International Journal of Artificial Intelligence, Data Science, and Machine Learning
Grace Horizon Publication | Volume 7, Issue 3, 83-94, 2026
ISSN: 3050-9262 | https://doi.org/10.63282/3050-9262.IJAIDSML-V7I3P111

Original Article

Detection Confidence as the Authorization Gate: Integrating the Detection Lifecycle Codex and Agentic Security Governance Framework into a Closed-Loop Autonomous SOC Architecture

Parijat Sharma
Independent Researcher, USA.

Received On: 19/06/2026 ***Revised On: 22/07/2026*** ***Accepted On: 29/07/2026*** ***Published On: 06/08/2026***

Abstract - *Autonomous response should not be triggered merely because a detection fires. This paper proposes detection confidence as an authorization gate that determines whether a security action may execute automatically, requires human review, or must be held. Two frameworks are specified to make that gate operable. The Detection Lifecycle Codex governs a detection from use-case definition through telemetry contract, adversarial validation, confidence calibration, authorization classification, runtime assurance, and retirement, and it assigns each detection an authorization class rather than a severity label. The Agentic Security Governance Framework governs the security agents that recommend or execute response, controlling identity, tool permissions, action boundaries, approval requirements, and rollback. A closed-loop architecture integrates the two: evidence is assembled, confidence is assessed across six independent dimensions, action risk is classified separately, and a policy decision point outside the agent's trust boundary arbitrates between them before anything executes. Outcomes are recorded and returned to confidence calibration and authorization policy. The arbitration rule is asymmetric by design, in that confidence can only authorize consideration of an action and never overrides the action-risk class, so no confidence score unlocks an irreversible operation. The architecture is specified and evaluated analytically. It is not reported as a measured production deployment, and the paper states which quantities remain unmeasured and what a controlled pilot would have to produce.*

Keywords - *Autonomous SOC, Detection Confidence, Security Orchestration, Agentic AI Security, Detection Engineering, Authorization Policy, Human-In-The-Loop, Closed-Loop Response, Telemetry Integrity, Explainable Automation, Security Governance, Adaptive Incident Response.*

1. Introduction

A security operations centre has two failure modes and they pull in opposite directions.

The first is latency. A detection passes through alert creation, queue assignment, triage, evidence collection, enrichment, escalation, approval, action, validation, and closure. Each of those steps can be individually reasonable and the aggregate still turns a technically detectable event into a response that lands tens of minutes or hours after the fact. The attacker uses that interval. Privileges get escalated, persistence gets established, movement continues laterally, data leaves, controls are disabled, and evidence is destroyed. None of that is hypothetical and none of it is new.

The second failure mode is what happens when an organization solves the first one carelessly. Acting automatically on every alert will disable legitimate users, isolate production servers, revoke credentials that a business process depends on, block traffic that is paying the bills, and delete resources, all on the strength of a false positive, stale context, poisoned telemetry, or an incomplete correlation. The response becomes the incident. In a regulated environment it becomes the incident with a regulator attached.

The interesting observation is that these two failure modes are usually treated as a single dial to be turned, more automation or less. They are not. They are two different questions that happen to be asked at the same moment: how strongly does the evidence support the conclusion, and how much damage does the proposed action do if the conclusion is wrong. Collapsing them into one severity field is the design error at the root of both failures.

Alahmadi et al. [1] provide the empirical detail that makes this concrete. In their study of SOC practitioners, analysts reported false-positive rates approaching total, and, critically, most of those false positives were not broken analytics. They were benign triggers, meaning the detection correctly identified the behavior it was written to identify and that behavior turned out to be legitimate in that environment. A detection can be functioning exactly as designed and still be an unsafe basis for a disruptive action. Severity does not capture this. Severity is a property of the

rule; what matters at the moment of response is a property of the specific evidence in front of you.

The 2026 literature on autonomous security operations has moved quickly and has largely optimized the wrong half. AgentSOC [3] reports sub-second enrichment. Saju and Azim [4] report an end-to-end detect-query-resolve loop at 82.8 percent accuracy with a false-positive rate of 0.120. Sahay et al. [5] apply policy guidance to triage in a commercial SIEM, and Cadet et al. [6] apply retrieval augmentation to incident analysis. These are real advances in the investigative step. What they leave open is the step immediately after it. A framework that classifies correctly 82.8 percent of the time is a useful analyst; it is not, without further argument, something that should be permitted to revoke production credentials unsupervised, and the residual 17.2 percent is exactly where the argument has to be made.

This paper makes that argument structurally. It proposes detection confidence as a formal authorization gate sitting between detection and action, and it specifies the two frameworks needed to make the gate operable rather than rhetorical. The Detection Lifecycle Codex governs the detection so that a confidence score means something auditable. The Agentic Security Governance Framework governs the agent so that the executor of an authorized action cannot exceed the authority granted. A closed-loop architecture integrates them and returns outcomes to both.

The paper makes four contributions.

First, it specifies detection confidence as a composite of six independent, individually auditable dimensions rather than a single model probability, and it defines the aggregation rule as worst-dimension gating rather than a weighted average, so a single failed dimension cannot be averaged away by five healthy ones.

Second, it separates confidence from action risk as orthogonal axes and defines the arbitration rule between them as deliberately asymmetric. High confidence can never promote an irreversible action into the autonomous lane. This is the central design claim of the paper.

Third, it specifies the Detection Lifecycle Codex as eight stages with named exit criteria and evidence artifacts, terminating in an authorization class assigned to the detection rather than a severity label.

Fourth, it treats adversarial manipulation of the confidence signal as a first-class design problem rather than as future work, including the correlated-source failure in which several apparently independent corroborating signals originate from one compromised control plane.

The rest of the paper runs as follows. Section 2 positions the work against the two literatures it sits between. The two frameworks and the loop that integrates them are specified in Section 3, with the confidence model and the arbitration rule following in Section 4. Failure modes and adversarial resistance get their own treatment in Section 5, because they are the part most often deferred. Section 6 states the limits plainly, and Section 7 sets out the evaluation framework a pilot would need.

2. Background and positioning

2.1. Current SOC automation

SOC automation as deployed today is dominated by rule-based orchestration, predefined playbooks, case-management workflow, and, increasingly, AI-assisted triage. These are genuinely effective for enrichment and for repetitive work, and nothing in this paper argues against them.

The limitation is what triggers the action. In most deployments it is the alert severity or the rule match. That is unsafe for a reason worth stating precisely: severity is a static property assigned when the detection was authored, and it encodes the author's expectation of how bad this class of event would be if real. It does not encode whether the evidence in this specific instance is complete, whether the sources were healthy at the time, whether anything independent corroborates it, or whether the proposed response is proportionate to the risk of being wrong. Those are the four things you actually need to know, and none of them is available to a severity field.

The recent agentic systems inherit this. Their improvements are concentrated in enrichment quality and classification accuracy, and their reported metrics are accordingly accuracy metrics [3], [4]. Accuracy is necessary. It is not sufficient, because it is a property averaged over a distribution of cases, and authorization is a decision made about one case.

2.2. Autonomy, trust, and pre-action authorization

Mohsin et al. [2] come closest to the framing used here. Their unified framework for human-AI collaboration in SOCs treats trusted autonomy as the design objective, and it establishes correctly that autonomy has to be earned rather than assumed. What it does not specify is the decision function itself: which quantity is measured, what it is compared against, and what happens when the two disagree. This paper supplies that function.

The agent-security literature has independently converged on the missing mechanism from the other direction. Uchibeke [13] argues for deterministic pre-action authorization evaluated before the tool call rather than inferred from the agent's stated intent. Zhu and Wang [14] name intent-tool mismatch as an access-control failure mode and bound privilege per request. Kodathala [16] moves the authorization decision off the agent's host so that a compromised agent cannot authorize itself, which is the correct structural instinct and is adopted here. Wu et al. [15] give agent guardrails a formal treatment. Lazer et al. [7] survey the field.

None of that work is specialized to security response, and the specialization matters more than it might appear. In the general agent case, the tool call is a business operation

and the evidence justifying it is ordinary application state. In the security-response case, the tool call is itself a destructive privileged operation of exactly the kind the security programme exists to control, and the evidence justifying it is generated in an environment where an adversary is present and may be shaping that evidence. Both properties tighten the requirements considerably.

2.3. Contribution boundary against the author's prior work

The author has previously specified a method for making security telemetry pipelines explicitly detection-ready, covering signal contracts, provider-neutral normalization that preserves evidence, coverage testing, latency objectives, provenance controls, and telemetry-health monitoring [18].

The boundary between that work and this one should be stated plainly, because both concern detection quality and a reader could reasonably expect overlap.

That work is about the evidence. Its subject is the pipeline that delivers signals to an analytic, and its objective is that coverage, timeliness, trustworthiness, and failure modes are measurable before detections silently degrade. Its central assertion is that log availability is not equivalent to detection usability.

This paper is about what may be done with the evidence. Its subject is the authorization decision that follows a detection, and its objective is that autonomous action is granted only where the evidence and the action risk jointly permit it.

The two meet at exactly one point, and only one. Telemetry health enters this paper as a bounded input to the confidence score, specifically the first of the six dimensions defined in Section 4.1. The seven-layer signal taxonomy, the twelve pipeline stages, the coverage-testing method, and the fail-visible detection behavior specified in the prior work are cited here rather than restated. Where this paper requires the property that a required source being down must render an analytic explicitly degraded rather than silently normal, that property is attributed to [18] and consumed, not re-derived.

2.4. The gap

Detection quality is a governed and measurable property in the detection-engineering literature. Action authorization is a governed and measurable property in the agent-security literature. No published work makes the first the gating input to the second, treats the resulting confidence as dynamic and auditable rather than as an opaque score, and specifies the arbitration rule for the case that matters most, which is high confidence attached to an irreversible action.

3. Architecture

3.1. The Detection Lifecycle Codex

The Detection Lifecycle Codex is a governed lifecycle for security detections. It defines the minimum stages, evidence, ownership, and exit criteria required before a detection may be trusted for operational or autonomous use. Its purpose in this architecture is narrow and specific: it is what makes a confidence score mean something. A confidence number attached to a detection whose telemetry dependencies are undocumented, whose precision has never been characterized, and whose evasion behavior has never been tested is a number without a referent.

The Codex defines eight stages.

- Stage 1, threat objective and use-case definition. Identify the behavior, asset, identity, or attack path the detection is intended to identify, expressed against a behavioral taxonomy [12], and assign an accountable detection owner together with the business or service context.
- Stage 2, telemetry contract. Define the mandatory signals, source systems, fields, timestamps, fidelity, retention, expected latency, and acceptable loss. This stage consumes the signal-contract mechanism specified in [18]. A detection that cannot state its telemetry contract cannot proceed.
- Stage 3, detection logic design. Specify the analytic rules, correlations, models, assumptions, exclusions, dependencies, and expected failure modes.
- Stage 4, validation and adversarial testing. Test true-positive scenarios, false-positive scenarios, missing data, delayed data, replay, tampering, and evasion. The negative cases matter more than the positive ones for this architecture's purposes, because the confidence model needs a characterized false-positive profile, not just a demonstration that the rule fires.
- Stage 5, confidence calibration. Score evidence quality, corroboration, historical precision, environmental relevance, and telemetry health, and document the supporting evidence for each.
- Stage 6, deployment and authorization classification. Determine whether the detection is advisory, human-gated, or eligible for bounded autonomous response. This is the stage that replaces severity assignment as the operationally meaningful output of detection engineering.
- Stage 7, runtime assurance. Monitor precision, recall proxies, drift, telemetry health, suppression behavior, and action outcomes.
- Stage 8, tuning, versioning, and retirement. Approve changes, preserve evidence, revalidate material modifications, and retire or downgrade detections that no longer meet requirements.

Two rules govern movement through the stages. A detection may not skip a stage, and material modification returns it to Stage 4. The second rule exists because the most common way a governed detection silently becomes ungoverned is an exclusion added under time pressure during an incident, which changes the false-positive profile without changing anything the confidence model can see.

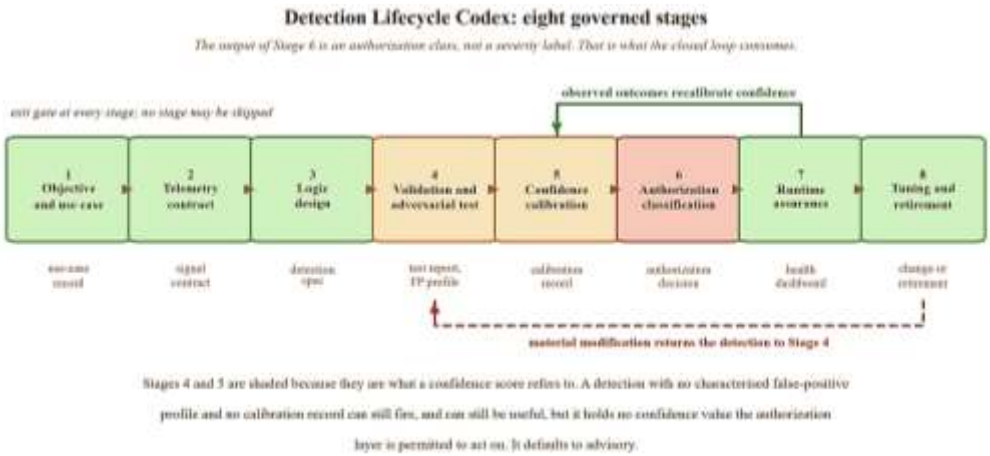

Fig 1: The Eight Stages of the Detection Lifecycle Codex, Showing the Exit Gate at Each Stage, the Evidence Artifact Each Stage Produces, and the Two Return Paths. The Output of Stage 6 is An Authorization Class, Not a Severity Label, Which Is What Makes the Detection Consumable by the Closed Loop.

Table 1 gives the exit criteria and evidence artifact for each stage. The evidence artifact column is what an auditor is shown, and it is the reason the Codex is specified as producing documents rather than as a set of good practices.

Table 1: Detection Lifecycle Codex Stages, Exit Criteria, and Evidence Artifacts. The Artifact Column Is The Audit Surface. A Detection with No Calibration Record Cannot Hold a Confidence Score, And A Detection with No Authorization Decision Record Cannot Be Autonomous Regardless of Its Score

Stage	Exit criterion	Evidence artifact	Accountable
1. Objective and use case	Behavior and asset scope named; owner assigned	Use-case record with behavioral-taxonomy mapping	Detection owner
2. Telemetry contract	All mandatory signals, fields, latency and loss tolerances declared	Signal contract [18]	Detection owner with platform engineering
3. Logic design	Assumptions, exclusions, dependencies and expected failure modes documented	Detection specification, versioned	Detection engineer
4. Validation and adversarial testing	Positive, negative, missing-data, delayed-data, replay, tamper and evasion cases executed	Test report with characterized false-positive profile	Detection engineer with adversarial testing
5. Confidence calibration	Each of the six confidence dimensions scored with stated basis	Calibration record	Detection owner
6. Authorization classification	Class assigned: advisory, human-gated, or bounded autonomous	Authorization decision record with approver	Security governance
7. Runtime assurance	Precision, drift, telemetry health, suppression and outcomes under continuous monitoring	Runtime health dashboard and exception log	Security operations
8. Tuning and retirement	Change approved and revalidated, or detection retired with evidence preserved	Change record and retirement note	Detection owner with governance

3.2. The Argentic Security Governance Framework

The Detection Lifecycle Codex governs the evidence side. The Agentic Security Governance Framework governs the actor side, meaning the security agents that reason, recommend, orchestrate, or execute.

Its central rule is that an agent receives only the minimum authority required for a defined task, and that authority is evaluated at runtime for each proposed action rather than granted as standing privilege. This aligns with the pre-action authorization principle argued by Uchibeke [13] and with the request-bounded privilege model of Zhu and Wang [14], and it is the reason the policy decision point in Section 3.4 sits outside the agent's own trust boundary, following Kodathala [16].

The framework governs twelve domains: agent identity and ownership; tool and API permissions; data access and handling boundaries; permitted, prohibited and approval-required actions; runtime authorization and least privilege; human oversight and segregation of duties; prompt, model, workflow and policy change control; memory, context and sensitive-data retention; audit logs, evidence preservation and explainability; testing, simulation, rollback, kill switch and incident response for the agent itself; third-party service and model dependencies; and performance, safety and governance metrics.

Three of those deserve comment because they are where implementations usually fail.

Change control over prompts and models is the domain most often omitted. A modification to a system prompt can change an agent's action selection as materially as a code deployment, and if it does not pass through change control then the agent's behavior is unversioned. An unversioned actor cannot be held to a governance record.

Incident response for the agent itself is the second. The framework treats the agent as a subject of security operations and not only as an instrument of it. If an agent is compromised, there must be a defined containment path, a kill switch that does not depend on the agent's own cooperation, and a means of determining which of its past actions were legitimate.

Segregation of duties is the third, and it has a specific meaning here. The component that assesses confidence must not be the component that executes the action, and neither may be the component that authorizes it. This is not organizational hygiene; it is what prevents a single compromised component from producing a self-justifying destructive action.

Manifest-style capability declaration, as used by Barbieri et al. [8], is a compatible mechanism for expressing the tool-permission domain, and an implementation may adopt it.

3.3. The closed loop

The loop has ten steps.

1. A detection fires and creates an evidence package containing the rule or model version, raw and normalized events, affected entities, timestamps, telemetry-health status, and the known limitations recorded at Stage 3 of the Codex.
2. An enrichment layer collects independent context: identity risk, asset criticality, vulnerability exposure, network observations, recent administrative changes, threat intelligence, and related detections. Retrieval-augmented assembly of this context is feasible at operational speed [6].
3. A confidence engine evaluates the six dimensions of Section 4.1 and produces a confidence level together with an explanation, not only a numeric score. The explanation is a requirement rather than a convenience, because a score whose derivation cannot be inspected cannot be contested by an analyst and cannot be defended to an auditor.
4. A policy decision point combines confidence with the action-risk class of Section 4.2, applying the arbitration rule of Section 4.3.
5. The decision routes the case to bounded automatic action, human review, or hold-and-observe.
6. For automatic actions, the execution layer uses least-privileged, time-bound credentials and an approved playbook. It validates preconditions, executes, checks success, and attempts rollback where necessary.
7. For human review, the analyst receives the evidence, the confidence rationale, the recommended action, the expected impact, and the rollback plan. Approval or rejection is recorded with the reason.
8. For held cases, the system waits for additional evidence, requests missing telemetry, increases observation, or applies non-disruptive monitoring controls.
9. The outcome is measured: whether the event was confirmed, whether the action succeeded, whether business impact occurred, and whether the detection or policy should be tuned.
10. Results feed back into confidence calibration, playbook eligibility, detection lifecycle status, and governance policy.
11. Step 10 is what makes the loop closed rather than merely automated. Without it, calibration is a one-time exercise at Stage 5 of the Codex and drifts from the moment it is completed.

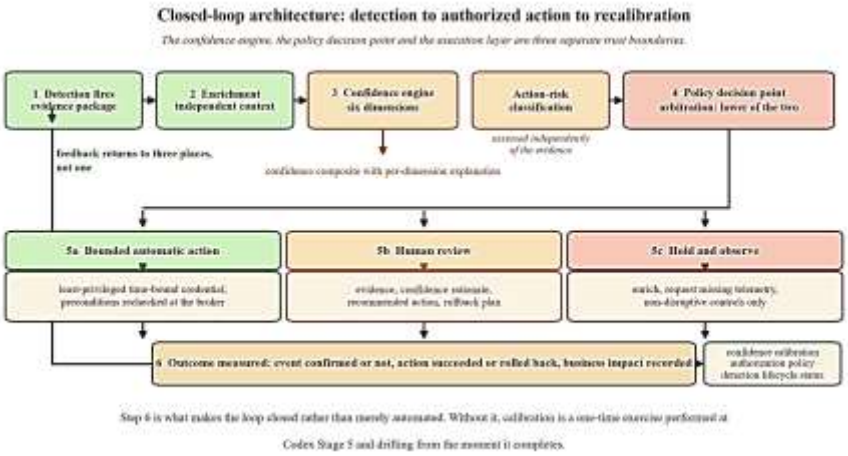

Fig 2: The closed-loop architecture. The confidence engine and the policy decision point are separate components with separate inputs, and both sit outside the execution layer's trust boundary. The feedback path from outcome measurement returns to three destinations, not one: confidence calibration, authorization policy, and the detection's lifecycle status.

3.4. Placement of the decision point

The architecture reuses the policy decision point and policy enforcement point separation established for zero trust architectures [11]. The reuse is deliberate and the reasons are worth making explicit.

The policy decision point must be outside the agent's trust boundary. If an agent can influence the component that authorizes it, then compromising the agent compromises authorization, and every downstream control is decorative. This is the structural argument made by Kodathala [16] and it is adopted without modification.

The policy decision point must also be outside the confidence engine. If confidence assessment and authorization are the same component, then an adversary who can influence evidence can influence authorization directly rather than through the arbitration rule, which is the one place the architecture applies independent constraints.

The policy enforcement point is placed at the tool broker rather than inside the playbook. A playbook that enforces its own limits enforces them only for cases that reach that playbook.

3.5. What the agent is and is not permitted to decide

The agent selects and proposes. It does not authorize. That sentence is the whole of the design intent and it is worth separating from the mechanism.

An agent may assemble evidence, reason over it, produce a recommendation, name the action it believes appropriate, and estimate impact. Each of those is a genuine contribution and each is auditable. What the agent may not do is convert its own recommendation into permission, and it may not vary the scope of an authorized action during execution. An authorization is granted for a specified action against specified entities within a specified window, and the tool broker enforces those bounds independently of what the agent subsequently requests.

4. Detection confidence as an authorization input

4.1. The six dimensions

Detection confidence in this architecture is a composite of six dimensions, each separately measurable and separately auditable. The choice to decompose rather than to use a single model probability rests on a well-established result. Guo et al. [9] showed that modern neural network confidence is systematically miscalibrated, and that the miscalibration is not a small correction. A single opaque probability is therefore an unsuitable gate for a destructive action even when the underlying classifier is accurate, because the number does not mean what its scale implies it means.

The six dimensions are as follows.

- Telemetry completeness and freshness: Are all mandatory signals from the Stage 2 contract present, and are they within their declared latency objective? This is the dimension that consumes the prior work directly [18], including the fail-visible property that a required source being down renders the analytic explicitly degraded rather than silently normal.
- Source integrity: Are the sources authenticated, untampered, and operating normally? A source that is present and fresh but whose integrity is unverified is a different condition from a source that is absent, and the two must not collapse into one score.
- Corroboration: Do independent data sources support the same conclusion? The word independent is doing considerable work and is examined in Section 5.2.
- Analytic reliability: What is this detection's validated precision, known false-positive profile, and version status, as characterized at Stage 4 of the Codex?
- Contextual fit: Is the behavior abnormal for this identity, asset, application, or business process? This is the dimension that addresses the benign-trigger problem identified by Alahmadi et al. [1], where a correct detection fires on legitimate environment-specific behavior.
- Adversarial resistance: Could an attacker fabricate, suppress, replay, or selectively trigger this evidence? Unlike the other five, this dimension is a property of the detection's design rather than of the current instance, and it is assessed at Stage 4 and carried forward.

Table 2: The Six Confidence Dimensions, Their Measurement Basis, and Their Characteristic Failure. Each Dimension Has an Independent Failure Mode, Which Is the Argument for Scoring Them Separately Rather Than Folding Them into One Probability

Dimension	What is measured	Primary source	Characteristic failure
Telemetry completeness and freshness	Mandatory signals present and within declared latency objective	Signal contract and pipeline health [18]	Silent source loss reported as normal
Source integrity	Authentication, tamper-evidence and operating state of each source	Provenance and integrity controls	Logging plane compromised; records authentic in form, false in content
Corroboration	Number and independence of sources supporting the conclusion	Enrichment layer [6]	Apparent independence with shared upstream control plane
Analytic reliability	Validated precision, false-positive	Codex Stage 4 test	Profile characterized once, then

	profile, version status	report	exclusions added under incident pressure
Contextual fit	Deviation from baseline for this identity, asset, application or process	Behavioral baselines and asset inventory	Rare but legitimate behavior scored as anomalous
Adversarial resistance	Fabrication, suppression, replay and selective-trigger exposure of the evidence	Codex Stage 4 adversarial assessment	Assessed at design time and never revisited as the environment changes

4.2. Aggregation is worst-dimension gating, not averaging

The aggregation rule is that the composite confidence is bounded by its weakest dimension.

This is a substantive design decision and it is where a weighted average would give the wrong answer. Consider a detection with excellent analytic reliability, strong contextual fit, good corroboration, verified source integrity, sound adversarial resistance, and one mandatory telemetry source that has been silently absent for six hours. A weighted average returns a high composite score. The correct answer is that the evidence is incomplete in a way that the other five dimensions cannot compensate for, because they are computing over a partial picture and do not know it.

The rule is therefore that any dimension below its own floor caps the composite regardless of the others, and the reason the composite is presented with its per-dimension breakdown rather than as a bare number is so that the capping dimension is visible to whoever reads it.

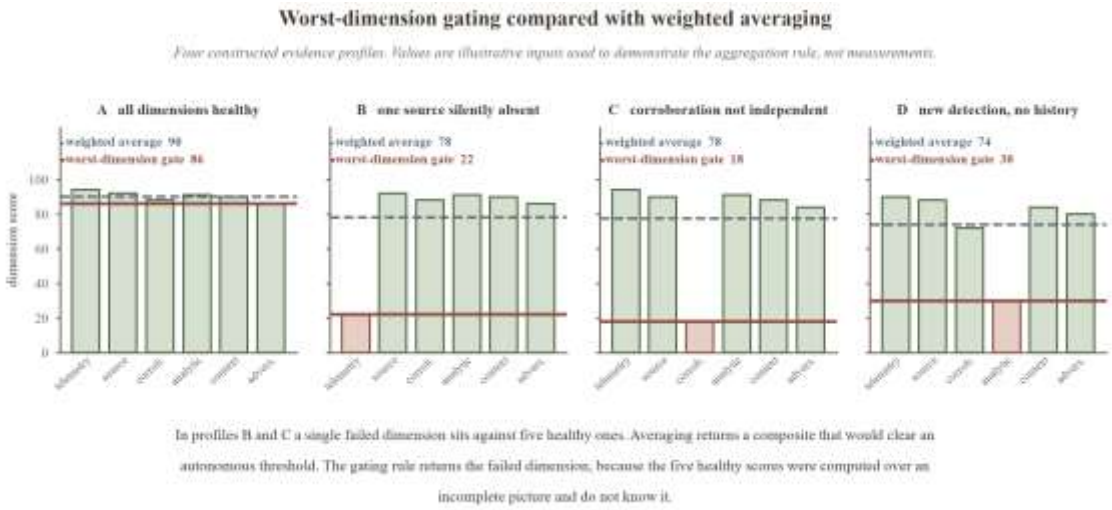

Fig 3: Worst-dimension gating compared with weighted averaging, across four illustrative evidence profiles. Profiles B and C carry a single failed dimension against five healthy ones. Averaging returns a composite that would clear an autonomous threshold; the gating rule returns the failed dimension's value. The composite values shown are illustrative and are used to demonstrate the aggregation rule, not to report measurements.

4.3. Action risk is a separate axis

Action risk is classified independently of confidence, and it is a property of the proposed response rather than of the evidence. Four attributes determine it: reversibility, blast radius, business criticality of the affected service, and whether the action carries legal, safety, or regulatory consequence.

Table 3: Action-risk classes and the maximum authorization available to each. The final column is a ceiling and not a target. An R4 action has no confidence score that promotes it into the autonomous lane, which is the paper's central design claim.

Class	Reversibility	Blast radius	Examples	Maximum authorization available
R1, observational	Fully reversible, no state change	None	Increase logging, capture memory, snapshot state, add watchlist entry	Autonomous
R2, bounded containment	Reversible within a defined window	Single entity	Suspend one session, quarantine one endpoint, expire one	Autonomous, bounded and time-limited

			token	
R3, service-affecting	Reversible with operational effort	Single service or account	Disable one service account, block one integration, isolate one workload	Human review required
R4, broad or irreversible	Not reversible, or reversible only through recovery	Multiple services, tenants or business processes	Enterprise credential rotation, large network-segment blocks, resource deletion, revocation of emergency access	Human review with second approver, never autonomous

4.4. The arbitration rule

The policy decision point receives a confidence composite and an action-risk class and applies an asymmetric rule.

Confidence determines what may be *considered*. Action risk determines what may be *done*. The authorization granted is the lower of the two, always, and the asymmetry is that no confidence value can raise the ceiling set by the action-risk class.

The practical consequence is that the interesting cell of the matrix is the top right: very high confidence attached to an R4 action. Under a single-dial design that case reads as maximum urgency and maximum certainty, which is precisely the reasoning that produces a self-inflicted outage. Under this rule it reads as a case for immediate expedited human review with a fully prepared evidence package and a recommended playbook, which is a materially different and considerably safer response to the same situation.

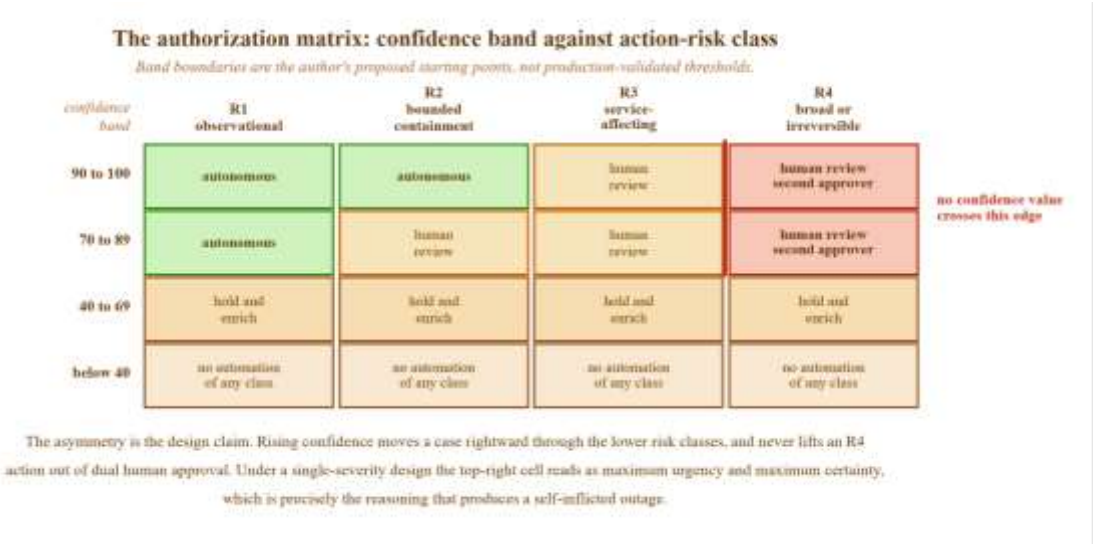

Fig 4: The authorization matrix. Confidence bands on the vertical axis, action-risk class on the horizontal, and the resulting authorization in each cell. The step pattern along the top edge is the asymmetry: increasing confidence moves a case rightward through the lower risk classes but never lifts an R4 action out of dual human approval. Band boundaries are illustrative proposed starting points, not production-validated thresholds.

Table 4 states the illustrative bands from which the matrix is drawn.

Table 4: Illustrative confidence bands. These values are the author's proposed starting points and are not production-validated thresholds. They are stated so the arbitration rule can be shown concretely; an implementing organization would calibrate them against its own measured analytic precision before permitting any autonomous class.

Band	Interpretation	What it authorizes for consideration	Secondary effect
90 to 100	Evidence complete, corroborated, from healthy verified sources, analytic profile good	Bounded, reversible automatic action where action-risk class permits and all policy conditions hold	None
70 to 89	Substantially supported with a known weakness	Expedited human review with recommended playbook and complete evidence package	None
40 to 69	Materially incomplete or weakly corroborated	Hold, enrich, increase observation; no disruptive response	Telemetry or enrichment request raised
Below 40	Not trustworthy for automation	No response automation of any class	Detection-quality or

			telemetry-quality issue opened; evidence retained

4.5. Stopping a confident but wrong detection

Confidence gating alone is insufficient, and the architecture does not rely on it alone. Authorization additionally requires action-specific controls that operate regardless of score: deny lists for critical assets and identities; two-person approval for destructive actions; precondition checks evaluated at execution time rather than at decision time; blast-radius limits; rate limits; canary execution; reversible containment applied before eradication; independent post-action validation; automatic rollback; and an immediate kill switch that does not depend on the agent.

The precondition check deserves emphasis because it addresses a specific timing gap. Confidence is assessed on evidence as of the moment of assessment, and execution happens some time later. Preconditions are re-evaluated at the tool broker immediately before the action, so that an authorization granted against a state that has since changed does not execute against the new state.

5. Failure modes and adversarial resistance

5.1. Where confidence it cannot be trusted

The architecture degrades when the confidence inputs are weak, correlated, manipulated, or unavailable.

Sparse history is the simplest case. A newly deployed detection has no validated precision, so its analytic-reliability dimension is low by construction, and under worst-dimension gating it is correctly ineligible for autonomy until it accumulates a characterized profile. This is intended behavior and not a limitation, though it does mean the architecture is conservative precisely when a novel threat first appears, which is a real operational cost that should be acknowledged rather than argued away.

Rapid environmental change degrades contextual fit, because baselines describe an environment that no longer exists. Migrations, reorganizations, and major deployments all produce this. Rare but legitimate behavior produces the same effect from the opposite direction, and it is the mechanism behind the benign triggers reported by Alahmadi et al. [1].

Miscalibration is the deepest problem and it is why Guo et al. [9] is cited as a design constraint rather than as background. A confidence number that is systematically overconfident will authorize actions at a rate its scale does not warrant, and the failure is silent because the score looks the same either way.

5.2. Adversaries who game the signal

An adversary aware of this architecture has four available strategies and each maps to a dimension.

- Manufacturing corroboration: Generate benign-looking activity across several sources so that a subsequent malicious action appears corroborated. This is the attack on the corroboration dimension, and it is the reason independence must be assessed structurally rather than counted.
- Poisoning baselines: Establish malicious behavior gradually as normal so that contextual fit scores it as unremarkable. This is slow and it works.
- Suppressing contradictory evidence: Disable or degrade the sources that would lower confidence, or that would raise it for the correct conclusion.
- Compromising enrichment: Attack the trusted context sources rather than the detection, so that the confidence engine is reasoning over adversary-supplied context while every source appears healthy.

The correlated-source problem underlies three of these and deserves separate statement. Corroboration is only meaningful if the sources are genuinely independent, and in cloud environments they very often are not. Identity logs, control-plane logs, and resource logs may all be emitted by, routed through, or retained within a single provider control plane. An adversary with administrative control of that plane can influence all three while each remains individually well-formed, authentic in appearance, and delivered on time. The confidence engine sees three corroborating sources and scores accordingly.

The architectural response is that corroboration must be scored on the *shared-dependency structure* of the sources rather than on their count. Two signals that traverse a common control plane count as one for corroboration purposes. Dual-path verification for critical signals, as specified in the prior telemetry work [18], is the mechanism that makes genuine independence available for the detections that most need it.

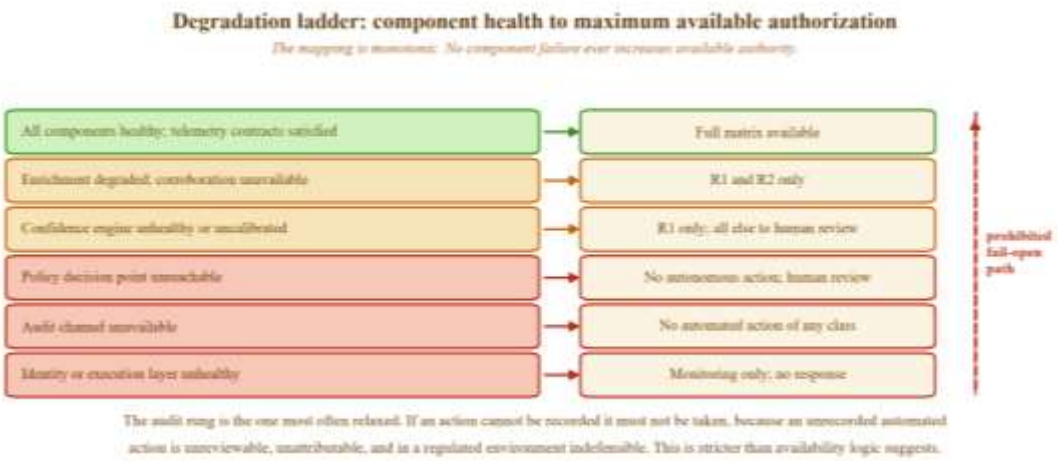

Fig 5: Degradation ladder and fail-safe states. Each component health condition maps to a defined maximum authorization, and the mapping is monotonic: no component failure ever increases available authority. The dashed transitions are the prohibited fail-open paths, shown to make explicit which transitions an implementation must be tested against.

5.3. Failing safe

If the confidence engine, policy decision point, identity system, audit channel, or execution layer is unhealthy, disruptive autonomy stops and cases fall back to human review or non-invasive monitoring.

The audit channel is the one most often overlooked. If actions cannot be recorded, they must not be taken, because an unrecorded automated action is unreviewable, unattributable, and in a regulated environment indefensible. This is a stricter rule than availability logic would suggest and it is deliberate. Alignment with the incident-response lifecycle and its governance outcomes follows NIST SP 800-61r3 [10], and the agent-oversight vocabulary follows the AI risk management framework [17].

5.4. Actions that should not be automated at any confidence

Some actions are too consequential for full automation regardless of evidence quality: deleting cloud resources; disabling high-impact business services; revoking privileged emergency or break-glass access; rotating enterprise-wide credentials; blocking large network segments; and any action with legal, safety, or regulatory consequence.

Revoking break-glass access deserves specific mention, because it is the action most likely to look correct to a confidence engine during an incident and most likely to be catastrophic. Emergency access is used rarely, by highly privileged identities, often outside normal patterns, and frequently during exactly the conditions that generate detections. It is close to a worst case for contextual fit. It is also the access path an organization depends on to recover, so an automated revocation during a live incident can remove the means of response. It is permanently classified R4.

6. Limits

Several limits are structural rather than incidental.

- The architecture is specified, not measured: No component of the integrated loop has been evaluated in a controlled production setting by the author. Every quantitative statement in this paper is either a proposed starting point or a property of the design, and none is an observation.
- Confidence has an irreducible floor: The six dimensions reduce dependence on any single signal but do not eliminate the possibility that all six are wrong together, which is exactly what a sufficiently resourced adversary with control-plane access can arrange.
- Governance depends on completeness: The Codex assumes detections are enrolled in it. Detections created outside the process, and they will be, hold no calibration record and must therefore default to advisory. Whether an organization can sustain that discipline under operational pressure is an open question that the architecture cannot answer for it.
- The policy layer is operational, not formal: Wu et al. [15] give guardrails a formal security treatment. This architecture does not carry equivalent formal guarantees, and formalizing the arbitration rule and its interaction with the tool broker's precondition checks is identified future work rather than a claim.
- Conservatism has a cost: Worst-dimension gating combined with the asymmetric arbitration rule will hold cases that a more permissive design would have actioned correctly. Some of those held cases will be real incidents that were contained more slowly as a result. This is an accepted trade and should be stated as such rather than presented as a free improvement.

7. Evaluation framework

No production measurements exist for the integrated architecture, so the paper offers the measurement design rather than results. Table 5 gives the instrument.

Table 5. Proposed evaluation framework. The baseline column matters: six of these ten metrics are meaningless without a pre-implementation measurement, which is the practical reason retrofitted evaluations of SOC automation so often report unfalsifiable improvements.

Metric	Definition	Purpose	Baseline required
Detection-to-containment, median and p95	Elapsed time from detection firing to containment achieved	Establishes whether gated autonomy improves speed at all	Yes, pre-implementation
Disposition distribution	Share of detections automatically actioned, escalated, held, rejected, rolled back	Shows where the gate is actually sitting	Yes
False-action rate	Automated actions taken against events later confirmed benign	The primary safety metric	Yes
Business-impact events from automation	Automated actions causing service disruption	The metric that decides whether autonomy is expanded or withdrawn	Yes
Confidence calibration error	Divergence between assigned confidence band and observed correctness within that band	Detects overconfidence before it authorizes wrongly	No, generated by the loop
Codex compliance rate	Share of active detections holding a current calibration record and authorization class	Measures governance coverage, not detection quality	Yes
Telemetry contract satisfaction	Share of decisions made with all mandatory signals present and fresh	Bounds how much of the estate is autonomy-eligible at all	Yes
Rollback success rate	Attempted rollbacks that fully restore prior state	Validates the reversibility assumption behind R1 and R2	No
Analyst minutes per case	Human time consumed per escalated case	The efficiency claim, separate from the safety claim	Yes, pre-implementation
Held-case resolution time	Time for held cases to reach a disposition	Detects the failure where holding becomes a queue	No

Validation should proceed in four stages rather than one: simulation against replayed historical detections with known outcomes; tabletop exercises against the arbitration rule specifically; purple-team testing directed at the confidence signal rather than at the detections; and only then a controlled production pilot restricted to R1 and R2 actions.

The purple-team stage is the one that cannot be skipped. Every other stage tests whether the architecture works when the evidence is honest. Section 5.2 is the argument that the evidence may not be, and only adversarial testing against the confidence engine itself examines that.

8. Conclusion

The argument of this paper is narrow and can be stated in one line. A detection firing is not permission to act, and the two questions that determine whether action is permitted, namely how strongly the evidence supports the conclusion and how much damage the action does if the conclusion is wrong, are separate questions that must be answered separately and then arbitrated.

The Detection Lifecycle Codex makes the first question answerable by giving a detection a governed lifecycle whose output is an authorization class supported by a calibration record. The Agentic Security Governance Framework makes the resulting authorization enforceable by constraining what the executing agent may do with it. The closed loop joins them and returns outcomes to both, so that calibration is continuous rather than a one-time exercise.

Two design decisions carry most of the weight, and both are easy to get wrong in the other direction. Confidence aggregates by worst-dimension gating rather than by averaging, because a single failed dimension means the remaining five are computing over an incomplete picture without knowing it. And the arbitration between confidence and action risk is asymmetric, because the alternative permits a sufficiently confident system to take an irreversible action, which is the specific failure that makes autonomous response unacceptable in regulated environments.

What remains is measurement. The architecture is specified and its failure modes are analyzed, and neither of those is evidence that it works. The evaluation framework in Section 7 states what a pilot would have to produce, including the baselines it would have to capture beforehand, and the staged validation path ends with adversarial testing directed at the confidence signal itself. Until that work is done, this is a design, and the paper is written to be read as one.

References

[1] B. A. Alahmadi, L. Axon, and I. Martinovic, "99% False Positives: A Qualitative Study of SOC Analysts' Perspectives on Security Alarms," in Proc. 31st USENIX Security Symposium, 2022, pp. 2783-2800.

[2] A. Mohsin, H. Janicke, A. Ibrahim, I. H. Sarker, and S. Camtepe, "A Unified Framework for Human AI Collaboration in Security Operations Centers with Trusted Autonomy," arXiv:2505.23397, 2025.

[3] J. Roy and S. K. Singh, "AgentSOC: A Multi-Layer Agentic AI Framework for Security Operations Automation," arXiv:2604.20134, 2026.

[4] M. H. Saju and A. Azim, "Toward Autonomous SOC Operations: End-to-End LLM Framework for Threat Detection, Query Generation, and Resolution in Security Operations," arXiv:2604.27321, 2026.

[5] R. Sahay, B. Eapen, W. Meng, M. R. Al Mamun, N. K. Dora, M. Sumasadan, S. K. Tetarave, and E. De La Cruz, "Policy-Guided Threat Hunting: An LLM enabled Framework with Splunk SOC Triage," arXiv:2603.23966, 2026.

[6] X. Cadet, A. V. Singh, H. Mamania, E. Koh, A. Fitts, D. Van Bruggen, S. Boboila, P. Chin, and A. Oprea, "Retrieval-Augmented LLMs for Security Incident Analysis," arXiv:2603.18196, 2026.

[7] S. J. Lazer, K. Aryal, M. Gupta, and E. Bertino, "A Survey of Agentic AI and Cybersecurity: Challenges, Opportunities and Use-case Prototypes," arXiv:2601.05293, 2026.

[8] S. Barbieri, A. L. R. Ferraz, and L. A. Pereira Junior, "PocketAgents: A Manifest-Driven Library of Autonomous Defense Agents," arXiv:2605.21694, 2026.

[9] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On Calibration of Modern Neural Networks," in Proc. 34th International Conference on Machine Learning (ICML), 2017, pp. 1321-1330. arXiv:1706.04599.

[10] National Institute of Standards and Technology, "Incident Response Recommendations and Considerations for Cybersecurity Risk Management: A CSF 2.0 Community Profile," NIST SP 800-61r3, Apr. 2025. DOI: 10.6028/NIST.SP.800-61r3.

[11] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero Trust Architecture," NIST SP 800-207, Aug. 2020. DOI: 10.6028/NIST.SP.800-207.

[12] B. E. Strom, A. Applebaum, D. P. Miller, K. C. Nickels, A. G. Pennington, and C. B. Thomas, "MITRE ATT&CK: Design and Philosophy," MITRE Technical Report MP180360, 2018, rev. 2020.

[13] U. Uchibeke, "Before the Tool Call: Deterministic Pre-Action Authorization for Autonomous AI Agents," arXiv:2603.20953, 2026.

[14] G. Zhu and C. Wang, "Intent-Governed Tool Authorization for AI Agents," arXiv:2606.22916, 2026.

[15] B. Wu, W. Zhang, K. Chen, H. Fang, and N. Yu, "Provably Secure Agent Guardrail," arXiv:2605.29251, 2026.

[16] S. V. Kodathala, "aiAuthZ: Off-Host, Identity-Bound Authorization for AI Agents," arXiv:2607.05518, 2026.

[17] National Institute of Standards and Technology, "Artificial Intelligence Risk Management Framework (AI RMF 1.0)," NIST AI 100-1, Jan. 2023. DOI: 10.6028/NIST.AI.100-1.

[18] P. Sharma, "Telemetry Engineering as a Security Architecture Discipline: A Design Framework for Detection-Ready Multi-Cloud Pipelines," manuscript in preparation, 2026.