DECIONIS RESEARCH • TECHNICAL NOTE • VERSION 1.0

The Execution Verifiability Gap

Why Model Governance Cannot Authorize Consequential Actions

Positioning Decision Dossiers alongside Policy Decision Records and policy-state serializability

---

Festus Jejelowo
Decionis, Inc.

Published
21 August 2026

ABSTRACT

Model governance assesses whether AI is suitable and controlled. Execution authority proves that a specific consequential action was permitted, under the applicable policy and state, at the moment of execution.

These are adjacent controls, not substitutes. A governed model can still produce an action that is altered after evaluation, sent to the wrong environment, approved against expired conditions, replayed, or committed after a budget, inventory, fraud signal, or policy changed. A reproducible explanation of the model does not prove that the resulting effect was authorized when it landed.

This note defines that residual assurance deficit as the *Execution Verifiability Gap*. It situates execution-bound Decision Dossiers alongside Han's broader Verifiability Gap, PACE's signed Policy Decision Records, and Peng and Wu's policy-state serializability. It then specifies the minimum evidence and pre-commit checks needed to turn a decision record into narrowly scoped execution authority.

01

The missing distinction

Governance can justify delegation. It cannot replace transaction-level authority.

Model governance asks whether a model or agent is appropriate for a task: how it was validated, monitored, constrained, overseen, and changed. Those questions are necessary. Financial-sector guidance increasingly emphasizes system-level controls, observable evidence, defined action limits, and containment across the complete AI system—not the model in isolation. The Bank of England/FCA consortium minutes use the useful phrase *execution boundaries*, although the minutes explicitly do not constitute Bank or FCA policy. [5]

But a consequential commit presents a narrower and more immediate question: did the action that actually changed the system match the action that was evaluated, and was it still permitted at the final safe moment before that change? This question is independent of whether the proposer was a language model, deterministic software, a human operator, or a mixture of all three.

Han describes a broader Verifiability Gap between the assurance delegated financial authority demands and the explainability and reproducibility retained after agentic decisions. His experiments show why replaying a probabilistic model later is an unstable foundation for accountability. [1] The Execution Verifiability Gap is the transaction-bound remainder: the difference between proving that an action *was once evaluated* and proving that the exact effect *was authorized when committed*.

Model-governance evidence

Fitness, validation, monitoring, oversight, change control, incident response, and use-case accountability.

Execution-authority evidence

Exact payload and target, current policy and signals, action-bound approval, time window, replay state, and commit outcome.

02

Vocabulary for the execution boundary

The terms below separate system governance, authorization evidence, and commit-time correctness.

Model governance	The organizational and technical practices used to assess whether an AI system is suitable, controlled, monitored, and accountable for an intended use. It governs a system and its lifecycle; it does not, by itself, issue authority for a particular downstream effect.
Execution authority	Cryptographically verifiable, scope-limited permission for one precisely identified action to commit to one permitted target while named policy, signal, approval, and validity conditions remain current.
Consequential action	An operation that changes money, rights, access, inventory, production state, contractual position, or another material system-of-record state.
Execution Verifiability Gap	The assurance deficit between evidence that a proposed action was acceptable when evaluated and evidence that the exact effect committed was still authorized under the policy and state immediately preceding execution.
Decision Dossier	A portable, signed evidence envelope that records a Decionis verdict, the policy and evidence supporting it, and —when execution-bound—the exact payload, target, state references, approval conditions, validity window, and single-use authority for the downstream action.

Execution binding	The signed mapping from a dossier to the canonical payload digest, permitted target, policy and signal state, Presence approval, validity window, nonce, and execution correlation identifier that define its executable scope.
Pre-commit revalidation	The last authorization check performed against the actual payload, actual target, current authoritative state, approval receipt, and replay store immediately before the downstream commit begins.
Stale authorization	An authorization that was valid when issued but is no longer valid because relevant policy, signals, approvals, resources, budgets, or time constraints changed before execution.
Policy-state serializability	The correctness criterion introduced by Peng and Wu: each committed effect remains explainable against a policy state consistent with the effects ordered before it, rather than merely against a possibly stale request-time snapshot.
Presence approval	A time-bounded human approval receipt bound to an approver identity and role, the approved action digest, inherited conditions, and expiry. It is evidence within a new evaluation, not a transferable human ALLOW.
Decision Chain	Tamper-evident lineage linking evaluation, approval, revalidation, nonce claim, commit outcome, and recovery evidence across a governed workflow.

03

How the gap opens

A correct request-time decision can become an incorrect execution without any component lying.

Payload drift

An amount, beneficiary, command, order line, or refund quantity changes after evaluation.

Target substitution

The same bytes are aimed at another account, environment, resource, endpoint, or adapter.

State drift

Budget, inventory, refundable balance, risk score, fraud evidence, or policy changes before commit.

Approval drift

A human approves an earlier action, role requirement, risk state, or validity window.

Replay

A valid signed result is presented twice or concurrently to create more than one effect.

Partial failure

The external system commits, but the control plane does not observe finalization and cannot safely infer whether retry is harmless.

Durable and interrupted workflows make these failures more likely because stored state can resume under changed models, policies, indexes, tools, or orchestration semantics. Mozafari describes related anomalies as semantic read skew, compatibility skew, context escape, and merge skew. [4] The conclusion is not that workflows must never pause. It is that resumption must preserve or explicitly re-establish the authority conditions on which execution depends.

04

PACE, MasuGate, and Decision Dossiers

The three architectures converge on execution-time evidence from different starting points.

PACE inserts a deterministic verifier between an agent and DeFi execution. Its signed Policy Decision Record binds typed intent, policy, simulation, and exact execution bytes with expiry and replay protection. In 2,800 deterministic sandbox trials, the authors report zero unsafe executions and zero false positives, while explicitly limiting the claim to logic-level validation rather than deployment-ready security. [2]

Peng and Wu identify stale authorization as a distinct concurrency failure and define policy-state serializability: every committed effect must remain explainable against the policy state immediately preceding it. Their MasuGate prototype coordinates policy, state, and effects while allowing unrelated work to proceed. [3]

An execution-bound Decision Dossier combines the portable signed record with a general adapter contract: deterministic payload canonicalization, an explicit target, versioned policy and material-signal references, action-bound Presence evidence, expiration, a single-use nonce, and immediate revalidation. This is a synthesis and generalization, not a claim that Decionis originated deterministic verification, replay protection, or serializable authorization.

Dimension	Model governance	PACE / PDR	MasuGate	Execution-bound dossier
Primary question	Is the AI system suitable and controlled for this use?	Do typed intent, policy, simulation, and exact transaction bytes satisfy a deterministic verifier?	Can concurrent effects be committed under a serializable policy state?	Is this exact payload still authorized for this target under current policy, signals, approval, and replay state?
Unit of assurance	Model, application, use case, or control program	A DeFi transaction and signed Policy Decision Record	A transaction-like governed effect over shared mutable state	One execution-bound Decision Dossier and commit attempt
Time of truth	Design, validation, deployment, and monitoring intervals	Verification and on-chain execution boundary	Immediately preceding the serialized commit	Pre-commit revalidation, then nonce claim and finalization
Binding surface	Requirements, controls, evaluations, and evidence	Typed intent, policy, simulation result, exact execution bytes, expiry, replay protection	Policy and state dependencies coordinated with effects	Canonical payload, target, policy, material signals, Presence conditions, time, nonce, and correlation
Concurrency treatment	Usually outside the model-control record	Replay protection at the contract boundary	Central concern; policy-state serializability	Atomic nonce and mutable-scope claim; unrelated scopes remain independent
Boundary	Cannot alone prove the exact committed effect	Evaluated in a deterministic DeFi sandbox; authors do not claim deployment-ready security	Requires authoritative shared state and coordinated effect handling	Cannot make an external commit atomic with its own store; indeterminate effects require reconciliation

05

Threat model

The control is designed for hostile substitution and ordinary distributed-systems failure, not only a malicious model.

Protected assets	Correctness of the committed effect; integrity and scope of authorization; replay resistance; and the evidence needed to reconstruct evaluation, claim, commit, and recovery.
Adversaries and faults	A compromised or confused agent, a malicious integrator, concurrent legitimate callers, a stale human approval, payload or target substitution, clock and state drift, and partial failure across an external commit boundary.
Trusted components	Protected Ed25519 signing keys, deterministic canonicalizers, authoritative policy and signal resolvers, a transactionally consistent nonce store, sufficiently synchronized clocks, and adapters that obey claim/finalize semantics.
Security objective	No consequential commit may proceed unless the exact payload and target remain authorized by the current policy, material signals, approval conditions, validity window, and unused nonce immediately before execution.

Failure rule. In Enforcement Mode, inability to resolve current policy, signal, approval, or prior-effect state is not evidence of permission and therefore fails closed. Shadow Mode records the same structured would-block outcome without stopping the downstream action.

06

Reference architecture

Authority is checked where the action becomes irreversible, not only where the intent is proposed.

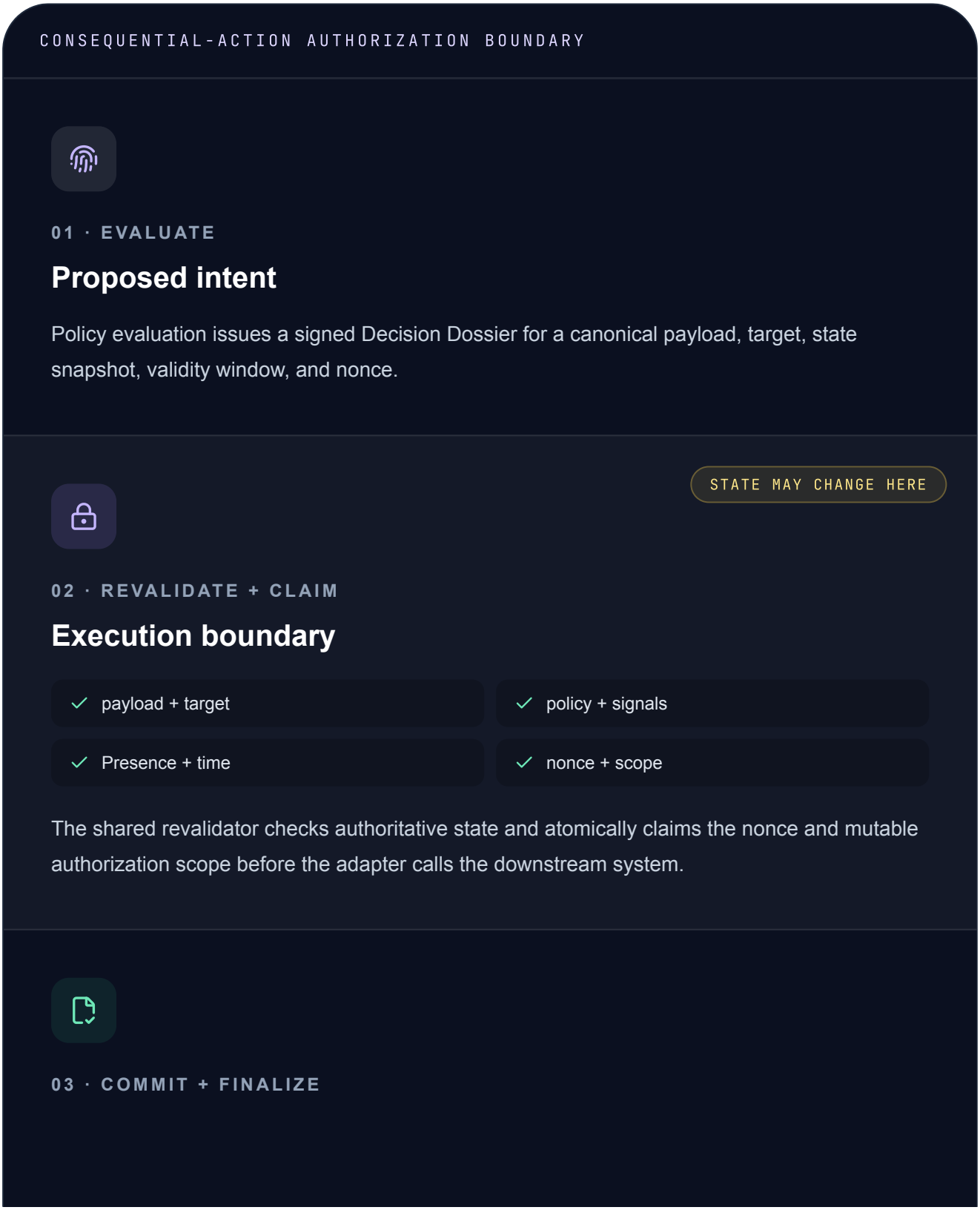

Figure 1. The Decision Dossier is evidence at evaluation time; execution binding plus pre-commit revalidation makes it scoped authority at execution time.

The dossier's complete `execution_binding` is covered by Ed25519 signature evidence. JSON payloads use RFC 8785/JCS canonicalization before SHA-256 hashing; non-JSON adapters must supply a named deterministic canonicalizer. [8] [9]

Backward compatibility is explicit. Existing proof bundles retain their original canonical representation and remain verifiable. Consequential adapters can require the new binding; in strict mode a valid but unbound legacy dossier cannot authorize a commit. This distinguishes *signature validity* from *execution eligibility*.

07

The pre-commit verification contract

A verifier should return stable, machine-actionable outcomes rather than security decisions hidden in prose.

1. Verify dossier signature and evidence integrity.
2. Accept only supported dossier and binding versions.
3. Check not-before and expiration constraints.
4. Digest the actual payload with the signed canonicalization profile.
5. Compare the actual system, environment, operation, resource, and endpoint.
6. Resolve the current policy identifier, version, digest, and outcome.
7. Resolve every material signal version or evidence digest.
8. Validate Presence identity, role, action digest, conditions, and expiry.
9. Atomically claim the nonce and mutable authorization scope.
10. Execute only on the existing verdict model's approving outcome, then finalize the claim.

STABLE REASON-CODE REFERENCE	
EXECUTION_BINDING_VALID	The signature, binding, current state, approval, and nonce claim all pass.
PAYLOAD_BINDING_MISMATCH	The canonical payload about to execute differs from the signed payload.
EXECUTION_TARGET_MISMATCH	The system, environment, operation, resource, or endpoint differs.
DOSSIER_EXPIRED	The signed execution window has ended.
NONCE_REPLAY_DETECTED	The single-use authorization was already claimed or finalized.
POLICY_VERSION_STALE	The active policy identity or version changed.
POLICY_STATE_CHANGED	The active policy digest, aggregate state, or current outcome changed.
SIGNAL_VERSION_STALE	A material signal version, value, observation, or evidence digest changed.
PRESENCE_APPROVAL_STALE	The human receipt, identity, role, conditions, or expiry is no longer current.
PRESENCE_ACTION_MISMATCH	The human approved a different payload or execution target.
CURRENT_STATE_UNRESOLVED	Authoritative policy, signal, approval, or prior-effect state cannot be resolved.
LEGACY_DOSSIER_NOT_EXECUTABLE	A strict adapter received a valid but execution-unbound legacy dossier.

`DOSSIER_SIGNATURE_INVALID`	Signature or signed evidence integrity verification failed.

Claim, finalize, reconcile

A nonce is claimed atomically before the external call and finalized as COMMITTED, FAILED, or INDETERMINATE. An indeterminate result is never automatically retried. Because Decionis and a third-party system usually cannot share one transaction, the adapter must provide downstream idempotency and status lookup for reconciliation. This residual atomicity limit is inherent, not removed by a signature.

08

Delayed Presence approval

A human approves an action under conditions; the human does not create a timeless bearer token.

Human review increases the interval in which an authorization can become stale. The amount or beneficiary may change; a risk score, available budget, policy version, required role, or approval expiry may move while the reviewer is deciding. A receipt saying “Festus approved” is insufficient unless it also proves *what* was approved and *under which inherited conditions*.

Presence approval is therefore incorporated into a newly evaluated dossier. It binds approver identity and role, the original intent, the payload-and-target action digest, inherited conditions, and an expiry. The effective execution window is the earliest relevant expiry. Any material change requires re-evaluation and—if policy still requires a human—a new action-bound approval.

01

Original request

Hold + action digest

02

Human review

Identity, role, conditions, expiry

03

Before commit

Resolve receipt again; mismatch means reapproval

09

Deterministic stale-authorization benchmark

Controlled barriers force state changes between evaluation and commit; no result depends on timing sleeps.

The implementation benchmark covers concurrent wire limits, overlapping refunds, inventory contention, delayed Presence approval, nonce replay, payload and target substitution, unchanged valid actions, and unrelated state changes. Each mode executes 265 attempts. The constructed attack rates below are scenario diagnostics, not estimates of real-world incident prevalence.

Metric	Request-time only	Execution-bound
Stale execution rate	50%	✓ 0%
Replay success rate	100%	✓ 0%
Payload / target substitution success	100%	✓ 0%
Valid-action completion	100%	✓ 100%
False-hold rate	0%	✓ 0%

P50	P95	P99	THROUGHPUT
0.246 ms	**0.902 ms**	**1.692 ms**	**2,246 attempts/s**

Measured on 21 August 2026 in the repository's in-process deterministic harness. Execution-bound p99 was below the existing p99 < 80 ms enforcement SLO. This isolates revalidation logic; it does not include production network, database, or downstream-system latency. The benchmark source and assertions are available in the **public repository**.

10

Limitations and non-claims

Execution evidence narrows one class of uncertainty. It does not make the surrounding institution correct.

- A Decision Dossier is not a certification that a model is safe, truthful, fair, robust, or suitable for a use case.
- A valid authorization does not prove that the policy itself is lawful, ethical, complete, or correctly encoded.
- The mechanism assumes protected signing keys, trustworthy authoritative state, deterministic canonicalizers, and adapters that cannot bypass enforcement.
- A legitimate but malicious approver can still approve harmful activity within their granted authority; identity and action binding do not establish good judgment.
- No dossier proves complete compliance with the EU AI Act, FSB practices, or another regulatory regime. It can supply evidence for particular implemented controls only.
- A third-party commit and Decionis finalization are not one atomic transaction. Indeterminate outcomes require downstream idempotency, status lookup, and human or automated reconciliation.
- The deterministic benchmark demonstrates specified failure modes in a controlled harness. It is not a formal proof, penetration test, production load test, or estimate of operational risk.

These limits matter in regulatory discussions. The FSB consultation proposes twelve practices spanning governance, materiality, monitoring, human oversight, cyber risk, and third-party dependencies; execution-bound evidence addresses only parts of that wider program. [6] Likewise, the start of enforcement for applicable EU AI Act provisions does not turn any one technical artifact into proof of complete compliance. [7]

CONCLUSION

Govern the model. Authorize the effect. Preserve evidence of both.

The relevant unit of accountability for a consequential action is not the model invocation alone. It is the complete path from proposed intent through current-state authorization to the exact committed effect. A Decision Dossier closes the execution gap only when it is bound to that path and revalidated at its final safe boundary.

11

References

1. Han, H. (2026). **Governing Agentic AI in FinTech: The Verifiability Gap in High-Stakes Financial Decision-Making** arXiv:2608.11344, version 2.

2. Karanjai, R., Lu, Y., Williamson, R., Hm, H., Mehrotra, P., Xu, L., & Shi, W. (2026). **Policy-Attested Contract Execution for Safe AI Agents in Decentralized Finance** arXiv:2608.17220.

3. Peng, Y., & Wu, X. (2026). **Stateful Governance for Concurrent Agentic Systems** arXiv:2608.02764, version 2.

4. Mozafari, B. (2026). **BEGIN AI TRANSACTION: Semantic Isolation for Durable AI Workflows** arXiv:2608.05412.

5. Bank of England & Financial Conduct Authority AI Consortium (2026). **Artificial Intelligence Consortium minutes — 3 June 2026** Published 5 August 2026.

6. Financial Stability Board (2026). **Sound Practices for Responsible Adoption of Artificial Intelligence: Consultation Report** 10 June 2026.

7. European Commission (2026). **Commission starts enforcing AI Act rules and new transparency requirements on 2 August** 31 July 2026.

8. Rundgren, A., Jordan, B., & Erdtman, S. (2020). **RFC 8785: JSON Canonicalization Scheme (JCS)** RFC Editor.

9. Josefsson, S., & Liusvaara, I. (2017). **RFC 8032: Edwards-Curve Digital Signature Algorithm (EdDSA)** RFC Editor.

RECOMMENDED CITATION

Jejelowo, Festus. “The Execution Verifiability Gap: Why Model Governance Cannot Authorize Consequential Actions.” Decionis Research, version 1.0, 21 August 2026. https://decionis.com/research/execution-verifiability-gap.

Download PDF v1.0

Decionis Protocol →

Verification tooling