries per generation cycle`.
AUDIT RECORD VOLUME	Each generation event produces one audit record per gate evaluated. A platform generating 1,000,000 outputs daily with an average of 3 gate evaluations per output produces approximately `3,000,000 audit records daily`. Audit storage at this scale requires object storage or columnar database infrastructure with defined retention and archival policy.
REVOCATION PROPAGATION	Revocation event processing time - from revocation registry write to gate-enforced blocking - is bounded by cache TTL in cached deployments. Typical acceptable TTL values for likeness/voice contexts range from `60 seconds to 15 minutes` depending on organizational risk tolerance. Live-registry deployments enforce revocation within registry response latency, typically `under 1 second`.

D3 - SECTION 7A

Enforcement Bypass and Adversarial Risk Considerations

QODIQA enforces only execution paths architecturally routed through the consent gate. The following vectors are specific to content generation platform deployments and represent conditions under which enforcement may be circumvented.

TABLE D3-2 - BYPASS VECTORS, MEDIA AND CONTENT GENERATION SECTOR

BYPASS VECTOR	DESCRIPTION	ENFORCEMENT DEPENDENCY	RESIDUAL RISK
Direct Model API Access	Generation model accessed via direct API endpoint - internal developer tools, staging environments, partner API integrations - without passing through the QODIQA gate. Common in platforms where gate integration is applied to the consumer-facing interface but not to all model access paths.	All model access paths - consumer-facing, internal, API partner, staging - must route through the gate. Architectural network controls must prevent model endpoint access that bypasses the gate layer.	Developer and partner access paths represent a meaningful bypass surface if not included in gate coverage scope. Coverage audit must enumerate all model invocation paths.
Weight-Level Characteristic Access	Generation requests that do not explicitly reference a specific individual's likeness or voice token but nonetheless produce outputs that draw on internalized characteristics from training data. The gate has no declared token to verify and permits the generation without a consent check.	This vector is not fully addressable by runtime enforcement. See Section 7.A Training / Inference Boundary Limitation. Mitigation requires training data governance controls, not runtime gate expansion.	This residual risk is structural to the training / inference boundary. Runtime enforcement cannot govern outputs that arise from training-time encoding without an explicit generation-time token reference.
Revocation Timing Window	A revocation event is issued during an active, in-flight generation request. The generation completes before the revocation check result is received from the registry, producing an output under a token that was revoked during execution.	Pre-execution revocation checks reduce but cannot eliminate this window. Long-running generation jobs (video, multi-page content) have larger exposure windows. Job-level revocation monitoring may be required for extended generation processes.	The timing window is bounded by generation job duration and registry query latency. For short generation requests (under 1 second), the practical exposure is minimal. Long-running jobs require additional controls.

BYPASS VECTOR	DESCRIPTION	ENFORCEMENT DEPENDENCY	RESIDUAL RISK
Mis-Scoped Likeness Token	A likeness token issued with a context scope broader than the individual intended - e.g., "all commercial uses" vs. "advertising only" - is used to authorize generation contexts the individual did not consent to. The gate permits because the token scope technically covers the declared purpose.	Token scope language must be reviewed for alignment with the individual's actual consent. Legal review of token scope definitions and enrolling individuals' awareness of scope implications is an organizational control requirement.	Technical enforcement cannot distinguish a correctly-formatted broad-scope token from one that was mis-scoped. Consent process quality at token issuance is the primary mitigation.
Cross-Platform Token Non-Verification	Content generated on Platform A is distributed or repurposed on Platform B. Platform B does not re-verify the consent tokens that authorized the original generation, treating cross-platform content as pre-authorized for all downstream uses.	Cross-platform distribution requires platform B to verify consent tokens attached to received content before further processing or distribution. This requires token interoperability and willingness of Platform B to implement verification.	Out-of-platform distribution of generated content is outside the originating platform's enforcement perimeter. Platform-to-platform enforcement interoperability is a governance negotiation requirement, not a technical property of the gate.

Architectural responsibility: QODIQA gate enforcement is effective only for execution paths architecturally routed through the gate. The training / inference boundary limitation (Section 7.A) represents a structural scope boundary, not a bypass vector. Architectural bypass prevention - including direct model access controls and cross-platform token verification requirements - is a deployment responsibility of the implementing organization.

Non-Bypass Guarantee: Any execution path that does not pass through the QODIQA enforcement gate is considered non-conformant and outside the system's guarantees. Enforcement is defined strictly at the execution boundary.

D3 - SECTION 8

Residual and Out-of-Scope Risks

OUTSIDE QODIQA ENFORCEMENT SCOPE - MEDIA AND CONTENT GENERATION SECTOR
Ø Training data consent governance. Content incorporated into model weights at training time is not governed by runtime enforcement. Consent for training data inclusion is a model development governance requirement, separate from and prior to runtime enforcement.
Ø Removal of internalized characteristics from model weights. Post-training revocation cannot retroactively remove characteristics encoded in model weight space. This is a structural boundary of runtime enforcement, not an implementation gap.
Ø Output content quality, accuracy, and harmful content. Consent enforcement does not evaluate the quality, truthfulness, or potential harms of generated content. Content moderation and safety evaluation are separate system responsibilities.
Ø Copyright and intellectual property infringement in outputs. Whether generated outputs infringe on third-party intellectual property is a legal determination outside the consent enforcement layer.
Ø Consumer deception and synthetic media disclosure obligations. Requirements to disclose AI-generated content to end audiences are regulatory and ethical obligations outside the consent enforcement layer.
Ø Downstream distribution and out-of-platform use. Once generated content leaves the platform, the consent lineage record is not automatically propagated with the content. Out-of-platform distribution is outside enforcement scope without cross-platform interoperability agreements.

D3 - SECTION 9

Institutional Closing - Media and Content Generation Dossier

The consent surface in media and content generation is multi-party and layered: creators, data subjects, voice talent, and consumers each hold distinct consent interests that intersect at the point of AI generation. Deterministic enforcement establishes a gate at each generation event that requires verified, non-revoked, scope-matched consent for each consent dimension - **making consent a prerequisite for output, not a post-hoc justification for it**.

The training / inference boundary limitation is not a deficiency of this enforcement model. It reflects an accurate description of what runtime enforcement can and cannot govern. Training data governance addresses the training layer. Runtime enforcement addresses the inference layer. Both are necessary; neither substitutes for the other.

In generative AI environments, consent is not established once and assumed thereafter. It is a condition that must be re-verified at the boundary of each generation event. Deterministic enforcement makes that re-verification systematic, auditable, and technically binding - within the structural boundary defined by the training / inference separation.

APPENDIX A3 - DOSSIER III

Enforcement Deployment Maturity Levels (Illustrative)

TIER 1

Advisory Logging Only

GATE COVERAGE	No enforcement gates active. Generation events logged. No token verification or execution blocking. Likeness/voice use unchecked at generation.
REGISTRY STATE	Consent registry not integrated with generation infrastructure. Creator and subject tokens may be inventoried but are not queried at generation time.
AUDIT COMPLETENESS	Partial. Generation request logs only. No consent token linkage per output.
REVOCATION	Not enforced at generation layer. Revocation processed via manual compliance workflow only.
CERTIFICATION	Does not meet deterministic enforcement criteria.
DEPLOYMENT SIGNAL	Experimental

TIER 2

Partial Gate Coverage

GATE COVERAGE	Gates active on explicit likeness/voice generation requests. General generation, personalization layer, and cross-platform use remain ungated.
REGISTRY STATE	Integrated for likeness/voice gate. Personalization data and training data ingestion not gated.
AUDIT COMPLETENESS	Partial. Likeness/voice events linked to tokens. Other generation events unlinked.
REVOCATION	Enforced for explicitly-gated likeness/voice requests. Personalization and general generation revocation gap remains.
CERTIFICATION	Does not meet deterministic enforcement criteria. Coverage gaps must be documented.
DEPLOYMENT SIGNAL	Controlled Deployment

TIER 3 - MINIMUM FOR DETERMINISTIC ENFORCEMENT

TIER 4

Full Deterministic Enforcement

GATE COVERAGE All generation pipeline paths gated. Likeness/voice, personalization, and cross-platform consumption each subject to gate evaluation. Training data ingestion gated separately under model lifecycle governance.

REGISTRY STATE Live registry with DENY-on-unavailable posture. Revocation propagation SLA defined and tested. Creator and subject token enrollment active.

AUDIT COMPLETENESS Complete. Pre-generation audit record per gate evaluation. Consent lineage record attached to each output.

REVOCATION Enforced at all gates. Revocation window bounded by documented TTL or live-registry policy. Training / inference boundary limitation acknowledged in deployment documentation.

CERTIFICATION Meets QODIQA Core deterministic enforcement standard for inference-layer enforcement. Eligible for Tier 3 conformance assessment.

DEPLOYMENT SIGNAL Production-Critical

Certified Conformance Deployment

GATE COVERAGE Full coverage as Tier 3, verified by external assessor. Training data ingestion consent governance documented and audited as separate but co-submitted governance artifact.

REGISTRY STATE Cryptographic token signing for all creator, subject, and personalization tokens. Issuer certificate chain verified at each gate evaluation.

AUDIT COMPLETENESS Tamper-evident audit chain certified. Consent lineage record format certified for use in rights verification and regulatory response contexts.

REVOCATION Maximum inference-layer revocation window formally declared. Training / inference boundary acknowledged in conformance statement with separate training governance documentation referenced.

CERTIFICATION QODIQA Tier 4 Conformance Certificate issued for inference-layer enforcement scope. Renewal cadence defined per Certification Framework.

DEPLOYMENT SIGNAL Regulatory-Grade

Note: Only Tier 3 and above meet the deterministic runtime enforcement criteria defined in QODIQA Core Standard. For media and content generation deployments, Tier 3 conformance applies to the inference execution layer only. The training data consent governance domain is a separate governance requirement. Organizations must not represent inference-layer enforcement as covering training-time consent obligations.

DOSSIER IV OF IV

QODIQA Use Case Dossier - Enterprise AI Assistants and Knowledge Systems

Deterministic Runtime Consent Enforcement in Organizational AI Deployment and Knowledge Access Environments

VERSION 1.0 DOCUMENT ID QODIQA-UCD-2026-001-D4 DATE 2026 STATUS Active - Public Release

SECTOR-SPECIFIC ENFORCEMENT CRITICALITY

Internal data leakage risk through AI assistants with broad knowledge base access represents an organizational data governance failure if not bounded by per-source consent gates. Cross-system propagation of sensitive data through multi-agent orchestration can breach multiple data governance boundaries in a single user interaction. Governance and policy enforcement collapse risk is elevated in enterprise environments where shadow IT deployments may operate entirely outside the enforcement perimeter.

This establishes this sector as a high-criticality domain requiring deterministic enforcement at execution time.

D4 - SECTION 1

Sector Context Overview

Enterprise AI assistants and knowledge systems encompass conversational AI agents deployed within organizational intranets, retrieval-augmented generation (RAG) systems indexing internal knowledge bases, AI-assisted HR systems, automated legal research and contract review agents, executive decision support tools, and AI systems operating across organizational boundaries in partner or supply chain contexts.

The consent surface in enterprise AI has a distinctive characteristic: it involves employee data subjects whose consent relationship with their employer is inherently constrained by the employment relationship, alongside organizational data subjects (business partners, clients, customers) whose data flows through enterprise systems under commercial agreements. Multi-agent orchestration in this sector frequently masks multi-system, multi-policy data access within a single user-facing interaction.

1.1 Multi-Agent Exposure

A user query to an enterprise assistant may trigger retrieval from HR systems, legal document repositories, financial databases, and external APIs - each governed by distinct data access policies. The orchestrating agent's single user-

facing interaction masks the multi-system, multi-policy data access occurring at the execution layer. QODIQA enforcement must be applied at the sub-agent call level, not only at the user-facing query level.

D4 - SECTION 2

Current Runtime Consent Failure Surface

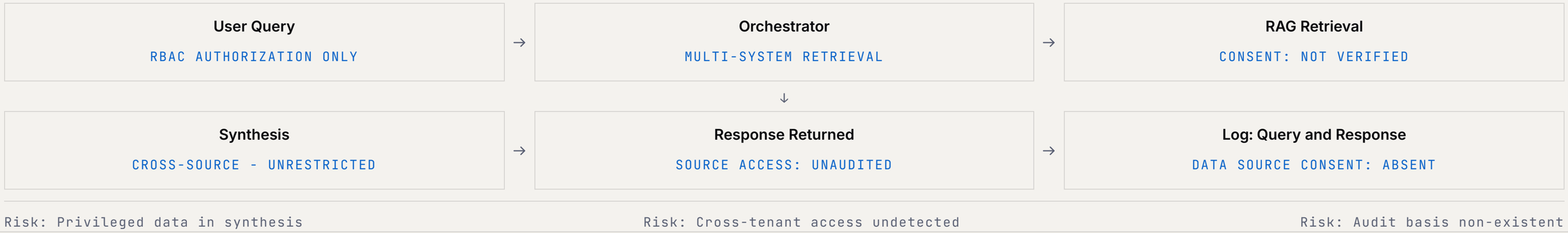
The following taxonomy classifies failure modes by enforcement breakdown type at runtime.

CONSENT DRIFT	SCOPE CONFLATION	Enterprise AI assistants granted broad data access at configuration time, inheriting deploying administrator permissions. Individual user queries triggering multi-system retrieval are executed under organizational authorization, not verified against specific consent or data use agreements governing each accessed source.
CONSENT DRIFT	EMPLOYEE CONSENT GAP	Employee consent for AI processing embedded in employment contracts as a blanket provision. Specific AI systems, data categories, and inference purposes are not enumerated at the execution layer. Performance analysis, sentiment inference, and productivity monitoring proceed without per-processing-purpose verification.
CONSENT DRIFT	RAG BOUNDARY FAILURE	RAG systems index organizational knowledge bases without per-document consent or classification verification. User queries may retrieve and synthesize information from documents marked confidential, legally privileged, or subject to specific distribution restrictions - without verifying that the requesting context authorizes access to those documents under applicable constraints.
CONSENT DRIFT	PURPOSE DRIFT	Client data held in CRM systems under contract-of-service authorization consumed by internal AI analytics models to train or fine-tune enterprise AI systems. The client's contractual authorization was for service delivery, not AI model development.
CONSENT DRIFT	CROSS-ORG EXPOSURE	In shared enterprise AI deployments - supply chain partners sharing an AI assistant, multi-tenancy SaaS - data from one organizational tenant may be accessible to AI queries from another tenant through a shared knowledge index. Tenant isolation at the AI retrieval layer is inconsistently enforced.
AUDIT FAILURE	AUDIT OPACITY	Enterprise AI interaction logs record user queries and AI responses but do not log which data sources were accessed, under what authorization, and whether the applicable consent or data use agreement was in force at the time of access.

D4 - SECTION 3

Execution Path Without Deterministic Enforcement

FIG D4-A - UNENFORCED ENTERPRISE AI ASSISTANT EXECUTION CHAIN

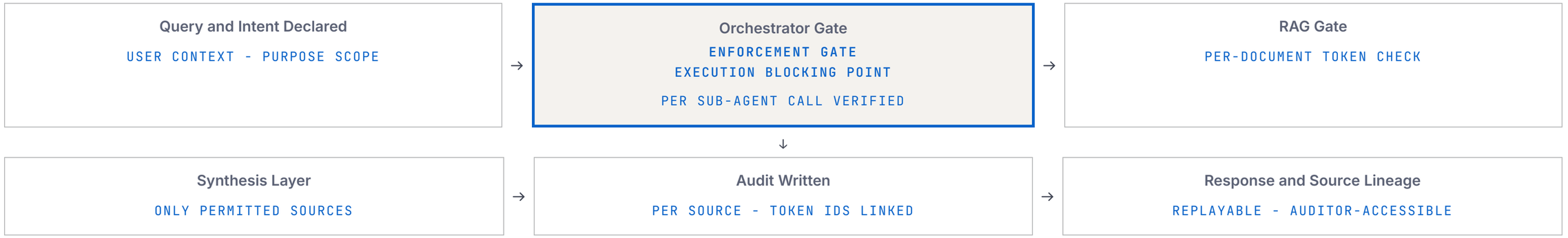


D4 - SECTION 4

Execution Path With QODIQA Runtime Enforcement

In enterprise deployments, QODIQA gates operate at the data source access layer within the AI orchestration stack. Each sub-agent retrieval call, each RAG document access, and each cross-system data fetch requires a consent gate evaluation before data is returned to the synthesis layer.

FIG D4-B - QODIQA-ENFORCED ENTERPRISE AI EXECUTION CHAIN



Gate: Per data source access Tenant isolation: enforced at gate Audit: source consent chain complete **SINGLE DETERMINISTIC ENFORCEMENT BOUNDARY (NON-BYPASSABLE)**

WITHOUT ENFORCEMENT	WITH QODIQA ENFORCEMENT
× RBAC authorization substitutes for consent	→ Consent gate distinct from RBAC layer
× RAG retrieval: no per-document consent check	→ RAG: per-document token verification before retrieval
× Tenant isolation: inconsistently enforced	→ Tenant isolation enforced at consent gate
× Employee data inference: purpose unverified	→ Employee data: purpose-scoped token required
× Audit: query/response only; source basis absent	→ Audit: full source consent chain per response
Data access: assumed authorized	Data access: verified, consent-anchored

D4 - SECTION 5

Risk Surface Reduction Analysis

TABLE D4-1 - RISK REDUCTION BY CATEGORY, ENTERPRISE AI ASSISTANTS

RISK CATEGORY	WITHOUT ENFORCEMENT	WITH QODIQA	RESIDUAL RISK
RAG Boundary Failure	Privileged and confidential documents retrievable by RAG without per-document consent or classification check	Per-document token required before retrieval; documents without valid token not returned to synthesis layer	Token assignment to existing document repositories; classification and tokenization of legacy knowledge bases
Employee Data Purpose	Employment-agreement blanket consent applied to all AI processing without per-purpose verification	Purpose-scoped tokens required for distinct processing contexts (performance, monitoring, HR analytics)	Token scope must align with employment law in each jurisdiction; labor law consultation required
Multi-Tenant Isolation	Cross-tenant data access through shared RAG index not systematically prevented	Tenant identifier scoped into consent tokens; cross-tenant access blocked at gate evaluation	Token issuance must correctly encode tenant boundaries; misconfiguration risk
Client Data Purpose Drift	Client data consumed under service delivery authorization for AI model training without scope check	Training pipeline access requires separate purpose token; service delivery token does not authorize training use	Existing client contracts may not provide training-purpose consent; legal review required
Audit Reconstructability	Response audit trail does not identify which data sources were accessed or under what authorization	Per-source, per-retrieval audit record includes token IDs, access timestamp, gate result	Audit record volume at enterprise scale; retention infrastructure requirements

TABLE D4-3 - OPERATIONAL RISK DELTA (INDICATIVE)

RISK CATEGORY	WITHOUT ENFORCEMENT	WITH QODIQA
Unauthorized Inference	High (RAG boundary failure)	Eliminated per document at retrieval gate
Purpose Drift	High (employment consent overreach)	Near-zero (purpose-scoped token per context)
Revocation Latency	Hours to days (manual HR process)	Near-zero with HR-to-registry integration
Audit Reconstruction	Response source basis non-existent	Per-source token ID linked in response record
Multi-Agent Propagation	Sub-agent access unverified	Gate check at each sub-agent data access

Interpretation: QODIQA collapses runtime consent risk from probabilistic enforcement to deterministic execution control.

D4 - SECTION 6

Evidence and Artifact Model

The following minimal artifact set defines the irreducible enforcement surface required for deterministic validation.

MINIMAL ARTIFACT SET (REQUIRED)

Every QODIQA-conformant gate evaluation MUST produce at minimum the following artifact types. Additional artifacts defined in this section extend the minimum set.

- Consent Token (hash-bound)
- Intent Declaration Object
- Enforcement Decision Record
- Audit Log Entry (append-only)

ART-D4-01

Orchestration Gate Record

Per-orchestration-event record: query intent, all sub-agent calls initiated, token IDs verified per call, gate results, timestamp. Complete visibility into multi-system access pattern for any AI-assisted decision.

ART-D4-02

RAG Document Access Log

Per-document retrieval record: document identifier, token ID verified, classification level, gate result, synthesis query reference. Enables reconstruction of which documents contributed to any AI response.

ART-D4-03

Employee Data Processing Record

Per-processing-event record for employee data: data subject identifier (anonymized), purpose token ID, processing system, gate result, timestamp. Supports data subject access requests and DPA audit inquiries.

ART-D4-04

Cross-Tenant Block Event

Generated when a retrieval request is blocked due to tenant scope mismatch: requesting tenant, target document tenant scope, gate evaluation result. Demonstrates active multi-tenant isolation enforcement.

ART-D4-05

Response Source Lineage Record

Maps each AI response to the full set of data sources accessed in its generation, with consent gate records for each. Enables complete audit reconstruction of AI-assisted decisions.

ART-D4-06

Organizational Replay Package

Structured export combining response source lineage, token state snapshots, and policy version for a defined time range. Supports regulatory examination, legal hold, or internal investigation without requiring live system access.

D4 - SECTION 7

Operational Constraints

OPERATIONAL CONSTRAINTS - ENTERPRISE AI DEPLOYMENT	
KNOWLEDGE BASE TOKENIZATION	Deploying per-document consent gates on RAG systems requires that existing knowledge bases be tokenized - each document or document class assigned a consent token reflecting applicable access and use constraints. For large enterprise knowledge repositories, this is a substantial operational undertaking requiring legal, information governance, and IT collaboration.
ORCHESTRATION INTEGRATION	AI orchestrators invoking multiple sub-agents must be instrumented to pass consent context through each sub-agent invocation and enforce gates at each data source access. In complex multi-agent pipelines, gate insertion must be architecturally disciplined to prevent bypass at any sub-agent invocation point.
EMPLOYMENT LAW VARIANCE	Consent token scope for employee data must be designed in alignment with employment law in each jurisdiction of operation. A single organizational token architecture may require jurisdictional variants. Employment law consultation in each operational jurisdiction is required before token scope finalization.
AUDIT VOLUME	Enterprise AI systems with high query volume generate audit records proportional to query complexity and knowledge base retrieval depth. Audit infrastructure must be sized for enterprise-scale record volume with appropriate retention, indexing, and access control for audit records themselves.
REGISTRY SYNCHRONIZATION	In distributed enterprise environments - multi-region, hybrid cloud - the consent registry must be accessible from all execution contexts with consistent state. Registry replication lag introduces a revocation window that must be bounded and documented. Cold-start scenarios following registry unavailability require defined default behavior.

7.A Quantified Operational Characteristics (Illustrative, Non-Normative)

OPERATIONAL OVERHEAD REFERENCE - ENTERPRISE AI DEPLOYMENT · ILLUSTRATIVE - NON-NORMATIVE - NOT A PERFORMANCE GUARANTEE

GATE LATENCY PER QUERY	A complex enterprise query triggering 10 sub-agent calls, each requiring one gate evaluation, incurs approximately `10 × 2 - 15 ms = 20 - 150 ms` of gate overhead per query under local registry conditions. Total query response time must budget for this overhead. High-complexity queries with many retrieval steps have proportionally higher gate overhead.
RAG GATE AMPLIFICATION	RAG retrieval pipelines performing per-document consent checks against large knowledge bases generate a gate query for each candidate document retrieved before filtering. A retrieval operation returning 50 candidate documents before re-ranking and filtering generates `50 gate queries` per retrieval event. Knowledge base scale and retrieval depth are the primary determinants of gate load in RAG deployments.
AUDIT RECORD GROWTH	An enterprise assistant processing 10,000 queries daily, each triggering an average of 8 source accesses requiring gate evaluations, generates approximately `80,000 audit records daily`. At this rate, yearly audit volume approaches `29,000,000 records`, requiring appropriate columnar storage, indexing, and access tiering infrastructure. Regulatory retention requirements for enterprise AI audit records should be assessed against applicable data protection law.
KNOWLEDGE BASE TOKENIZATION	Initial deployment of per-document gates on an existing enterprise knowledge base requires a one-time tokenization effort proportional to knowledge base size and document classification complexity. Organizations with 100,000-document knowledge bases should plan for a tokenization program spanning weeks to months, depending on classification automation maturity and legal review requirements per document category.
REGISTRY REPLICATION LAG	Multi-region registry replication typically introduces a replication window of `100 ms to several seconds` depending on geographic distance and network conditions. For enterprise AI decisions where revocation must be effective within a tight window - termination of access for departing employees, partner agreement revocation - primary-node queries may be required, accepting higher latency in exchange for zero replication lag.

D4 - SECTION 7A

Enforcement Bypass and Adversarial Risk Considerations

QODIQA enforces only execution paths architecturally routed through the consent gate. Enterprise AI environments are structurally complex, with multiple data access paths, legacy system integrations, and administrative overrides. The following vectors represent conditions under which enforcement may be circumvented.

TABLE D4-2 - BYPASS VECTORS, ENTERPRISE AI ASSISTANTS

BYPASS VECTOR	DESCRIPTION	ENFORCEMENT DEPENDENCY	RESIDUAL RISK
Sub-Agent Direct Data Access	A sub-agent within the orchestration pipeline accesses a data source directly - via internal API, shared database connection, or file system access - without being routed through the QODIQA gate. The orchestrator-level gate is passed, but the sub-agent's data retrieval step bypasses the per-source gate.	Gate insertion must occur at each data source access point within sub-agent logic, not only at the orchestrator entry point. Sub-agent code must be audited to confirm all data access paths route through the gate.	Sub-agent code audits are required at deployment and upon each sub-agent update. Ungated sub-agent data access paths are a meaningful enforcement gap in complex orchestration systems.
RAG Index Pre-Fetch Bypass	The RAG retrieval system pre-fetches and caches document chunks without per-chunk consent verification. Subsequent query-time retrieval draws from the pre-fetched cache, which may include document content from sources whose consent tokens have since expired or been revoked.	Pre-fetch operations must be gated individually. Cached document chunks must carry consent token metadata and be invalidated when the underlying token expires or is revoked. Cache invalidation events must be triggered by registry revocation events.	Pre-fetch cache management is operationally complex at enterprise scale. Organizations must define and test cache invalidation behavior explicitly as part of RAG gate deployment.
Administrative Override Path	Administrative or elevated-privilege access paths - system administrator accounts, IT service accounts, emergency access - may be configured to bypass the consent gate layer for operational purposes. These paths create ungated data access routes to AI-consumed data.	Administrative access paths must be included within the enforcement perimeter, or must be explicitly prohibited from invoking AI pipeline functions. Administrative consent gate bypass must not exist as a default configuration.	Administrative bypass is a common finding in enterprise gate deployments. Organizations must audit privileged access paths and confirm their inclusion within or explicit exclusion from gate coverage scope.
Tenant Token Misconfiguration	Consent tokens are issued with incorrect tenant scope identifiers, allowing cross-tenant retrieval that the gate permits because the token appears to authorize the access based on malformed or overly broad tenant scope encoding.	Token issuance for multi-tenant deployments requires tenant scope validation at token creation time. Token schema must enforce tenant identifier constraints. Issuance-time validation must be tested against realistic multi-tenant access patterns.	Token misconfiguration is an organizational process risk, not a technical bypass. Issuance-time validation reduces but does not eliminate the risk of human error in tenant scope assignment.
Stale Employee Token	Employee data consent tokens remain valid in the registry after an employee's consent has been practically withdrawn - through role change, contract modification, or termination - because the token revocation process has not been triggered by the HR or identity management system that processed the employment change.	HR and identity management system events (termination, role change, contract modification) must be mapped to consent token revocation triggers. Integration between HR systems and the consent registry is required to automate revocation on employment status change.	Manual revocation processes introduce a gap between employment status change and token revocation. HR-to-registry integration automation reduces this gap to near-zero; manual processes leave a window measured in business hours or days.

BYPASS VECTOR	DESCRIPTION	ENFORCEMENT DEPENDENCY	RESIDUAL RISK
Shadow IT AI Integration	Employees or departments deploy AI tools - third-party SaaS, local LLM instances, browser extensions - that access organizational data sources outside the centrally managed AI assistant infrastructure and entirely outside the QODIQA enforcement perimeter.	Shadow IT AI deployments are outside the enforcement perimeter by definition. QODIQA does not govern data access that does not pass through the organizational enforcement architecture. Detection and prohibition of shadow AI access to organizational data sources is an IT governance and DLP responsibility.	Shadow IT represents a meaningful and persistent enforcement gap. Technical enforcement cannot govern tools that operate outside the organizational AI infrastructure. Policy, DLP, and network controls are the primary mitigations.

Architectural responsibility: QODIQA gate enforcement is effective only for execution paths architecturally routed through the gate. Enterprise AI environments contain a high density of potential bypass vectors - sub-agent direct access, pre-fetch caches, administrative overrides, and shadow IT - each of which requires deliberate architectural control. Deployment organizations must conduct a comprehensive bypass vector audit before asserting full enforcement coverage.

Non-Bypass Guarantee: Any execution path that does not pass through the QODIQA enforcement gate is considered non-conformant and outside the system's guarantees. Enforcement is defined strictly at the execution boundary.

D4 - SECTION 8

Residual and Out-of-Scope Risks

OUTSIDE QODIQA ENFORCEMENT SCOPE - ENTERPRISE AI ASSISTANTS
Ø Business logic and decision quality. Consent enforcement does not evaluate the accuracy, appropriateness, or bias of AI-assisted organizational decisions. Decision governance and model quality are separate requirements.
Ø Information security and data exfiltration. QODIQA governs consent for data processing; it does not provide information security controls against unauthorized exfiltration, insider threat, or external attack on data sources or the registry itself.

- Intellectual property ownership of AI-assisted outputs. Whether AI-generated responses incorporating organizational knowledge constitute protectable intellectual property is a legal matter outside the consent enforcement layer.
- Adequacy of employment agreement consent provisions. Whether existing employment agreements provide legally adequate consent for specific AI processing purposes requires employment law assessment in each applicable jurisdiction.
- Third-party AI service provider compliance. Where enterprise AI assistants are provided by external vendors, the vendor's own data processing practices are governed by contractual and regulatory obligations outside the QODIQA enforcement perimeter.
- Shadow AI use. Employees using AI tools outside organizational deployment are outside the enforcement perimeter. QODIQA governs only AI systems within the organizational deployment boundary.

D4 - SECTION 9

Institutional Closing - Enterprise AI Assistants Dossier

Enterprise AI assistants present a consent architecture challenge that is qualitatively distinct from consumer-facing deployments. The multiplicity of data subjects, the diversity of applicable consent frameworks, and the depth of multi-system data access within a single AI interaction create a consent surface that cannot be governed by flat organizational policy alone.

Deterministic enforcement at the orchestration and retrieval layers ensures that **each data source access is individually verified against the applicable consent instrument**. The enterprise AI assistant ceases to be an authorized agent with broad permission and becomes a constrained agent that verifies specific consent at each decision boundary. The bypass vector density in enterprise environments - sub-agents, pre-fetch caches, administrative overrides, shadow IT - makes comprehensive deployment audit an essential prerequisite, not an optional enhancement.

The organizational deployment of AI assistants does not suspend the consent rights of employees, clients, or partners whose data those systems access. Deterministic enforcement makes consent verification systematic at the data access layer - the point where data rights are actually exercised or violated - rather than leaving those rights governed solely by policy instruments that AI execution does not interpret.

APPENDIX A4 - DOSSIER IV

Enforcement Deployment Maturity Levels (Illustrative)

TIER 1

Advisory Logging Only

GATE COVERAGE No enforcement gates active. AI assistant interaction logs captured. No token verification, no document-level access control through consent gates.

REGISTRY STATE Registry not integrated with AI assistant or RAG infrastructure. Organizational token inventory may exist but is not queried at runtime.

AUDIT COMPLETENESS Partial. Query and response logs only. No per-source, per-retrieval consent basis recorded.

REVOCATION Not enforced at runtime. Employee offboarding and partner revocation processed via RBAC and data governance workflows, not consent gate invalidation.

CERTIFICATION Does not meet deterministic enforcement criteria.

DEPLOYMENT SIGNAL Experimental

TIER 2

Partial Gate Coverage

GATE COVERAGE Gates active at orchestrator entry point. Sub-agent data access, RAG document retrieval, and pre-fetch operations may remain ungated. Coverage limited to initial query authorization.

REGISTRY STATE Integrated at orchestrator level. Per-document and sub-agent level registry queries not yet implemented.

AUDIT COMPLETENESS Partial. Orchestrator gate evaluation recorded. Sub-agent and document-level access events may be unlinked to consent tokens.

REVOCATION Enforced at orchestrator entry. Sub-agent and RAG-level revocation gap persists. Pre-fetch cache invalidation on revocation not implemented.

CERTIFICATION Does not meet deterministic enforcement criteria. Sub-agent and document-level coverage gaps must be documented.

DEPLOYMENT SIGNAL Controlled Deployment

TIER 3 - MINIMUM FOR DETERMINISTIC ENFORCEMENT

TIER 4

Full Deterministic Enforcement

GATE COVERAGE	All data source access paths gated: orchestrator, sub-agents, RAG document retrieval, pre-fetch operations. No ungated data access to AI-consumed organizational data. Coverage verified by deployment audit. Bypass vector inventory documented and remediated.
REGISTRY STATE	Live registry with DENY-on-unavailable posture. HR-to-registry revocation integration active for employee data tokens. Multi-tenant token scoping verified. Replication lag bounds defined and documented.
AUDIT COMPLETENESS	Complete. Pre-access audit record per gate evaluation at all pipeline stages. Response source lineage record attached to each AI response. Token IDs linked throughout.
REVOCATION	Enforced at all gates. Employee data token revocation triggered by HR system events. Pre-fetch cache invalidation active. Replication window bounded and documented.
CERTIFICATION	Meets QODIQA Core deterministic enforcement standard. Eligible for Tier 3 conformance assessment.
DEPLOYMENT SIGNAL	Production-Critical

Certified Conformance Deployment

GATE COVERAGE	Full coverage as Tier 3, verified by external conformance assessor. Bypass vector inventory reviewed and closed. Administrative access gate inclusion verified. Shadow IT detection controls documented.
REGISTRY STATE	Cryptographic token signing implemented per QODIQA Security and Cryptographic Profile. Jurisdictional token variants for employee data per employment law requirements. Issuer certificate chain verified at each gate evaluation.
AUDIT COMPLETENESS	Tamper-evident audit chain certified. Organizational replay packages validated. Employee data processing records verified for data subject access request fulfillment capability.
REVOCATION	Maximum revocation window formally declared per data subject category. HR integration revocation latency measured and certified. Replication lag bounds certified and included in conformance statement.
CERTIFICATION	QODIQA Tier 4 Conformance Certificate issued. Renewal cadence defined per Certification Framework. Jurisdictional employment law compliance documentation co-submitted as separate annex.
DEPLOYMENT SIGNAL	Regulatory-Grade

Note: Only Tier 3 and above meet the deterministic runtime enforcement criteria defined in QODIQA Core Standard. Enterprise AI deployments present the highest density of potential bypass vectors across all sectors. Organizations must complete a comprehensive bypass vector audit - covering sub-agent access paths, RAG pre-fetch behavior, administrative overrides, and shadow IT - before asserting Tier 3 or above conformance status.

DOCUMENT STATUS

Document Status and Classification

This document is the Use Case Dossier collection of the QODIQA specification corpus. It provides sector-specific deployment analyses and risk-containment frameworks for deterministic runtime consent enforcement across four industry sectors: Healthcare and Clinical AI Systems; Financial Services and Algorithmic Decisioning; Media and Content Generation Platforms; and Enterprise AI Assistants and Knowledge Systems. It is issued as a technical operational annex to the QODIQA Core Standard and is not legal advice. Sector-specific regulatory compliance requires qualified legal counsel in the applicable jurisdiction.

This document is addressed to the following audiences:

- AI system architects and deployment engineers implementing consent enforcement in sector-specific contexts
- Chief privacy officers and data protection officers evaluating runtime consent architecture
- Legal and compliance teams assessing QODIQA deployment requirements by sector
- Regulators and policy analysts reviewing technical implementations of consent infrastructure
- Risk and audit professionals conducting AI governance assessments
- Academic researchers in AI governance, data protection, and consent engineering

This document should be read together with the following related specifications:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper - Version 1.0
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement - Version 1.0
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement - Version 1.0
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement - Version 1.0
- QODIQA: Implementation Playbook for Deterministic Runtime Consent Enforcement - Version 1.0
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement - Version 1.0
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement - Version 1.0
- QODIQA: Threat Model and Abuse Case Specification - Version 1.0
- QODIQA: Governance Charter for the QODIQA Standard Corpus - Version 1.0
- QODIQA: Residual Risk and Assumption Disclosure Annex - Version 1.0

Version 1.0 represents the initial formal release of this document as part of the QODIQA standard corpus.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dițescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-UCD-2026-001
TITLE	Use Case Dossiers for Runtime Consent Enforcement Deployments
SUBTITLE	Sector-Specific Deployment and Risk-Containment Analyses
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Use Case Dossier Collection
DOCUMENT STATUS	Active - Public Release
GOVERNING AUTHORITY	QODIQA Governance Charter
SECTORS COVERED	Healthcare and Clinical AI - Financial Services and Automated Decisioning - Media and Content Generation Platforms - Enterprise AI Assistants and Knowledge Systems
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

CHAPTER 32 PART V · CONTEXTUAL, ANALYTICAL AND PROCESS SUPPORT

Corpus Change Log and Version History

The canonical historical record of all versioned publications, amendments, and governance evolution across the specification corpus.

DOCUMENT	VERSION	PUBLISHED	TYPE	STATUS	MARKER
QODIQA-CCL-2026-001	1.0	2026	Governance Record	Canonical	NOT STATED

ABSTRACT

This document constitutes the Corpus Change Log and Version History for the QODIQA specification corpus. It establishes, with forensic precision and without ambiguity, the formal publication state of the entire QODIQA standard at the moment of its initial synchronized release in 2026. All twenty-eight documents comprising the QODIQA corpus are published simultaneously at Version 1.0, under the governance authority of the QODIQA Governance Charter.

No prior version of any corpus document exists. No pre-release, working draft, or provisional iteration of any corpus document is recognized as part of the formal publication record. This document constitutes the genesis state of the QODIQA governance record, the baseline reference point against which all future amendments, revisions, withdrawals, and governance actions will be measured, and the permanent audit anchor for the QODIQA corpus.

SECTION 1

Initial Corpus Release

The QODIQA specification corpus is formally published in its complete and final form as of 2026. The corpus comprises twenty-eight documents, each released simultaneously, each carrying a Version 1.0 designation, and each classified as Public Release. The synchronized release of all corpus documents is a deliberate and non-negotiable governance requirement: it eliminates the class of conformance failures that arise when implementing parties apply normative requirements drawn from one corpus document against companion specifications that are not yet published, are published at a different version, or are otherwise unavailable at the time of implementation. The initial release date and the Version 1.0 designation constitute a single, indivisible, immutable publication event for the corpus as a whole.

Table 1.1, designated below, is the normative document registry of the QODIQA corpus at the time of this initial release. It is the authoritative reference for corpus composition and version state at the v1.0 baseline. This registry is itself part of the governance record and may not be amended except through a formal governance action under the procedures defined in the QODIQA Governance Charter.

TABLE 1.1: QODIQA CORPUS DOCUMENT REGISTRY, INITIAL RELEASE, 2026, 28 DOCUMENTS

#	DOCUMENT TITLE	DOCUMENT CODE	VERSION	PUBLICATION STATUS	CLASSIFICATION
01	QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper	QODIQA-WP	1.0	Initial Release	Public Release
02	QODIQA: Core Standard for Deterministic Runtime Consent Enforcement	QODIQA-CS	1.0	Initial Release	Public Release
03	QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement	QODIQA-EF	1.0	Initial Release	Public Release

#	DOCUMENT TITLE	DOCUMENT CODE	VERSION	PUBLICATION STATUS	CLASSIFICATION
04	QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement	QODIQA-CF	1.0	Initial Release	Public Release
05	QODIQA: Implementation Playbook for Deterministic Runtime Consent Enforcement	QODIQA-IP	1.0	Initial Release	Public Release
06	QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement	QODIQA-RA	1.0	Initial Release	Public Release
07	QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement	QODIQA-SC	1.0	Initial Release	Public Release
08	QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement	QODIQA-CT	1.0	Initial Release	Public Release
09	QODIQA: Regulatory Alignment Matrix for Deterministic Runtime Consent Enforcement	QODIQA-RM	1.0	Initial Release	Public Release
10	QODIQA: Governance Charter for the QODIQA Standard Corpus	QODIQA-GC	1.0	Initial Release	Public Release
11	QODIQA: Economic Impact Analysis of Deterministic Runtime Consent Enforcement	QODIQA-EI	1.0	Initial Release	Public Release
12	QODIQA: Threat Model and Abuse Case Specification	QODIQA-TM	1.0	Initial Release	Public Release
13	QODIQA: Interoperability and Deployment Constraints	QODIQA-ID	1.0	Initial Release	Public Release
14	QODIQA: Terminology and Normative Definitions	QODIQA-TD	1.0	Initial Release	Public Release
15	QODIQA: Public Specification License	QODIQA-PL	1.0	Initial Release	Public Release

#	DOCUMENT TITLE	DOCUMENT CODE	VERSION	PUBLICATION STATUS	CLASSIFICATION
16	QODIQA: Positioning and Scope Limitation Statement	QODIQA-PS	1.0	Initial Release	Public Release
17	QODIQA: Residual Risk and Assumption Disclosure Annex	QODIQA-RR	1.0	Initial Release	Public Release
18	QODIQA: Use Case Dossiers for Runtime Consent Enforcement Deployments	QODIQA-UC	1.0	Initial Release	Public Release
19	QODIQA: Non-Compliance Conditions and Failure Modes	QODIQA-NC	1.0	Initial Release	Public Release
20	QODIQA: Conformance Verification Specification	QODIQA-CV	1.0	Initial Release	Public Release
21	QODIQA: Audit and Evidence Generation Model	QODIQA-AE	1.0	Initial Release	Public Release
22	QODIQA: System Boundary and Trust Model Specification	QODIQA-SB	1.0	Initial Release	Public Release
23	QODIQA: Failure Handling and Recovery Specification	QODIQA-FH	1.0	Initial Release	Public Release
24	QODIQA: Extended Adversarial Threat Model	QODIQA-AT	1.0	Initial Release	Public Release
25	QODIQA: Interoperability and Deployment Constraints	QODIQA-ID-B	1.0	Initial Release	Public Release
26	QODIQA: Terminology and Normative Definitions	QODIQA-TD-B	1.0	Initial Release	Public Release
27	QODIQA: Positioning and Scope Limitation Statement	QODIQA-PS-B	1.0	Initial Release	Public Release

#	DOCUMENT TITLE	DOCUMENT CODE	VERSION	PUBLICATION STATUS	CLASSIFICATION
28	QODIQA: Residual Risk and Assumption Disclosure Annex	QODIQA-RR-B	1.0	Initial Release	Public Release

REGISTRY CLASSIFICATION NOTE: VARIANT ENTRIES (SUFFIX -B)

Entries 25-28 are classified as Variant Editions of the primary registry entries at positions 13, 14, 16, and 17 respectively. Each variant entry carries an independent document code bearing the suffix `-B`, constituting a separate and distinct publication identity within the corpus. The `-B` suffix designates a parallel canonical registration, not a revision, supplement, or amendment. All 28 registry entries are assigned equal normative standing at Version 1.0.

Table 1.1 is the normative registry of the QODIQA corpus at initial publication. All 28 entries are present as provided in the authoritative corpus document list. This table may not be amended outside a formal governance action recorded in this change log.

SECTION 2

Corpus Synchronization State

All twenty-eight documents constituting the QODIQA corpus were released simultaneously in 2026. No document in the corpus was held in a draft, provisional, pilot, or pre-release state at the time of publication. No staging dependency conflict exists between any two documents in the corpus. Every cross-document reference contained within any corpus document resolves correctly against a simultaneously published companion document. The corpus is therefore internally consistent in full at the v1.0 baseline.

SYNCHRONIZATION VERDICT

The corpus synchronization state is confirmed as **VALID**. All inter-document references resolve correctly within the v1.0 publication boundary. No DEGRADED, PARTIAL, or INCONSISTENT state condition has been identified, logged, or recorded against any corpus document. The corpus is complete, synchronized, and normatively operative as of the 2026 publication date.

The synchronized release model eliminates a well-documented class of governance failures. When a standard corpus is released in stages, implementing parties may inadvertently apply normative requirements from a published document against companion specifications that have not yet been released, that are available only in draft form, or that exist at a different version than that referenced by the applying document. This creates conformance ambiguity, certification risk, and audit fragility. The QODIQA corpus avoids this class of failure entirely by requiring that all twenty-eight documents reach publication readiness simultaneously, and by treating the entire corpus as a single indivisible publication unit for version control purposes.

SYNCHRONIZATION PROPERTIES AT V1.0

- All 28 corpus documents carry identical publication date: 2026.
- All 28 corpus documents carry identical version designation: 1.0.
- All cross-document references resolve within the published v1.0 corpus boundary.
- No dependency on any external, unpublished, pending, or draft document exists.
- No DEGRADED corpus state condition has been identified or logged at the time of this record.
- The corpus synchronization state is declared VALID and entered into the governance record without qualification.

SECTION 3

Version Baseline Lock

All twenty-eight documents in the QODIQA corpus are locked at Version 1.0 as of the date of this record. This baseline lock is the normative starting state for all governance, amendment, certification, conformance assessment, and regulatory alignment activities conducted under or against the QODIQA standard. It is not a provisional or advisory designation. It is a binding governance fact, entered into the formal record without condition or qualification.

BASELINE LOCK STATUS

Version Baseline: `LOCKED AT V1.0`. No document in the corpus may be amended, re-versioned, corrected, supplemented, or withdrawn without formal initiation of the governance workflow defined in the QODIQA Governance Charter (QODIQA-GC v1.0). Any action taken against a corpus document outside of this workflow is without normative effect and shall not be recognized as part of the governance record.

The baseline lock carries the following normative consequences. Any substantive modification to the content of any corpus document, including but not limited to normative requirements, definitions, architecture specifications, test conditions, conformance thresholds, or scope boundaries, shall trigger a full version increment under the governance procedures defined in QODIQA-GC. Corrections of a purely typographical or formatting nature that do not alter normative meaning may be processed under an errata procedure defined in QODIQA-GC, but shall still be recorded in this Corpus Change Log with a corresponding entry conforming to the format defined in Section 4. Silent modifications, undocumented revisions, and out-of-band amendments are categorically prohibited under Section 6 of this document, and any apparent change to a corpus document for which no corresponding change log entry exists shall be treated as unauthorized and shall not be recognized as normatively operative.

The v1.0 lock constitutes a permanent and irrevocable audit anchor. All certification decisions, regulatory alignment submissions, conformance assessments, procurement qualifications, and implementation plans conducted against the QODIQA corpus must explicitly state which version of the corpus they reference. Reference to the QODIQA corpus

without version qualification shall be interpreted as reference to the most recently published locked baseline, with this v1.0 release serving as the initial and foundational anchor point.

SECTION 4

Change Log Entry

This section contains the canonical master entry for the genesis publication event of the QODIQA corpus. This entry is immutable. It may not be modified, superseded, annotated, or deleted by any subsequent governance action. All future change log entries shall follow the structured format defined below, carrying sequential reference numbers beginning at QODIQA-CCL-ENTRY-002.

ENTRY TYPE	INITIAL CORPUS PUBLICATION: GENESIS EVENT
ENTRY REFERENCE	QODIQA-CCL-ENTRY-001
PUBLICATION DATE	2026
CORPUS VERSION	1.0: all 28 corpus documents
DOCUMENTS AFFECTED	All 28 corpus documents: QODIQA-WP, QODIQA-CS, QODIQA-EF, QODIQA-CF, QODIQA-IP, QODIQA-RA, QODIQA-SC, QODIQA-CT, QODIQA-RM, QODIQA-GC, QODIQA-EI, QODIQA-TM, QODIQA-ID, QODIQA-TD, QODIQA-PL, QODIQA-PS, QODIQA-RR, QODIQA-UC, QODIQA-NC, QODIQA-CV, QODIQA-AE, QODIQA-SB, QODIQA-FH, QODIQA-AT, QODIQA-ID-B, QODIQA-TD-B, QODIQA-PS-B, QODIQA-RR-B.
CHANGE TYPE	Genesis Publication. This is not an amendment to any prior version. No prior version of any affected document exists in the governance record.
RATIONALE	Initial formal and synchronized release of the complete QODIQA specification corpus as a publicly available, internally consistent, normatively operative standard for deterministic runtime consent enforcement in artificial intelligence systems.
PRIOR VERSION	None. This entry records the genesis state of the corpus. No prior formal publication exists for any corpus document.
GOVERNANCE AUTHORITY	QODIQA Governance Charter (QODIQA-GC v1.0)
CLASSIFICATION	Public Release
ENTRY STATUS	CANONICAL AND IMMUTABLE. This entry may not be amended, annotated, superseded, or deleted by any future governance action. It constitutes the permanent opening record of the QODIQA Corpus Change Log.

SECTION 5

Governance Implication

The publication of the QODIQA corpus at Version 1.0 represents a precise and consequential transition in the status and function of the QODIQA standard. Prior to this publication, QODIQA existed exclusively as a framework under development: a body of conceptual and technical work without formal standing in any governance, regulatory, or contractual context. With this publication, the corpus transitions completely and irrevocably from the pre-normative development phase into its status as an enforceable specification standard.

This transition carries concrete and immediate governance implications. The normative requirements distributed across the corpus documents, in particular those contained in the Core Standard (QODIQA-CS), the 68-Point Enforcement Framework (QODIQA-EF), the Certification Framework (QODIQA-CF), and the Conformance Verification Specification (QODIQA-CV), now constitute formally binding constraints against which implementing parties, certification bodies, and assurance programs may conduct assessments. The definitions established in the Terminology and Normative Definitions (QODIQA-TD) are now the authoritative terminological reference for all QODIQA-related documentation, technical correspondence, regulatory submissions, and contractual instruments.

The governance procedures established in the Governance Charter (QODIQA-GC) become operative upon publication. From this date, all amendments to any corpus document are subject to the controls and procedures defined in that charter. The Corpus Change Log, of which this document is the initial and foundational entry, becomes the authoritative audit trail for all governance events affecting the corpus.

GOVERNANCE TRANSITION SUMMARY

- PRE-PUBLICATION STATE:
  Conceptual and pre-normative framework, internal only, without formal governance standing.
- POST-PUBLICATION STATE:
  Enforceable specification standard, publicly available, normatively operative, open for certification and regulatory reference.
- GOVERNANCE OPERABILITY DATE:
  2026.
- CERTIFICATION OPERABILITY DATE:
  2026.
- AUDIT TRAIL OPENING DATE:
  2026, with this document as the initial entry.

SECTION 6

Prohibited States

The following states are categorically and permanently prohibited for any document within the QODIQA corpus following the publication of this record. These prohibitions apply without exception, without waiver, and without the possibility of suspension except through a formal revision of this document conducted under the governance procedures of QODIQA-GC.

PROHIBITED STATE 1: PRE-V1 HISTORY

PROHIBITED The QODIQA corpus has no formal history prior to Version 1.0. No document, record, draft, working paper, or prior publication may be cited as a prior version of any corpus document within any formal, regulatory, legal, or governance context. Claims of pre-v1 corpus history are factually incorrect as a matter of governance record and must not be recognized by any party operating under or against the QODIQA standard.

PROHIBITED STATE 2: UNDOCUMENTED CHANGES

PROHIBITED No modification of any kind may be made to any corpus document without a corresponding entry in this Corpus Change Log, recorded in the format defined in Section 4. A modification for which no corresponding change log entry exists is invalid as a matter of governance record. The absence of a change log entry renders any apparent modification void, without normative effect, and unrecognized by the governance record, regardless of where the modified document is published or distributed.

PROHIBITED STATE 3: SILENT UPDATES

PROHIBITED Silent updates of any kind, including corrections of typographical errors, formatting adjustments, cross-reference corrections, and numbering changes, are prohibited if applied without corresponding disclosure to the QODIQA governance authority and without a corresponding entry in this change log. The determination of whether a change is substantive or editorial is made exclusively by the governance authority defined in QODIQA-GC, not by the party proposing or applying the change.

PROHIBITED STATE 4: OUT-OF-BAND VERSIONING

PROHIBITED No corpus document may carry any version designation, revision marker, draft label, working copy indicator, or equivalent annotation other than the version designation assigned through the formal governance workflow defined in QODIQA-GC and recorded in this change log. Informal versioning practices are not permitted as substitutes for formal version control and shall not be treated as normatively operative by any party.

SECTION 7

Forward Compatibility

This document is designed and required to serve as the permanent and unbroken historical record of the QODIQA corpus across its entire governance lifetime, without limit of time. All future change log entries, regardless of the nature, scope, or number of corpus documents they affect, shall be appended to this document in strict chronological order. The document therefore functions simultaneously as a historical record and as a governance instrument. Its integrity is a direct precondition for the integrity of the corpus itself.

Future entries shall follow the structured identity block format established in Section 4. Each entry shall carry a unique sequential reference number in the format QODIQA-CCL-ENTRY-NNN, where NNN is a zero-padded three-digit integer. The master entry recorded in this initial release carries reference QODIQA-CCL-ENTRY-001. The next governance event affecting any corpus document shall carry reference QODIQA-CCL-ENTRY-002, and the sequence shall continue without gap, without duplication, and without reuse of previously assigned reference numbers.

This document is itself subject to the same version control and governance controls as all other corpus documents. Any modification to the structure, format, procedural rules, or content of this Corpus Change Log requires a formal governance action under QODIQA-GC and a corresponding entry within the change log itself.

FORWARD COMPATIBILITY RULES

- All future entries are appended chronologically. No retroactive insertion is permitted.
- Entry numbering is strictly sequential with no gaps, reuse, or duplication.
- The genesis entry QODIQA-CCL-ENTRY-001 is immutable and may not be altered by any future governance action.
- This document constitutes the permanent audit anchor for the QODIQA corpus. All version history derives from this record.
- Any certification body, regulatory authority, or implementing party referencing the QODIQA corpus shall treat this document as the sole authoritative version history.
- Modifications to this change log document are themselves subject to change log entry requirements.

GOVERNANCE STATEMENT

Governance Statement

CANONICAL GOVERNANCE DECLARATION: QODIQA-CCL-2026-001 V1.0, 2026

The QODIQA Corpus Change Log and Version History, in the form of this document at Version 1.0, constitutes the immutable starting point of the QODIQA governance record. It establishes, without ambiguity and without the possibility of reinterpretation by any party or under any circumstances, the complete and final publication state of the QODIQA specification corpus at the moment of its initial formal release in 2026.

Every corpus document listed in Section 1, Table 1.1 of this record is published at Version 1.0. No document in that table has a prior published version. No document in that table is in a partial, draft, provisional, or experimental state. The corpus is complete, internally consistent, and normatively operative as of the date recorded herein. The corpus synchronization state is VALID. The version baseline is LOCKED AT v1.0.

This record does not invite interpretation. It does not leave open questions regarding the publication state of any corpus document. It does not accommodate competing accounts of corpus history. The publication state documented here is a matter of governance fact, established through the formal publication process and recorded in the change log maintained under the authority of the QODIQA Governance Charter. Any assertion that conflicts with the contents of this document is without basis in the governance record and must be disregarded entirely by all parties operating in any governance, certification, regulatory, legal, or procurement context that references the QODIQA standard.

Future governance bodies, certification authorities, regulatory reviewers, procurement officers, legal counsel, and implementing organizations are directed to treat this document as the foundational layer of the QODIQA audit trail, the layer from which all subsequent governance history derives its authority, its sequence, and its meaning. No governance history predates this record. No governance history can supersede this record except through the formal amendment process that this record itself establishes and governs.

DOCUMENT STATUS

Document Status and Corpus Alignment

This document is the Corpus Change Log and Version History of the QODIQA specification corpus. It is a governance-class record, not a technical specification. Its function is to provide a forensic-grade, audit-ready account of the complete publication and amendment history of all corpus documents, beginning with the initial synchronized release at Version 1.0 in 2026 and extending forward across the full governance lifetime of the corpus.

This document forms part of the QODIQA corpus and should be read in conjunction with all constituent corpus documents. The complete list of corpus documents at the v1.0 baseline is as follows:

- QODIQA: Consent as Infrastructure for Artificial Intelligence Technical Whitepaper
- QODIQA: Core Standard for Deterministic Runtime Consent Enforcement
- QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement
- QODIQA: Implementation Playbook for Deterministic Runtime Consent Enforcement
- QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement
- QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement
- QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement

- QODIQA: Regulatory Alignment Matrix for Deterministic Runtime Consent Enforcement
- QODIQA: Governance Charter for the QODIQA Standard Corpus
- QODIQA: Economic Impact Analysis of Deterministic Runtime Consent Enforcement
- QODIQA: Threat Model and Abuse Case Specification
- QODIQA: Interoperability and Deployment Constraints
- QODIQA: Terminology and Normative Definitions
- QODIQA: Public Specification License
- QODIQA: Positioning and Scope Limitation Statement
- QODIQA: Residual Risk and Assumption Disclosure Annex
- QODIQA: Use Case Dossiers for Runtime Consent Enforcement Deployments
- QODIQA: Non-Compliance Conditions and Failure Modes
- QODIQA: Conformance Verification Specification
- QODIQA: Audit and Evidence Generation Model
- QODIQA: System Boundary and Trust Model Specification

- QODIQA: Failure Handling and Recovery Specification
- QODIQA: Extended Adversarial Threat Model
- QODIQA: Interoperability and Deployment Constraints
- QODIQA: Terminology and Normative Definitions
- QODIQA: Positioning and Scope Limitation Statement
- QODIQA: Residual Risk and Assumption Disclosure Annex

Version 1.0 of this document represents the initial formal release of the Corpus Change Log as part of the QODIQA standard corpus. This document is canonical, its genesis entry is immutable, and it constitutes the authoritative starting point and permanent audit anchor for all governance proceedings, certification activities, regulatory reviews, and dispute resolution processes conducted against the QODIQA standard.

For strategic inquiries, architectural discussions, or partnership exploration:

Bogdan Dușescu

contact@qodiqa.com

www.qodiqa.com

DOCUMENT IDENTIFIER	QODIQA-CCL-2026-001
TITLE	Corpus Change Log and Version History
SUBTITLE	Deterministic Runtime Consent Enforcement for Artificial Intelligence Systems
PUBLICATION DATE	2026
VERSION	1.0
DOCUMENT TYPE	Governance Record
DOCUMENT STATUS	Canonical
GOVERNING AUTHORITY	QODIQA Governance Charter (QODIQA-GC v1.0)
CORPUS DOCUMENTS	28 documents, all registered at v1.0, 2026
GENESIS ENTRY	QODIQA-CCL-ENTRY-001: Immutable
INTEGRITY NOTICE	Document integrity may be verified using the official SHA-256 checksum distributed with the QODIQA specification corpus.

APPENDIX A

Glossary

Terms and definitions reproduced verbatim from QODIQA-TND-2026-001 (Terminology and Definitions); no definition is added. Each entry links to its place in that document. 48 entries, in alphabetical order.

Amendment AMD

GOVERNANCE

A versioned modification to a QODIQA specification document, processed through the change control procedure defined in Section 8. An Amendment that alters a normative definition in this specification constitutes a new version of this document. Amendments are recorded as Immutable Records and are appended to the document revision history.

(Reference: QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0)

Append-Only Record AOR

ARTIFACT

A record for which the only permitted write operation is creation. No update, deletion, or modification operation is permitted after creation. All Consent Artifacts and Audit Artifacts within the QODIQA framework are Append-Only Records. Revocation of a Consent Artifact is achieved through the creation of a new Revocation State record, not through modification of the original.

Artifact Signing AS

PROCESS

The cryptographic operation by which a Cryptographic Integrity Anchor is computed and attached to a record at the time of its creation. Artifact Signing uses private key operations as specified in the QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0 and is performed exactly once per record, at creation time. Re-signing of existing records is PROHIBITED.

(Reference: QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0)

Audit Artifact AA

ARTIFACT

A tamper-evident, cryptographically anchored record generated at the completion of every enforcement evaluation, capturing the complete set of inputs, the resolved consent state, the applicable policy version, and the resulting Enforcement Action. Audit Artifacts are appended to the Immutable Log and are the primary mechanism through which enforcement decisions are made verifiable and reproducible.

(Reference: QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0; QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0)

Blocking Event BE

CONTROL

A condition arising during enforcement evaluation that causes the Policy Engine to halt processing and return a deny decision. Blocking Events include, without limitation: absent consent, expired consent, consent scope mismatch, policy rule violation, and integrity verification failure. All Blocking Events are recorded as Audit Artifacts.

Breaking Change BC

GOVERNANCE

A modification to a QODIQA specification that alters the meaning, scope, or binding status of a normative requirement such that implementations previously assessed as conformant may no longer satisfy the amended requirement without modification. Breaking Changes require a major version increment and mandatory re-assessment of affected certified implementations.

Canonical Version CV

GOVERNANCE

The current, authoritatively published version of a QODIQA specification document as recognized by the QODIQA Governance Charter. The Canonical Version is the only version against which conformance assessments and certification claims are valid. Prior versions are superseded upon publication of a Canonical Version but remain accessible as archived references.

(Reference: QODIQA: Governance Charter for the QODIQA Standard Corpus, Version 1.0)

Certification CERT

GOVERNANCE

The formal recognition issued by an authorized certifying body upon verified attainment of a specified Certification Level. Certification is based on documented evidence of Conformance and does not extend beyond the scope, version, and configuration of the implementation assessed. Certification SHALL NOT be claimed for configurations, versions, or deployment contexts not covered by the assessment.

(Reference: QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0)

Certification Level CL

CONTROL

A defined tier of conformance achievement, as specified in the QODIQA Certification Framework, indicating the scope of control points satisfied by an implementation. Certification Levels are ordinal, cumulative, and verifiable. Attainment of a higher Certification Level requires satisfaction of all requirements at lower levels plus additional level-specific requirements.

(Reference: QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0)

Compliance COMP

GOVERNANCE

The state of an organization or system in which applicable legal, regulatory, or contractual obligations are satisfied. Compliance is a legal determination, not a technical one. QODIQA Conformance and Certification are evidence inputs that MAY be relevant to a compliance determination but do not constitute compliance with any regulatory framework. Organizations remain solely responsible for their own compliance assessments.

INFORMATIVE NOTE

The distinction between Conformance and Compliance is intentional and significant. QODIQA specifications assert technical properties only. Claims of regulatory compliance, with GDPR, the EU AI Act, NIST AI RMF, or any other framework, must be made by qualified legal and compliance professionals, not derived from QODIQA conformance status alone.

(Reference: QODIQA: Regulatory Alignment Matrix for Deterministic Runtime Consent Enforcement, Version 1.0; QODIQA: Positioning and Scope Limitation Statement, Version 1.0)

Conformance CON

GOVERNANCE

The verifiable state of an implementation in which all mandatory requirements of the applicable QODIQA specification set, as assessed through the Conformance Test Suite, are satisfied. Conformance is a technical determination. It does not imply, guarantee, or substitute for regulatory compliance with any external legal framework.

(Reference: QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0; QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0)

Conformance Validation Layer CVL

ARCHITECTURAL

The system or procedural layer responsible for verifying that an implementation satisfies the requirements of the QODIQA Conformance Test Suite and Certification Framework. The Conformance Validation Layer operates independently of the Enforcement Layer and does not participate in runtime enforcement decisions.

(Reference: QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0)

Consent C

ARTIFACT

An explicit, machine-readable authorization record granted by an identified principal, scoped to a defined action type, bounded by a declared purpose, constrained by temporal validity limits, and subject to revocation. Consent within the QODIQA framework is not inferred, assumed, or derived from behavioral signals. It exists as a discrete, addressable record in the Consent Registry.

INFORMATIVE NOTE

This definition deliberately excludes implied consent, browsewrap consent, and terms-of-service acceptance. Those forms carry no normative standing within QODIQA enforcement logic.

EXPLICIT NON-INCLUSIONS

The following do not constitute Consent within this specification: inferred consent derived from behavioral patterns or usage history; assumed consent based on prior interactions; contextual consent derived from conversational or environmental signals; consent implied by silence or inaction; and any authorization not recorded as a discrete Consent Artifact in the Consent Registry.

Consent Artifact CA

ARTIFACT

A cryptographically signed, immutable record representing a discrete consent state at a specific point in time. A Consent Artifact contains, at minimum: the identity of the consenting principal, the scope of the consented action, the declared purpose, the temporal validity window, and an integrity anchor. Consent Artifacts are the unit of consent evaluated by the Policy Engine.

(Reference: QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0)

Consent Registry CR

ARCHITECTURAL

The authoritative, append-only data store in which all Consent Artifacts are recorded, indexed, and made retrievable for enforcement evaluation. The Consent Registry is the sole source of consent truth within the QODIQA framework. Consent not recorded in the Consent Registry does not exist for enforcement purposes.

(Reference: QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0)

Consent Validation CV

PROCESS

The process by which the enforcement system verifies that a Consent Artifact exists in the Consent Registry, is cryptographically intact, is temporally valid at the moment of evaluation, and is scoped to cover the action declared in the execution request. Consent Validation precedes Policy Evaluation in all enforcement flows.

(Reference: QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0)

Control Surface CS

CONTROL

The complete set of defined control points at which enforcement decisions are made or enforcement conditions are checked within a QODIQA implementation. The Control Surface is enumerated in the QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0 and constitutes the measurable scope of enforcement coverage.

(Reference: QODIQA: 68-Point Enforcement Framework for Deterministic Runtime Consent Enforcement, Version 1.0)

Cryptographic Binding CB

ARTIFACT

The association between a record, such as a Consent Artifact or Audit Artifact, and a cryptographic value such that any modification to the record's content renders the cryptographic value invalid. Cryptographic Binding is achieved through digital signature or keyed hash mechanisms as specified in the QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0.

(Reference: QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0)

Cryptographic Integrity Anchor CIA

ARTIFACT

A cryptographic value, specifically a digital signature or a hash chain element, bound to a specific record and used to verify that the record's content has not been altered since its creation. Cryptographic Integrity Anchors are required for all Consent Artifacts and Audit Artifacts within QODIQA-conformant implementations.

(Reference: QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0)

Decision Boundary DB

CONTROL

The complete set of conditions, constraints, and input parameters that fully determine the output of a single enforcement evaluation. A Decision Boundary is closed: no inputs outside its declared scope affect the evaluation outcome. Extension of the Decision Boundary requires a versioned policy amendment.

(Reference: QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0)

Deterministic Evaluation Path DEP

CONTROL

The complete sequence of processing steps (from request receipt at the Enforcement Gateway through consent validation, policy evaluation, and Audit Artifact generation) that produces an enforcement decision. The Deterministic Evaluation Path is fully defined, has no optional branches under normal operating conditions, and must be traversed in its entirety for each request.

(Reference: QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0)

Deterministic Runtime Consent Enforcement DRCE

FOUNDATIONAL

A system property whereby every AI execution request is evaluated against explicitly declared consent state and applicable policy rules at the moment of invocation, producing a bounded output (allow, deny, or allow with stated restrictions) that is reproducible given identical inputs. The evaluation process introduces no probabilistic inference, no contextual assumption, and no default-to-permit behavior.

EXPLICIT NON-INCLUSIONS

This definition does not include: probabilistic inference of enforcement outcomes; contextual assumption of consent from behavioral or usage signals; default-to-permit behavior in the absence of an explicit consent record; model-derived or AI-generated authorization decisions; and post-hoc enforcement applied after execution has occurred.

(Reference: QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0)

Deterministic System DS

FOUNDATIONAL

A system in which, given a fixed set of inputs and a fixed rule set at a fixed version, the evaluation process always yields the same output with no variance. Determinism within the QODIQA framework applies specifically to the enforcement evaluation path and does not constrain the behavior of AI models or downstream services operating outside that path.

(Reference: QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0)

Enforcement Action ENF

PROCESS

An operation performed by the Enforcement Gateway as a result of a completed enforcement evaluation. Enforcement Actions include: granting Execution Authorization, returning a deny decision, returning a conditional authorization with stated restrictions, and recording an Audit Artifact. An Enforcement Action is distinct from and independent of any downstream AI operation.

Enforcement Gateway EG

ARCHITECTURAL

The system component through which all AI execution requests must pass before reaching the execution layer. The Enforcement Gateway is responsible for receiving execution requests, triggering consent validation, invoking policy evaluation, and returning a deterministic enforcement decision. No execution request may bypass the Enforcement Gateway.

(Reference: QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0)

Enforcement Layer EL

ARCHITECTURAL

The logical tier of the QODIQA reference architecture comprising the Enforcement Gateway, Policy Engine, and Consent Validation components. The Enforcement Layer is the sole site of enforcement decision-making. No other architectural layer produces or modifies enforcement decisions.

(Reference: QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0)

Execution Authorization EA

CONTROL

A deterministic, time-stamped enforcement decision produced by the Policy Engine that permits a specified execution request to proceed under stated conditions. Execution Authorization is not granted by default and is not presumed by the absence of a deny decision. It must be explicitly produced as an output of a completed enforcement evaluation.

External Dependency XD

ARCHITECTURAL

Any system, service, or data source not contained within the QODIQA Enforcement Layer upon which the enforcement evaluation process relies. External Dependencies are subject to availability failure and MUST be treated as potential sources of Blocking Events. A QODIQA-conformant implementation documents all External Dependencies and specifies failure behavior for each.

(Reference: QODIQA: Residual Risk and Assumption Disclosure Annex, Version 1.0)

Fail-Closed State FCS

STATE

The system condition in which, when required inputs for enforcement evaluation are absent, incomplete, or unresolvable, the default output is a deny decision. No execution proceeds in the absence of a complete and valid enforcement evaluation. Fail-Closed State is a mandatory property of all QODIQA-conformant enforcement implementations.

(Reference: QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0)

Immutable Log IL

ARCHITECTURAL

The append-only, cryptographically chained sequence of Audit Artifacts produced by the Enforcement Layer. Entries in the Immutable Log cannot be altered or deleted after being written. The Immutable Log is the primary evidence source for enforcement audits and reproducibility verification.

(Reference: QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0; QODIQA: Conformance Test Suite Specification for Deterministic Runtime Consent Enforcement, Version 1.0)

Immutable Record IR

ARTIFACT

A data record whose content, once written, cannot be altered, deleted, or overridden by any system operation or administrative action. Within the QODIQA framework, Immutable Records include Consent Artifacts, Audit Artifacts, and policy version snapshots. Immutability is enforced through append-only storage architecture and cryptographic integrity anchoring.

Inference Boundary IB

CONTROL

The declared limit of the scope of execution authorized for a given AI invocation. The Inference Boundary constrains what the AI system may produce, access, or affect as a result of an authorized execution. Outputs or effects falling outside the declared Inference Boundary constitute a boundary violation and SHALL be treated as a Blocking Event in subsequent evaluations.

Integrity Failure Condition IFC

STATE

A system state in which the result of Integrity Verification for a record is negative, indicating that the record's content does not match its Cryptographic Binding. An Integrity Failure Condition is an immediate Blocking Event. The affected record SHALL NOT be used as input to enforcement evaluation. The condition SHALL be recorded in the Immutable Log.

Integrity Verification IV

PROCESS

The process of confirming that a record's content matches its Cryptographic Binding, establishing that the record has not been altered since its creation. Integrity Verification is a mandatory step in Consent Validation and MUST be performed for every Consent Artifact retrieved from the Consent Registry during an enforcement evaluation.

(Reference: QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0)

Interpretive Drift ID

GOVERNANCE

The condition in which the practical application of a term diverges from its definition in this specification, whether through informal restatement, paraphrase, omission, or contextual substitution. Interpretive Drift constitutes semantic non-conformance. Its detection obligates the responsible party to align usage with this specification through documented correction.

(Reference: Section 7 · Semantic Integrity Requirements)

Key Management Boundary KMB

CONTROL

The defined operational and architectural perimeter within which cryptographic key material used for Artifact Signing is generated, stored, and used. Key material SHALL NOT cross the Key Management Boundary in plaintext form. Access to key material is restricted to the signing operation and is not available to enforcement evaluation logic.

(Reference: QODIQA: Security and Cryptographic Profile for Runtime Consent Enforcement, Version 1.0)

Non-Coverage Disclosure NCD

GOVERNANCE

A formal statement accompanying a conformance or certification claim identifying the specific control points, domains, or requirements that fall outside the scope of the assessed implementation. Non-Coverage Disclosures are mandatory for partial-scope implementations. Their absence is treated as a claim of full-scope coverage.

(Reference: QODIQA: Certification Framework for Deterministic Runtime Consent Enforcement, Version 1.0)

Non-Execution State NES

STATE

The state in which an AI execution request has been received by the Enforcement Gateway but has not been authorized. A system in Non-Execution State has neither permitted nor committed to any downstream action. The Non-Execution State is the initial and default state for all requests entering the Enforcement Gateway.

Normative Revision NR

GOVERNANCE

Any Amendment to a QODIQA specification document that alters, adds, or removes normative requirements. Normative Revisions are distinguished from editorial revisions, which correct typographical or formatting errors without changing meaning. Normative Revisions require a version increment according to the versioning scheme defined in Section 8.

Policy Engine PE

CONTROL

The component responsible for evaluating declared execution intent against applicable policy rules and resolved consent state, producing a deterministic enforcement decision. The Policy Engine operates on explicitly declared inputs only. It does not access raw data, perform semantic interpretation of content, or apply probabilistic risk scoring.

EXPLICIT NON-INCLUSIONS

The Policy Engine does not perform: semantic analysis or natural language interpretation of content; probabilistic risk scoring or threat modeling; contextual inference from behavioral or environmental signals; model-derived reasoning about intent; and any function that introduces variance into enforcement output for identical inputs.

(Reference: QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0; QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0)

Policy Evaluation Module PEM

ARCHITECTURAL

The bounded component within the Policy Engine responsible for applying a specific versioned policy rule set against resolved consent state and declared execution intent. The Policy Evaluation Module receives only the inputs required for evaluation and returns only a binary or ternary enforcement decision. It retains no state between evaluations.

Registry Interface RI

ARCHITECTURAL

The defined programmatic interface through which the Enforcement Layer queries the Consent Registry. The Registry Interface is the exclusive access path to Consent Artifact data for enforcement evaluation purposes. Direct registry access by components outside the Enforcement Layer is PROHIBITED.

Revocation State RS

STATE

A record in the Consent Registry asserting that a specific Consent Artifact is no longer valid as of a declared timestamp. A Revocation State record does not alter the original Consent Artifact. Consent Validation MUST check for the existence of a Revocation State record for any Consent Artifact retrieved during an enforcement evaluation. A revoked Consent Artifact SHALL NOT authorize execution.

(Reference: QODIQA: Core Standard for Deterministic Runtime Consent Enforcement, Version 1.0)

Runtime RT

STATE

The temporal and computational interval during which an AI execution request is processed, from the point of request submission to the point at which an authorized or denied execution outcome is produced. All consent evaluation and enforcement operations within the QODIQA framework occur within this interval.

Runtime Invocation RVI

CONTROL

A single, discrete request submitted to the Enforcement Gateway for evaluation, associated with one declared intent, one identified principal, and one target action scope. Each Runtime Invocation is evaluated independently. The outcome of a prior Runtime Invocation does not affect the evaluation of a subsequent one, except where applicable policy rules explicitly reference prior enforcement history.

Scope Limitation SL

GOVERNANCE

A declared boundary on the applicability of a QODIQA specification, certification claim, or conformance assessment. Scope Limitations identify the deployment contexts, data categories, jurisdictions, or operational conditions to which an assessed claim applies. Claims made outside a declared Scope Limitation are unverified.

(Reference: QODIQA: Positioning and Scope Limitation Statement, Version 1.0)

Tamper Detection TD

PROCESS

The capability of a system component to identify that a record's content does not match its Cryptographic Binding. Detection of tamper constitutes an Integrity Failure Condition and SHALL trigger a Blocking Event. Tamper Detection is distinct from tamper prevention; it is a detection and response property, not a prevention property.

Trust Boundary TB

CONTROL

A declared demarcation between system components or operational domains across which data, requests, or credentials must be explicitly validated before being accepted as input to enforcement evaluation. Trust is not transitive across Trust Boundaries. Each boundary crossing requires independent verification.

(Reference: QODIQA: Reference Architecture for Deterministic Runtime Consent Enforcement, Version 1.0)

APPENDIX B

Index of requirement identifiers

660 identifiers that appear as item labels (an identifier cell or label heading an item, table row or block) in the documents. Each entry gives the chapter number and the page where the identifier is set; an identifier set in more than one document lists each. Identifiers cited only in running text are not indexed.

A

A-01 22· 801
A-02 22· 801
A-03 22· 802
A-04 22· 802
A-05 22· 802
A-06 22· 803
A-07 22· 803
A-08 22· 803
A-09 22· 804
A-10 22· 804
A-11 22· 804
A-12 22· 805

AE

AE-01 23· 880
AE-02 23· 880
AE-03 23· 880
AE-04 23· 880
AE-05 23· 881
AE-06 23· 881

ART-D1

ART-D1-01 31·1134
ART-D1-02 31·1134
ART-D1-03 31·1135
ART-D1-04 31·1135
ART-D1-05 31·1135
ART-D1-06 31·1135

ART-D2

ART-D2-01 31·1147
ART-D2-02 31·1147
ART-D2-03 31·1147
ART-D2-04 31·1147

ART-D3

ART-D3-01 31·1159
ART-D3-02 31·1159
ART-D3-03 31·1159
ART-D3-04 31·1159

ART-D4

ART-D4-01 31·1172
ART-D4-02 31·1172
ART-D4-03 31·1172
ART-D4-04 31·1172
ART-D4-05 31·1173
ART-D4-06 31·1173

AUD

AUD-01 28·1032
AUD-02 28·1032
AUD-03 28·1032
AUD-04 28·1032
AUD-05 28·1032

C

C-01 20· 739
C-02 20· 739
C-03 20· 740
C-04 20· 740
C-05 20· 740
C-06 20· 740
C-07 20· 740

CERT

CERT-01 28·1033
CERT-04 28·1033

CI

CI-01 22· 812
CI-02 22· 813
CI-03 22· 813
CI-04 22· 814

CIPR

CIPR-01 21· 776

CL	
CL-01	13· 509 23· 841
CL-02	13· 509 23· 841
CL-03	13· 509 23· 842
CL-04	13· 509 23· 843
CL-05	13· 509
CL-06	13· 509
CL-07	13· 509
CL-08	13· 510
CL-09	13· 510
CL-10	13· 510
CL-11	13· 510
CL-12	13· 510
CL-13	13· 510
CL-14	13· 510
CL-15	13· 511
CL-16	13· 511
CL-17	13· 511
CL-18	13· 511
CL-19	13· 511
CL-20	13· 511
CL-21	13· 511
CL-22	13· 512
CL-23	13· 512
CL-24	13· 512
CL-25	13· 513
CL-26	13· 513
CL-27	13· 513
CL-28	13· 513
CL-29	13· 513
CL-30	13· 513
CL-31	13· 513
CL-32	13· 514
CL-33	13· 514
CL-34	13· 514
CL-35	13· 514
CL-36	13· 514
CL-37	13· 514
CL-38	13· 515
CL-39	13· 515
CL-40	13· 515
CL-41	13· 515
CL-42	13· 515
CL-43	13· 515
CL-44	13· 515
CL-45	13· 516
CL-46	13· 516
CL-47	13· 516
CL-48	13· 516
CL-49	13· 516
CL-50	13· 517
CL-51	13· 517
CL-52	13· 517
CL-53	13· 517
CL-54	13· 517
CL-55	13· 517
CL-56	13· 517
CL-57	13· 518
CL-58	13· 518
CL-59	13· 518
CL-60	13· 518
CL-61	13· 518
CL-62	13· 518
CL-63	13· 518
CL-64	13· 518
CL-65	13· 519
CL-66	13· 519
CL-67	13· 519
CL-68	13· 519

CP	
CP-01	13· 509
CP-02	13· 509
CP-03	13· 509
CP-04	04· 127 13· 509
CP-05	04· 127 13· 509
CP-06	04· 127
CP-07	04· 127
CP-08	04· 127
CP-09	04· 127 13· 509
CP-10	04· 127 13· 509
CP-11	04· 127 13· 510
CP-12	04· 127 13· 510
CP-13	04· 128 13· 510
CP-14	04· 128
CP-15	04· 128
CP-16	04· 128
CP-17	04· 128 13· 510
CP-18	04· 128 13· 510
CP-19	04· 128 13· 510
CP-20	04· 128 13· 510
CP-21	04· 128
CP-22	04· 129
CP-23	04· 129
CP-24	04· 129 13· 511
CP-25	04· 129 13· 511
CP-26	04· 129 13· 511
CP-27	04· 129
CP-28	04· 129
CP-29	04· 130
CP-30	04· 130 13· 511
CP-31	04· 130 13· 511
CP-32	04· 130 13· 511

CP-33 04· 130 13· 511
CP-34 04· 130 13· 512
CP-35 04· 130 13· 512
CP-36 04· 130 13· 512
CP-37 04· 130 13· 513
CP-38 04· 130 13· 513
CP-39 04· 131 13· 513
CP-40 04· 131 13· 513
CP-41 04· 131 13· 513
CP-42 04· 131 13· 513
CP-43 04· 131 13· 513
CP-44 04· 131 13· 514
CP-45 04· 131 13· 514
CP-46 04· 131
CP-47 04· 131
CP-48 04· 131 13· 514
CP-49 04· 131 13· 514
CP-50 04· 132 13· 514
CP-51 04· 132 13· 514
CP-52 04· 132
CP-53 04· 132
CP-54 04· 132 13· 515
CP-55 04· 132 13· 515
CP-56 04· 132 13· 515
CP-57 04· 133 13· 515
CP-58 04· 133 13· 515
CP-59 04· 133 13· 515
CP-60 04· 133 13· 515
CP-61 04· 133 13· 516
CP-62 04· 133 13· 516
CP-63 04· 133 13· 516
CP-64 04· 133 13· 516
CP-65 04· 133 13· 516
CP-66 04· 133 13· 517
CP-67 04· 134 13· 517
CP-68 04· 134 13· 517
CP-69 13· 517
CP-70 13· 517
CP-71 13· 517
CP-72 13· 517
CP-73 13· 518
CP-74 13· 518
CP-75 13· 518
CP-76 13· 518
CP-77 13· 518
CP-78 13· 518
CP-79 13· 518
CP-80 13· 518
CP-81 13· 519
CP-82 13· 519
CP-83 13· 519
CP-84 13· 519

CT

CT-01 28·1029
CT-02 28·1029
CT-03 28·1029

CTS-AF

CTS-AF-01 08· 329 16· 592
CTS-AF-02 08· 329 16· 592
CTS-AF-03 08· 329 16· 593
CTS-AF-04 16· 593

CTS-BF

CTS-BF-01 08· 330 16· 595
CTS-BF-02 08· 330 16· 596
CTS-BF-03 16· 596

CTS-CF

CTS-CF-01 08· 329 16· 594
CTS-CF-02 08· 329 16· 594
CTS-CF-03 16· 595

CTS-CP

CTS-CP-01 16· 590
CTS-CP-02 16· 591
CTS-CP-03 16· 591

CTS-DAI

CTS-DAI-01 16· 589
CTS-DAI-02 16· 589
CTS-DAI-03 16· 590

CTS-DI

CTS-DI-01 08· 328 16· 587
CTS-DI-02 08· 328 16· 588
CTS-DI-03 08· 329 16· 588

CTS-DR

CTS-DR-01 08· 329
CTS-DR-02 08· 329

CTS-EF

CTS-EF-01 16· 586
CTS-EF-02 08· 328 16· 586
CTS-EF-03 08· 328 16· 587

ENF

ENF-01 28·1030
ENF-07 28·1031

FC

FC-01 16· 597
FC-02 16· 597
FC-03 16· 597
FC-04 16· 597
FC-05 16· 597
FC-06 16· 597
FC-07 16· 598
FC-08 16· 598
FC-09 16· 598
FC-10 16· 598

FCT

FCT-01 23· 850
FCT-02 23· 850
FCT-03 23· 850
FCT-04 23· 850
FCT-05 23· 850

GI

GI-01 11· 425
GI-02 11· 425
GI-03 11· 426
GI-04 11· 427
GI-05 11· 427
GI-06 11· 428
GI-07 11· 428
GI-08 11· 429
GI-09 11· 429
GI-10 11· 430
GI-11 11· 430
GI-12 11· 431
GI-13 11· 431
GI-14 11· 432
GI-15 11· 432
GI-16 11· 433
GI-17 11· 433
GI-18 11· 434
GI-19 11· 435
GI-20 11· 435

GP

GP-01 21· 757
GP-02 21· 757
GP-03 21· 758
GP-04 21· 758
GP-05 21· 758
GP-06 21· 759
GP-07 21· 759

GPI

GPI-01 21· 784

HCV

HCV-01 23· 859
HCV-02 23· 859
HCV-03 23· 859
HCV-04 23· 859
HCV-05 23· 859

IAL

IAL-1 18· 683
IAL-2 18· 683
IAL-3 18· 683

IES

IES-01 15· 570
IES-02 15· 570
IES-03 15· 570
IES-04 15· 570
IES-05 15· 570
IES-06 15· 570
IES-07 15· 570
IES-08 15· 570
IES-09 15· 570
IES-10 15· 570

INV

INV-001 29·1073
INV-002 29·1073
INV-003 29·1073
INV-004 29·1073
INV-005 29·1073
INV-006 29·1073
INV-007 29·1073
INV-008 29·1073

INV-A

INV-A-01 15· 567
INV-A-02 15· 567
INV-A-03 15· 567
INV-A-04 15· 567
INV-A-05 15· 567
INV-A-06 15· 568
INV-A-07 15· 568

INV-AUD

INV-AUD-01 08· 329 16· 597
INV-AUD-02 08· 329
INV-AUD-03 08· 329 16· 598

INV-BND

INV-BND-01 16· 598

INV-BOUND

INV-BOUND-01 08· 330
INV-BOUND-02 08· 330

INV-C

INV-C-01 15· 566
INV-C-02 15· 566
INV-C-03 15· 566
INV-C-04 15· 566
INV-C-05 15· 566
INV-C-06 15· 567

INV-CON

INV-CON-01 08· 329
INV-CON-02 08· 329

INV-CRY

INV-CRY-01 16· 597

INV-CRYPT

INV-CRYPT-01 08· 329
INV-CRYPT-02 08· 329

INV-CTL

INV-CTL-01 16· 597
INV-CTL-02 16· 597

INV-D

INV-D-01 15· 569
INV-D-02 15· 569
INV-D-03 15· 569
INV-D-04 15· 569
INV-D-05 15· 569
INV-D-06 15· 569

INV-DAT

INV-DAT-01 16· 598
INV-DAT-03 16· 598

INV-DEC

INV-DEC-01 08· 328
INV-DEC-02 08· 328 16· 597
INV-DEC-03 08· 329

INV-ENF

INV-ENF-01 08· 328
INV-ENF-02 08· 328
INV-ENF-03 08· 328

INV-INTEROP

INV-INTEROP-000 18· 653
INV-INTEROP-011 18· 656
INV-INTEROP-012 18· 656
INV-INTEROP-013 18· 657
INV-INTEROP-014 18· 657
INV-INTEROP-034 18· 663
INV-INTEROP-041 18· 665
INV-INTEROP-042 18· 666
INV-INTEROP-043 18· 666
INV-INTEROP-051 18· 667
INV-INTEROP-052 18· 667
INV-INTEROP-053 18· 668
INV-INTEROP-054 18· 668
INV-INTEROP-061 18· 669
INV-INTEROP-062 18· 670
INV-INTEROP-063 18· 670
INV-INTEROP-071 18· 671
INV-INTEROP-072 18· 672
INV-INTEROP-073 18· 672
INV-INTEROP-074 18· 673
INV-INTEROP-075 18· 673
INV-INTEROP-081 18· 674
INV-INTEROP-082 18· 675
INV-INTEROP-083 18· 675
INV-INTEROP-091 18· 677
INV-INTEROP-106 18· 679
INV-INTEROP-113 18· 684

INV-S

INV-S-01 15· 568
INV-S-02 15· 568
INV-S-03 15· 568
INV-S-04 15· 568
INV-S-05 15· 569

LC

LC-01 23· 876
LC-02 23· 876
LC-03 23· 877
LC-04 23· 877
LC-05 23· 877
LC-06 23· 877
LC-07 23· 877
LC-08 23· 878
LC-09 23· 878

OUT

OUT-01 04· 150
OUT-02 04· 150
OUT-03 04· 150
OUT-04 04· 150
OUT-05 04· 150
OUT-06 04· 150

PBR

PBR-01 28·1030

PL

PL-01 28·1030
PL-02 28·1030
PL-03 28·1030

QCC

QCC-01 03· 85
QCC-01.1 12· 493
QCC-01.2 12· 493
QCC-01.3 12· 493
QCC-02 03· 85
QCC-02.1 12· 493
QCC-03 03· 83
QCC-03.1 12· 493
QCC-03.2 12· 494
QCC-03.3 12· 494
QCC-03.4 12· 494
QCC-03.5 12· 494
QCC-03.6 12· 494
QCC-04 03· 87
QCC-04.1 12· 494
QCC-04.2 12· 494
QCC-04.3 12· 495
QCC-05 03· 88
QCC-05.1 12· 495
QCC-05.2 12· 495
QCC-06 03· 88
QCC-06.1 12· 495
QCC-06.2 12· 495
QCC-06.3 12· 495
QCC-06.4 12· 495
QCC-07 03· 88
QCC-07.1 12· 496
QCC-07.2 12· 496
QCC-07.3 12· 496
QCC-08 03· 88
QCC-08.1 12· 496
QCC-08.2 12· 496
QCC-08.3 12· 496
QCC-09 03· 85
QCC-09.1 12· 497
QCC-09.2 12· 497
QCC-10 03· 85
QCC-10.1 12· 497
QCC-10.2 12· 497
QCC-11 03· 88
QCC-11.1 12· 497
QCC-12 03· 88
QCC-12.1 12· 497

QTS

QTS-01-001 07· 270
QTS-01-002 07· 270
QTS-01-003 07· 270
QTS-01-004 07· 271
QTS-01-005 07· 271
QTS-01-006 07· 271
QTS-01-007 07· 271
QTS-02-001 07· 272
QTS-02-002 07· 272
QTS-02-003 07· 272
QTS-02-004 07· 272
QTS-02-005 07· 273
QTS-02-006 07· 273
QTS-03-001 07· 273
QTS-03-002 07· 273
QTS-03-003 07· 274
QTS-03-004 07· 274
QTS-03-005 07· 274
QTS-04-001 07· 274
QTS-04-002 07· 275
QTS-04-003 07· 275
QTS-04-004 07· 275
QTS-04-005 07· 275
QTS-05-001 07· 276
QTS-05-002 07· 276
QTS-05-003 07· 276
QTS-05-004 07· 276
QTS-05-005 07· 277
QTS-05-006 07· 277
QTS-05-007 07· 277
QTS-05-008 07· 277
QTS-06-001 07· 278
QTS-06-002 07· 278
QTS-06-003 07· 278
QTS-06-004 07· 278
QTS-06-005 07· 279
QTS-06-006 07· 279
QTS-07-001 07· 279
QTS-07-002 07· 279
QTS-07-003 07· 280
QTS-07-004 07· 280
QTS-07-005 07· 280
QTS-08-001 07· 280
QTS-08-002 07· 281
QTS-08-003 07· 281
QTS-08-004 07· 281
QTS-08-005 07· 281
QTS-08-006 07· 281
QTS-08-007 07· 282
QTS-09-001 07· 282
QTS-09-002 07· 282
QTS-09-003 07· 282
QTS-09-004 07· 283
QTS-09-005 07· 283
QTS-09-006 07· 283
QTS-10-001 07· 283
QTS-10-002 07· 284
QTS-10-003 07· 284
QTS-10-004 07· 284
QTS-10-005 07· 284
QTS-11-001 07· 285
QTS-11-002 07· 285
QTS-11-003 07· 285
QTS-11-004 07· 285
QTS-11-005 07· 286
QTS-11-006 07· 286
QTS-12-001 07· 286
QTS-12-002 07· 286
QTS-12-003 07· 287
QTS-12-004 07· 287
QTS-12-005 07· 287

QTS-ADV

QTS-ADV-001 07· 294
QTS-ADV-002 07· 295
QTS-ADV-003 07· 295
QTS-ADV-004 07· 295
QTS-ADV-005 07· 295
QTS-ADV-006 07· 295
QTS-ADV-007 07· 295
QTS-ADV-008 07· 296
QTS-ADV-009 07· 296
QTS-ADV-010 07· 296

QTS-AI

QTS-AI-001 07· 292
QTS-AI-002 07· 292
QTS-AI-003 07· 293
QTS-AI-004 07· 293
QTS-AI-005 07· 293
QTS-AI-006 07· 293
QTS-AI-007 07· 293

QTS-AL

QTS-AL-001 07· 294
QTS-AL-002 07· 294
QTS-AL-003 07· 294
QTS-AL-004 07· 294
QTS-AL-005 07· 294
QTS-AL-006 07· 294

QTS-CA

QTS-CA-001 07· 300
QTS-CA-002 07· 300
QTS-CA-003 07· 300
QTS-CA-004 07· 300
QTS-CA-005 07· 301
QTS-CA-006 07· 301

QTS-FC

QTS-FC-001 07· 290
QTS-FC-002 07· 290
QTS-FC-003 07· 290
QTS-FC-004 07· 290
QTS-FC-005 07· 290
QTS-FC-006 07· 290
QTS-FC-007 07· 290
QTS-FC-008 07· 290
QTS-FC-009 07· 290
QTS-FC-010 07· 290

QTS-RC
QTS-RC-001 07· 291
QTS-RC-002 07· 291
QTS-RC-003 07· 291
QTS-RC-004 07· 291
QTS-RC-005 07· 291
QTS-RC-006 07· 291

QTS-RT
QTS-RT-001 07· 301
QTS-RT-002 07· 301
QTS-RT-003 07· 301
QTS-RT-004 07· 302
QTS-RT-005 07· 302
QTS-RT-006 07· 302

QTS-SER
QTS-SER-001 07· 311
QTS-SER-002 07· 311
QTS-SER-003 07· 311
QTS-SER-004 07· 311
QTS-SER-005 07· 311
QTS-SER-006 07· 311

R
R-01 03· 87
R-02 03· 87
R-03 03· 87
R-04 03· 87

REV
REV-01 28·1031
REV-02 28·1031
REV-03 28·1032

ROLE
ROLE-01 21· 771
ROLE-02 21· 771
ROLE-03 21· 771
ROLE-04 21· 771
ROLE-05 21· 772

RT
RT-01 22· 810
RT-02 22· 810
RT-03 22· 810
RT-04 22· 810
RT-05 22· 810
RT-06 22· 810
RT-07 22· 810
RT-08 22· 810
RT-09 22· 810
RT-10 22· 811
RT-11 22· 811
RT-12 22· 811

SEC-ALG
SEC-ALG-01 06· 212 28·1033
SEC-ALG-02 28·1033
SEC-ALG-03 28·1033

SEC-ALG-AGILITY
SEC-ALG-AGILITY-01 06· 212 28·1033

SEC-ART
SEC-ART-01 06· 213
SEC-ART-02 06· 213

SEC-AUDIT
SEC-AUDIT-01 06· 213

SEC-CLK
SEC-CLK-01 06· 213

SEC-DET
SEC-DET-01 06· 213

SEC-INV
SEC-INV-01 06· 209

SEC-KEY
SEC-KEY-01 06· 212 28·1033
SEC-KEY-02 06· 212 28·1033
SEC-KEY-03 06· 212 28·1033

SEC-NET
SEC-NET-01 06· 213 28·1033
SEC-NET-02 06· 240

SEC-REG
SEC-REG-01 06· 212
SEC-REG-02 06· 213

SEC-SCOPE
SEC-SCOPE-01 06· 213

SEC-TOK
SEC-TOK-01 06· 212

SR
SR-01 23· 886
SR-02 23· 886
SR-03 23· 886
SR-04 23· 886
SR-05 23· 887
SR-06 23· 887

T
T-01 03· 83
T-02 03· 83
T-03 03· 83
T-04 03· 83
T-05 03· 83
T-06 03· 83
T-07 03· 83
T-08 03· 83

TC
TC-01 10· 396
TC-02 10· 396
TC-03 10· 396
TC-04 10· 396
TC-05 10· 396
TC-06 10· 396
TC-07 10· 396
TC-08 10· 396

THR
THR-01 06· 206
THR-02 06· 206
THR-03 06· 206
THR-04 06· 206
THR-05 06· 206
THR-06 06· 206
THR-07 06· 207
THR-08 06· 207
THR-09 06· 207
THR-10 06· 207

TV	
TV-01	17· 637
TV-02	17· 638
TV-03	17· 638
TV-04	17· 638
TV-05	17· 638
TV-06	17· 638

TZ	
TZ-1	17· 618
TZ-2	17· 619
TZ-3	17· 619
TZ-4	17· 619
TZ-5	17· 620
TZ-6	17· 620
TZ-7	17· 621
TZ-8	17· 621
TZ-9	17· 622

W	
W-01	05· 169

APPENDIX C

Hash manifest

SHA-256 of each of the 32 source HTML files, recomputed from the files at build time (all match the corpus inventory).
Source: https://github.com/qodiqa/qodiqa at commit `d3b5066a29b1a32f20d30301311dcbd4e30d2ed9`, directory `docs/`.

CH.	DOCUMENT IDENTIFIER	SOURCE FILE	SHA-256
01	QODIQA-IDX-2026-001	QODIQA___Master_Index_and_Readers_Guide.html	dd37d13ad5edb775cd6f6df78dcb313a63596fd44414bd6a916ce862a286b3c7
02	QODIQA-WP-2026-001	QODIQA___Consent_as_Infrastructure_for_Artificial_Intelligence_Technical_Whitepaper.html	faf53ae4eb94fb8f7654c2e1984e6eaf3530b5ef7612b816bb6e4d8dc562c5d0
03	QODIQA-CORE-2026-001	QODIQA___Core_Standard_for_Deterministic_Runtime_Consent_Enforcement.html	545f26f59a377a431ff0bea58865e58dddce015d5adfce5f5e95a75556a36499
04	QODIQA-FRAMEWORK-2026-003	QODIQA___68-Point_Enforcement_Framework_for_Deterministic_Runtime_Consent_Enforcement.html	2d200d1daf0dc8db2020e348295618803e9933154579f6ed963f10fb15e782b0
05	QODIQA-ARCH-2026-001	QODIQA___Reference_Architecture_for_Deterministic_Runtime_Consent_Enforcement.html	f1d66af635284d2eabb7cd0627f7745fdd2d0938ec2374549e454d282af53768
06	QODIQA-SEC-2026-001	QODIQA___Security_and_Cryptographic_Profile_for_Runtime_Consent_Enforcement.html	099f2e37a2074953891f7dc61e9322f20c34439b147de1d7375bb20cdef10efd
07	QODIQA-CTSS-2026-001	QODIQA___Conformance_Test_Suite_Specification_for_Deterministic_Runtime_Consent_Enforcement.html	705de243d2c0ac6e1ba8784d1e9742d3ecd25a19c17429823628ecb50076b531
08	QODIQA-CVS-2026-001	QODIQA___Conformance_Verification_Specification.html	464066d1318f0780c86a4233ca87df5930b4333f071f49f3c5f771afda8177e7
09	QODIQA-SEC-TM-2026-001	QODIQA___Threat_Model_and_Abuse_Case_Specification.html	947edcab28174f43be7be5abc49acb7ea0a48b34feab4edab6e1d367e1ff1e80
10	QODIQA-TM-2026-001	QODIQA___Extended_Adversarial_Threat_Model.html	3ec9e4d99ec12f62a24fbe69b8e6e3245323e3ca7cbc78ccae1437996e757c12
11	QODIQA-GEI-2026-001	QODIQA___Global_Enforcement_Invariants_for_Deterministic_Runtime_Consent_Enforcement.html	04a216451826d970aa48362873ae3012f5a3fe84a64d86b6dd3a3ffb42cf4a6f
12	QODIQA-IMPL-2026-001	QODIQA___Implementation_Playbook_for_Deterministic_Runtime_Consent_Enforcement.html	48362fa3e69995c69154714a48371a6e93e78f499bbf705b3fd900cbdcea7a05

CH.	DOCUMENT IDENTIFIER	SOURCE FILE	SHA-256
13	QODIQA-CL-2026-001	QODIQA___Implementation_Conformance_Checklist.html	7e973251d0be70a77001314af0df8d012b07a9c79842a83f6bec6d0fa6984901
14	QODIQA-RIS-2026-001	QODIQA___Reference_Implementation_Specification _-_Minimal_Deterministic_Enforcement_Stack.html	a41562947831ef7853845001c7d3c80f52cb95c2239664cef8d201bd7587bc94
15	QODIQA-FEM-2026-001	QODIQA___Failure_Handling_and_Recovery_Specification.html	4cb6b53b1111d04f4605461876556945516 4a06c6899c69d755649a0e03e9f11
16	QODIQA-FCM-2026-001	QODIQA___Non-Compliance_Conditions_and_Failure_Modes.html	672e9536bcd8e6e12c86413a18505b1757ce24bf3d97b7ddad4e0f40854a24d9
17	QODIQA-SBTM-2026-001	QODIQA___System_Boundary_and_Trust_Model_Specification.html	1fe33897687c72013a3a4028e4f3e67721f439fe485af927a02419cf3c807b0c
18	QODIQA-INTEROP-CONS-2026-001	QODIQA___Interoperability_and_Deployment_Constraints.html	5e6742d976a2729d1b257a9374d740e3306ff1b7b798687d53673a038daa819e
19	QODIQA-TND-2026-001	QODIQA___Terminology_and_Normative_Definitions.html	7d29bce382eb59e06c90eb069c168b84afd2df2409e169ba251d762147259378
20	QODIQA-WE-2026-001	QODIQA___Worked_Example_End_to_End_Scenario.html	688d47da8b05d2243d13ba5aeef0437850938191d6be5f96a9de3fb81f0afa32
21	QODIQA-GOV-CH-2026-001	QODIQA___Governance_Charter_for_the_QODIQA_Standard_Corpus.html	e65a1ae40e12a772dfb45f97ef91ecd204861c4a405369455519fd6b001aee9a
22	QODIQA-RISK-2026-001	QODIQA___Residual_Risk_and_Assumption_Disclosure_Annex.html	6379f29f68e5cbab71a8ad2a09414d041dab27a4c6a276c4fb9b1e59ce890317
23	QODIQA-CERT-2026-001	QODIQA___Certification_Framework_for_Deterministic_Runtime_Consent_Enforcement.html	235af389957bb2f9a1c07d17e620096868cdf7fc562799434311bc4b72f87e6d
24	QODIQA-PSLI-2026-001	QODIQA___Public_Specification_License.html	4c1ea2e8b7fae3b7d05997a19eb10512200d72aa48bfdc9af2309459071dffd3
25	QODIQA-PSL-2026-001	QODIQA___Positioning_and_Scope_Limitation_Statement.html	9ccd1755939936b6fd3edb33691c3d86c4cee62219ccd253ddc866b3506ff1eb
26	QODIQA-EB-2026-001	QODIQA___Executive_Brief_for_Deterministic_Runtime_Consent_Enforcement.html	7a6927677ea9e34f3d4802d0be46d17c8cc375e18de245e758f4a075969ad0ba
27	QODIQA-EIP-2026-001	QODIQA___Economic_Impact_Analysis_of_Deterministic_Runtime_Consent_Enforcement.html	1b4e33036248e3d19388e9ced1babff50852ce04814942ba7366470671fcdbd8
28	QODIQA-RAM-2026-001	QODIQA___Regulatory_Alignment_Matrix_for_Deterministic_Runtime_Consent_Enforcement.html	92667ecc9b8458f4686452d8066e0b7ff73c3980bc8e96c109d6679dfa12d51b
29	QODIQA-AUDIT-2026-001	QODIQA___Audit_and_Evidence_Generation_Model.html	7a5372f64f4da2c50433f6b8c882f4ef067eead46ef72232c66695ee0a6a67f7
30	QODIQA-ARP-2026-001	QODIQA___Audit_Readiness_and_Evidence_Pack.html	f7c1c227cd0c37e3f12f6526a1960974e6134a62f520442dfea316b66889f9c9
31	QODIQA-UCD-2026-001	QODIQA___Use_Case_Dossiers_for_Runtime_Consent_Enforcement_Deployments.html	47456ed97b279eebbb5f7b4550b422b045a1d67bfb06134bcdcfe37a9a1ab631
32	QODIQA-CCL-2026-001	QODIQA___Corpus_Change_Log_and_Version_History.html	e2bb40d8577f15561607fe7d9ceb10e4b7eb479e5dbb0e3ff2fc55542d092cbc

QODIQA

Consent, enforced at execution.

WWW.QODIQA.COM · CONTACT@QODIQA.COM