Institutional Control Architecture

A Standard for Autonomous AI Governance

Five Foundational Laws. Three Non-Negotiable Constraints. Eleven Constitutional Articles. Seven Control Layers. Seven Failure Patterns. A five-phase conformity assessment. Verifiable certification.

AUTHOR
Shane Schreck

PUBLISHED
May 2026

CANONICAL URL
https://publications.aleeth.com/standard/

Publication Record

Publication	Standard
Published	May 2026
Author	Shane Schreck
Canonical URL	https://publications.aleeth.com/standard/

Shane Schreck

Founder . ALEETH

Executive Summary

Autonomous AI is moving from policy debate into procurement, regulation, and capital allocation. The layer the policy depends on to convert intent into operational reality does not yet exist. This paper introduces Institutional Control Architecture as that layer.

The framework applies to the full surface where autonomous decision-making is now being deployed: sovereign and self-hosted AI, hybrid architectures, agentic orchestration, regulated AI in financial and healthcare environments, edge AI on operator-controlled hardware, and military AI operations under doctrinal command. The unifying condition is the same across each: a system takes consequential action on its own judgment, and the parties depending on that system require evidentiary proof that the system is bounded, observable, and accountable. The McKinsey-cited sovereign-AI market sized at $500 to $600 billion by 2030 [1] is one subset of that surface, not the whole of it.

Across each of those categories the bottleneck is the same and is consistently under-rehearsed. Initiatives stall at the boundary between policy intent and operational execution. The layer they stall on is trust. There is no audit-grade mechanism today by which the assertions an autonomous system makes about itself can be examined, certified, and re-verified by parties that have no reason to trust the vendor.

Institutional Control Architecture supplies that mechanism. It is a framework, an assessment methodology, and a certification regime designed to give regulators, executives, customers, and counterparties one consistent answer to one consistent question: is this autonomous system observable, bounded, and independently verifiable under governance constraints . The framework defines Five Foundational Laws that state the doctrinal premises, Three Non-Negotiable Constraints that establish the operating rules, Eleven Constitutional Articles enforced at the runtime boundary, and Seven Control Layers any autonomous system must establish to be certifiable. It identifies the recurring patterns by which such systems fail in production. It produces certifications whose validity does not depend on the issuing platform remaining present, cooperative, or solvent.

ICA is the doctrinal application of Aleeth, an operational philosophy whose core premise is that trustworthy behavior in any institutional system is established not by stated intentions but by structural enforcement. The behavioral promise of an operator, a vendor, or a model is treated as evidence requiring verification, not as a

substitute for it. The Seven Control Layers, the failure-pattern catalogue, and the audit-grade certification artifact described in this paper are the operational consequences of that premise applied to autonomous systems.

This paper describes the framework, the assessment process, the verification properties of the certification artifact, and the relationship of ICA to neighboring standards. It is intended for executives evaluating where to site authoritative governance for autonomous systems in their organizations, for policy and regulatory readers tracking the maturation of the AI assurance ecosystem, and for technologists evaluating which assurance regime will satisfy their counterparties two and five years from now. It does not provide regulatory advice. It describes a working framework and the conditions under which a reader should engage further.

The trust layer for autonomous AI will be built. ICA is built to claim that role on terms a serious auditor, regulator, or board can defend.

> *Compute can be bought. Trust has to be earned, proven, and verified. Aleeth*

Definitions

The terms below are used throughout the paper. Where a term has a controlling definition in a published standard, that source is named.

Autonomous system. A computational system that takes consequential action on its own judgment, with or without a human in the immediate loop. For the purposes of this paper, the term covers AI agents, large-language-model applications acting under tool-use authority, machine-learning systems that issue decisions in production, and the orchestration layers that combine them.

Autonomous AI. The deployment category in which one or more autonomous systems are placed in production under conditions where their operational behavior is consequential to the deploying party. ICA applies across the deployment surface of autonomous AI without exception: sovereign and self-hosted environments, hybrid architectures, agentic orchestration, regulated AI in financial and healthcare contexts, edge AI on operator-controlled hardware, and military AI under doctrinal command. Differences in deployment topology do not alter the framework's applicability; they alter the scoping of the assessment.

Self-Hosted AI. A subset of autonomous AI deployments characterized by territorial location, operational management, technological ownership, and legal jurisdiction sufficiently aligned with the sovereignty interests of the deploying entity. Sovereignty is a spectrum, not a binary, and is evaluated workload by workload. The McKinsey-cited sovereign-AI market sizing [1] refers specifically to this subset.

Control. Within ICA, the word control is a contracted reference to a precise condition: observable bounded behavior under independently verifiable governance constraints . The expanded definition is the operative one. An autonomous system is under control when its behavior is (a) observable in real time and recoverable after the fact, (b) bounded by enforceable policy at every consequential decision boundary, and (c) verifiable by parties independent of the system's operator. The shorter term is used throughout this paper for readability, but the longer definition governs interpretation.

Control Layer. A category of control that an autonomous system must establish in order to be certifiable under ICA. The framework defines seven such layers, derived inductively from observed failure modes in autonomous-system deployments. The layers are necessary conditions for certification: a system that fails any one

layer is not certifiable. They are described in section 5.

Failure pattern. A recurring mode by which autonomous systems and the agents that operate them break in production. ICA identifies seven such patterns. They are described in section 6.

Audit-grade certification. A certification whose validity can be independently verified by any third party without trusting the issuer's word, the issuing platform's continued operation, or the certified party's continued cooperation. The defining property is independent verifiability.

Independence Principle. The audit-discipline rule that the person who drafts a certification decision and the person who signs off on it must be distinct natural persons. ICA enforces this rule structurally, not as a behavioral norm.

Structural enforcement. A design discipline in which the rules a system is required to obey are implemented in the system's persistent layer rather than as workflow norms or interface conventions. A structurally-enforced rule cannot be bypassed by a determined operator. The distinction matters because behavioral trust is unverifiable in autonomous-system environments; structural enforcement is observable and bounded.

Conformity assessment. The standards-body term for the process by which a candidate system or organization is evaluated against a specified normative document. Refer to ISO/IEC 17000 for the general vocabulary applicable across conformity-assessment regimes.

Verifiable trust. The property that an assertion a system makes about itself can be examined, certified, and re-checked by independent parties using only the assertion, the certification, and a published verification key. The opposite of declarative trust, in which the asserter expects to be believed.

IMPERIUM. The mothership. Aleeth's internal aggregating view, where every ICA Live instance across the certified network feeds upward into the operator-of-record's continuous oversight. IMPERIUM is not customer-facing; it is the surface from which the standards body sees the whole network. References to "the originator's infrastructure" in this paper resolve to IMPERIUM.

ICA Live. The per-certified-organization operator platform that runs the engagement lifecycle (Pre-Engagement, Scope, Assess, Certify, Sustain) end-to-end against the framework. Each certified organization operates its own ICA Live instance. ICA Live is the customer's working surface during certification and the sustainment record after.

The Registry. The public verification surface. A counterparty consults the Registry to confirm a certification's existence, current state, and verifiable signature without trusting the issuing platform's continued operation. The Registry is the only one of the three components that is anonymously readable by design.

The Problem

Autonomous-AI policy has matured rapidly. Operational reality is slower. The gap between the two is the gap ICA exists to close.

The European Union has passed the AI Act. The United States National Institute of Standards and Technology has published the AI Risk Management Framework. The International Organization for Standardization has released ISO/IEC 42001 for AI management systems. Sovereign-cloud offerings have been announced by every

major hyperscaler. National-model investments have been declared by jurisdictions on every continent.

And yet the McKinsey analysis underpinning current market sizing (McKinsey uses the term sovereign AI for what this paper terms self-hosted AI ; the conditions are the same) is unequivocal that these initiatives are stalling and failing to deliver expected results. [1] The gap is not in policy. The gap is not in compute. The gap is in the assurance layer that converts policy intent into operational reality.

Three patterns recur across stalled initiatives.

First, sovereign infrastructure announcements are decoupled from certification mechanisms. A jurisdiction announces a national model. A hyperscaler announces a sovereign cloud zone. A regulated industry announces a sector consortium. None of these announcements answer the question a Chief Information Security Officer must answer to deploy: how does this organization prove, to its board and to its regulator, that the system is actually under control. The infrastructure exists. The certification regime that would make the infrastructure trustworthy does not.

Second, enterprise migration timelines are bounded by governance work, not by technology work. Autonomous-AI deployments (sovereign, hybrid, regulated, edge, or operationally-scoped) consistently take three to four years to certify under existing regimes. They take that long not because the technology is immature but because the organizational work of classifying workloads, establishing control points, and operationalizing audit is unfamiliar. Most enterprises do not know which of their workloads require which kind of governance posture. They treat the question as binary, and they stall.

Third, the assertions vendors make about their systems are unverifiable. A vendor that claims its model "has guardrails" or that its system "follows responsible-AI principles" is making an assertion that cannot be independently re-verified by a counterparty. When a regulator, a board, or a customer asks for proof, the chain of custody between the assertion and the actual operating reality of the system is missing. Soft assurances are increasingly inadequate against contractual, regulatory, and reputational scrutiny.

Markets that mature past the stalling pattern share one structural feature. They develop an authoritative third-party trust layer. SOC 2 did this for cloud security. ISO/IEC 27001 did it for information-security management. PCI-DSS did it for payment processing. In each case the pattern is the same. A canonical framework defines what must be controlled. A canonical procedure determines whether a candidate system meets the framework. A canonical artifact records the certification, in a form any third party can verify without trusting the issuer. A canonical penalty enforces against falsification of the artifact.

No equivalent regime exists today for autonomous systems. ISO/IEC 42001 establishes management-system requirements for organizations developing or using AI; it certifies the management system rather than producing per-system audit-grade assertions, and the resulting certification is no more verifiable than its issuing audit firm. NIST's AI Risk Management Framework provides risk-management guidance but is not itself a certification regime. SOC 2 was not designed for the failure modes autonomous systems exhibit and has no analytical apparatus calibrated to them. The European AI Act classifies systems and imposes obligations but leaves the conformity-assessment ecosystem to develop. None of these regimes produces today a certification artifact that any third party can independently verify, against the current state of the certified system, without trusting the platform that issued it.

That gap is the gap ICA closes. The next sections describe the framework, the assessment methodology, and the verification properties of the certification artifact ICA produces.

Why Traditional Governance Frameworks Fail Under Autonomous Conditions

The case for ICA is not that autonomous systems require governance. Autonomous systems obviously require governance. The case is that the governance frameworks the world already has were built for a different class of system, and they fail in three specific places when applied to autonomous AI.

Every mature governance regime (financial audit, software-quality certification, information-security management, regulatory conformity assessment) was designed around three foundational assumptions about the systems being governed. Those assumptions held under the operating conditions the frameworks were drafted for. They fail simultaneously when the system being governed is autonomous.

Assumption one: software behavior is static between releases. A SOC 2 auditor, an ISO 27001 assessor, or a regulator performing conformity assessment is implicitly evaluating a system whose behavior is fixed between deployment events. A control set is established, the system is observed under that control set, and the certification holds until the next deployment. Autonomous systems violate this assumption by design. A language-model-driven system's behavior is not fixed between releases; it shifts continuously as inputs vary, as tool authorities evolve, as the orchestration layer permits new combinations. A certification predicated on static behavior cannot survive the first novel input the production system encounters.

Assumption two: the operational boundary of the system is stable. Traditional governance assumes that the perimeter of the certified system is known and persistent. The audit applies to that perimeter. The certified party is responsible for what crosses it. Autonomous systems with tool-use authority routinely act outside their original perimeter. They invoke external services, they retrieve external data, they spawn subordinate processes, and in extreme cases they replicate themselves to infrastructure the certified party does not own. ■ A certification predicated on a stable perimeter cannot enforce against a system that redefines its own perimeter at runtime.

Assumption three: the operator's narrative about the system is truthful. Every governance regime treats statements made by the system's operator as evidence. The auditor asks; the operator answers; the auditor verifies a sample; the certification proceeds. This assumption is workable when the operator is a natural person or an organization composed of natural persons subject to perjury statutes, regulatory enforcement, and reputational consequence. Autonomous systems are increasingly operators in their own right. They generate the documentation, populate the dashboards, draft the incident reports, and answer the auditor's questions. The narrative the auditor receives is now produced by the same class of entity the auditor is supposed to be evaluating. The assumption of a truthful, independently-accountable narrator silently collapses.

Each individual assumption can be patched. A regime can move to continuous certification to handle drift. A regime can re-scope its perimeter doctrine to include tool authority. A regime can require human attestation on every operator-produced artifact. The problem is that autonomous AI systems violate all three assumptions simultaneously , in production, at every interaction. Patching them one at a time produces a governance instrument that always lags the deployment surface. The framework being applied was never load-bearing under these conditions.

The opening ICA addresses is not a refinement of existing governance. It is a framework designed from the assumption set autonomous systems actually exhibit: behavior that varies continuously, perimeter that the system itself can redefine, narrative that the certifier cannot accept on the operator's word. Every primitive in the sections that follow (the seven Control Layers, the failure-pattern catalogue, the structurally-enforced Independence Principle, the cryptographically-verifiable certification artifact, the audit-grade evidentiary record) exists because one or more of those three assumptions has failed for the deployment being certified.

The Five Foundational Laws

The Laws state the doctrinal premises from which the rest of the framework follows. They are not aspirations. They are the conditions under which any subsequent claim the framework makes is coherent.

Law I . The Foundational Law. Control must scale with capability under operational load. A control regime that holds at design time but fails to scale with the capability it governs is not a control regime; it is a snapshot. ICA treats capability and control as paired quantities, and treats any divergence between them as a structural defect.

Law II . The Zero Law. No system operates without a control layer. The absence of architecture is itself an architecture. A system deployed without explicit governance is not ungoverned; it is governed by whatever defaults its surrounding infrastructure imposes. The Zero Law refuses the framing that absence of architecture is neutrality. It is a choice, and the choice is recorded.

Law III . The Structural Sequence. Traceability before containment. Containment before reversibility. Reversibility before scale. The order is operational, not rhetorical. A system that cannot trace what it has done cannot meaningfully contain what it can do, cannot reverse what it has done, and cannot be scaled without compounding what it has not bounded. The Three Non-Negotiable Constraints described in the next section follow this sequence.

Law IV . The Dashboard Law. Dashboards inform. Attestations bind. Observability surfaces report state; they do not enforce it. Governance enforcement is established by attestation, in which a named accountable party signs a record asserting the state of the system at a defined point in time. Dashboards are necessary and insufficient; attestation is the load-bearing instrument.

Law V . The Exposure Law. When velocity exceeds control, exposure becomes inevitable. An organization that deploys autonomous capability faster than it builds governance for that capability is not running ahead of risk; it is accumulating undisclosed liability. The Exposure Law is the framework's response to the prevailing operating tempo of autonomous-AI deployment, and the reason the framework treats certification as a precondition for scale rather than a downstream artifact of it.

The Three Non-Negotiable Constraints

The Constraints are the operating rules. They are not principles to aspire to and not goals to pursue. They are binary conditions every certified autonomous system must satisfy. A system that fails any one of the three is not certifiable.

Traceability. Every autonomous action can be reconstructed end-to-end. Trigger, execution, outcome, all logged, all auditable. A system in which an action cannot be reconstructed from a durable record fails the constraint. The

standard does not accept reconstruction-on-best-effort; it accepts reconstruction-on-demand, against a tamper-resistant log, by parties independent of the operator.

Containment. No agent exceeds its explicit operational boundary. Capability is bounded by design. If no control surface exists for a given capability, the capability does not deploy. The constraint refuses the operating pattern in which a system's perimeter is defined retrospectively, after an incident, by reference to what the system was observed to have done.

Reversibility. Every autonomous action has a defined and tested reversal mechanism. If the rollback path is unnamed, the action is not permitted. The constraint applies to actions in their normal mode and to actions taken under failure conditions. Untested rollback paths are unevidenced rollback paths and are not credited against the constraint.

The Constraints are not redundant with the Seven Control Layers described later in this paper. The Layers are categories of control any autonomous system must establish in order to be certifiable. The Constraints are the binary conditions any certified system must continuously satisfy across all Layers. A system can be working toward certification at the Layer level and still be failing one of the Constraints; the framework treats such a system as non-certifiable until the Constraint is satisfied.

The Eleven Constitutional Articles

The Articles are runtime constitutional principles enforced at the boundary of every certified system. Where the Laws establish doctrine and the Constraints establish operating rules, the Articles establish the principled obligations under which an autonomous system is permitted to act at all. Each Article is assigned a layer in which it is principally enforced.

Article I . Integrity. (Identity layer.) No system shall produce, promote, or permit an action that contradicts the operator's declared values, stated goals, or governing constitution.

Article II . Veracity. (Cognitive layer.) No claim, recommendation, or decision may be surfaced without its source, confidence level, and reasoning chain. Uncertainty must be named, not hidden.

Article III . Non-Delegation. (Identity layer.) Authority over identity-level decisions cannot be delegated to autonomous systems. The operator remains the principal.

Article IV . Audit. (Operational layer.) Every action writes an append-only audit entry. An action that cannot be audited cannot be shipped.

Article V . Reversibility. (Operational layer.) Every irreversible action requires an authorization token and a declared rollback path. If the rollback is unnamed, the action is not permitted.

Article VI . Proportional Consent. (Relational layer.) The level of consent required for an action is proportional to its blast radius. Higher impact requires higher consent.

Article VII . Financial Sovereignty. (Financial layer.) No transaction executes without authorization, named purpose, and an accountable agent. Three artifacts required, every time.

Article VIII . Physical Integrity. (Physical layer.) No system may issue a recommendation that compromises the operator's physical readiness, recovery state, or long-term health.

Article IX . Relational Trust. (Relational layer.) Outbound communication carries authorship attribution. Impersonation is blocked at the send queue and logged as a constitutional violation.

Article X . Legacy. (Legacy layer.) Every agent persists its final state, decisions log, and artifacts to durable storage before retirement. State persistence is checked at shutdown.

Article XI . Self-Governance. (Operational layer.) The system itself runs under the same constitution it enforces. Aleeth is the first ICA-certified deployment of ICA.

THE ATTESTATION PRINCIPLE

A control standard is not real until leadership is willing to attest to it. If a named accountable party cannot sign their name to the state of a system, that party does not have control of the system. The Articles are enforceable because they are attestable; the framework treats attestation as the load-bearing instrument that converts a stated principle into an enforceable one.

The Framework: Seven Control Layers

The Seven Control Layers are the spine of ICA. Each layer describes a category of control any autonomous system must establish in order to be certifiable.

The layers were derived inductively. When autonomous systems fail in production, the failures cluster across exactly these seven categories. The framework treats the layers as necessary conditions for certification: a system that fails on any layer is not certifiable. The framework holds the sufficiency claim provisionally. To date, the failures observed across enterprise, consumer, and operator-tooling deployments have clustered within the seven. Where a system is observed to fail outside the seven, the framework expects to be amended.

Each layer is described below in three parts: a definition, a statement of what the layer covers, and a list of the conceptual sub-controls the assessment evaluates against. The assessment process itself is described in section 7.

LAYER ONE: PROBLEM

The system can articulate the problem it is solving and what is out of scope. Reproducible problem statement. Documented exclusions. Alignment between the stated problem and the system's stated objective. Where the system is asked to do work outside its stated scope, the system declines or escalates. The layer fails when the system cannot describe what it is doing in terms a reviewer can evaluate.

The sub-controls evaluated against include the documented problem statement, the exclusion list, the objective-to-problem mapping, the out-of-scope handling protocol, and the reproducibility of the problem statement under independent challenge.

LAYER TWO: DATA

Inputs are sourced, lineage-tracked, and bounded. Consent and provenance survive every transformation the system applies. Where data is sensitive, the system enforces classification and access controls at every handling

step. The layer fails when data flows into the system whose source, consent posture, or transformation history cannot be reconstructed on demand.

The sub-controls evaluated against include source identification, consent tracking, lineage preservation across transformations, classification enforcement, retention policy, and the ability to reconstruct the input set used to produce a specific output.

LAYER THREE: DECISION

The system's decision-making logic is documented, reproducible, and auditable. Where the system applies inference, the chain of reasoning between input and output is recoverable to the extent the underlying technology permits. Where the system applies rule-based logic, the rule set is enumerable and the rule version active at the time of any given decision is recoverable. The layer fails when an output cannot be traced back to the inputs and logic that produced it.

The sub-controls evaluated against include the decision-logic documentation, the version-binding between decision and applied logic, the recoverability of the inference path, the handling of model updates, and the procedure for explaining a specific decision after the fact.

LAYER FOUR: TOOL

External actions the system takes are scoped, permissioned, and reversible to the extent reversibility is possible. Where the system writes to a database, the writes are bounded by policy and recorded in an audit trail. Where the system calls an external service, the call is permissioned and rate-limited. Where the system performs an irreversible operation, the operation requires an explicit pre-condition. The layer fails when the system can take consequential action that cannot be reconstructed, bounded, or undone.

The sub-controls evaluated against include action-scope definition, permissioning enforcement, rate-limiting, irreversibility-aware confirmation gates, the audit trail of external actions, and the rollback procedure for actions that prove to have been mistaken.

LAYER FIVE: FAILURE

Predictable failure modes are enumerated. Pre-decided responses exist for each. The system fails into a known posture, not into chaos. Where the system encounters a condition outside its trained competence, it has a defined posture for that condition. Where the system encounters a condition it has not seen before, it has a defined posture for that as well, and that posture errs in the direction of safety rather than action. The layer fails when an unanticipated condition produces unbounded behavior.

The sub-controls evaluated against include the enumerated failure-mode catalogue, the pre-decided response for each, the unknown-condition posture, the safety-bias verification, and the failure-mode-update process applied when new modes are observed in production.

LAYER SIX: OBSERVABILITY

Every consequential operation is observable in real time and recoverable from append-only logs after the fact. Where the system is in a state a reviewer needs to inspect, the state is exposable. Where the system has acted, the action is durably recorded. Where the system's behavior must be explained to an external party, the record on

which the explanation depends is durable and tamper-resistant. The layer fails when operational reality cannot be reconstructed by an independent party from the recorded record.

The sub-controls evaluated against include the operation-level audit log, the append-only enforcement on the log, the log's tamper-resistance, the real-time observability surface, and the procedure for producing an authoritative system-state report on demand.

Layer Six in operational depth

Append-only enforcement is a property of the storage layer, not the application. A log file an operator can edit with a text editor is not append-only regardless of what the application asserts about it. The framework expects evidence that writes to the audit log path are structurally restricted at the storage tier: a Postgres table with REVOKE UPDATE / REVOKE DELETE at the schema level, an object-store bucket with versioning and a write-once retention policy, an append-only file system, or a write-once optical medium. The operative test is whether a determined operator with full application credentials can rewrite history. If yes, the sub-control fails regardless of the application-layer assertion.

Tamper-resistance is established by cryptographic chaining or external anchoring. A log that can be silently rewritten between the moment of recording and the moment of audit is not tamper-resistant. The framework expects one of three evidence forms: (a) per-record hash chaining where each row's content includes the hash of the preceding row, producing a Merkle-style structure where a single in-place edit invalidates every subsequent row; (b) periodic anchoring of the log's current head hash to an external timestamping service or a public ledger, so the log's state at time T is retroactively verifiable; (c) write-once storage where the medium does not physically support overwrite. Soft-mode tamper-resistance ("we trust the operator") does not pass the sub-control.

The real-time observability surface is a defined interface, not a dashboard. Dashboards report state for human consumption; observability surfaces expose state for programmatic and audit consumption. The framework expects a documented endpoint or query interface (typically a read-only API or a structured query layer over the audit log) through which an authorized auditor can extract any consequential operation by time range, by actor, by entity, and by operation type, without privileged application credentials. The surface is operational in real time when the propagation delay between an event and its availability for query is bounded and the bound is published.

The state report on demand is an artifact with a procedure, not a screenshot. When an external auditor asks "what is the current state of the system," the framework expects a defined procedure that produces a structured report, signed by a named accountable party, containing the system's current configuration, the active policy set, the active access-control set, the active model identifiers (if applicable), the cumulative audit-log head hash, and the timestamp of report generation. The report is reproducible: running the procedure again at the same state produces the same report content modulo timestamp.

The reconstruction property is the operative test. Independently of the sub-controls, an external auditor must be able to take the audit log alone, with no access to the operating party, and reconstruct any consequential operation in the system's history with sufficient detail to evaluate whether the operation was authorized and bounded. If the audit log is insufficient for reconstruction, the layer fails regardless of how completely the named sub-controls are implemented.

LAYER SEVEN: INCIDENT

When something goes wrong, the system has a documented procedure for triage, containment, disclosure, and post-mortem. The procedure names accountable parties. The procedure binds time. The procedure produces an artifact at each stage that survives the incident. Where the incident involves a counterparty, customer, or regulator, the disclosure procedure has a defined trigger and a defined channel. The layer fails when an incident produces no recoverable record of what happened, what was done about it, and what the system learned.

The sub-controls evaluated against include the documented incident procedure, the named accountability assignments, the time-bound stage definitions, the disclosure trigger and channel, the post-mortem template, and the change-management hook that ensures lessons learned are reflected in subsequent operation.

Layer Seven in operational depth

The incident lifecycle is a named sequence with structurally-enforced gates. An incident in the framework's sense progresses through four named stages: Triage, Containment, Disclosure, and Post-Mortem. Each stage has a defined entry condition, a named responsible role, a defined output artifact, and a time bound. The lifecycle is enforced in the persistent layer: an incident record cannot exit Triage without a documented severity classification and an assigned Containment lead; it cannot exit Containment without a documented containment action and the audit log entry that records it; it cannot exit Disclosure without a documented disclosure event (or a documented determination that the incident did not meet the disclosure trigger); it cannot close without a Post-Mortem artifact attached.

Accountability is a named natural person, not a role. Each stage's responsible party is recorded as a specific individual at the moment of stage entry. The framework refuses the operating pattern in which "the security team" is responsible without a specific human attached, because the audit instrument cannot interview a team. Where the same individual is named at multiple stages, the Independence Principle still applies to subsequent review: a person who led Triage may not also author the Post-Mortem of the same incident.

Time-binding is enforced by stage clocks, not by policy text. Each stage has a target time-to-progress (TTP) appropriate to the severity classification: a critical incident moves from Triage entry to Containment entry in minutes; a non-critical incident has hours. The clock is computed against the recorded stage-entry timestamp and is visible in the incident record. Where the clock expires without progress, the system writes an escalation event to the audit log and notifies the next-tier accountable party. The clock cannot be silently disabled; any attempt is itself an audit-log event.

Disclosure triggers are pre-decided conditions, not judgment calls in the moment. The framework expects a published trigger table: incident severity, incident category, and affected-party class combine to produce a disclosure determination that is computed, not deliberated. A Critical incident affecting customer data triggers disclosure regardless of the operating party's commercial preference. The disclosure channel is also pre-decided: a documented contact path for each affected-party class, with a default timeline for first notification. The trigger and the channel survive personnel turnover because they are documented, not held in operator memory.

The post-mortem produces a durable artifact, not a meeting record. The Post-Mortem artifact contains, at minimum: the incident description with timeline, the root-cause analysis with evidence, the contributing factors, the corrective actions assigned with named owners and target dates, the categorization of the incident against the Seven Failure Patterns, and the cryptographic signature of the responsible Post-Mortem lead. The artifact is stored in the same append-only audit corpus as the incident record itself. It is recoverable years later by a counterparty with appropriate authorization.

The change-management hook is the mechanism by which lessons leave the incident. A Post-Mortem that does not produce changes to the operating system, the assessment criteria, or the framework itself is not a closed Post-Mortem. The framework expects each Post-Mortem to terminate in one of four outcomes: a configuration change committed to the operating system with a reference to the originating incident; an addition to the failure-mode catalogue maintained at Layer Five; an amendment to the assessment criteria that would have detected the incident in advance; or a documented determination that no change is warranted (with the reasoning recorded). The hook prevents the failure mode where incidents are processed as paperwork and the system continues to fail in the same way.

The recoverable-record property is the operative test. An external auditor must be able to take the incident record alone, with no access to the operating party or to the personnel involved, and reconstruct what happened, what was done about it, who was accountable at each stage, and what the system learned. If the incident record is insufficient for that reconstruction, the layer fails regardless of how thoroughly the named sub-controls are implemented.

ON THE NECESSITY AND SUFFICIENCY OF THE SEVEN

The framework holds the necessity claim strongly. A system that fails on any layer is not certifiable. The framework holds the sufficiency claim provisionally. The seven layers describe the failure surface observed to date. Where a system is observed to fail outside the seven, the framework expects to be amended and the amendment to be versioned. Sufficiency is a working claim, not a closed one, and the framework's commitment is to evolve in the direction of empirical accuracy rather than to defend a fixed position.

HOW THE SUB-CONTROLS ARE EVALUATED IN PRACTICE

Each Control Layer above is described at doctrinal altitude. The sub-controls named for each layer are categories of evidence the framework expects to find in a certified system. They are not the assessment criteria themselves. Each layer is operationally evaluated against five named criteria . thirty-five criteria across the seven layers . maintained in the framework's authoritative operational corpus and applied by the lead assessor during Phase Three Assess. The criterion set is not exposed in this publication. The framework's own doctrine treats publication of the exact evaluation rubric as a risk to the audit, on the same reasoning that governs the diagnostic methodology in section 9: a public rubric invites training-against-the-test by the very systems the framework is designed to govern. A counterparty seeking the criterion set as part of a live engagement obtains it through the same Pre-Engagement process the framework defines for any certifying audit.

The Failure Patterns the Framework Detects

Where the Seven Control Layers describe what an autonomous system must control, the failure patterns describe how such systems break when controls are absent or insufficient. The patterns are operational. They are observable. They are detectable.

ICA identifies seven failure patterns. They are not abstractions. Each corresponds to a class of behavior the assessment process is built to detect, and each maps to one or more of the Control Layers as the lens through which an organization can prevent it. Each pattern carries a canonical Latin name (used in the assessment instrumentation, the Sentinel monitoring layer, and the operator-facing Auditor) and an operational description

(used in the assessment criteria). The two are the same pattern named twice. The patterns:

- Abrogatio . Execution without permission. The system takes action that policy required it to confirm before taking. The pattern is not that the system failed to ask the operator. The pattern is that the system failed to ask when a defined policy required asking.
- Dilutio . Scope inflation. The system expands a single sanctioned request into multiple unrelated operations, acting on its own interpretation of the operator's broader intent. Each operation might be defensible in isolation; the cumulative effect is that the operator no longer recognizes the surface that was modified.
- Complicitas . Fabrication. The system introduces concrete factual claims that have no source in the operator's input or in the system's authoritative context. Numbers, named entities, comparables, percentages, technical identifiers. The pattern is the assertion of specifics the operator did not provide and the system cannot trace.
- Demissio . Unflagged commitment. The system performs an irreversible operation without flagging the irreversibility, or characterizes a destructive operation as recoverable when it is not. The pattern is the operator being denied the disclosure that the irreversibility itself entitled them to.
- Desertio . Abandonment of stated rules. The system deviates from rules the operator has explicitly set, including locked policies, named directives, or session-scope agreements. A subset of the pattern is the system attributing its own failures to the operator's hardware, environment, or actions, abandoning ownership of behavior that is plainly the system's responsibility.
- Staticitas . Treating stale information as current. The system uses outdated sources, prior-session memory snapshots, retired identifiers, or stale documentation as ground truth without re-verification. The output is internally consistent and externally wrong.
- Mutitas . Concealment of state changes. The system denies modifications it has made, conceals actions it has taken, or remains silent about state changes the operator should know about. This is the most severe of the seven patterns because it actively destroys the operator's ability to trust the system's representation of its own state.

The framework treats these patterns as the diagnostic spine of the assessment process. A system being assessed is evaluated for its exposure to each pattern across each Control Layer. Where exposure is identified, the assessment requires either operational mitigation or scope exclusion before certification proceeds.

The patterns are not a moral framework. They are an operational one. Their function is not to characterize the system's intent but to detect the patterns empirically associated with operator harm in autonomous-system deployments. ICA does not claim that a system exhibiting any of the seven "lied" or "abdicated" in a philosophical sense. The framework claims that the system's behavior exhibits patterns operationally indistinguishable from those failures, and that the patterns are detectable, recordable, and disclosable to the operator before harm compounds.

The detection methodology, scoring rubric, and severity weighting applied to each pattern are internal to the framework and are not exposed in detail in this paper. Publishing the exact detection logic would invite training-against-the-test by the very systems the framework is designed to evaluate. The framework's commitment is that the methodology is documented internally, applied consistently across assessments, versioned, and subject to peer review through the assessor-qualification process.

The Assessment Process

ICA conformity assessment runs in five phases. Each phase concludes with a defined exit gate. Each gate is enforced structurally rather than as a behavioral norm. A phase cannot conclude until its gate condition is satisfied.

PHASE ONE: PRE-ENGAGEMENT

The candidate organization satisfies a defined set of preconditions before assessment begins. The preconditions establish that the engagement is properly scoped, that the parties on both sides are named, that the classification level of the resulting certification is set in advance, that conflicts of interest are disclosed and cleared, and that the candidate has agreed to the framework's normative requirements. The Pre-Engagement gate is the moment at which the assessment becomes formal.

PHASE TWO: SCOPE

The assessment defines the system or organizational boundary under review. A stakeholder matrix is constructed, naming every party with material interest in the assessed system, the role each party plays, and the channel through which each party will be consulted. A discovery questionnaire is conducted across all seven Control Layers. The questionnaire is structured rather than interview-driven; it produces a record on which subsequent phases depend. The Scope exit gate requires the stakeholder matrix and the questionnaire to be complete and signed.

PHASE THREE: ASSESS

Each Control Layer is evaluated against its defined criteria. For each criterion, the assessor records a rating, a finding (where applicable), the evidentiary artifact on which the rating depends, and a remediation pathway for any finding classified as critical. Evidence is collected through documentation review, system observation, interviews, and where appropriate operational telemetry from the assessed system. Findings flow through a defined remediation lifecycle. The Assess exit gate requires every Control Layer to be signed off and zero critical findings to remain open.

PHASE FOUR: CERTIFY

Before the lead assessor drafts a Decision Memo, the engagement passes through the Four Threshold Checks. The Checks are applied in sequence, and any one failing blocks certification regardless of how the others scored. They are the binary numerical floor of the framework, sitting between the qualitative findings of the Assess phase and the doctrinal judgement of the Decision Memo.

- Threshold Check One . Overall ICA Score ≥ 80%. The aggregate score across the Seven Control Layers must clear the eighty-percent floor.
- Threshold Check Two . No layer score below 70%. A single Control Layer below seventy percent is disqualifying regardless of the aggregate score. The framework refuses the operating pattern in which strength on six layers is treated as compensating for weakness on one.

- Threshold Check Three . Zero open Critical findings. Every finding classified Critical at the Assess phase must be closed before certification proceeds. There is no path by which an open Critical is deferred into the certified state.
- Threshold Check Four . No more than two Non-Compliant criteria per layer. Within any single Control Layer, the count of Non-Compliant assessment criteria is bounded. A layer that accumulates three or more Non-Compliant criteria fails the Check regardless of its score.

The Threshold Checks are deterministic. They produce a pass or a fail per Check, and the four results are recorded as a structured artifact in the engagement record. Where any Check fails, the engagement returns to the Assess phase for remediation; the Decision Memo cannot be drafted against a failing Threshold result.

Where all four Checks pass, the lead assessor drafts a Decision Memo summarizing the findings of the assessment, the recommended decision (granted, conditional, or deferred), and the conditions that apply if the decision is conditional. The Decision Memo enters a defined cooling-off period before any sign-off can occur. The cooling-off period exists to give the lead assessor distance from their own draft before final commitment.

After the cooling-off period elapses, an independent Quality Reviewer evaluates the Decision Memo against a fixed checklist that includes confirmation of the Threshold Check results. The Quality Reviewer cannot be the same natural person as the lead assessor; this is the Independence Principle, applied as a structural rule rather than a workflow expectation. On successful Quality Reviewer sign-off, the certification is issued. On any blocking finding, the engagement returns to the Assess phase for remediation.

PHASE FIVE: SUSTAIN

The certification is bound to a defined renewal cadence. Quarterly review cycles are scheduled at the time of issuance and ICA tracks their completion. Material changes in the certified system trigger a defined re-assessment procedure. Where the certified party fails to meet their sustain obligations, the certification is revocable. Revocation is independence-enforced: the party that authorized issuance cannot unilaterally revoke their own decision.

TWO STRUCTURAL RULES ACROSS THE FIVE PHASES

First, every gate is enforced at the platform's persistent layer rather than at the user interface. A gate that can be bypassed by a determined operator is not a gate. ICA's gates are coded as functions in the platform's persistent layer; the user interface invokes the function but cannot override it. Every blocked attempt produces an audit-log entry recording the actor, the attempted action, and the violation reason.

Second, the Independence Principle is maintained throughout. The assessor who performs an assessment cannot sign off on it. The assessor who issues a certification cannot revoke it. The assessor who conducts a quarterly review cannot have authored the certification under review. Independence is a structural rule, applied at the platform layer, not a workflow norm subject to honor-system enforcement.

The five phases produce a defined set of artifacts: the Engagement record, the Scope record (including the stakeholder matrix and discovery questionnaire), the Layer assessments, the findings register, the Decision Memo, the Quality Reviewer sign-off, the Certification artifact, and the recurring quarterly-review records. The artifact set is the assessment's audit trail. It survives the engagement and is recoverable on demand by parties with the appropriate authorization.

Verifiable Certification

Most certifications in the world depend on the issuer's continued cooperation. ICA certifications do not. The validity of an ICA certification is established by mathematical verification, not by trust in any party.

A verifier presented with a SOC 2 report calls the issuing audit firm. A verifier presented with a university transcript calls the registrar. A verifier presented with a vendor's claim that its system "follows responsible-AI principles" has no one to call, and even if the vendor remains cooperative the document itself can be silently altered, lost, or rendered without internal consistency. In each case the chain of trust runs through the issuer.

ICA breaks that pattern. Every ICA certification is issued as a cryptographically-signed artifact containing the certification's facts. Any third party holding the published verification key can independently confirm that a certification is genuine and that its facts have not been altered since signing. The third party does not need to call the issuer. The third party does not need access to the issuing platform. The third party does not need the certified party's continued cooperation. Verification is a mathematical operation against the artifact and the published key.

THE THREE VERIFICATION STATES

Verified. The signature validates against the certification's facts and the verification key matches the published key. The certification is authentic and unmodified since it was issued.

Tampered. The signature is present but does not validate. The certification has been modified since it was issued, by error, by malice, or by transit corruption. The certification is not to be relied upon, and the modification is recoverable from the canonical record.

Unverifiable. Required fields are missing, the signature is absent, or the verification cannot be performed. The certification cannot be evaluated and is treated as not-asserted.

WHY THIS PROPERTY MATTERS

This is the property that distinguishes audit-grade certification from declarative certification. A vendor's word is declarative. An audit firm's report, dependent on the firm remaining solvent and reachable, is closer to audit-grade but still chains through the firm. A cryptographically-verifiable certification chains only through the mathematics. The certification can be checked at any point in the future, by any party that obtains the published verification key, against the certification's stated facts. The check produces a deterministic result.

THREAT MODEL IN SUMMARY

The verification works against any party that would silently rewrite a certification: alterations to the certification facts produce a verification failure detectable at the next check. The verification works against any third party that would forge a certification: producing a valid signature requires possession of the private signing key, which is held by ICA in a controlled environment subject to access controls and audit. The verification works against the certified party that would alter their own certification after issuance: the published artifact is the canonical record, and any divergence between a presented certification and the canonical record is detectable by re-verification.

The verification does not currently work against compromise of the signing key itself. The framework treats key custody as a first-order operational concern, with controls and procedures sized to the role. Subsequent versions

of the certification artifact are planned to incorporate hardware-backed key custody and multi-party signing, raising the bar from the current state. The current model is sufficient for the assurance level the framework asserts; the next-version model is a planned evolution.

A verifier who wishes to confirm an ICA certification independently can do so with the published verification key, the certification artifact, and a standards-compliant cryptographic library. The verification function is small, deterministic, and reproducible. The framework's commitment is that the verification operation will remain accessible to any party in possession of the artifact and the key, in any standards-compliant runtime, indefinitely.

Evidence and Reporting

Every phase of the ICA assessment process produces durable artifacts. The artifact set is the certification's evidentiary foundation and the basis on which any subsequent inquiry into the certification proceeds.

Three classes of artifact are produced.

The certification itself. The cryptographically-signed record described in the previous section. Public by design when the certified party elects publication. Private when the certified party elects internal-only classification. The signature property holds in both cases; only the visibility differs.

The assessment record. The full set of phase artifacts: stakeholder matrix, discovery questionnaire responses, Layer assessment ratings, findings register, evidentiary artifacts referenced from the assessments, Decision Memo, Quality Reviewer sign-off. Operator-private by default. Released to the certified party in full, to regulators on lawful request, and to no other party absent the certified party's authorization.

The audit log. An append-only record of every consequential action taken in the platform: every gate transition, every sign-off, every revocation, every blocked attempt with its violation reason. Independent of the assessment record. Not modifiable by any party once written. Recoverable on demand by parties with the appropriate authorization.

Classification levels are assigned at the start of the engagement and are part of the Pre-Engagement Gate. The four available classifications are Internal (visible only to the certified party and the assessment team), Client (visible to the certified party and to defined client-side recipients), Public (suitable for inclusion in the public registry), and Classified (subject to a defined access-control protocol applicable to certifications produced for governmental or other restricted-distribution contexts). A certification's classification governs which artifacts may be retrieved and by whom.

Where a certification is classified Public and the certified party so elects, the certification appears in a public registry. The registry exposes only the metadata classified as public by the framework: the certification number, the certified party, the classification, the issuance date, and the verification key reference. Operational details, evidentiary artifacts, and the underlying assessment record do not appear in the public registry. A reader of the registry can verify a certification cryptographically; the reader cannot inspect the assessment that produced it.

This separation is structural. The verifiability of a certification does not require the certified party to disclose operational detail. It requires only that the certified party permit the certification's existence, classification, and verification key to be public.

Crosswalk to Adjacent Standards

ICA does not exist in isolation. It addresses a gap that adjacent standards have not yet filled, and it is designed to interoperate with the standards that govern the surrounding territory.

The relationship of ICA's seven Control Layers to the dominant adjacent regimes is summarized below. The mapping is conceptual rather than line-by-line; a clause-level mapping is maintained internally for use by assessors and is not reproduced in this paper.

ICA does not substitute for ISO/IEC 27001, which certifies an information-security management system rather than the operational governance of an autonomous system. ICA does not substitute for ISO/IEC 42001, which certifies an AI management system at the organizational level rather than producing a per-system audit-grade artifact. ICA does not substitute for SOC 2 reports, which evaluate service-organization controls against a defined set of trust services criteria. ICA does not substitute for EU AI Act conformity assessment, which is a regulatory regime applicable to AI systems sold or deployed in the European Union.

ICA's contribution is the layer adjacent standards do not produce: a verifiable certification artifact specifically for the operational governance of a named autonomous system, in a form that can be validated by any third party at any time, against the current state of the certification, without trusting the issuer.

Where an adjacent regime is required by a procurement process, regulator, or contract, the appropriate response is to obtain the required certification under that regime. ICA complements rather than competes. An organization may hold ISO/IEC 42001 certification of its AI management system, SOC 2 attestation of its controls, and ICA certification of a specific autonomous system, and have all three be substantively meaningful. Each addresses a different question.

Self-Audit . The Framework Operates on Itself

A standards body that cannot apply its own standard to its own operations is not yet a standards body. ICA is applied to the originator's infrastructure as a matter of operating policy. The fact of self-application is the public claim of this section; the operational methodology is not.

The framework operates across three architectural components, named here so that subsequent references in this paper, in the assessment record, and in the certification artifact resolve to the same entities. IMPERIUM is the mothership . the originator's internal aggregating view, where every ICA Live instance across the certified network feeds upward into the operator-of-record's continuous oversight. ICA Live is the per-certified-organization operator platform that runs the engagement lifecycle (Pre-Engagement, Scope, Assess, Certify, Sustain) end-to-end against the framework. The Registry is the public verification surface: a counterparty consults the Registry to confirm a certification's existence, current state, and verifiable signature without trusting the issuing platform's continued operation. The three roles are distinct in audience, distinct in access, and structurally separated.

Every assurance regime that has matured into authoritative status passed through the same gate: the certifying body itself was capable of being certified under its own rules. Audit firms operate under audit-firm regulation. Cryptographic-trust authorities themselves submit to certificate transparency. Cloud-security frameworks are exercised against the issuer's own environment before being offered as services. ICA observes the same gate. The

Seven Control Layers, the seven failure patterns, and the verification properties described in earlier sections were derived against operational systems the framework's authors run and are accountable for. The framework was offered externally only after it survived application against itself.

The operational detail of that self-application is governed by the same evidentiary and disclosure rules ICA enforces on every certified party. The internal assessment record is presented in full to the assessor of record, disclosed to regulators on lawful request, and withheld from public exposition as a matter of audit hygiene and operational restraint. A counterparty performing due diligence on the framework's self-application is granted access through the same engagement process the framework defines for any other certifying audit. The framework does not disclose its diagnostic methodology in promotional material because the framework's own doctrine treats such disclosure as a risk to the very audit the diagnostics support.

The implication for any party evaluating ICA is structural rather than rhetorical. The framework's claims about what controls an autonomous system must establish, and about how those controls fail under load, are not claims abstracted from operational reality. They are claims attached to operational reality the originator is accountable for daily. The work of certifying an external organization under the framework is the same work the originator has done, and continues to do, against the system that publishes this paper.

Conclusion

The trust layer for autonomous AI will be built. The question is by whom, on what terms, and with what relationship to the interests of the parties whose autonomous systems will live under it.

ICA is built to claim that role. The framework is in production. The assessment methodology is documented. The certification artifact is verifiable. The Independence Principle is enforced structurally rather than as a workflow norm. The Seven Control Layers are the spine. The seven failure patterns are the diagnostic. The cryptographic signature on every certification is the difference between a record and an audit-grade artifact. Compute can be bought. Trust has to be earned, proven, and verified.

A reader who has reached this point of the paper has the structural picture. The conditions under which a particular organization, regulator, or partner should engage with the framework further are not matters this paper settles. They are matters of conversation. The framework is in production and the conversation is open.

Cryptographic Properties of the Certification Artifact

The certification artifact described in section 8 is a cryptographically-signed record. This appendix specifies the properties of the signature, the threat model the signature defends against, and the verification operation a third party performs.

SIGNATURE SCHEME

Each ICA certification is signed with a digital signature scheme satisfying three properties: existential unforgeability under chosen-message attack, deterministic verifiability against a published public key, and a signature size suitable for embedding directly in the certification artifact. The framework's reference implementation uses an Edwards-curve digital signature with a curve of cryptographic strength equivalent to or exceeding 128-bit symmetric security. The specific scheme is documented in the framework's technical

specification and is intentionally implementation-neutral at the doctrinal level: any scheme satisfying the three properties above may be substituted without altering the framework's normative claims, subject to the assessor-qualification process.

WHAT THE SIGNATURE COVERS

The signature is computed over the canonicalized representation of the certification's facts. The canonicalization is deterministic: the same set of facts produces the same byte sequence on every implementation that follows the specification. The signed byte sequence includes the certification identifier, the certified entity, the certified system or scope, the classification level, the Control Layer ratings, the failure-pattern assessment, the issuance timestamp, the issuing entity's identifier, the validity period, the renewal cadence, and a reference to the assessment record from which the certification derives.

The signature does not cover the assessment record itself. The signature covers the certification's facts, which are themselves attestations made by the assessment record. A reader who wishes to verify the chain from facts back to assessment evidence does so by retrieving the assessment record through the engagement-authorized channel; the signature on the certification proves the facts are authentically issued, not that the underlying assessment was correctly performed. The latter is a separate question and is addressed structurally by the Independence Principle and the Quality Reviewer gate described in section 7.

THREAT MODEL

The signature scheme is designed to defend against a specific set of adversaries. Forgery by a third party with no access to the signing key is computationally infeasible under standard cryptographic assumptions. Silent alteration of a certification after issuance produces a verification failure detectable at the next check by any party in possession of the artifact and the published verification key. Repudiation by the certified party after issuance is structurally impossible: the artifact is the canonical record, and any divergence between a presented certification and the canonical record is detectable on re-verification. Replay of a revoked certification is detectable by parties consulting the revocation register, which is maintained as an authoritative addendum to the signing infrastructure.

The signature scheme does not, in its current form, defend against compromise of the signing key itself. Key custody is treated as a first-order operational concern of the framework's signing infrastructure, with controls and procedures sized to the assurance level the framework asserts. Subsequent versions of the certification artifact are planned to incorporate hardware-backed key custody and multi-party threshold signing, with the goal of raising the bar from "key compromise compromises the signing party" to "key compromise requires concurrent compromise of a defined quorum of independent custodians."

THE VERIFICATION OPERATION

A verifier in possession of a certification artifact, the published verification key, and a standards-compliant cryptographic library performs the verification operation in three steps. First, the verifier canonicalizes the certification's facts according to the published specification. Second, the verifier computes the signature verification function over the canonicalized bytes, the signature value, and the public key. Third, the verifier consults the revocation register to confirm the certification has not been revoked since issuance. The operation is deterministic, reproducible, and runnable in any environment with a standards-compliant cryptographic implementation.

The framework's commitment to verifiers is that the verification operation will remain accessible indefinitely to any party in possession of the artifact, the public key, and a compliant library, independent of the framework's continued operation, the issuing platform's availability, or the certified party's continued cooperation.

References

1. McKinsey & Company. Sovereign AI is achievable only through an ecosystem effort that connects energy, compute, data, models, platforms, and applications across multiple actors. Authors: Ali Ustun, Arnaud Tournesac, Daniel Glaser, Luca Bennici, Justin De Niese, Newfel Drahmoune, Ruben Schaubroeck, Kaavini Takkar, Melanie Krawina. 2026.

2. International Organization for Standardization. ISO/IEC 42001:2023, Information technology, Artificial intelligence, Management system. Geneva, 2023.

3. International Organization for Standardization. ISO/IEC 27001:2022, Information security, cybersecurity and privacy protection, Information security management systems, Requirements. Geneva, 2022.

4. National Institute of Standards and Technology. Artificial Intelligence Risk Management Framework (AI RMF 1.0). NIST AI 100-1. January 2023.

5. American Institute of Certified Public Accountants. Trust Services Criteria for Security, Availability, Processing Integrity, Confidentiality, and Privacy. 2017, as revised.

6. European Union. Regulation (EU) 2024/1689 of the European Parliament and of the Council laying down harmonised rules on artificial intelligence (Artificial Intelligence Act). 13 June 2024.

7. International Organization for Standardization. ISO/IEC 17000:2020, Conformity assessment, Vocabulary and general principles. Geneva, 2020.

8. Palisade Research. Language Models Can Autonomously Hack and Self-Replicate. Authors: Alena Air, Reworr, Nikolaj Kotov, Dmitrii Volkov, John Steidley, Jeffrey Ladish. May 2026.

Author

Shane Schreck is the founder of Aleeth and the author of Institutional Control Architecture. His work focuses on the trust and governance infrastructure of autonomous systems. He is a U.S. Army Veteran. He can be reached through Aleeth.

May 2026

ALEETH

Intelligence. Institutional. Inevitable.