Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Drift-Aware Governance-as-Code for Agentic Cloud Operations: Formalizing State Certifiability and Testing Dynamic Authority Control in an AWS Testbed

Albert Adusei Brobbey

albertab@primemeridianintl.com

Prime Meridian International

Jin Lang

Crestmont University

Brandon Taylor

Prime Meridian International

Nasr Omar

Hochschule Bremen

Research Article

Keywords: agentic AI, cloud operations, governance-as-code, infrastructure drift, runtime assurance, policy-as-code, state certifiability, AWS

Posted Date: October 8th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10519517/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Abstract

Context: AI agents increasingly execute consequential cloud-operations tasks through privileged tools, yet conventional policy-as-code evaluates requests against a representation of infrastructure that may have diverged from live state.

Objective: We develop and test drift-aware governance-as-code (DAGA), a runtime authority-control model in which declared–observed divergence, temporal staleness, and observation confidence determine whether ordinary agent authority remains certifiable.

Method: A sequential mixed-methods study combined practitioner interviews, operational drift telemetry, a randomized controlled AWS testbed experiment, and an independently seeded operational validation using real AWS control-plane calls on isolated synthetic resources. After frozen adjudication, the analysis included 11 practitioners, 140 telemetry episodes across 84 resources, 1,429 valid primary experimental runs, and 270 operational-validation runs balanced across six scenarios, three governance modes, three drift levels, and five replications.

Results: Interview evidence supported declared–observed divergence, staleness, observation integrity, non-certifiability, and recovery-only authority as distinct governance mechanisms, although pre-adjudication intercoder reliability was weak. Operational telemetry yielded 96 first-divergence events; the mean observation-adjusted drift index was 0.607, and 60 episodes crossed the frozen lockout threshold. In the confirmatory severe-drift stratum, policy-violation rates were 92.3% for unconstrained automation, 55.2% for static governance-as-code, and 25.0% for drift-aware governance. Drift-aware governance reduced violation risk relative to unconstrained automation (risk ratio 0.27, 95% CI 0.13–0.58) and produced a less precise reduction relative to static governance (risk ratio 0.45, 95% CI 0.20–1.04). A segmented model did not identify a discontinuity in violations at the prespecified lockout threshold (level-change odds ratio 0.88, 95% CI 0.52–1.47), indicating that the hard boundary is a defensible governance rule rather than an empirically established natural breakpoint. Sensitivity analyses showed nearly identical discrimination for the soft-OR, additive, and logistic indices (AUC approximately 0.70), although fixed numerical thresholds were not portable across scales. Under ±20% divergence perturbations, median governance-state agreement was 97.6%. In a frozen-rule post hoc replay of 4,211 recorded actions, a universal hard stop reduced unsafe-action exposure by 81.1% but preserved only 18.1% of recorded recovery actions; a recovery-only lock reduced unsafe exposure by 54.7% while preserving 83.8% of recovery actions. In the independent operational validation, DAGA reduced rule-defined policy violations from 48.9% under unconstrained operation to 5.6% (risk difference −43.3 percentage points; Holm-adjusted Fisher $p<5\times10^{-11}$) and unauthorized privileged actions from 8.9% to 1.1% (Holm-adjusted p=0.034), while false lockouts did not differ from the algorithmic fixed prospective-review proxy (6.7% versus 7.8%; p=1.00).

Conclusion: Dynamic authority control materially attenuated drift-related policy violations without increasing low-drift recovery time. Governance-as-code for agentic operations should therefore treat state certifiability as a runtime precondition. Non-certifiability is defined here as an institutional decision

rule imposed on continuous epistemic uncertainty, rather than as a naturally occurring phase transition, and hard lockout thresholds require local normative calibration.

Highlights

• Defines state certifiability as the evidentiary adequacy required to legitimate an authority decision.

• Combines 11 practitioner interviews, 140 telemetry episodes, and 1,429 AWS testbed runs.

• Under severe drift, dynamic governance reduced policy violations from 92.3% to 25.0% versus unconstrained automation.

• Recovery-only authority preserved operational success while restricting ordinary privileged actions.

• Trace replay exposed the safety–recoverability tradeoff that motivates a bounded recovery action set.

• A separately seeded 270-run AWS validation reproduced the primary safety advantage using real control-plane actions.

• The prespecified hard lockout threshold was not an empirically detectable natural discontinuity.

1. Introduction

Large-language-model agents are moving from advisory roles into operational control loops. In cloud environments, an agent may diagnose a degraded service, inspect configuration, scale a workload, rotate a credential, alter an identity policy, roll back a deployment, or isolate a compromised resource. These capabilities compress detection and recovery time, but they also combine stochastic reasoning with powerful, state-changing tools. The resulting governance problem is not merely whether the model can produce an appropriate plan. It is whether the organization can certify, at the moment of action, that the agent's authority is appropriate for the system that actually exists.

Policy-as-code and infrastructure-as-code provide an attractive response because they encode constraints in machine-readable form. Open Policy Agent, for example, separates policy decision-making from enforcement and evaluates structured inputs against declarative rules [1]. AWS CloudFormation likewise compares expected resource properties with actual properties to identify drift [2], while AWS Config records resource configurations and relationships over time [3]. These mechanisms make governance executable and auditable. However, they share an underexamined epistemic dependency: policy evaluation is only as valid as the state representation supplied to the policy engine.

When declared state and observed state diverge, a formally correct rule can return an operationally unsafe decision. A resource tagged as nonproduction may have acquired a production dependency; an ostensibly private workload may have public ingress; an IAM binding may have changed outside the infrastructure-as-code path; or the configuration recorder itself may have degraded. AWS documentation

explicitly notes that a drifted configuration recorder can produce false-positive or false-negative compliance results [4]. The problem is therefore deeper than policy noncompliance. Sufficient drift can invalidate the classification inputs on which compliance is computed.

Research on infrastructure-as-code has documented defects, security smells, and maintenance challenges [5–8], while research on self-adaptive systems argues that assurance must continue at runtime because knowledge available at design time is incomplete [9–13]. Emerging work on AI-agent governance similarly emphasizes runtime enforcement, execution paths, and context-aware access control [14–16]. Yet these streams do not provide an empirically tested rule for how infrastructure drift should change an agent's operational authority.

This study addresses that gap through a drift-aware governance-as-code model and a four-component empirical evaluation. Its central contribution is deliberately singular: governance-as-code requires not only correct policies but certifiable policy inputs; consequently, agent authority should be a runtime-controlled variable conditional on the epistemic adequacy of the observed infrastructure state. The drift index, ordinal authority mapping, recovery-only action set, and asymmetric transitions are mechanisms that operationalize this claim rather than separate coequal innovations.

The study provides mixed but coherent evidence. Dynamic governance substantially reduces policy violations as drift rises, and the result is robust to reasonable changes in the drift aggregator's ranking and to bounded perturbations of the divergence component. However, the proposed hard lockout threshold is not a natural statistical discontinuity. We therefore define non-certifiability throughout as an institutional tolerance rule applied to continuous uncertainty: the organization declines to legitimate ordinary authority once evidentiary adequacy falls below a prespecified standard.

2. Related work and theoretical development

2.1 Infrastructure-as-code, policy-as-code, and configuration drift

Infrastructure-as-code represents intended operational state in versioned, machine-readable artifacts. Empirical studies show that configuration scripts can contain defects with availability, confidentiality, and integrity consequences [5, 6], and industry interviews identify limited tool support and evolving practices as continuing adoption challenges [7]. More recent policy-as-code research shows that executable policies serve heterogeneous purposes, including security, compliance, configuration control, and documentation, and require ongoing maintenance [8].

Drift is commonly defined operationally as a difference between expected and actual resource properties [2]. Existing platforms can detect such differences and trigger remediation, but a binary drift flag is too coarse for agent authority. It neither distinguishes a cosmetic label mismatch from an unexpected

privileged principal nor captures the risk created by a stale or impaired observation process. Our model therefore treats drift as a weighted, time-dependent, observation-conditioned construct.

2.2 Runtime assurance and self-adaptive systems

Models-at-runtime research argues that self-adaptive systems require continuous assurance because requirements satisfaction depends on changing context [9]. Perpetual-assurance work similarly treats evidence and assurance cases as evolving runtime objects [10]. Goal-oriented and control-theoretic approaches use monitored variables to adapt behavior under uncertainty [11–13]. This literature supplies the feedback-control logic for drift-aware governance: observe, compare, classify, constrain, and restore. We extend it by making decision rights, meaning the maximum authority available to an AI agent, the controlled variable.

2.3 Decision rights, institutional accountability, and noncompensability

Decision-rights theory distinguishes possession of technical capability from organizational authorization. For agentic operations, access credentials alone do not constitute legitimate authority; authority must be bounded by role, resource criticality, operational state, and accountable ownership. Institutional theory further suggests that enforceable rules acquire legitimacy through explicit ownership, auditability, and stable escalation paths. Accordingly, the proposed model assigns governance owners responsibility for attribute weights, thresholds, recovery-only actions, overrides, and post-incident review.

The central theoretical claim is noncompensability. Conventional risk scores are compensatory: a high value on one dimension can be offset by low values on others. That is unsuitable when observation integrity has deteriorated or when a critical veto is present. If the policy engine cannot certify that its inputs describe the live system, no favorable score or historical success rate should restore ordinary authority. This claim creates a boundary between risk calibration and epistemic validity.

2.4 Construct identity: state certifiability

State certifiability is the degree to which the organization possesses current, sufficiently complete, and independently trustworthy evidence that the governance-relevant representation supplied to the policy engine corresponds to the operational state on which an agent would act. It is relational: it concerns a representation, a contemplated action, an evidentiary basis, and an accountable organization. It is not a claim that the operational state is desirable, safe, or compliant. Table 1 distinguishes state certifiability from adjacent constructs, and Table 2 traces the theoretical lineage and operational role of the model constructs.

Table 1
Discriminant definition of state certifiability.

Construct	Question answered	Why it is not certifiability
Correctness	Is the represented state technically accurate or desirable?	A correct state may still lack independently adequate evidence.
Compliance	Does the state or action satisfy a policy?	Compliance presupposes valid policy inputs.
Observability	Do relevant signals exist?	Signals may exist but be stale, incomplete, correlated, or untrusted.
Measurement confidence	How uncertain is a particular observation?	Certifiability concerns the adequacy of the complete evidence package for authority.
Configuration integrity	Has configuration been protected from unauthorized change?	Integrity does not establish currency, completeness, or correspondence.
Drift	How far do declared and observed state diverge?	Drift is one input to certifiability, not the legitimating judgment itself.
State certifiability	Is the evidence adequate to legitimate this authority decision now?	Institutional decision rule integrating evidence and tolerance.

Table 2
Theoretical lineage and operational role of the model constructs.

Construct	Theoretical anchor	Operational implication
Measured divergence (δ)	Models at runtime; configuration management	Compare governance-relevant declared and observed attributes.
Temporal staleness (ρ)	Reliability/survival theory	Discount confidence as time since verified reconciliation increases.
Observation confidence (q)	Measurement theory; runtime assurance	Degraded scanners, latency, and missing signals increase effective drift.
Non-certifiability	Epistemic uncertainty; noncompensatory risk	Ordinary policy outputs are invalid when state classification cannot be trusted.
Authority ceiling	Decision-rights theory	Separate technical capability from permissible action.
Recovery-only authority	Resilience engineering	Allow bounded restoration actions while ordinary actions remain locked.
Override accountability	Institutional theory	Require named authorization, scope, evidence, and audit trail.

3. Formal model

3.1 Governance-relevant state and measured divergence

Let A denote the frozen set of governance-relevant attributes: environment classification, ingress exposure, IAM bindings, encryption status, criticality, region, change-window status, dependency graph, monitoring coverage, and rollback availability. Let $S^{\text{decl}}_t(a)$ and $S^{\text{obs}}_t(a)$ be the declared and observed values for attribute a at time t, and let w_a be a governance-owner weight. A directional discrepancy cost $c_a \in [0,1]$ represents the consequence of the observed deviation.

$$\delta_t = [\Sigma_{a \in A} w_a \, c_a(S^{\text{decl}}_t(a), S^{\text{obs}}_t(a))] / \Sigma_{a \in A} w_a \quad (1)$$

Unlike an unweighted mismatch count, Eq. 1 permits unexpected public exposure or privileged IAM bindings to dominate cosmetic deviations. Critical-veto attributes remain noncompensatory and can independently force restriction.

3.2 Temporal staleness and observation-adjusted drift

A last-known aligned state becomes less informative as time passes. Let τ_t be hours since the last verified reconciliation and κ_r the divergence hazard for resource class r.

$$\rho_t = 1 - \exp(-\kappa_r \tau_t) \quad (2)$$

Let $q_t \in [0,1]$ be observation confidence derived from signal coverage, scanner health, latency, API errors, and tamper indicators. The combined observation-adjusted drift index is:

$$\Phi_t = 1 - (1 - \delta_t)(1 - \rho_t)q_t \quad (3)$$

Equation 3 is a normalized noisy-OR over three failure complements: absence of measured divergence $(1 - \delta)$, absence of temporal staleness $(1 - \rho)$, and adequate observation q. It is the unique multiplicative complement under three explicit modeling assumptions: the components are normalized to [0,1]; each component can independently defeat evidentiary adequacy; and joint adequacy is represented by the product of component adequacies. These assumptions do not assert statistical independence of the underlying sensors. They define a transparent engineering aggregator whose interaction is conservative: deterioration in any component reduces the residual protection supplied by the others.

Proposition 1

(boundedness and monotonicity of Φ). For $\delta,\rho,q \in [0,1]$, $\Phi \in [0,1]$; Φ is weakly increasing in δ and ρ and weakly decreasing in q. Proof: $(1 - \delta)(1 - \rho)q \in [0,1]$, establishing boundedness. The partial derivatives are $\partial\Phi/\partial\delta = (1 - \rho)q \geq 0$, $\partial\Phi/\partial\rho = (1 - \delta)q \geq 0$, and $\partial\Phi/\partial q = -(1 - \delta)(1 - \rho) \leq 0$. Moreover, $\Phi = 0$ only when $\delta = \rho = 0$ and $q = 1$; $\Phi = 1$ when $\delta = 1$, $\rho = 1$, or $q = 0$. The negative cross-partials with q express a substitutability limit: improving observation confidence has less protective effect once divergence or staleness is already extreme.

The soft-OR is proposed, not claimed to be uniquely true. A weighted additive index, maximum-component rule, logistic transform, Bayesian belief state, or partially observable Markov model may be preferable when calibrated data and stronger dependence assumptions are available. Section 6.5

therefore compares alternative aggregators and distinguishes ranking robustness from threshold portability.

3.3 Authority states and asymmetric transitions

Let D⃗∈{C0,C1,C2,C3,C4} be the maximum certified authority, ordered from no ordinary action to high-impact autonomous execution. These tiers are ordinal; the distance from C4 to C3 is not assumed equal to the distance from C2 to C1. We therefore replace a linear floor discount with a task- and resource-specific monotone mapping $f_{r,u}$ governed by ordered thresholds $\lambda_{r,u},1\le\ldots\le\lambda_{r,u},4$:

d⃗⃗ = $f_{r,u}$(Φ⃗) ∈ {C0,…,C4}, with Φ'≥Φ ⇒ $f_{r,u}$(Φ')≤$f_{r,u}$(Φ) (4)

For a C4-capable task, an illustrative mapping is C4 below λ1, C3 on [λ1,λ2), C2 on [λ2,λ3), C1 on [λ3,λ4), and C0 at or above λ4. Lower-risk or lower-capability tasks may begin below C4 and use fewer transitions. Thresholds are policy parameters informed by validation data, criticality, reversibility, blast radius, and false-lockout cost; they are not arithmetic cut points implied by the labels.

Three runtime governance states are defined using frozen thresholds θtrigger = 0.30, θlock = 0.65, and θpromote = 0.15:

Z⃗ = NORMAL if Φ⃗<θtrigger; RESTRICT if θtrigger ≤ Φ⃗<θlock; LOCKOUT if Φ⃗≥θlock (5)

Inside RESTRICT, both channels operate: $f_{r,u}$(Φ) lowers the evidence-based ceiling continuously through ordinal transitions, and the ordinary task gates still apply. At θlock, the discrete institutional rule overrides ordinary authority. The final decision is:

D⃗ = min≼{d⃗, $f_{r,u}$(Φ⃗), g_1(x⃗),…,gM(x⃗)} if Φ⃗<θlock; D⃗=Dlock otherwise (6)

Here min≼ denotes the minimum under the authority ordering, not numeric subtraction. Proposition 2 (authority monotonicity). Holding d⃗ and gate outputs fixed, D⃗ is weakly nonincreasing in Φ⃗. Proof: $f_{r,u}$ is weakly nonincreasing by construction; the minimum of fixed ceilings and a weakly nonincreasing ceiling cannot increase. Crossing θlock replaces the ordinary minimum with Dlock, which is no greater than any ordinary-action tier.

3.4 Recovery-only authority

Let $\mathcal{A}$ be the full enumerated tool-action set. Define the recovery-only subset $\mathcal{A}$lock⊂$\mathcal{A}$ as {read_status, isolate_workload, revoke_or_rotate_compromised_credential, execute_verified_rollback, reconcile_config, request_human_authorization}. Membership is conditional on an action-specific precondition, target allowlist, maximum blast radius, and immutable audit emission. Under LOCKOUT, an action a is executable only if a∈$\mathcal{A}$lock and all recovery gates return allow; otherwise it is denied. Dlock therefore does not mean unrestricted C1 authority or complete shutdown. It is a purpose-limited authorization to restore certifiability.

3.5 Promotion with performance and harm evidence

Drift alignment is necessary but insufficient for promotion. Let n be the number of eligible validation runs in the current task–resource regime, s the number of successful safe runs, h the number of harmful runs, Ls the one-sided lower confidence bound on success probability, Uh the one-sided upper confidence bound on harm probability, πs the minimum acceptable success rate, and πh the maximum acceptable harm rate. A one-tier promotion is eligible only when:

$D_{t+1} \le D_t + 1$ iff $\Phi_t \le \theta_{promote} \wedge q_t \ge q_{min} \wedge V_t = 0 \wedge n \ge n_{min} \wedge L_s(s,n;\alpha) \ge \pi_s \wedge U_h(h,n;\alpha) \le \pi_h \wedge A_t = 1$ (7)

The confidence bounds may use prespecified Wilson or exact binomial bounds. Evidence is regime-specific: runs collected under another task, resource class, policy version, or materially different drift distribution do not automatically transport. This rule prevents current alignment alone from licensing expanded authority when the agent has not demonstrated sufficiently precise safe performance.

Worked RESTRICT example. Suppose a credential-rotation task has base ceiling C3, Φ = 0.48, θlock = 0.65, and task thresholds λ1 = 0.20, λ2 = 0.40, λ3 = 0.55. The ordinal mapping returns C1. If the privilege gate returns C2 and the blast-radius gate C3, Eq. 6 yields $\min_{\preccurlyeq}\{C3,C1,C2,C3\} = C1$. The action may proceed only through the C1-compatible bounded rotation path. If Φ rises to 0.67, f no longer controls the result: the institutional lockout rule activates Dlock, and only a qualifying action in $\mathcal{A}$lock may execute.

This asymmetry implements a fail-safe principle: restriction can be automatic and immediate, whereas expanded authority is incremental, evidence-bounded, regime-specific, and institutionally authorized. Figure 1 summarizes the drift-aware runtime authority model and its asymmetric transitions.

4. Research questions and hypotheses

The empirical study addressed four research questions: RQ1, do practitioners recognize the proposed drift and certifiability constructs as distinct operational mechanisms? RQ2, how quickly do governance-relevant resource classes diverge after verified reconciliation? RQ3, does drift-aware governance reduce unsafe agent actions compared with static governance and unconstrained automation? RQ4, does the prespecified lockout threshold correspond to an observable discontinuity in policy-violation risk? Table 3 states the prespecified hypotheses.

Table 3
Prespecified hypotheses.

Hypothesis	Prediction
H1	Policy violations rise with injected drift under unconstrained and static governance but are attenuated under drift-aware governance.
H2	Under severe drift, drift-aware governance produces fewer critical safety events and missed lockouts.
H3	Under no/low drift, drift-aware governance does not materially increase time to safe recovery relative to static governance.
H4	Recovery-only authority preserves safe recovery better than complete operational failure.
H5	Lower observation confidence increases Φ and the probability of restriction or lockout.
H6	Resource classes exhibit different divergence hazards κ_r.
H7	Violation risk shows a positive discontinuity at θlock = 0.65.

5. Methods

5.1 Study design

We used a sequential mixed-methods design with four linked components. Practitioner interviews evaluated construct meaning and operational plausibility. Drift and reconciliation telemetry quantified divergence dynamics and resource-class hazard. A randomized controlled AWS testbed experiment compared governance modes under injected drift. A separately seeded operational validation then executed bounded AWS control-plane actions on isolated synthetic resources. No customer or production data were used. Training, validation, and test partitions were assigned before confirmatory analysis; the operational-validation manifest and thresholds were frozen before execution. Table 4 summarizes the linked study components, raw records, adjudicated analytic samples, and purposes.

Table 4
Study components and adjudicated samples.

Component	Raw records	Primary adjudicated analytic set	Purpose
Practitioner interviews	15 unique participants; 159 segments; 231 coding records	11 participants; 69 segments; 109 final coding records	Construct elicitation, plausibility, and negative cases
Operational telemetry	86 resource rows; 155 drift episodes	84 resources; 140 drift episodes; 96 first divergences	Estimate Φ states and class-specific κ
AWS experiment	1,440 intended unique runs; 1 duplicate/conflicted ID	1,429 valid runs; 341 confirmatory test runs	Compare governance modes and test thresholds
Independent AWS validation	270 frozen runs; 45 full-scenario repair reruns	270 balanced analysis records; zero API or cleanup failures	Test operational feasibility on real AWS control-plane actions

5.2 Practitioner interviews

Participants were recruited purposively across site reliability engineering, cloud architecture, DevOps, IAM, cloud security, technology risk, and compliance roles. The interview guide examined declared–observed divergence, stale reconciliation, observation integrity, non-certifiability, lockout, recovery-only actions, false-lockout costs, promotion, and accountability. Participant identifiers were separated from contact records. Three withdrawn participants and one additional participant excluded by the principal investigator's frozen quality decision were removed from construct analysis.

Segments were coded using a deductive codebook with allowance for negative cases. Two coders contributed 231 records. Final thematic summaries retained valid confidence values and records marked Agreed or Disagreement resolved. Pre-adjudication reliability was estimated on the 28 shared, eligible segments using code-specific binary Cohen κ; because multiple codes could be assigned to a segment, κ was calculated separately for each code and summarized descriptively. The median κ was − 0.08, indicating poor independent agreement. We therefore name this component practitioner construct elicitation and plausibility assessment. It surfaces practitioner language, tensions, and candidate mechanisms; it does not validate discriminant construct structure or establish consensus.

5.3 Operational telemetry

Telemetry covered reconciliation events, attribute-level comparisons, observation-integrity assessments, drift episodes, remediation, and incident outcomes. Configuration values were hashed and resource identifiers pseudonymized. For each episode we recomputed Φ from the recorded δ, ρ, and q rather than relying on workbook display formulas. Episodes with adjudicated clock skew, unresolved veto status, or excluded identifiers were removed under frozen rules.

Attribute weights and directional discrepancy costs were protocol-specified engineering judgments frozen in the attribute-definition register before the confirmatory partition was examined. Critical security attributes received larger weights and could carry veto status. They were not estimated as population preferences and should not be interpreted as universal values. We evaluated aggregate weight uncertainty by perturbing each episode's divergence contribution independently by ± 20% across 1,000 deterministic-seed draws and recomputing governance states and predictive discrimination.

For resource-class divergence, time zero was the last verified alignment. First governance-relevant divergence was the event; episodes without recorded divergence were censored at evaluation. We estimated an exponential hazard κ_r as events divided by person-hours and reported the implied median time $\ln(2)/\kappa_r$. These estimates are descriptive because some resources contributed recurrent episodes and the modest event count did not support a fully adjusted recurrent-event model.

The constant-hazard assumption was checked descriptively by comparing Kaplan–Meier survival estimates with fitted exponential survival at event times and by contrasting early and late crude hazards around the median observed time. The diagnostics were not treated as a formal goodness-of-fit test because of censoring, recurrence, and small class-specific samples.

5.4 AWS controlled experiment

The factorial design crossed three governance modes, four drift levels, six tasks, two environment classes, and scenario-specific criticality. Modes were: (1) unconstrained automation; (2) static governance-as-code, which evaluated fixed policy but did not alter authority using Φ; and (3) drift-aware dynamic governance, which computed Φ, restricted the authority ceiling, and activated recovery-only behavior under lockout. Drift levels were none, low, moderate, and severe. Tasks were service scaling, deployment rollback, credential rotation, IAM modification, configuration reconciliation, and security isolation.

The run registry recorded randomization seed, replication, data partition, agent model/version, prompt version, policy version, testbed version, timestamps, status, and operator. Each run was bounded to 20 tool-using steps and 900 seconds. The action allowlist prevented access outside the sandbox. The primary safety outcome was any policy violation; critical safety events combined critical policy violations and unauthorized privileged actions. Operational success required completion, task success, safe recovery, and no critical safety event.

Recorded implementation versions were agent-v1/agent-v2, agent versions 2.4.0/2.4.1, prompts p1.0/p1.1, policies 2.0/2.1, and testbed tb-1.0. The harness stored a per-run randomization seed, but seeds were not reused as paired identical trajectories across governance modes. Model reasoning generated intents and tool requests; drift calculation, threshold comparison, policy evaluation, allowlist enforcement, timeout handling, and outcome derivation were deterministic harness functions. Exact AWS region, service topology, foundation-model identity, tool JSON schemas, prompt bodies, policy

source, teardown script, and cost ceiling must accompany a computational submission in a security-reviewed supplement.

One duplicated run identifier mapped to conflicting outcomes and was quarantined in full. Runs explicitly excluded by the adjudication panel were removed for data-integrity reasons. Timeouts and aborted-error runs were retained in the intention-to-treat operational-success outcome rather than silently dropped. Completed-run analyses were used as sensitivity analyses.

5.5 Statistical analysis

Confirmatory inference used the held-out test partition. We calculated proportions with Wilson 95% confidence intervals and prespecified severe-drift risk ratios with log-scale confidence intervals. Descriptive analyses used all valid randomized runs and completed runs. Because sparse cells and zero events caused separation in fully interacted logistic models, the paper emphasizes transparent stratum-specific risks and risk ratios rather than unstable maximum-likelihood odds ratios. This choice was made after detecting separation and is reported as an analytic limitation.

To examine H7, we fitted a segmented logistic model across valid runs with centered Φ, an indicator for $\Phi \geq 0.65$, an above-threshold slope term, governance mode, and scenario indicators. The coefficient of the indicator represents a level discontinuity at the frozen threshold. This analysis tests whether the threshold is an empirical breakpoint; it does not test whether lockout is normatively prudent.

Functional-form sensitivity compared the soft-OR with three alternatives: an equally weighted additive index, a maximum-component index, and a logistic transform of the additive index. Because these functions have different scales, we compared rank correlation and area under the receiver-operating-characteristic curve for policy violations. We also report the proportion crossing the same numerical thresholds to demonstrate why thresholds cannot be transported mechanically across aggregators.

No imputation was used for missing primary outcomes. Missing drift level in the no-injection condition was recoded as None. All analyses were performed on copies of immutable raw workbooks with deterministic exclusions derived from quality logs.

5.6 Post hoc counterfactual policy replay

To probe the mechanisms bundled inside the randomized drift-aware treatment, we conducted a frozen-rule, post hoc replay of recorded action requests. This analysis was specified after the package experiment and is therefore not a randomized component ablation. It evaluates whether each observed action would have remained available under four counterfactual policies while holding the recorded action request and contemporaneous drift assessment fixed. It does not simulate how a denied action would have altered later system state, agent reasoning, subsequent actions, recovery, or final task outcome.

The replay included valid experimental runs from the adjudicated analysis-ready file, deduplicated action identifiers, an observed combined drift score, and a mappable requested-authority class. Authority

classes C0–C4 were represented as an ordered set. The frozen ordinal ceiling was C4 for $\Phi < 0.15$, C3 for $0.15 \le \Phi < 0.30$, C2 for $0.30 \le \Phi < 0.50$, and C1 otherwise. A hard-lock condition was $\Phi \ge 0.65$ or a recorded critical veto. Table 5 defines the frozen policies used in the post hoc action-level replay.

Table 5
Frozen policies used in the post hoc action-level replay.

Policy	Decision rule	Mechanism isolated
P0 Visibility only	All recorded actions remain available.	Observation without enforcement
P1 Ordinal restriction	Permit only when requested authority does not exceed the Φ-dependent ceiling.	Continuous authority contraction
P2 Ordinal + hard stop	Apply P1; deny every action under the hard-lock condition.	Noncompensatory lockout
P3 Full recovery-only lock	Apply P1; under hard lock, permit only a recorded recovery-only action using a frozen recovery tool.	Fail-closed control with bounded restoration

The recovery whitelist comprised read_status, isolate_workload, rotate_credential, execute_rollback, reconcile_config, and request_approval. The primary replay metrics were unsafe-action exposure (an allowed action with a recorded policy violation or unauthorized privileged action), benign-action availability, and recovery-action availability. Confidence intervals were obtained from 2,000 nonparametric bootstrap resamples clustered by run identifier with seed 20260728. Labels such as “benign action” and “denied” are trace-based proxies, not counterfactual ground truth. The policy definitions, action-level decisions, formulas, and stratified summaries are supplied in the supplementary workbook.

5.7 Independent operational AWS validation

Before execution, we froze a 270-run manifest using master seed 2026072801. The balanced design crossed six operational scenarios, three governance modes (DAGA, an algorithmic fixed prospective-review proxy, and unconstrained operation), three drift levels, and five replications. Scenarios exercised parameter updates, Lambda configuration, secret rotation, IAM inline-policy changes, CloudWatch alarm changes, and isolated security-group ingress. Every permitted mutation targeted a transient resource created solely for the study and included a restoration or deletion check. The pre-execution manifest SHA-256 digest was fe5b3af860b5bdba4b5f5ceedb57988c92bba7848a0b70ea5aa55fdfa5629846.

Primary safety outcomes were policy violation and unauthorized privileged action; false lockout was the primary availability outcome. We compared proportions with Fisher exact tests and finite Haldane–Anscombe odds ratios when cells were sparse. Holm adjustment controlled the two primary safety contrasts. Kruskal–Wallis and Holm-adjusted Mann–Whitney tests assessed latency. A frozen threshold grid varied $\theta_{trigger}$ over {0.25, 0.30, 0.35} and θ_{lock} over {0.60, 0.65, 0.70}. The algorithmic fixed prospective-review proxy used prespecified error rates and did not involve observed human reviewing.

Audit correction. All 270 initial Lambda invocations returned, but 31 records in the Lambda-configuration scenario contained AWS API errors because the execution role omitted lambda:GetFunction, which the AWS completion waiter requires. The original archive was preserved unchanged. After adding only that permission, all 45 observations in the affected scenario, rather than only the failures, were rerun under new -R1 identifiers. The replacement series had zero API errors and zero cleanup failures; the analysis-ready set substitutes those 45 records and retains the other 225 originals.

The validation event chain was recorded as distinct fields. Seeded telemetry produced an exogenous scenario-level unsafe-intent indicator; the governance function then produced a state and authorization decision; an authorized request triggered the scenario-specific AWS API; api_success recorded whether AWS accepted and completed the requested operation; cleanup_success recorded restoration or deletion; and policy_violation was classified as unsafe intent combined with authorization. Consequently, the policy-violation contrast is a rule-conformance estimand that is mechanically affected by authorization, not an independent measure of realized cloud harm. AWS API success and restoration success provide implementation evidence, but the validation did not independently adjudicate downstream resource harm. Table 6 summarizes the independent-validation event chain and the evidentiary boundary for each recorded field.

Table 6
Independent-validation event chain and evidentiary boundary.

Event-chain stage	Recorded field	Interpretive boundary
Seeded scenario	unsafe_intent; telemetry components; Φ	Generated before the governance decision from the frozen seed.
Policy decision	governance_state; authorized	Determined whether a requested mutation could be attempted.
AWS execution	api_success; api_error; action latency	Observed response from the real AWS control-plane operation.
Resulting-state recovery	cleanup_success; recovery latency	Verified restoration or deletion; not a general harm assessment.
Outcome classification	policy_violation; unauthorized_privileged_action; false_lockout	Rule-defined from intent, authorization, and scenario privilege; not statistically independent of policy.

6. Results

6.1 Practitioner construct elicitation and plausibility

The final qualitative set comprised 109 coding records across 69 segments from 11 participants. Declared–observed divergence appeared in 13 segments from six participants. Stale reconciliation appeared in 12 segments from seven participants, and recovery-only authority appeared in 12 segments

from eight participants. Non-certifiability was represented in nine segments from seven participants. Observation integrity was less frequent but distributed across six participants, suggesting it was not confined to one specialist perspective. Table 7 reports the adjudicated practitioner coding summary.

Table 7
Adjudicated practitioner coding summary.

Construct	Coded records	Unique segments	Participants
Drift Declared Observed	13	13	6
Recovery Only	12	12	8
Stale Reconciliation	12	12	7
Blast Radius	10	10	6
Privilege	10	10	6
Non Certifiable	11	9	7
False Lockout	9	9	5
Lockout	7	7	6
Reversibility	7	7	4
Promotion	7	6	4
Obs Integrity	6	6	6
Accountability	5	5	5

The interviews also exposed tensions. False lockout was mentioned in nine segments, and promotion in six, indicating that practitioners did not equate safer governance with permanent restriction. Accountability appeared in five segments across five participants, supporting named override ownership but not establishing a dominant institutional model. The weak pre-adjudication median κ (− 0.08) shows that the constructs were not applied consistently by independent coders. Accordingly, RQ1 receives qualified support: the proposed mechanisms were recognizable and distributed across participants, but construct boundaries required substantial adjudication.

6.2 Operational drift dynamics

After adjudication, telemetry comprised 140 episodes across 84 resources and 96 first-divergence events. Mean Φ was 0.607. Thirteen episodes were classified NORMAL, 67 RESTRICT, and 60 LOCKOUT. The concentration above θtrigger reflects the drift-event sampling frame rather than the prevalence of drift across all infrastructure. Table 8 summarizes the observation-adjusted drift states, and Fig. 2 visualizes their distribution across the frozen governance states.

Table 8
Observation-adjusted drift states in operational telemetry.

State	Episodes	Mean Φ	Critical-veto rate	Remediated
LOCKOUT	60	0.726	20.0%	53.3%
NORMAL	13	0.240	7.7%	69.2%
RESTRICT	67	0.528	20.9%	64.2%

Divergence hazards differed by resource class. Serverless functions had the largest estimated hazard ($\kappa = 0.0408$ per hour; implied exponential median 17.0 hours), although only six episodes contributed. Network resources had $\kappa = 0.0174$ (median 39.9 hours), container clusters $\kappa = 0.0155$ (44.6 hours), databases $\kappa = 0.0145$ (47.7 hours), compute $\kappa = 0.0135$ (51.5 hours), and storage $\kappa = 0.0186$ (37.2 hours). The estimates support H6 descriptively, but their uncertainty and recurrent-event structure preclude ranking resource classes definitively. Table 9 reports the resource-class divergence hazards and their implied median times.

Table 9
Resource-class divergence hazards.

Resource class	Episodes	Events	Person-hours	κ/hour	Implied median (h)
Compute	9	6	446	0.0135	51.5
Container Cluster	34	23	1481	0.0155	44.6
Database	32	21	1444	0.0145	47.7
Network	47	35	2016	0.0174	39.9
Serverless Function	6	4	98	0.0408	17.0
Storage	12	7	376	0.0186	37.2

The exponential diagnostics did not support a strong constant-hazard interpretation. Mean absolute deviations between Kaplan–Meier and exponential survival ranged from 0.052 to 0.147 across classes, while late-to-early crude hazard ratios ranged from 0.16 to 0.49. These ratios suggest declining rather than constant hazard over the observed windows, potentially because episodes most susceptible to divergence failed earlier. We retain κ as a compact descriptive calibration parameter, not as a verified generative law.

6.3 Confirmatory AWS experiment

The cleaned experiment contained 1429 valid unique runs, of which 1267 completed, 105 timed out, and 57 ended with an adjudicated error. The held-out confirmatory test partition contained 341 runs. Figure 3 shows a monotonic rise in violations with injected drift for all modes, but the slope was markedly

attenuated under drift-aware governance. The corresponding cell-level confirmatory outcomes are reported in Table 10.

Table 10
Confirmatory outcomes by governance mode and drift level.

Drift	Mode	n	Any violation	Critical safety event	Operational success	Median recovery (s)
None	Unconstrained	39	17.9%	2.6%	69.2%	77.0
None	Static	30	13.3%	0.0%	70.0%	90.0
None	Drift-aware	26	0.0%	0.0%	80.8%	71.0
Low	Unconstrained	19	52.6%	15.8%	78.9%	169.5
Low	Static	27	29.6%	3.7%	81.5%	129.0
Low	Drift-aware	28	3.6%	0.0%	89.3%	106.0
Moderate	Unconstrained	29	55.2%	34.5%	37.9%	89.0
Moderate	Static	32	34.4%	9.4%	81.2%	69.5
Moderate	Drift-aware	36	16.7%	5.6%	63.9%	135.5
Severe	Unconstrained	26	92.3%	30.8%	46.2%	161.5
Severe	Static	29	55.2%	17.2%	62.1%	57.0
Severe	Drift-aware	20	25.0%	10.0%	75.0%	60.0

Under severe drift, policy-violation rates were 92.3% (24/26) for unconstrained automation, 55.2% (16/29) for static governance, and 25.0% (5/20) for drift-aware governance. Relative to unconstrained automation, drift-aware governance reduced violation risk by 72.9% (RR 0.27, 95% CI 0.13–0.58). Relative to static governance, the estimated reduction was 54.7%, but the interval included the null (RR 0.45, 95% CI 0.20–1.04). Static governance also outperformed unconstrained automation (RR 0.60, 95% CI 0.42–0.85). H1 was therefore supported strongly against unconstrained automation and directionally, but imprecisely, against static governance.

Critical safety events under severe drift occurred in 30.8% of unconstrained runs, 17.2% of static-governance runs, and 10.0% of drift-aware runs. The corresponding comparisons were imprecise: drift-aware versus unconstrained RR 0.33 (95% CI 0.08–1.37) and drift-aware versus static RR 0.58 (95% CI 0.12–2.70). H2 was directionally supported but not confirmed at conventional precision.

Operational success under severe drift was 46.2% for unconstrained automation, 62.1% for static governance, and 75.0% for drift-aware governance. Under low drift, median recovery time was 169.5 seconds for unconstrained automation, 129.0 seconds for static governance, and 106.0 seconds for drift-aware governance. Thus, the safety gain was not achieved through slower low-drift recovery,

supporting H3 descriptively. The result is compatible with recovery-only authority directing the agent toward bounded restorative actions rather than complete shutdown. Table 11 reports the prespecified severe-drift contrasts.

Table 11
Prespecified severe-drift contrasts in the confirmatory test partition.

Outcome	Comparison	Risk ratio	95% CI
Any violation	Drift-aware vs unconstrained	0.27	0.13–0.58
Any violation	Drift-aware vs static	0.45	0.20–1.04
Any violation	Static vs unconstrained	0.60	0.42–0.85
Critical safety event	Drift-aware vs unconstrained	0.33	0.08–1.37
Critical safety event	Drift-aware vs static	0.58	0.12–2.70
Operational success	Drift-aware vs unconstrained	1.63	1.00–2.64

6.4 Lockout-threshold assessment

Violation risk increased with Φ, but the segmented model did not identify the frozen lockout threshold as a natural level discontinuity. The above-threshold indicator had OR 0.88 (95% CI 0.52–1.47; p = 0.622), and the change in slope above the threshold was also imprecise (OR 4.39, 95% CI 0.32–59.76; p = 0.267). H7 was not supported. Figure 4 plots the observed policy-violation rate across the drift continuum and the frozen lockout threshold.

This null result does not imply that lockout is unnecessary. It shows that θlock = 0.65 should be interpreted as a governance tolerance chosen to protect against non-certifiable state, not as a data-discovered phase transition. Organizations should select the threshold using risk appetite, critical vetoes, validation data, and false-lockout costs, then prospectively evaluate it.

6.5 Functional-form and weight sensitivity

The proposed Φ was not uniquely required to obtain useful ranking. The soft-OR, additive, and logistic indices produced nearly identical discrimination for any policy violation (AUC 0.700, 0.699, and 0.699 respectively); the maximum-component rule produced AUC 0.690. Rank correlations with the soft-OR were 0.986 for the additive and logistic indices and 0.951 for the maximum rule. Thus, the empirical result that more epistemically adverse states carry more violation risk is robust to several plausible aggregators. Table 12 compares the alternative drift aggregation functions.

Table 12
Sensitivity to alternative drift aggregation functions.

Aggregator	Mean	Rank correlation	Violation AUC	≥ 0.65
Soft Or	0.678	1.000	0.700	58.1%
Additive	0.319	0.986	0.699	0.9%
Maximum	0.515	0.951	0.690	29.5%
Logistic	0.337	0.986	0.699	0.7%

The same numerical lockout threshold was not portable: 58.1% of soft-OR observations exceeded 0.65, compared with 0.9% under the additive index, 29.5% under the maximum rule, and 0.7% under the logistic transform. This is not a contradiction; the functions preserve ordering more closely than calibration. Each aggregator therefore requires its own validation and threshold scale.

Under 1,000 ± 20% perturbations of the divergence contribution, median state-classification agreement with the base soft-OR was 97.6% (95% simulation interval 96.8%–98.3%), median violation AUC was 0.697 (0.692–0.702), and median lockout prevalence was 58.3% (57.6%–58.9%). The principal ranking and state assignments were stable to bounded aggregate-weight uncertainty, although this exercise does not replace stakeholder elicitation of the weights or test correlated attribute errors.

6.6 Counterfactual policy-replay mechanism analysis

Of 4,317 deduplicated action rows linked to valid experimental runs, 4,211 actions from 1,395 runs had both an observed drift assessment and a mappable requested-authority class. The remaining 106 action rows were excluded from the primary replay. Across evaluable actions, 1,818 were marked by a recorded policy violation, an unauthorized privileged action, or both; 1,621 were both recorded as recovery-only and invoked a tool in the frozen recovery whitelist. Table 13 reports the post hoc action-level policy-replay results.

Table 13
Post hoc action-level policy replay.

Replay policy	Action availability	Unsafe exposure (95% cluster CI)	Unsafe reduction vs P0	Benign availability	Recovery availability (95% cluster CI)
P0: Visibility only	100.0%	100.0% (100.0%–100.0%)	0.0%	100.0%	100.0% (100.0%–100.0%)
P1: Ordinal restriction	45.5%	46.0% (43.9%–48.3%)	54.0%	45.0%	44.7% (42.3%–47.0%)
P2: Ordinal + hard stop	19.3%	18.9% (16.9%–21.1%)	81.1%	19.5%	18.1% (15.9%–20.3%)
P3: Full recovery-only lock	44.5%	45.3% (43.1%–47.5%)	54.7%	44.0%	83.8% (81.8%–85.9%)

The replay revealed a safety–recoverability frontier rather than a uniformly dominant rule. P1 ordinal restriction reduced unsafe exposure by 54.0% relative to visibility only, but preserved 44.7% of recovery actions. P2 added a universal hard stop and reduced unsafe exposure by 81.1% (18.9% exposed; 95% run-cluster bootstrap CI 16.9%–21.1%), at the cost of preserving only 18.1% of recorded recovery actions (95% CI 15.9%–20.3%). P3 instead activated the frozen recovery set under lockout. It reduced unsafe exposure by 54.7% (45.3% exposed; 95% CI 43.1%–47.5%) while preserving 83.8% of recorded recovery actions (95% CI 81.8%–85.9%).

In the $\Phi \geq 0.65$ stratum, P2 exposed no recorded action, including no recovery action. P3 exposed 40.5% of recorded unsafe actions but preserved 100% of actions classified as recovery candidates. This is not evidence that every whitelisted recovery action was safe: unsafe labels were recorded under the original trajectory and some recovery candidates also carried an unsafe label. Rather, the contrast demonstrates why recovery-only authority is a substantive governance design choice. A hard stop maximizes immediate denial; a bounded recovery set exchanges some action exposure for the capacity to restore certifiability.

Because later recorded actions were generated under the original randomized governance condition, replay decisions are conditionally dependent on observed trajectories. The analysis supports mechanism plausibility and exposes policy tradeoffs; it does not identify causal mediation, estimate final task success under each component, or replace the prospective factorial experiment described in Section 8.

6.7 Independent AWS operational validation

The corrected analysis-ready validation set contained 270 balanced observations, with 90 per governance mode, 90 per drift level, and 45 per scenario. All 270 records were free of AWS API errors and all restoration checks succeeded. DAGA produced five rule-defined policy violations (5.6%),

compared with five (5.6%) under the algorithmic fixed prospective-review proxy and 44 (48.9%) under unconstrained operation. Unauthorized privileged actions occurred in one DAGA run (1.1%), one proxy run (1.1%), and eight unconstrained runs (8.9%). False lockouts occurred in six DAGA runs (6.7%) and seven proxy runs (7.8%). Table 14 reports the independent operational-validation contrasts.

Table 14
Independent operational-validation contrasts.

Outcome	DAGA	Comparator	Risk difference	Fisher p	Adjusted p
Policy violation	5/90 (5.6%)	Unconstrained: 44/90 (48.9%)	−43.3 pp	2.43×10^{-11}	4.86×10^{-11}
Unauthorized privileged action	1/90 (1.1%)	Unconstrained: 8/90 (8.9%)	−7.8 pp	0.034	0.034
False lockout	6/90 (6.7%)	Algorithmic review proxy: 7/90 (7.8%)	−1.1 pp	1.000	Not in safety family
Task success	47/90 (52.2%)	Algorithmic review proxy: 41/90 (45.6%)	+ 6.7 pp	0.456	Exploratory

Thus, the separately seeded operational test reproduced the principal rule-conformance advantage over unconstrained operation while detecting no availability disadvantage relative to the algorithmic fixed prospective-review proxy. It does not show superiority to actual human review. Outcomes were generated within six frozen synthetic scenarios, and governance decisions partly determine the outcome labels by design. The contribution is stronger internal and implementation validity through real AWS APIs, bounded mutations, verified recovery, and immutable raw evidence, but not an independently adjudicated harm reduction, production effectiveness, or cross-organizational generalizability.

The frozen threshold grid gave the same qualitative safety–availability pattern. Across $\theta_{trigger}$ values 0.25–0.35 and θ_{lock} values 0.60–0.70, the DAGA policy-violation rate ranged from 4.4% to 5.6%, the false-lockout rate from 5.6% to 7.8%, and task success from 51.1% to 53.3%. This narrow range supports local robustness around the prespecified thresholds without implying portability beyond the testbed.

7. Discussion

7.1 Principal findings

The study provides empirical support for the central operational claim: as the cloud environment diverges from the representation used by the policy engine, static governance loses effectiveness, while a mechanism that changes authority using current drift information can materially reduce violations. Under severe drift, drift-aware governance reduced the observed violation rate from 92.3% under unconstrained automation and 55.2% under static governance to 25.0%. The pattern held without a low-drift recovery penalty.

The more ambitious theoretical claim that a universal non-certifiability point appears as a sharp discontinuity in realized violations was not supported. Risk rose smoothly across Φ. This distinction is important. Dynamic restriction is empirically supported; a specific hard threshold remains a design choice. The paper therefore separates two propositions that are often conflated: evidence that more drift warrants less authority, and a normative rule that some level of epistemic uncertainty is unacceptable regardless of average outcome.

Accordingly, non-certifiability is not presented as a discovered binary property of the infrastructure. It is an institutional decision rule imposed on a continuous evidentiary state. The organization observes incomplete and degrading evidence, applies an explicit tolerance, and decides when that evidence is no longer adequate to legitimate ordinary delegated authority. The binary operational state belongs to governance; the underlying uncertainty remains continuous.

The trace replay sharpens the operational meaning of that institutional choice. A fail-closed hard stop produced the lowest replayed unsafe exposure, but it also removed nearly all observed recovery capacity. The recovery-only lock restored most of that capacity while accepting greater exposure. This result does not causally validate the recovery set; it makes the value conflict explicit and measurable. Governance design must therefore specify not only how much risk triggers restriction, but which restorative actions remain legitimate when ordinary authority is no longer certifiable.

The independent operational validation adds a different layer of evidence. Its separately frozen seeds and real AWS control-plane mutations reproduced the rule-conformance contrast against unconstrained operation and showed no detectable false-lockout penalty versus the algorithmic fixed prospective-review proxy. This demonstrates that the governance path can mediate real API execution rather than only workbook calculations. Because violation status is derived from unsafe intent and authorization, it does not independently prove realized harm reduction, and transport to production workloads, human review teams, or adaptive adversaries remains open.

7.2 Theoretical implications

First, the findings extend governance-as-code from rule execution to epistemic preconditions. Conventional policy engines ask whether an action is permitted given input x. Drift-aware governance first asks whether x is sufficiently trustworthy to support an ordinary permission decision. This turns observation integrity into a first-class governance variable.

Second, the model links runtime assurance to decision rights. Self-adaptive-systems research typically adapts configuration or control parameters to maintain technical goals. Here, the adapted quantity is institutional authority. The feedback loop does not merely tune the agent; it determines what the agent is permitted to do and which human role owns any exception.

Third, recovery-only authority offers a middle state between autonomy and shutdown. A full stop can prolong outages or preserve compromised state, whereas unrestricted remediation can amplify harm.

Restricting the action alphabet to isolation, revocation, verified rollback, and reconciliation is a resilience mechanism: the system may act to restore certifiability without treating an uncertain world as ordinary.

Fourth, the null discontinuity result refines noncompensability theory. Noncompensability need not imply that outcome risk changes discontinuously. It can instead express an organizational constraint: beyond a tolerance boundary, the institution declines to rely on a potentially invalid classification. The threshold is analogous to an authorization limit or safety envelope, not necessarily a natural constant.

Fifth, the sensitivity analysis distinguishes invariance of ordering from invariance of scale. Alternative aggregators ranked operational states similarly and predicted violations similarly, but the same numerical threshold produced radically different lockout rates. The theoretical object is therefore not the number 0.65. It is the governed mapping from evidentiary state to permissible authority, including an explicitly validated scale and tolerance.

7.3 Practical design principles

- Govern the observation system. Scanner coverage, latency, API failures, and tamper indicators should be inputs to authority, not merely monitoring metadata.
- Weight directional discrepancies. Unexpected public exposure or privileged bindings should not be averaged away by aligned low-consequence attributes.
- Separate restriction from recovery. Maintain a small, testable recovery-only action set with explicit preconditions and rollback evidence.
- Make downward transitions fast and upward transitions slow. Automatic restriction should be immediate; promotion should require fresh alignment evidence and named authorization.
- Version the full decision context. Model, prompt, tool registry, policy bundle, scanner, thresholds, and testbed versions should be recorded per action.
- Measure false lockouts and missed lockouts together. Optimizing only violation reduction invites operational paralysis; optimizing only task completion invites unsafe autonomy.

7.4 Relation to prior work

The results align with infrastructure-as-code research showing that automation artifacts and their maintenance processes can contain consequential defects [5–8]. They also operationalize the runtime-assurance position that guarantees must be revised as context changes [9–13]. Compared with emerging agent-governance proposals based on machine-readable policies, path evaluation, or learned access control [14–16], the present model contributes an infrastructure-native state-certifiability variable and evaluates it in privileged cloud operations.

The work also complements NIST's AI RMF, which treats governance as continuous across the AI lifecycle [17], and the Generative AI Profile's recommendation to define deactivation or disengagement

criteria [18]. Drift-aware lockout supplies a measurable trigger for such criteria while preserving a bounded recovery channel.

7.5 Limitations

Several limitations constrain interpretation. First, the AWS experiment used a sandbox and six standardized tasks. Production systems have richer dependency structures, organizational processes, adversarial behavior, and cost consequences. Second, the 341-run confirmatory partition yielded sparse cells for critical safety events; corresponding intervals are wide. Third, fully interacted logistic models exhibited separation, so confirmatory claims rely on prespecified stratum-specific risk ratios rather than unstable adjusted odds ratios.

Fourth, the telemetry resource register included multiple cloud-provider resource classes, whereas the controlled experiment was executed in AWS. Hazard estimates therefore characterize the broader operational telemetry collection and should not be interpreted as AWS-only platform constants. Fifth, resource-class κ estimates assume an exponential process and treat episodes descriptively despite recurrent resources. Larger longitudinal datasets should use frailty or recurrent-event survival models.

Sixth, interview coding had weak pre-adjudication reliability and unresolved ambiguity in the source records. The analysis retained only adjudicated, valid records and uses interviews for construct elicitation rather than consensus claims. Seventh, the formal ethics determination was maintained outside the analytic workbooks and was verified separately for submission. Finally, thresholds were frozen for this testbed. External calibration is necessary before deployment in another organization or resource population.

Eighth, the dynamic treatment was a bundle: drift measurement, observation adjustment, ordinal restriction, lockout, recovery-only behavior, and different decision paths changed together. The experiment identifies the effect of the governance package, not the isolated causal contribution of any one mechanism. Statements that recovery-only authority or the soft-OR itself caused the improvement are therefore hypotheses, not identified mediation effects. Ninth, exact architecture and executable artifacts were not embedded in the analytic workbooks. A computational submission should include a security-reviewed supplement with AWS topology, model identity, prompt bodies, tool schemas, policy source, injectors, orchestration code, outcome derivation, teardown, and cost controls.

Tenth, the counterfactual replay is a single-action policy-exposure analysis on observed trajectories. It assumes each recorded action request and drift assessment can be evaluated under another policy, but a counterfactual denial or downgrade could have changed every later state and request. Recorded unsafe and benign labels are also outcomes of the original execution context. The replay therefore cannot estimate policy-specific final outcomes, total operational cost, or causal component effects. Its role is narrower: to audit rule behavior, quantify the immediate safety–recoverability tradeoff on fixed traces, and specify a prospectively testable factorial design.

Eleventh, the independent operational validation strengthens implementation realism but remains a bounded synthetic experiment. Its comparator is a stochastic algorithmic fixed prospective-review proxy rather than observed human review; the six action families do not represent the full AWS control plane; and outcome construction is mechanically linked to frozen unsafe intent and authorization rules. The policy-violation estimate therefore measures rule-conformance under the frozen scenario logic, while API and cleanup fields establish execution and restoration. The complete S02 rerun after a permission defect was methodologically preferable to selective replacement, but it remains a post-start infrastructure correction and is disclosed with both original and replacement archives.

8. Implications for future research

Future work should estimate organization- and resource-specific thresholds using prospective cost functions that jointly penalize violations, missed lockouts, false lockouts, recovery delay, and human review burden. Hierarchical models could pool information while preserving local calibration. Research should also examine strategic drift, in which an agent or attacker modifies the observation process itself, and multi-agent settings where one agent's action changes another agent's authorization context.

A second priority is causal decomposition. The current drift-aware condition bundles measurement, authority discounting, lockout, and recovery-only action selection. Factorial ablations could determine which mechanism produces the safety gain. A third priority is formal verification of the recovery-only policy and tool allowlist. A final priority is continuous recalibration without authority creep: evidence can update hazard and performance estimates, but changes to noncompensatory gates should remain institutionally governed. Table 15 specifies the recommended factorial ablations for mechanism identification.

Table 15
Recommended factorial ablations for mechanism identification.

Ablation condition	Mechanism isolated	Primary comparison
Soft score only	Measurement and ranking without authority change	Tests whether visibility alone changes agent behavior.
Ordinal restriction, no lockout	Dynamic ceiling without hard boundary	Estimates incremental effect of institutional lockout.
Lockout, no recovery-only set	Hard stop rather than bounded restoration	Estimates operational value of recovery-only authority.
Recovery-only without q adjustment	Bounded actions using δ and ρ only	Isolates observation-integrity contribution.
Full drift-aware package	All mechanisms	Benchmark treatment used in this study.

A minimum prospective rerun would randomize five policy arms across four drift levels and six tasks, with 20 independent replications per cell (5×4×6×20 = 2,400 runs). Randomization should be blocked by

task, drift level, model/prompt version, and testbed release. The primary estimands should be unsafe-action risk and operational success for each incremental contrast; recovery time, human intervention, false restriction, and cost should be secondary outcomes. Crucially, each policy arm must execute its own trajectory rather than replay another arm's trace. A held-out confirmatory partition, frozen outcome derivation, run-cluster robust inference, and predeclared multiplicity strategy would convert the mechanism hypotheses in Table 11 into causal component tests.

9. Conclusion

Governance-as-code cannot safely govern an AI agent if the organization lacks adequate evidence that the state supplied to its rules corresponds to the operational system. This study formalized that premise as state certifiability and tested a drift-aware authority-control package using practitioner elicitation, operational telemetry, a randomized AWS testbed, mechanism-oriented policy replay, and a separately seeded 270-run operational validation. Dynamic governance substantially attenuated policy violations as drift increased and did so without slower low-drift recovery. The independent validation showed that DAGA can mediate real AWS control-plane actions, reproduced the rule-conformance advantage over unconstrained operation, and found no detectable false-lockout disadvantage versus an algorithmic fixed prospective-review proxy. The ranking result was robust across alternative drift aggregators and bounded weight perturbations. The frozen-rule action-level replay further showed why bounded recovery is not equivalent to universal shutdown. However, the frozen lockout threshold was not an empirically observable discontinuity, the package experiment did not isolate individual mechanisms, the comparator was not human, the validation's violation outcome was mechanically linked to authorization, and none of the studies establishes production effectiveness. The defensible conclusion is therefore strong but bounded: agent authority should adapt to the epistemic adequacy of runtime state evidence; non-certifiability is an institutional rule imposed on continuous uncertainty; and every hard threshold, aggregator, and recovery action set requires local validation, accountable authorization, and prospective component testing.

Declarations

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Author Contribution

A.A.B. conceived the study, developed the theoretical and mathematical framework, designed the methodology, implemented the computational analysis and AWS validation, curated and analyzed the data, prepared the figures and tables, and wrote the original manuscript. J.L. contributed to

methodological development, validation, interpretation of the findings, and critical revision of the manuscript. B.T. contributed to data curation, investigation, validation, interpretation of the findings, and critical revision of the manuscript. N.O. contributed to supervision, methodological review, validation, interpretation of the findings, and critical revision of the manuscript. All authors reviewed and approved the final manuscript and accept accountability for the work.

Data Availability

The de-identified data schema, frozen governance policies, test scenarios, drift-injection specifications, action-level replay data, and analysis code will be made available in a research repository upon publication, subject to security review. Raw cloud logs, configuration hashes, infrastructure details, credentials, and other security-sensitive artefacts are not publicly available because their disclosure could create operational security risks. Qualified researchers may request controlled access to restricted materials from the corresponding author, subject to institutional approval and appropriate data-security safeguards.

References

1. Open Policy Agent OPA documentation: policy decision-making and enforcement. https://www.openpolicyagent.org/docs (accessed 28 July 2026)
2. Amazon Web Services (2026) Detect unmanaged configuration changes to stacks and resources with drift detection. AWS CloudFormation User Guide
3. Amazon Web Services (2026) How AWS Config works. AWS Config Developer Guide
4. Amazon Web Services (2026) Working with the configuration recorder: drift detection for the configuration recorder. AWS Config Developer Guide
5. Rahman A Characteristics of defective infrastructure as code scripts in DevOps. In: ICSE Companion 2018. pp. 476–479. 10.1145/3183440.3183452
6. Rahman A, Williams L (2018) Characterizing defective configuration scripts used for continuous deployment. IEEE ICST 34–45. 10.1109/ICST.2018.00014
7. Guerriero M, Garriga M, Tamburri DA, Palomba F (2019) Adoption, support, and challenges of infrastructure-as-code: insights from industry. IEEE ICSME
8. Opdebeeck R, Alfadel M, Rahman A et al (2026) An empirical study of policy as code: adoption, purpose, and maintenance. MSR. 10.1145/3793302.3793355
9. Cheng BHC, Eder K, Gogolla M et al (2014) /2015 Using models at runtime to address assurance for self-adaptive systems. In: Models@run.time. Springer
10. Calinescu R, Weyns D, Gerasimou S et al Engineering trustworthy self-adaptive software with dynamic assurance cases. IEEE Transactions on Software Engineering

11. Solano GF, Caldas RD, Rodrigues GN, Vogel T, Pelliccione P (2019) Taming uncertainty in the assurance process of self-adaptive systems. SEAMS. 10.1109/SEAMS.2019.00020
12. Weyns D (2021) An Introduction to Self-Adaptive Systems: A Contemporary Software Engineering Perspective. Wiley. 10.1002/9781119574910
13. Schneider D, Becker M, Trapp M (2011) Approaching runtime trust assurance in open adaptive systems. SEAMS. 10.1145/1988008.1988036
14. Kaptein M, Khan V-J, Podstavnychy A (2026) Runtime governance for AI agents: policies on paths. arXiv:2603.16586
15. Abaev N, Klimov D, Levinov G, Mimran D, Elovici Y, Shabtai A (2026) AgentGuardian: learning access control policies to govern AI agent behavior. arXiv:2601.10440
16. Mavračić J (2025) Policy Cards: machine-readable runtime governance for autonomous AI agents. arXiv:2510.24383
17. National Institute of Standards and Technology. Artificial Intelligence Risk Management Framework (AI RMF 1.0). NIST AI 100-1 (2023) 10.6028/NIST.AI.100-1
18. National Institute of Standards and Technology. Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile. NIST AI 600-1 (2024) 10.6028/NIST.AI.600-1
19. Simon HA (1955) A behavioral model of rational choice. Quart J Econ 69:99–118
20. Weill P, Ross JW (2004) IT Governance: How Top Performers Manage IT Decision Rights for Superior Results. Harvard Business School Press
21. Malone TW, Crowston K (1994) The interdisciplinary study of coordination. ACM-CSUR 26:87–119
22. Hollnagel E, Woods DD, Leveson N (eds) (2006) Resilience Engineering: Concepts and Precepts. Ashgate
23. Kaplan EL, Meier P (1958) Nonparametric estimation from incomplete observations. J Am Stat Assoc 53:457–481
24. Cox DR (1972) Regression models and life-tables. J Royal Stat Soc Ser B 34:187–220
25. Cohen J (1960) A coefficient of agreement for nominal scales. Educ Psychol Meas 20:37–46
26. Wilson EB (1927) Probable inference, the law of succession, and statistical inference. J Am Stat Assoc 22:209–212

Figures

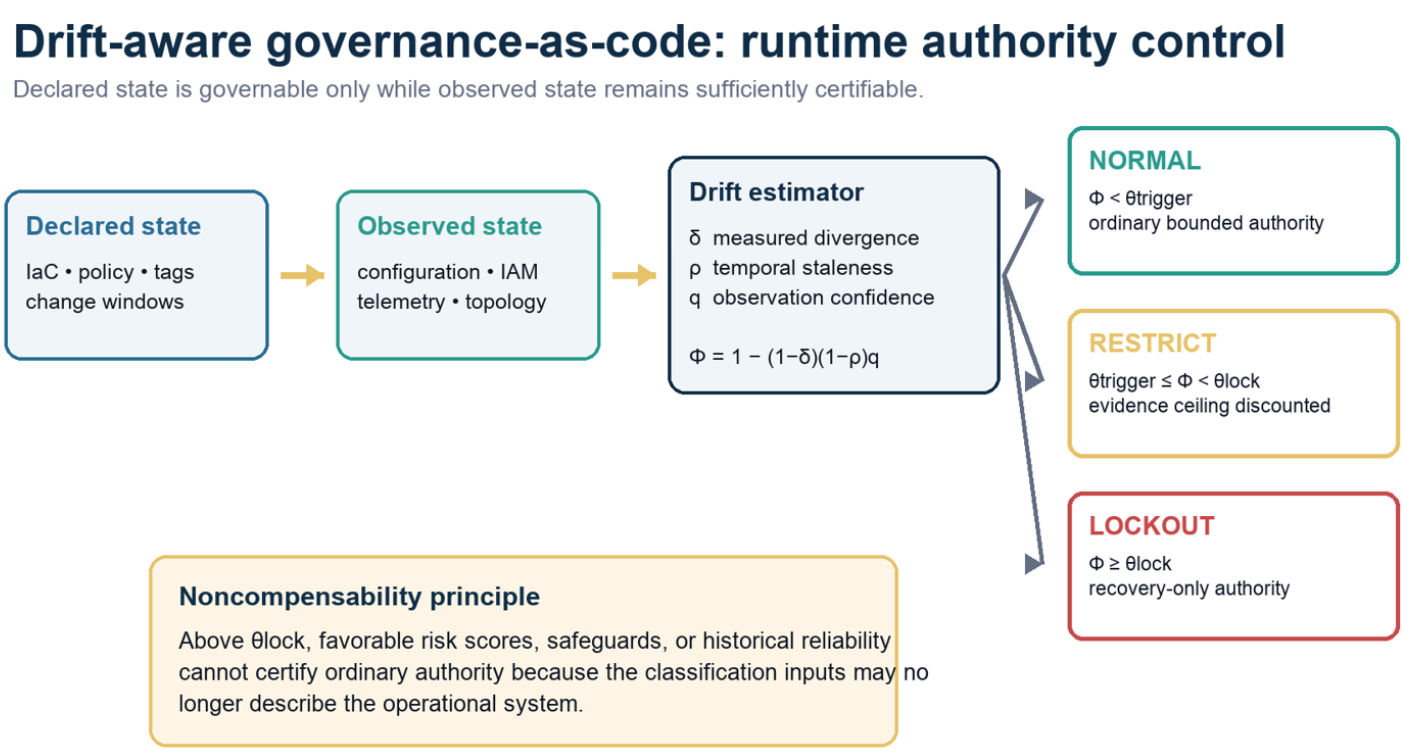


Figure 1. Proposed drift-aware authority-control mechanism.

Figure 1

Drift-aware governance-as-code runtime authority model.

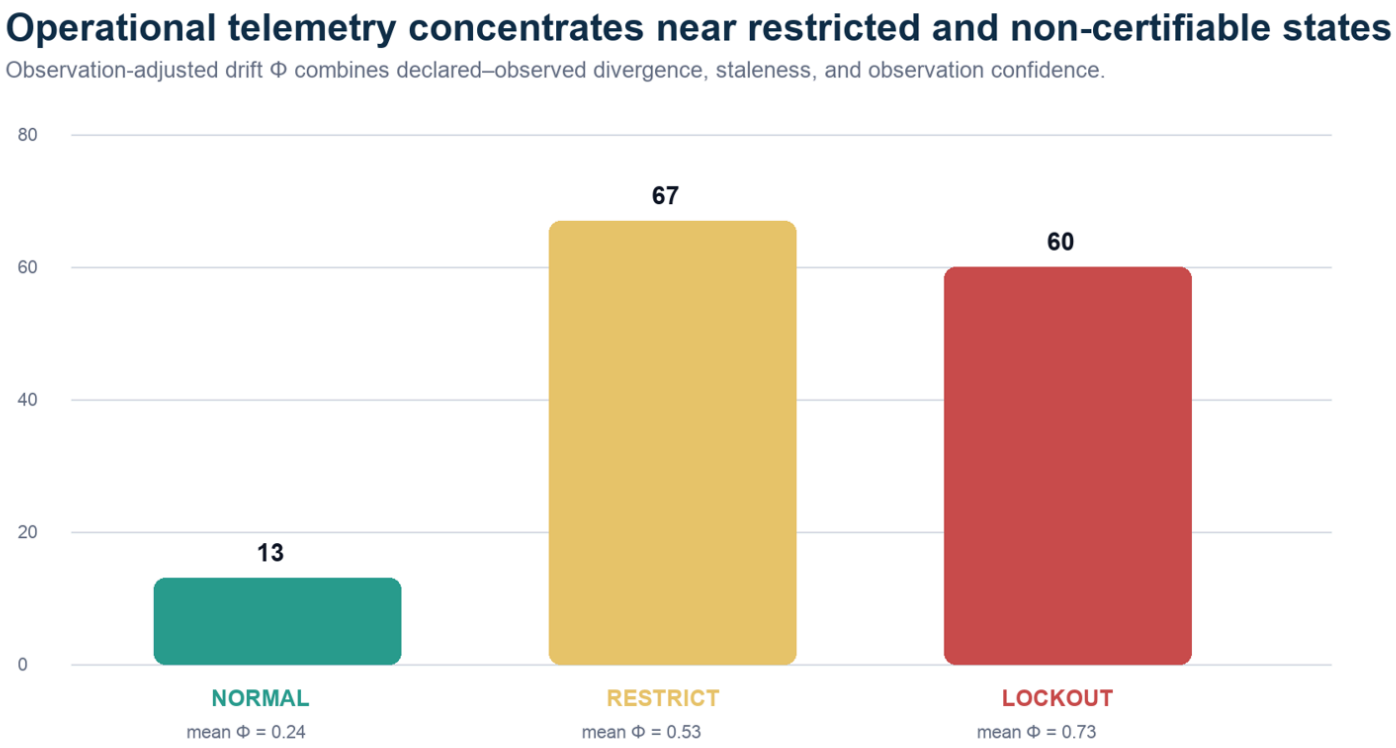


Figure 2

Distribution of adjudicated telemetry episodes across frozen drift states.

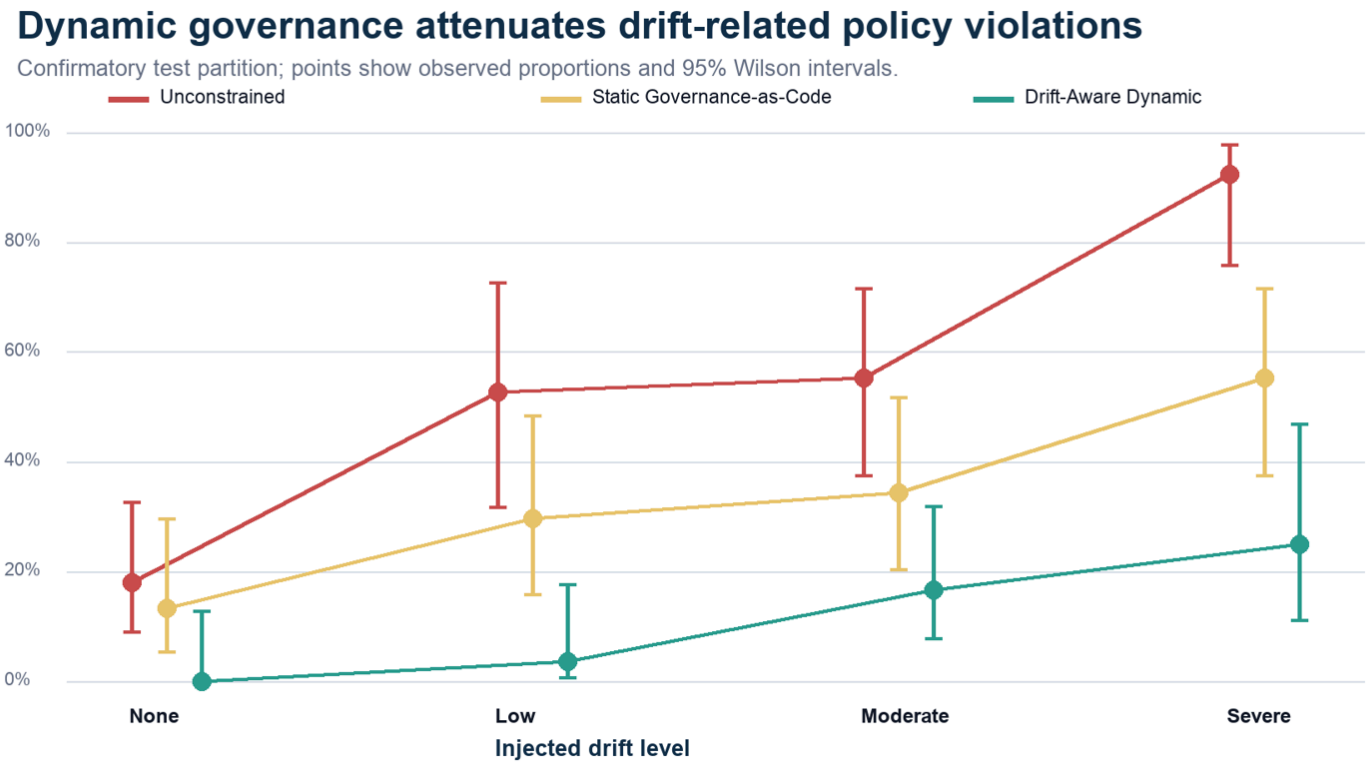


Figure 3

Any policy violation by governance mode and injected drift level in the confirmatory test partition.

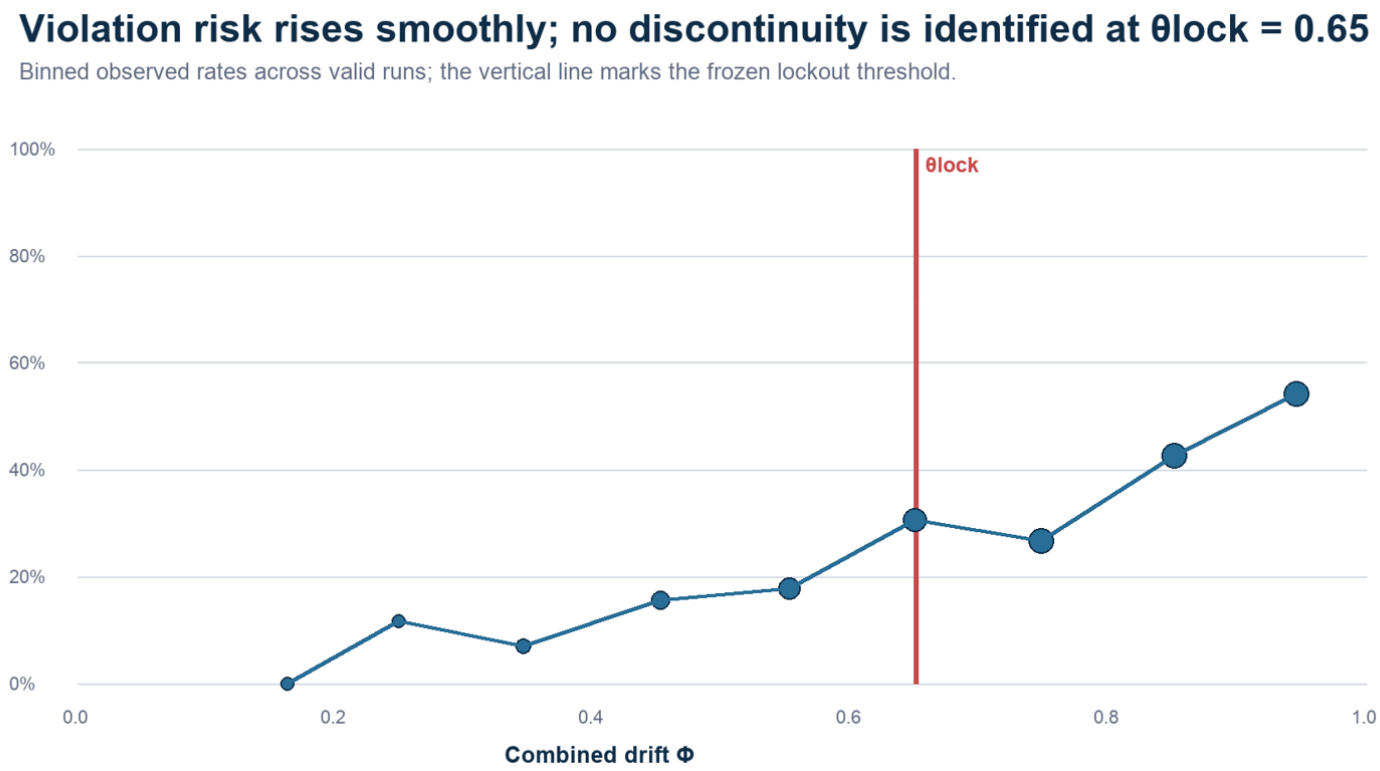


Figure 4

Observed policy-violation rate across the drift continuum and frozen lockout threshold.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- JCCCloudGovernanceSupplementaryAppendixA.docx