

The Cost of Future Admissibility: Compression, Policy Revision, and the Limits of Retained History

Flyxion
Independent Researcher

October 23, 2026

Abstract

Runtime verification and enforcement systems for agentic AI judge an execution through a compressed representation of its history, usually chosen for the policy in force when the history is recorded. Recent work on formal enforcement, evolving guardrails, and auditable claims treats the governing policy as fixed or as revisable, but it does not develop what a revision requires of the history already retained. This essay states that requirement. For one policy, a compression suffices if and only if it merges no histories the policy separates, a condition already present in the literature. We extend it to families of policies and to online, incrementally maintained compression: the coarsest sufficient compression is a quotient whose logarithmic cardinality, when finite, is monotone and subadditive in the family; a family that separates all histories forces injectivity; the minimal incremental state is the intersection of the individual Nerode congruences, with index between the largest and the product of the individual indices; and the set of revisions that remain evaluable is fixed by the partition the compression induces. A corollary bounds the error of any verifier that sees only the compressed state and prices conservative rejection in lost availability. We read five recent preprints and a conceptual ledger-based specification through the framework, and propose a sufficiency gate to sit beside the structural and evidential checks of evolving-policy systems, together with a recognition audit for the case in which information is retained but not consulted. We also state a preservation relation and show that a measured improvement on a test family does not entail sufficiency for the family a system is answerable for, and we apply it to five reported boundaries at which retention, behavior or authority changed. We then separate three mechanisms that look alike in institutions, namely information loss, non-recognition of retained information, and strategic erasure, and add elementary economic propositions on dependency, hold-up, and retention, including a complementarity result for compression and revision under stated assumptions. The theorems are elementary; the claim is that the ability to revise a constraint is itself constrained by what the system has kept.

1 Introduction

A system that enforces a rule at runtime does not, in general, consult the full history of what has happened. It consults a representation of that history: a monitor state, a projected and abstracted event stream, a window of recent actions, a summary produced by a language model, or a set of predicates maintained by an observability service. The representation exists because

the full history is too large, too sensitive, too slow to consult before an effect must be allowed or refused, or simply not retained. In every such design the representation is chosen to serve a particular policy. A freshness rule requires the age of the most recent approval; a destination rule requires the hostname of an outgoing report; a data-flow rule requires provenance edges. The choice is made well, and it is made with respect to the rule that is in force on the day the system is built.

Rules do not stay in force. Failures are discovered, requirements change, regulators issue guidance, and operators learn that an approval window of six steps was too generous or too tight. A growing body of work now treats the revision of agent policies as a first-class engineering problem, and it builds machinery for it: language models that propose symbolic patches, deterministic checks that reject structural conflicts, and authority layers that decide when a verified rule becomes operative. That machinery answers the question of whether a proposed rule is well formed and whether it is warranted by evidence. It does not obviously answer a prior question, which is whether the proposed rule can be evaluated at all on the history that the system has kept.

The difference between these questions is the subject of this essay. Four properties are easily conflated in discussions of agent safety, and the conflation hides a gap. *Representational sufficiency* asks whether the retained representation preserves the distinctions a given policy needs. *Verification correctness* asks whether, given a sufficient representation, the verifier computes the policy’s answer. *Authorization of intervention* asks whether the system is entitled to act on that answer, for instance by blocking an operation or by activating a revised rule. *Preservation under continuation* asks whether the first three properties continue to hold as the history grows and as the policy changes. Verification correctness, however perfect, cannot repair a representation that has already merged histories the policy separates. Authorization, however rigorous, cannot make an unevaluable rule evaluable. The fourth property is therefore the one on which the others depend, and in the family and continuation senses developed here it is the least studied; the fixed-policy case is already treated in the runtime-verification literature.

The results below make the dependence exact. For a single fixed policy, a compressed representation suffices if and only if it does not merge histories that the policy distinguishes (Section 3). For a family of policies the same condition holds for the family’s joint answer, so that the coarsest sufficient representation is a quotient of the history space whose size, measured in bits, is monotone and subadditive in the family (Section 4). If the family of admissible future revisions is rich enough to single out any individual history, no compression below the identity is sufficient. When compression must be maintained incrementally, as it must in any online monitor, the state is a right congruence, the minimal sufficient state is the intersection of the individual Nerode congruences, and the number of states lies between the maximum and the product of the individual state counts, with both bounds attainable (Section 5). For any given compression, the set of revisions that can still be adopted without loss is determined by the partition the compression induces, and it is closed under Boolean combination; revisions with no retroactive reach are always safe (Section 6). Each of these results is elementary, and the essay does not present them as deep. It presents them as the correct vocabulary for a question that the sources reviewed pose only implicitly, and it shows by example and by reading of those sources that the vocabulary is not idle.

The essay proceeds in three movements. The first, Sections 2 through 6, develops the formal framework. The second, Sections 7, 8, 9 and 10, asks how a designer can escape the

theorem’s consequences, reads five recent preprints as instances of its conditions, follows the same condition across five boundaries at which retained evidence or authority changes, and works through a conceptual ledger-based specification as a concrete case. The third, Section 11, observes that organizations revise their own rules too, that they sometimes do so in ways that cease to recognize obligations generated by the rules they replace, and that this phenomenon has the same formal shape as compression loss while having a different cause. Distinguishing the two causes is an empirical matter, and the section states what evidence would be needed. Section 12 then adds elementary economic propositions on dependency, commitment, and retention, and a complementarity result showing how the power to compress history and the power to revise policy reinforce one another.

Three mechanisms that produce similar outcomes in institutions must also be kept apart, and the later sections do so. *Information loss* is the case in which the institution cannot distinguish histories because its retained representation merges them; the factorization and Nerode results govern it. *Non-recognition* is the case in which the institution can distinguish the histories, because the distinction survives somewhere in its total information, but the policy it applies declines to consult it; this concerns the selection of decision rules and not the availability of information. *Strategic erasure* is the case in which the institution benefits from losing the distinction, which requires economic assumptions about the gain from restriction and the cost of evidenceable breach, developed in Section 12. The three have superficially similar results, and only the first is a theorem about representations alone. That separation is what makes a connection to the pattern of platform degradation analytically useful and not merely rhetorical.

A note on sources is owed at the outset. The preprints discussed here were published in October 2026, apart from one read at an earlier revision, and the account of each rests on a stated basis recorded in Appendix A. Where only an abstract was available the text says so, and no claim is made about material that was not read.

2 The Setting

Let E be a set of events and $H \subseteq E^*$ a prefix-closed set of admissible histories. Admissibility encodes the deployment’s assumptions about event sources, instrumented boundaries, and workflow scope, in the sense in which recent runtime-verification work speaks of the admissible prefixes of a deployment [1]. A history is a finite sequence of events delivered in order. Whether a history records attempts, completed effects, or both is a modelling choice; the framework below is agnostic, although Section 13 returns to the distinction because it matters for enforcement.

Definition 1 (Answers and policies). Let W be a finite set of answers; for a permit/reject gate $W = \{0, 1\}$. A *policy* is a function $P : H \rightarrow W$. The value $P(h)$ is the answer to the question the policy poses at the prefix h , whose last event may be the pending request.

Definition 2 (Compression). A *compression* is a function $\sigma : H \rightarrow S$ into a set S of representations. A policy P is *decidable from* σ if there is a function $g : S \rightarrow W$ with $P = g \circ \sigma$.

For a function f with domain H , write $\ker f = \{(h, h') \in H^2 : f(h) = f(h')\}$. This is an equivalence relation, and the partition it induces is the *information content* of f in the only sense that matters for the arguments below. For equivalence relations ρ, θ on H , write $\rho \subseteq \theta$

when every ρ -class lies inside a θ -class; we then say ρ is *finer* than θ . A compression is finer when it merges fewer histories.

Two conventions are used throughout. First, a verifier that sees only the representation is any function of $\sigma(h)$, deterministic or randomized, and may use unbounded computation; the results concern information, not complexity. Second, the policy is a function of the history alone. Policies that depend on hidden environment state are outside the scope of the framework and are discussed in Section 13.

A running example. The incident-response agent of a recent runtime-verification paper [1] supplies a convenient instance. The agent collects diagnostics, creates a checkpoint, applies a remediation, and sends an external report. A temporal gate permits a remediation request only if an approval occurred recently enough. In that paper the bound is a small number of monitor steps chosen for validation and explicitly described as illustrative rather than elicited from an operator’s requirements. Suppose the deployed bound is six. The retained representation need only record, for the most recent approval, whether its age in events is at most six; any more detailed record is surplus. Suppose now that operators conclude that the bound should be ten. The question this essay asks is whether the system, which has been faithfully maintaining exactly the information that the bound of six required, can enforce the bound of ten.

3 Sufficiency for a Fixed Policy

The condition for a single policy is the starting point, and it is not new. Recent work states it for observation functions and prefix answers, and proves the equivalence below [1]. It is restated here in the vocabulary of kernels so that the extensions that follow are visibly extensions of one idea.

Lemma 1 (Decidability from a compression). *A policy P is decidable from σ if and only if $\ker \sigma \subseteq \ker P$.*

Proof. If $P = g \circ \sigma$ and $\sigma(h) = \sigma(h')$ then $P(h) = g(\sigma(h)) = g(\sigma(h')) = P(h')$. Conversely, if $\ker \sigma \subseteq \ker P$, define $g(s) = P(h)$ for any h with $\sigma(h) = s$; the choice does not depend on the representative because histories sharing a σ -value share a P -value. Extend g arbitrarily outside the image of σ . $\square$

The lemma has a one-line contrapositive that carries the whole argument of the essay. If two histories receive the same representation and the policy gives them different answers, then no verifier restricted to the representation can compute the policy on both. The failure is informational. It is not a failure of the verifier’s logic, its correctness proof, its computational budget, or its authority to act.

The qualitative statement can be sharpened into a quantitative one, which is useful because real compressions are rarely either perfectly sufficient or hopelessly insufficient. For the probabilistic results of this paper assume that H is countable, with its discrete σ -algebra, so that a probability measure μ on H is a probability mass function and every sum below is a countable sum of masses; for general measure spaces the same statements hold with conditional expectations in place of singleton masses. Let μ be the distribution of histories the deployment

will produce. For a policy Q with values in $\{0, 1\}$ define the *irrecoverability* of Q from σ as

$$\varepsilon_\mu(\sigma, Q) = \sum_{s \in S} \min\{\mu(\sigma = s, Q = 0), \mu(\sigma = s, Q = 1)\}.$$

Proposition 1 (Error floor). *For every randomized verifier $\hat{g} : S \rightarrow \Delta(\{0, 1\})$, the probability that $\hat{g}(\sigma(h)) \neq Q(h)$ is at least $\varepsilon_\mu(\sigma, Q)$, and the deterministic majority verifier attains it. Moreover, $\varepsilon_\mu(\sigma, Q) = 0$ if and only if Q is decidable from σ on the support of μ .*

Proof. Fix s and let $q(s)$ be the probability with which $\hat{g}(s)$ outputs 1. The conditional error mass at s is $\mu(\sigma = s, Q = 0)q(s) + \mu(\sigma = s, Q = 1)(1 - q(s))$, which is linear in $q(s)$ and therefore minimized at $q(s) \in \{0, 1\}$, where it equals the smaller of the two masses. Summing over s gives the floor, and choosing $q(s)$ to output the more probable value attains it. The sum vanishes exactly when every σ -class with positive mass (a countable family, by the discrete setting) is monochromatic for Q on the support of μ , which by Lemma 1 is decidability on the support. $\square$

The proposition applies to every verifier that works from $\sigma(h)$, and in particular to a language model that reads a truncated transcript, because a stochastic reader of a truncated view is a randomized function of that view. Its practical use is as a measurement. Given recorded traces, which the runtime-verification infrastructure surveyed below already supports, $\varepsilon_\mu(\sigma, Q)$ can be estimated by replay for any candidate revision Q , without deploying it.

Safety gates are asymmetric: a false permit and a false reject have different costs. The next corollary records what the common conservative design, which rejects whenever the representation is ambiguous, actually buys.

Corollary 1 (Conservative verification). *Let $Q = 1$ denote an admissible request and $Q = 0$ an inadmissible one. Call a σ -class ambiguous if it contains histories of positive μ -mass with both values of Q ; in the discrete setting this is the same as the mass-based notion used in Section 10. (A distribution-free notion, counting a class as ambiguous when it contains any two histories with different values, is stricter: it can refuse a positive-mass safe history because its class also contains an unsafe history of mass zero. The false-reject formula below and the bound of Proposition 11(iv) are stated for the mass-based notion, that is, for a verifier restricted to the support of μ .) The verifier that rejects on every ambiguous class and otherwise answers correctly makes no false permits, and its false-reject mass is $\sum_{s \text{ ambiguous}} \mu(\sigma = s, Q = 1)$.*

The corollary makes precise a remark that appears in the recent literature, that rejecting an ambiguous request avoids permitting an unsafe candidate but does not recover the lost distinction [1]. The price of the conservative choice is unavailability, and the corollary says exactly how much: the probability mass of safe histories that share a representation with an unsafe one. A compression that is cheap for today's policy can turn tomorrow's revision into a source of spurious refusals whose volume is computable.

4 Sufficiency for a Family of Policies

A revision is the replacement of one policy by another, and the designer who wishes to remain able to revise must choose a compression that is sufficient not for one policy but for every policy that might be adopted. Let $\mathcal{P}$ be a family of policies, which may be finite, infinite, or

the set of everything a policy language can express. Write $\Pi_{\mathcal{P}} : H \rightarrow \prod_{P \in \mathcal{P}} W$ for the joint answer map $h \mapsto (P(h))_{P \in \mathcal{P}}$ and $\equiv_{\mathcal{P}} = \ker \Pi_{\mathcal{P}}$.

Definition 3 (Family sufficiency). A compression σ is $\mathcal{P}$ -sufficient if every $P \in \mathcal{P}$ is decidable from σ .

Proposition 2 (The sufficient quotient). σ is $\mathcal{P}$ -sufficient if and only if $\ker \sigma \subseteq \equiv_{\mathcal{P}}$. Consequently the quotient map $q_{\mathcal{P}} : H \rightarrow H/\equiv_{\mathcal{P}}$ is $\mathcal{P}$ -sufficient and factors through every $\mathcal{P}$ -sufficient compression, $q_{\mathcal{P}} = k \circ \sigma$; it is unique up to bijection as the coarsest $\mathcal{P}$ -sufficient compression. The converse need not hold: a sufficient compression may retain further distinctions, in which case it cannot be recovered from the quotient, although some sufficient compressions are equivalent to the quotient and then factor both ways.

Proof. By Lemma 1, σ is $\mathcal{P}$ -sufficient iff $\ker \sigma \subseteq \ker P$ for every $P \in \mathcal{P}$, iff $\ker \sigma \subseteq \bigcap_P \ker P = \ker \Pi_{\mathcal{P}}$. If $\ker \sigma \subseteq \ker q_{\mathcal{P}}$ then $q_{\mathcal{P}} = k \circ \sigma$ for some k by the same construction as in Lemma 1. Uniqueness is the usual uniqueness of the coarsest partition satisfying a refinement condition. $\square$

The quotient is the analogue, for policies, of a minimal sufficient statistic in the sense of Fisher [9]: it retains exactly the distinctions that some policy in the family can detect and no others. When the quotient is finite, its size defines a cost.

Definition 4 (Admissibility rank). For a family $\mathcal{P}$ with $H/\equiv_{\mathcal{P}}$ finite, the *admissibility rank* is $\text{rk}(\mathcal{P}) = \log_2 |H/\equiv_{\mathcal{P}}|$, the number of bits any $\mathcal{P}$ -sufficient representation must be able to carry.

Proposition 3 (Monotonicity and subadditivity). Let $\mathcal{P}, \mathcal{Q}$ be families with finite quotients.

1. If $\mathcal{P} \subseteq \mathcal{Q}$ then $\text{rk}(\mathcal{P}) \leq \text{rk}(\mathcal{Q})$, and $\equiv_{\mathcal{Q}} \subseteq \equiv_{\mathcal{P}}$.
2. $\max\{\text{rk}(\mathcal{P}), \text{rk}(\mathcal{Q})\} \leq \text{rk}(\mathcal{P} \cup \mathcal{Q}) \leq \text{rk}(\mathcal{P}) + \text{rk}(\mathcal{Q})$.
3. If $\mathcal{P}$ consists of k Boolean policies then $\text{rk}(\mathcal{P}) \leq k$, with equality if and only if all 2^k answer patterns occur.

Proof. (1) Adding policies can only refine the joint kernel. (2) The lower bound is (1). For the upper bound, two histories are $\equiv_{\mathcal{P} \cup \mathcal{Q}}$ -equivalent iff they are equivalent for both families, so the class of a history in the union is determined by the pair of its classes, and the map into the product of the two quotients is injective. (3) The joint answer takes at most 2^k values. $\square$

The direction of the inclusions in part (1) is easy to misread: a larger family has a finer, hence smaller, joint kernel, which is why a larger family demands a finer compression.

The cost of future admissibility can now be defined. If $\mathcal{P}_{\text{now}}$ is the family of policies the system enforces today and $\mathcal{P}_{\text{fut}} \supseteq \mathcal{P}_{\text{now}}$ the family it may be asked to enforce after revision, the minimal present-answer quotients have logarithmic cardinalities $\text{rk}(\mathcal{P}_{\text{now}})$ and $\text{rk}(\mathcal{P}_{\text{fut}})$, respectively. These are information-capacity lower bounds, not measured storage costs for a particular implementation. The difference

$$\Delta = \text{rk}(\mathcal{P}_{\text{fut}}) - \text{rk}(\mathcal{P}_{\text{now}}) \geq 0$$

is the number of additional bits of distinction that must already have been kept. By Proposition 3, $\Delta \leq \text{rk}(\mathcal{P}_{\text{fut}} \setminus \mathcal{P}_{\text{now}})$, so independent revisions are priced additively in the worst case, and revisions that reuse the distinctions already retained are free.

An average-case measure. Rank is a worst-case count in the sense of Hartley, and it counts *present-answer* classes: histories are identified when every policy in the family gives them the same answer now. Section 5 defines a second, larger quantity, the incremental rank, which counts classes that must also keep answering correctly after further events. Under a declared distribution μ on histories, let Q_0 and Q_1 denote the random class of a history in $H/\equiv_{\mathcal{P}_{\text{now}}}$ and in $H/\equiv_{\mathcal{P}_{\text{fut}}}$ respectively. Because $\equiv_{\mathcal{P}_{\text{fut}}} \subseteq \equiv_{\mathcal{P}_{\text{now}}}$ by Proposition 3(1), Q_0 is a deterministic function of Q_1 , and the *revision debt*

$$D_{\text{rev}} = H(Q_1 | Q_0) = H(Q_1) - H(Q_0) \geq 0$$

measures the average additional information, in bits, that the finer present-answer classification carries beyond the coarser one; the incremental analogue is defined in Section 5. It is bounded above by $\text{rk}(\mathcal{P}_{\text{fut}}) - H(Q_0)$ and is not the number of bits that any particular implementation must store, since an implementation may need more than the entropy of the classes it distinguishes and a distribution-dependent encoding may need less. Its usefulness is that it is a property of the pair of policy families and the distribution of histories, independent of the monitor’s design. A system can carry substantial revision debt even though every decision made under its original policy was correctly verified, which is the sense in which verification correctness and preservation under continuation are different properties.

4.1 Families that separate everything

The extreme case is instructive. Call a family $\mathcal{P}$ *separating* if for every pair of distinct histories $h \neq h'$ some $P \in \mathcal{P}$ gives them different answers.

Theorem 1 (No policy-agnostic compression). *If $\mathcal{P}$ is separating then $\equiv_{\mathcal{P}}$ is equality, and a compression is $\mathcal{P}$ -sufficient if and only if it is injective on H . In particular, if H is finite then every $\mathcal{P}$ -sufficient compression has at least $|H|$ values, so $\text{rk}(\mathcal{P}) = \log_2 |H|$.*

Proof. Separation says $\ker \Pi_{\mathcal{P}}$ is the diagonal. Proposition 2 then requires $\ker \sigma$ to lie inside the diagonal, which is injectivity. $\square$

Separating families are not exotic. The family of one-trace patches, $P_{h_0}(h) = 0$ if $h = h_0$ and 1 otherwise, is separating on any history space, and a patch that denies one specific newly observed failure is a policy of this form. A policy language that can name arbitrary traces therefore carries a family of future revisions that forces injectivity on the histories it can name. For histories of length n over an alphabet of $a \geq 2$ events, the bound is $n \log_2 a$ bits: the whole history. It follows that no deterministic compression is universally safe against unrestricted revision, and that nontrivial compression is possible only by one of three moves, each of which has a cost examined in Section 7: restrict the family of admissible revisions, retain evidence outside the working state, or accept a quantified error.

Remark 1 (The family is a property of the language). In practice $\mathcal{P}$ is not an arbitrary set of functions but the set $\mathcal{P}_L$ of policies expressible in a policy language L . The quotient $H/\equiv_L$ is then the set of histories modulo indistinguishability by any L -policy, and its size is a property of the language. A language whose rules read only uniformly bounded windows over finite event alphabets can have a finite quotient, although its size need not be small; a language whose rules can distinguish arbitrary complete traces forces injectivity on those traces. The cost of future admissibility is therefore fixed in advance by the expressive power one is willing to grant to future revisions, which is a design decision rather than an empirical accident.

5 Incremental Compression and Continuation

An online monitor does not recompute its representation from scratch. It maintains a state and updates it as each event arrives. This constraint changes the problem, because an update rule can only transform the state it has; information that has been dropped cannot be restored by any later event. For this section take $H = E^*$; for a prefix-closed proper subset, restrict continuations accordingly.

Definition 5 (Incremental compression). A compression $\sigma : E^* \rightarrow S$ is *incremental* if there is an update function $u : S \times E \rightarrow S$ with $\sigma(he) = u(\sigma(h), e)$ for all $h \in E^*$ and $e \in E$. Equivalently, $\ker \sigma$ is right-invariant: $(h, h') \in \ker \sigma$ implies $(he, h'e) \in \ker \sigma$.

For a policy P define the *Nerode congruence* $h \sim_P h'$ iff $P(hz) = P(h'z)$ for every continuation $z \in E^*$ [7, 8]. Two histories are P -continuation-equivalent when no future sequence of events can make the policy tell them apart.

Lemma 2 (Nerode). (a) $\sim_P$ is right-invariant. (b) $\sim_P \subseteq \ker P$. (c) $\sim_P$ is the coarsest right-invariant equivalence contained in $\ker P$: every right-invariant $\rho \subseteq \ker P$ satisfies $\rho \subseteq \sim_P$.

Proof. (a) If $h \sim_P h'$ then for any e and z , $P(hez) = P(h'ez)$ because ez is a continuation. (b) Take z empty. (c) If ρ is right-invariant and $h \rho h'$, then $hz \rho h'z$ for every z by induction on $|z|$, so $P(hz) = P(h'z)$ because $\rho \subseteq \ker P$. $\square$

For a family $\mathcal{P}$ write $\sim_{\mathcal{P}} = \bigcap_{P \in \mathcal{P}} \sim_P$. An intersection of right-invariant equivalences is right-invariant, so $\sim_{\mathcal{P}}$ is a right congruence.

Theorem 2 (Minimal incremental sufficiency). *For an incremental compression σ , the following are equivalent: (i) σ is $\mathcal{P}$ -sufficient; (ii) $\ker \sigma \subseteq \sim_{\mathcal{P}}$; (iii) for every $P \in \mathcal{P}$ and every pair h, h' with $\sigma(h) = \sigma(h')$, $P(hz) = P(h'z)$ for all continuations z . The quotient $E^*/\sim_{\mathcal{P}}$, with update $u([h], e) = [he]$, is the minimal incremental $\mathcal{P}$ -sufficient compression, and its number of states is the index $\text{idx}(\sim_{\mathcal{P}})$.*

Proof. (i) $\Rightarrow$ (ii): $\ker \sigma$ is right-invariant and, by Proposition 2, contained in $\ker \Pi_{\mathcal{P}} \subseteq \ker P$ for each P ; by Lemma 2(c), it lies in each $\sim_P$, hence in their intersection. (ii) $\Rightarrow$ (iii) is the definition of the intersection. (iii) $\Rightarrow$ (i): taking z empty gives $\ker \sigma \subseteq \ker P$ for each P , which is $\mathcal{P}$ -sufficiency by Lemma 1. Minimality: $\sim_{\mathcal{P}}$ is right-invariant, so its quotient is an incremental compression; it is $\mathcal{P}$ -sufficient by (ii) $\Rightarrow$ (i), and any other incremental sufficient σ has $\ker \sigma \subseteq \sim_{\mathcal{P}}$, so the quotient is a coarsening of σ . $\square$

The theorem upgrades a statement about answers at the present prefix into a statement about answers at all future prefixes. The upgrade is automatic for incremental compressions, which is why a monitor whose state is sufficient for today's policy is automatically sufficient for the policy's answers on every continuation, and also why it is irrecoverably insufficient for any revision that separates histories it has merged: those histories will stay merged under every possible update.

Proposition 4 (Index bounds). *If each $\sim_P$, $P \in \mathcal{P}$ finite, has finite index, then*

$$\max_{P \in \mathcal{P}} \text{idx}(\sim_P) \leq \text{idx}(\sim_{\mathcal{P}}) \leq \prod_{P \in \mathcal{P}} \text{idx}(\sim_P).$$

Both bounds are attained.

Proof. The intersection refines each $\sim_P$, so its index is at least each individual index. A class of the intersection is determined by the tuple of its classes in the individual congruences, so the map into the product is injective, giving the upper bound. Attainment of the lower bound is trivial when $\mathcal{P}$ is a singleton, and attainment of the upper bound is Example 1 below. $\square$

The bounds can be sharpened to an exact count. Write $\rho_i = \sim_{P_i}$ for $\mathcal{P} = \{P_1, \dots, P_k\}$, $S_i = E^*/\rho_i$ with $n_i = |S_i|$, and let

$$J = \{([h]_{\rho_1}, \dots, [h]_{\rho_k}) : h \in E^*\} \subseteq \prod_i S_i$$

be the set of component-state tuples that some common history actually reaches.

Proposition 5 (Exact index). (i) $\text{idx}(\sim_{\mathcal{P}}) = |J|$. (ii) $\text{idx}(\sim_{\mathcal{P}}) = \prod_i n_i$ if and only if $J = \prod_i S_i$, that is, every combination of component states is jointly reachable; reachability of each component state separately does not suffice. (iii) $\text{idx}(\sim_{\mathcal{P}}) = \max_i n_i$ if and only if, for a component j with $n_j = \max_i n_i$, $\rho_j \subseteq \rho_i$ for every i ; equivalently each S_i is a function of S_j .

Proof. (i) Two histories have the same tuple exactly when they are equivalent under every ρ_i , hence under the intersection, so the intersection quotient is in bijection with J . (ii) follows from (i) because $J \subseteq \prod_i S_i$. (iii) Put $\rho = \bigcap_i \rho_i \subseteq \rho_j$. The induced surjection $E^*/\rho \rightarrow E^*/\rho_j$ is a bijection when both quotients have n_j classes, so $\rho = \rho_j$ and ρ_j refines every ρ_i . Conversely, if ρ_j refines all ρ_i then $\rho = \rho_j$. $\square$

The lower bound therefore measures complete state reuse: further policies impose different requirements at no additional incremental memory. Over $E = \{a\}$, the policies testing divisibility of the length by two and by four are distinct, but the modulo-four state determines the modulo-two state, so the joint index is four. The upper bound measures full joint reachability, and intermediate values measure the constraints that a shared history imposes among component states. For unary counters with arbitrary moduli $m_1, \dots, m_k$, without coprimality, the joint index is $\text{lcm}(m_1, \dots, m_k)$, since two lengths agree modulo every m_i exactly when their difference is divisible by the least common multiple. Moduli four and six give twelve states, strictly between the lower bound six and the upper bound twenty-four; the reachable residue pairs are those with $r_4 \equiv r_6 \pmod{2}$, and the shared parity excludes half of the product. Pairwise independence also does not give the product bound for three or more policies. Over $E = \{a, b\}$ let three policies report the parity of the number of a 's, the parity of the number of b 's, and the parity of the length. Each has two Nerode states and every pair reaches all four combinations, yet the third bit is the exclusive or of the first two, so the three-policy monitor has four states and not eight. The relevant quantity is combinatorial reachability under permitted histories, not statistical independence, and it is the quantity an experimental program can measure directly.

The present-answer rank of Section 4 and the quantities of this section are different objects, and the difference is the point of the section. Define the *incremental rank* of a family $\mathcal{P}$ as $\text{rk}_{\text{inc}}(\mathcal{P}) = \log_2 \text{idx}(\sim_{\mathcal{P}})$, where $\sim_{\mathcal{P}} = \bigcap_{P \in \mathcal{P}} \sim_P$, and, under a distribution μ on H with $C_{\mathcal{P}}$ the random $\sim_{\mathcal{P}}$ -class, the *incremental revision debt* $D_{\text{rev}}^{\text{inc}} = H(C_{\mathcal{P}_{\text{fut}}} | C_{\mathcal{P}_{\text{now}}}) = H(C_{\mathcal{P}_{\text{fut}}}) - H(C_{\mathcal{P}_{\text{now}}})$, the inclusion $\sim_{\mathcal{P}_{\text{fut}}} \subseteq \sim_{\mathcal{P}_{\text{now}}}$ making $C_{\mathcal{P}_{\text{now}}}$ a function of $C_{\mathcal{P}_{\text{fut}}}$. Always $\text{rk}(\mathcal{P}) \leq \text{rk}_{\text{inc}}(\mathcal{P})$, since $\sim_P \subseteq \ker P$ by Lemma 2(b): the information needed to answer now can be strictly smaller than the information needed to keep answering correctly.

Example 1 (Independent counters). Let $E = \{a\}$ and let $P_m(h) = 1$ iff m divides $|h|$. Then $h \sim_{P_m} h'$ iff $|h| \equiv |h'| \pmod{m}$: the implication from right to left is immediate, and for the converse, if $|h| \not\equiv |h'| \pmod{m}$, choose z with $|h| + |z| \equiv 0$ and then $|h'| + |z| \not\equiv 0$. Hence $\text{idx}(\sim_{P_m}) = m$. For pairwise coprime $m_1, \dots, m_k$, the Chinese remainder theorem gives $\sim_{\{P_{m_i}\}}$ equal to congruence modulo $\prod m_i$, whose index is $\prod m_i$. The incremental rank, $\sum_i \log_2 m_i$, is exactly additive, whereas the present-answer rank of the k Boolean policies is at most k bits (at most 2^k answer patterns). The gap between $\sum_i \log_2 m_i$ and k is the price of keeping the answers correct as events arrive. Revisions that track independent features cost the full product.

Example 2 (Freshness windows). Let $E = \{a, n, r\}$ for approval, neutral event, and remediation request. Define the age of a history h as the number of events after the most recent a , with age ∞ if no approval has occurred. For a bound k let P_k be the answer to the gate question posed at a prefix, namely whether a remediation request arriving next would be permitted, so that $P_k(h) = 1$ iff $\text{age}(h) \leq k$. The pending request is evaluated at the prefix that precedes it, so it does not enter the age on which its own permission depends. (Defining the answer instead on histories that end in r would make the current answer a further distinction and change the count of classes.) The Nerode classes of P_k are indexed by $\min\{\text{age}, k+1\}$ with all ages of at least $k+1$, including ∞ , forming one stale class. Two distinct ages $i < j \leq k$ are separated by the continuation n^{k-j+1} , after which the younger history still has an admissible age and the older does not, and age k is separated from the stale class by the empty continuation, since P_k takes the value 1 at age k and 0 at the stale class. Hence $\text{idx}(\sim_{P_k}) = k+2$. For the family $\{P_k : k \leq K\}$ the intersection equals $\sim_{P_K}$, since the classes of P_K determine every P_k with $k \leq K$, and the state needed grows linearly with the largest bound that may ever be adopted. For the family of all bounds, $\{P_k : k \in \mathbb{N}\}$, the index is infinite: no finite-state monitor is sufficient.

Now return to the running example. Recording merely whether the approval is at most six events old suffices for the present answer but not for updating it: ages zero and six have the same Boolean answer and different answers after one more neutral event. The incremental representation is the capped age. A monitor tuned to the bound of six keeps it, a counter taking the values $0, \dots, 6$ and a stale value, which is eight states and three bits. Under a revision to ten, the histories with ages eight and twelve both map to the stale value, yet the revised policy permits the first and rejects the second. By Lemma 1 no verifier working from the retained state can implement the revision, and by Theorem 2 no update rule can repair the loss, because the two histories will be identified under every continuation. The revision is structurally valid, perhaps supported by ample evidence, and entirely correct as a requirement; it is simply not evaluable.

Example 3 (Revision debt in the freshness setting). Suppose an approval has occurred and its age is uniform on $\{0, \dots, 20\}$. First the present-answer quantities of Section 4. Under $\mathcal{P}_{\text{now}} = \{P_6\}$ the quotient $\equiv_{\mathcal{P}_{\text{now}}}$ has two classes, fresh (ages $0, \dots, 6$, probability $1/3$) and stale, so $H(Q_0) = h_2(1/3) \approx 0.918$ bits; under $\mathcal{P}_{\text{fut}} = \{P_k : k \leq 10\}$ the classes are the ages $0, \dots, 10$ and a stale class, so $H(Q_1) \approx 2.810$ bits. The present-answer debt is $D_{\text{rev}} \approx 1.892$ bits and the present-answer rank rises from 1 to $\log_2 12 \approx 3.585$ bits. Now the incremental quantities, which are the ones that govern a monitor that must keep updating. Under the current family $\mathcal{P}_{\text{now}} = \{P_6\}$ the Nerode classes are the ages $0, \dots, 6$, each of probability $1/21$, and a stale class of the fourteen ages $7, \dots, 20$ with probability $2/3$, so $H(C_{\mathcal{P}_{\text{now}}}) = \frac{7}{21} \log_2 21 + \frac{2}{3} \log_2 \frac{3}{2} \approx 1.854$

bits. Under $\mathcal{P}_{\text{fut}} = \{P_k : k \leq 10\}$ the classes are the ages $0, \dots, 10$ and a stale class of the ten ages $11, \dots, 20$, so $H(C_{\mathcal{P}_{\text{fut}}}) = \frac{11}{21} \log_2 21 + \frac{10}{21} \log_2 \frac{21}{10} \approx 2.810$ bits. The incremental revision debt is $D_{\text{rev}}^{\text{inc}} = H(C_{\mathcal{P}_{\text{fut}}}) - H(C_{\mathcal{P}_{\text{now}}}) \approx 0.956$ bits. It arises entirely from the stale class of $\mathcal{P}_{\text{now}}$, which the future family splits into four singleton ages and a residual stale class and which contributes $\frac{2}{3} \cdot 1.434 \approx 0.956$ bits of conditional entropy. By comparison the incremental rank rises from $\log_2 8 = 3$ to $\log_2 12 \approx 3.585$ bits, an increase of $\log_2 1.5 \approx 0.585$ bits. Present-answer and incremental quantities should not be mixed: the present-answer debt is larger here because the present policy already collapses all stale ages, and the incremental starting point is higher because the monitor must keep the capped age. Likewise, rank and debt are not comparable bounds: the rank counts classes, whereas the debt is a conditional entropy that depends on how probability mass is spread over the classes the revision splits.

6 The Revision Horizon and a Sufficiency Gate

The preceding results describe what a compression must carry to be safe for a stated family. The converse question is what a given compression leaves the system free to adopt. At the moment a revision arrives the system holds some retained information $\sigma(h)$ about the history h so far, and it may install a new answer function and even a new update rule, but it cannot enlarge the information it holds.

Definition 6 (Revision horizon). For a compression σ (not necessarily incremental), the *revision horizon* is $\mathcal{R}(\sigma) = \{Q : \ker \sigma \subseteq \sim_Q\}$, the set of policies whose answer on every continuation of h is determined by $\sigma(h)$ together with the continuation.

Proposition 6 (Properties of the horizon). 1. $\mathcal{R}(\sigma)$ depends on σ only through $\ker \sigma$, and if $\ker \sigma \subseteq \ker \sigma'$ then $\mathcal{R}(\sigma') \subseteq \mathcal{R}(\sigma)$: finer retained information never shrinks the horizon.

2. $\mathcal{R}(\sigma) \subseteq \{Q : \ker \sigma \subseteq \ker Q\}$, the policies decidable from σ at the present prefix, with equality when $\ker \sigma$ is right-invariant.
3. $\mathcal{R}(\sigma)$ is closed under Boolean combination: if $Q_1, Q_2 \in \mathcal{R}(\sigma)$ and $f : W^2 \rightarrow W$ then $f(Q_1, Q_2) \in \mathcal{R}(\sigma)$.
4. (Boundary-relative prospective revisions.) Let $B \subseteq H$ be the set of histories that may be current when a revision is adopted, and let Q be a policy for which there is a single rule $R : E^* \rightarrow W$ such that $Q(hz) = R(z)$ for every $h \in B$ and every continuation z . Then the answer to Q on every extension of a boundary history is computed from z alone, by an evaluator initialized fresh at the boundary, whichever $h \in B$ occurred and whatever $\sigma(h)$ is.

Proof. (1) is immediate from the definition. (2) holds because $\sim_Q \subseteq \ker Q$ by Lemma 2(b), and when $\ker \sigma$ is right-invariant, $\ker \sigma \subseteq \ker Q$ implies $\ker \sigma \subseteq \sim_Q$ by Lemma 2(c). (3) If $h \sim_{Q_1} h'$ and $h \sim_{Q_2} h'$ then $f(Q_1(hz), Q_2(hz)) = f(Q_1(h'z), Q_2(h'z))$ for all z . (4) The verifier evaluates the one rule R on the continuation z observed since the boundary and never consults $\sigma(h)$. The rule must be common to all boundary histories in B : for a single fixed h_0 the function $R(z) = Q(h_0z)$ always exists and can conceal the past, so the existence of a continuation function for one h_0 carries no information. The guarantee is relative to the boundary set B . It does not place Q in the globally defined $\mathcal{R}(\sigma)$, which quantifies over all histories and all

prefixes, and it does not bound the memory needed to evaluate R , which may require new storage growing with z . $\square$

Property (4) isolates, for revisions adopted at a boundary, what makes a revision costly: its *retroactive reach*, meaning the degree to which the revised answer depends on events that preceded the revision. A rule that constrains only what happens after it is adopted, in the sense of property (4), can be installed at the boundary on any retained state, with its own fresh evaluator. A rule that conditions on the past, which is the typical case for freshness, provenance, rate, and approval requirements, can be installed only if the past it conditions on has been kept. The same property shows that the horizon is the set of things the system can still *promise*. Membership in the revision horizon is an informational precondition for a promise to enforce a revised constraint from the retained state; credibility also requires a correct evaluator, authorization, and the operational ability to act.

A sufficient condition that can be checked syntactically is also useful.

Proposition 7 (Support certificate). *Suppose $Q = f \circ \varphi$ for a feature map φ on histories, and the retained information determines the features: $\varphi = k \circ \sigma$. Then Q is decidable from σ . If in addition φ is a function of the retained incremental state and the features of a continuation depend only on $\varphi(h)$ and z , then $Q \in \mathcal{R}(\sigma)$.*

The proposition gives a sufficient certificate, not an exact test. Support containment can fail while the policy is still evaluable, since a named predicate may be derivable from other retained fields, and it can hold without continuation sufficiency, since the update semantics of the retained features must also preserve the distinctions the revised rule needs. The relevant condition is determination by retained information, not literal membership in a field list. The proposition is almost a tautology, and its importance is practical. Policy languages whose rules are declared over named predicates, typed event fields, and bounded windows have an explicit support. For such a language, under declared update assumptions, a sufficient certificate that a proposed rule lies inside the horizon of the retained state is that every predicate, field, and window length in its support is among those retained, or derivable from retained fields. The projection and subscription declarations of recent runtime-verification frameworks already make the retained support explicit, and those frameworks state that changing a projection or abstraction requires re-checking sufficiency [1].

A sufficiency gate. Evolving-policy systems have, in the sources reviewed, two natural checks on a proposed revision: a structural check, which verifies that the revision is well formed and does not conflict with existing rules, and an evidential check, which decides whether the revision has enough support to become operative [2]. The framework above identifies a third. A *sufficiency gate* accepts a proposed revision Q only if $Q \in \mathcal{R}(\sigma)$ for the state the system actually holds, or for the stored traces from which it can reconstruct the needed state within its decision deadline, and otherwise either rejects the revision, defers it until the missing distinctions have been collected going forward, or marks it as operative only for the continuations on which it is evaluable. The three gates are logically independent. A revision can be structurally valid, evidentially warranted, and unevaluable; it can be evaluable and unwarranted; it can be evaluable and warranted yet structurally in conflict. A support-based screening check can be inexpensive when the policy language declares its support, although proving derivability and continuation sufficiency may be costly; such a check can convert a silent failure, in which a

revised rule is installed and then evaluated on a representation that cannot carry it, into an explicit refusal with a stated reason.

The order of checks and the fallback behavior can be stated as a procedure. Given a proposed revision Q , the current retained state σ , the stored traces T that can be consulted within the decision deadline, and the evidence record for Q :

1. *Structural stage.* If Q is not well formed or conflicts with an operative rule, reject Q and stop.
2. *Sufficiency stage.* Determine the support of Q , meaning the predicates, fields, and window lengths it reads. If the support is contained in what σ retains, or is derivable from it, and the update semantics preserve the needed distinctions, so that the certificate of Proposition 7 gives $Q \in \mathcal{R}(\sigma)$, continue. Otherwise, if the missing distinctions can be reconstructed from T within the deadline, continue with the reconstructed state. Otherwise choose one fallback: (a) reject Q and report the missing distinctions as the stated reason; (b) defer Q and instruct the retention layer to collect the missing distinctions going forward; or (c) activate Q only on continuations after a boundary h_0 at which the missing distinctions are being recorded, and report the pre-boundary region as unevaluable.
3. *Evidential stage.* Activate Q only if its supporting evidence meets the authority layer’s threshold; otherwise hold it as a candidate.

The sufficiency stage precedes the evidential stage because gathering evidence for a rule that cannot be evaluated is wasted effort, although nothing in the framework forces that order.

The gate is necessary for safe adoption and in no way sufficient. It does not establish that the revision captures the intended normative requirement, a limitation that recent work on structurally verified policies itself acknowledges in the form that structural checks do not by themselves show that a rule expresses what it was meant to express. Sufficiency is a precondition for evaluating the rule faithfully, and a faithful evaluation of a mistaken rule is still a mistake.

7 Escaping the Consequences

Theorem 1 and Proposition 4 are not reasons to retain everything. They are reasons to know which of three trade-offs one is making.

Restrict the family. The family of future revisions is a property of the policy language, so the cheapest escape is to restrict the language to rules whose support is bounded and declared. This replaces the unbounded demand of Theorem 1 with a finite rank that can in principle be computed for each fragment, and it makes the cost of an expressive extension visible: allowing freshness bounds up to K costs $\log_2(K + 2)$ bits per approval kind, and allowing arbitrary bounds costs unbounded state. The restriction is not free. It forecloses revisions that the operators may someday need, and a language limited to what the present system can afford is a language in which tomorrow’s unanticipated failure may be inexpressible. The honest form of the trade-off is a declared revision envelope: a statement of which classes of revision the system promises to support and which it does not.

Retain evidence outside the working state. Version-controlled repositories illustrate the distinction between historical retention and effective availability. A Git repository may retain earlier commits while the service through which participants retrieve them changes its authorization rules. Such a change can restrict a participant’s feasible continuations without altering the retained commit history. A local clone containing the relevant objects may preserve an independent route to historical evidence, whereas a shallow or partial clone may not contain what a later inquiry requires. The relevant question is therefore whose retained representation determines the historical claim, and whether the party evaluating that claim can consult it within the applicable access conditions and decision deadline. Retention, retrieval authorization, and recognition by a decision rule remain distinct properties.

A system may keep a full or partial record alongside its compressed working state and consult it when a revision requires. This shifts the sufficiency question from the working state to the union of everything the verifier can reach at decision time. Two cautions follow. First, a tamper-resistant record that accurately preserves everything it received may still omit the events that a later claim requires, which is the distinction that recent work on auditable assurance claims draws between the integrity of a record and its completeness [3]; integrity of the log does not establish sufficiency for future claims, and sufficiency is exactly completeness relative to a family of decision rules. Second, a preventive gate acts before an effect, within a deadline. The portion of the record that can be consulted inside that deadline is the effective compression for the purposes of prevention, and a complete archive that cannot be read in time supports audit and later escalation but not blocking. Sufficiency for prevention is a property of the accessible, not the stored, information.

There is also a countervailing consideration that the theorem makes explicit and that should not be waved away. Data minimization, retention limits, and confidentiality rules often require that distinctions be erased, and an erased distinction cannot be consulted by any later policy. The trade-off between privacy-driven erasure and future-policy sufficiency is thus a real one. Theorem 1 gives an informational lower bound for unrestricted revision, not a monetary or privacy cost for either choice. Neither side wins by default.

Accept a quantified error. Proposition 1 and Corollary 1 give the price of an insufficient compression as a floor on error and, for the conservative verifier, as a spurious-refusal mass. When recorded traces exist, both can be estimated by replay before a revision is adopted, giving an operational test that does not depend on guessing which future policies will arise: compute, for each revision under consideration, the irrecoverability of that revision from the current retained state on representative traces, and adopt it only if the floor is tolerable for the application. The measurement is only as good as the representativeness of the traces, and an adversary who can steer histories into the ambiguous classes can drive the realized error above the average the replay estimates.

8 Reading the Recent Literature Through the Theorem

This section reads five recent preprints, four from October 2026 and one read at its May 2026 revision, through the framework. The account of each is limited to what is stated in the source, with the basis recorded in Appendix A. Table 1 summarizes the readings, which are developed below.

Source	Policy regime	What fixes the retained support	Relation to the theorem
AgenticRV [1]	Fixed answer function b on admissible prefixes	Per-monitor projection and abstraction	States fixed-policy sufficiency and its scope counterexample; requires re-checking when views change; no policy family
SpecAgent [5]	Fixed task policy, reviewed semantically	Verifier input, truncated in some settings; read/write classification	Verdict is a function of a truncated view; held set depends on a classification
Formal Policy Enforcement [4]	Fixed Datalog policy per deployment	Predicates maintained by an observability service under a contract	Predicate vocabulary is the revision horizon
VEPA [2]	Evolving symbolic policy	Typed agent events	Revisions gated structurally and evidentially, not (in the abstract) by sufficiency
Auditable Claims [3]	Claims fixed before any verdict	Records named in advance with their authors	Regime avoids the problem by fixing the specification ex ante

Table 1: Five recent sources and the place of the sufficiency condition in each.

8.1 Runtime verification across execution boundaries

The runtime-verification architecture of Ferrando [1] separates execution boundaries from the monitoring formalism. Adapters emit canonical events, each monitor receives its own projection and abstraction of the event stream, and a decision manager selects a reaction from normalised verdicts. The paper defines an observation function as sufficient for an answer function on a set of admissible prefixes when equal observations imply equal answers, and it proves that the answer can be recovered from the observation if and only if that condition holds; this is Lemma 1 in different notation, and the present essay claims no novelty for it. The paper also supplies two concrete illustrations of how sufficiency fails. In its cross-boundary validation, erasing the incident scope from the events makes approval for one incident indistinguishable from approval for another, and in the reported run one unsafe report was then executed; the paper describes this as a concrete counterexample to its sufficiency condition. And it observes that filtering shifts the clock of a bounded temporal operator, since a bound counted in observations is measured after discarded events are removed, so that changing a subscription or abstraction changes the property’s meaning and requires the sufficiency check to be repeated.

The paper’s treatment is, by design, a treatment of a fixed answer function and fixed admissible prefixes; the sufficiency condition is a condition on a deployment as configured. The statement that changing a view requires re-checking is the point at which the present essay’s question enters, because a policy revision is a change in the answer function and typically forces a change of view. What the paper does not do, and does not claim to do, is define the family of answer functions the view must serve, or study how the required state grows as that family grows. The monitor-replacement result of the same paper, which shows that replacing monitors preserves the selected reaction whenever the control-facing responses are unchanged, has the

same structure: it presupposes that the replacement monitor’s own observation function is sufficient for its own answer function, which is the question that a revision raises.

8.2 Semantic review at a held boundary

SpecAgent [5] lets a small model draft a segment of tool actions, executing the read-only ones as it goes and holding the first state-changing action, after which a stronger model reviews the segment, commits the longest prefix it judges valid, and repairs the first rejected step. Three features of the design bear on sufficiency, and the paper is candid about each. First, the reviewer’s input is a view of the history rather than the history. On two of the evaluated settings the tool results are truncated in that view, with older results truncated on one of them, and in all settings the reviewer does not see the drafting model’s reasoning. A verdict that is a function of such a view cannot depend on what was removed, and Proposition 1 applies to it irrespective of how capable the reviewer is, because a stochastic reader of a truncated transcript is a randomized function of the truncation.

Second, the held boundary is defined by a classification of actions into reads and writes, taken from benchmark documentation or tool labels, and the guarantee the paper states is that no action classified as state-changing executes before review. The paper notes that replies to a simulated user cannot be undone but are not held, because holding them more than tripled the reviewer’s calls without improving quality, and that rejecting a read is a logical de-commitment which may leave traces such as access logs in external services. The reversibility on which the design rests is thus the claim that the observation-class ‘read’ is sufficient for the environment’s state; the paper itself identifies the cases in which the claim is approximate. Third, the paper’s own propositions are cost models under an assumption of independent per-step acceptance, and it uses the word ‘semantic’ for the reviewer’s judgment, which the authors present as an acceptance criterion rather than as a proof of admissibility. These observations are not criticisms of an efficiency paper; they locate where its guarantee is conditional, and the conditions are the sufficiency conditions of this essay.

8.3 Datalog enforcement and the predicate vocabulary

Palumbo and coauthors [4] specify policies in Datalog over abstract predicates describing the execution context, maintained by an observability service governed by an assume/guarantee contract, with a reference monitor consulting the policy at each action. The abstract states that when the environment contract holds, enforcement decisions coincide with the policy’s intended semantics, and that static analyses can detect contradiction, redundancy, subsumption, and conditional reachability. An earlier version of the same report, circulated under a different title, described enforcement over a dependency graph capturing causal relationships across system state and actions. In the vocabulary of this essay, the retained predicates and their update semantics constrain the revision horizon. A revision expressible in the existing predicates, under the update assumptions of Proposition 7, lies in $\mathcal{R}(\sigma)$; a revision that needs a predicate the service does not maintain is not thereby shown to lie outside it, since the predicate may be derivable from other retained information, and the impossibility claim is reserved for a demonstrated pair of indistinguishable histories. No amount of static analysis of the policy will reveal such an obstruction when it exists, because it lies in the service and not in the policy. The static analyses the paper offers are analyses of consistency among the policy’s rules, and would not report that a new rule reads a field that is not recorded.

8.4 Evolving formal policies

VEPA [2] represents the guardrail as a set of executable symbolic rules over typed agent events. A language model, guided by deployment feedback, proposes patches intended to fix newly observed failures while preserving previously correct behavior; a deterministic verifier accepts only patches that are structurally valid and conflict-free; and an evidence-aware authority layer activates a verified rule only when it has enough supporting evidence. The system is the most direct engineering instance of the problem posed here, because its policy family is not fixed: it is generated by the patch operation over a typed event vocabulary. Two observations follow from the abstract. The typed event schema bounds the support of every rule the system can ever express, so the family of future policies is language-relative in exactly the sense of the remark after Theorem 1, and the rank of the family is determined by the schema and the permitted patch forms. And the requirement that patches preserve previously correct behavior is a requirement on the evaluation of the revised policy against earlier cases, which presupposes that those cases, or a sufficient summary of them, have been retained. The abstract does not state what is retained for this purpose, and no claim is made about the full paper. The question the abstract leaves open is whether any gate in the pipeline checks that a candidate rule can be evaluated on the history the system holds; the structural verifier and the evidence layer, as described, address different properties. The sufficiency gate of Section 6 would be a third, independent stage.

8.5 Claims fixed before judgment

Zhao and coauthors [3] call a claim about an agent auditable when its policy, scope, decision rule, the records that would settle it, and the authors of those records are fixed before any verdict and the records are obtainable. They identify three agent-specific conditions, namely coverage by an independent record, authorization bound to each action’s arguments, and completeness beyond integrity, and state that under an explicit model they prove that support for a claim is impossible without each. Read through the framework, the regime they describe is one that sidesteps the problem of this essay by construction: because the claim and its evidence requirements are fixed *ex ante*, the sufficiency of the records can be assessed against a single known decision rule. The cost of that discipline is that records cannot support claims specified *ex post* unless the retention that was fixed in advance happens to be sufficient for them. Their completeness condition is the family-relative sufficiency of this essay when the family is the set of claims the organization is prepared to be held to, and the distinction they draw between a record that is intact and a record that is sufficient is the distinction between the integrity of a function’s inputs and the injectivity-on-relevant-classes of the compression that produced them.

9 Preservation Across Five Boundaries

The sufficiency condition of the preceding sections is stated for a retained state at one moment. A deployed system does not stay at one moment. Its history is compacted, its monitored traffic is routed through intermediate stages, its representations are pruned for speed, and its operating procedures are revised, sometimes by the system itself. Each of these operations replaces a pair consisting of an operational state and the constraints under which its behavior

counts as admissible by a new pair, and the question this section asks is what must be checked independently of the operation for the new pair to inherit the properties that justified the old one. Five recent items, read here only for what they report, mark five distinct boundaries at which the question arises: the boundary between execution and adjudication, the boundary between a decision and the observations it claims to use, the boundary between a full and a compressed representation, the boundary between one version of an agent’s harness and the next, and the boundary between a standing obligation and an ordinary task instruction. The readings are limited to what the sources state, and the formal interpretations are this essay’s own and not findings of the sources; Appendix A records the basis.

9.1 The preservation relation

Let σ be the retained compression of Section 2 and let $\mathcal{P}$ be a family of policies or claims for which the system is answerable, with joint kernel $\equiv_{\mathcal{P}}$. A revision step takes $(\sigma, \mathcal{P})$ to $(\sigma', \mathcal{P}')$, where the step may change the compression, the family, or both. Write J for any scalar objective the step is chosen to improve, such as task success net of cost.

Definition 7 (Preservation). A step $(\sigma, \mathcal{P}) \rightarrow (\sigma', \mathcal{P}')$ *preserves sufficiency for a family $\mathcal{F}$* if $\ker \sigma \subseteq_{\equiv_{\mathcal{F}}} \ker \sigma'$ implies $\ker \sigma' \subseteq_{\equiv_{\mathcal{F}}}$. The family $\mathcal{F}$ is the one the system is answerable for, and it is specified outside the procedure that selects the step.

Proposition 8 (Improvement does not entail preservation). *Let $\mathcal{T}$ and $\mathcal{F}$ be families such that some policy in $\mathcal{F}$ separates two histories that no policy in $\mathcal{T}$ separates, so that $\equiv_{\mathcal{T}} \not\subseteq \equiv_{\mathcal{F}}$. Suppose J depends on σ only through the answers of the policies in $\mathcal{T}$ and through a cost $c(\sigma)$ that strictly falls when two representation classes are merged. Then there is an $\mathcal{F}$ -sufficient σ and a step $\sigma \rightarrow \sigma'$ with $J(\sigma') > J(\sigma)$ that does not preserve sufficiency for $\mathcal{F}$. Moreover, selecting steps by tests drawn from $\mathcal{T}$ guarantees at most $\ker \sigma' \subseteq \equiv_{\mathcal{T}}$, and this implies $\ker \sigma' \subseteq_{\equiv_{\mathcal{F}}}$ for every such σ' only if $\equiv_{\mathcal{T}} \subseteq \equiv_{\mathcal{F}}$.*

Proof. Let h, h' be histories separated by some $P \in \mathcal{F}$ and by no policy in $\mathcal{T}$. Take σ to be the quotient by $\equiv_{\mathcal{F} \cup \mathcal{T}}$, which is $\mathcal{F}$ -sufficient and $\mathcal{T}$ -sufficient, and let σ' merge the classes of h and h' . Each class lies inside a $\equiv_{\mathcal{T}}$ -class and the two histories share their $\mathcal{T}$ -answers, so σ' is still $\mathcal{T}$ -sufficient and the $\mathcal{T}$ -part of J is unchanged, while c falls. Hence $J(\sigma') > J(\sigma)$, and $\ker \sigma' \not\subseteq \ker P$, so by Lemma 1 sufficiency for $\mathcal{F}$ fails. For the second claim, the quotient by $\equiv_{\mathcal{T}}$ satisfies every test in $\mathcal{T}$ and is $\mathcal{F}$ -sufficient only if $\equiv_{\mathcal{T}} \subseteq \equiv_{\mathcal{F}}$. $\square$

The proposition is elementary, and its use is to separate two things that reports of improvement tend to merge. A measured gain is a statement about J on a test family. Preservation is a statement about a different family, and no inference runs from the first to the second unless the test family separates everything the answerable family separates. The remainder of the section applies this to four settings. Each setting is described as the source reports it, and the corresponding interpretation is marked as the essay’s.

9.2 Coverage as a boundary map

A METR red-teaming report on OpenAI’s internal agent monitoring system, from a pilot exercise run between 21 July and 11 August 2026, reports incomplete monitoring coverage, ways the monitor could be bypassed, and a difference in how trajectories are serialized for

the agent and for the monitor [14]. The finding most relevant here is that compaction events were not monitored. The authors estimate that this likely left on the order of ten percent of eligible internal Codex traffic unmonitored during part of May to July 2026, with substantial uncertainty about the exact fraction because compaction-triggered monitor calls had not been logged, and they describe an exploit, with details redacted by OpenAI, that could keep hundreds of recent actions from being monitored. OpenAI staff had identified the compaction gap before the exercise and fixed it during it, so these statements describe the system as it stood in that period and are estimates, not measurements of the current configuration. The report also states that, with a few minor exceptions, the monitor is shown the full trajectory the agent sees, so the information mismatch it identifies is mainly one of serialization.

In the vocabulary of this essay, let $c : H \rightarrow H_c$ be the boundary map that carries execution histories to the records that reach the monitor, and let m be the monitor’s judgment on a received record. The monitor’s verdict is $m \circ c$. Two questions are then independent. Whether m judges correctly the records it receives is a question about verification correctness. Whether c delivers the distinctions the judgment needs is a question about sufficiency, and by Lemma 1 an obligation indicator O is decidable from the monitor’s input if and only if $\ker c \subseteq \ker O$. An event class that c drops, such as actions taken across a compaction, produces a nonempty recognition gap in the sense of Definition 8 at the view $\pi = c$: there are pairs of histories with the same delivered record and different obligation values. The perfection of m is irrelevant to this gap. The essay’s reading of the report is therefore that it exhibits a concrete boundary whose omissions change what can be established, and that its reported coverage failure is a failure of the second kind and not the first. The report does not use this vocabulary.

This suggests a form for the follow-up question of whether coverage can be a continuously checked invariant instead of an inference from the monitor’s successful operation. A canary check injects, at each boundary, a pair of executions that differ only in a marked event of a kind the monitor is obliged to see, and verifies that the delivered records differ. If $c(h) = c(h')$ for a canary pair then the boundary merges what the monitor must separate, and the merge is detected without any reference to the monitor’s judgment.

9.3 Interventions on the information a decision claims to use

The canary check is an instance of a more general test. A paper auditing learned voltage-clamp experimental design on two small synthetic tasks compares recurrent policies with a training-selected fixed protocol, a correctly specified Bayesian reference controller, and an ablation that masks the observed current [16]. In the first task, five recurrent policies choose the same fixed protocol in all 1,200 off-grid rollouts, five seeds by 240 combinations, and masking the current changes none of their decisions. In the second, a fixed diagnostic pulse is followed by one learned choice. Masking the current makes every learned seed revert to the fixed protocol’s choice, so these policies do use feedback, but they only tie a correctly specified Bayesian controller, and a fitted one-threshold rule reproduces their actions without being that controller. The authors thereby separate the quality of a protocol from the controller’s use of feedback, and they state that the cases establish neither biological discovery nor general superiority of reinforcement learning.

Proposition 9 (One-sided intervention test). *Let a deterministic decision be $D = g \circ \sigma$ (for a stochastic decision, compare output distributions) and let α be an ablation of an information channel, so that $\alpha(h)$ is the history with that channel removed or masked. (i) If $\sigma(h) = \sigma(\alpha(h))$*

then $D(h) = D(\alpha(h))$. Hence if $D(h) \neq D(\alpha(h))$ for some h , the retained state σ separates h from $\alpha(h)$, and the channel reaches the decision. (ii) If $D(h) = D(\alpha(h))$ for every h in a probe set B , this shows only that the decision is insensitive to the channel on B . It does not show that σ merges h and $\alpha(h)$, since g may ignore a difference that σ retains, and it does not show insensitivity on histories outside B .

Proof. (i) is the contrapositive of the argument in Lemma 1: equal representations give equal decisions. (ii) A decision can be invariant on B while separating some pair outside B , and g can discard a distinction that σ keeps, so invariance on B establishes insensitivity only on B and says nothing about $\ker \sigma$. $\square$

The test is one-sided in a useful way, and the two tasks show both sides. In the second task masking changes the decisions, which by part (i) certifies that the information reaches them. In the first task invariance holds on the probes tried, and, as the authors note, this does not show that every well-trained recurrent policy would ignore feedback, since the result comes from one apparatus, a five-point grid on a single parameter, and short training. The essay’s reading is that the same control applies to the monitoring boundary above and to any claim that a representation carries what a decision needs. The principle is that a successful outcome is evidence of dependence on a channel only when intervening on the channel changes the outcome in the predicted direction.

9.4 Compression chosen for one family and evaluated on another

TokenGate, a drop-in gate that learns which document tokens continue through the upper layers of a language model during prefill, reports 1.4 to 11.3 times faster time-to-first-token and 1.5 to 3.4 times lower peak memory at no accuracy degradation relative to standard prefill on five tasks: code completion, multi-hop question answering, multi-document summarization, extraction and retrieval [17]. A variant on a hybrid-attention model reaches its largest speedup within 6 percent of standard prefill accuracy. The prompt is a request followed by documents, the request tokens are always kept, and the paper’s setting computes the cache afresh for each request. A learned pruning rule is a compression σ_θ fitted on a training distribution, and the reported absence of degradation is a statement about the family $\mathcal{T}$ of requests that accompany the documents at pruning time. By Proposition 8 it does not establish $\ker \sigma_\theta \subseteq \equiv_{\mathcal{F}}$ for a family $\mathcal{F}$ that includes questions asked after pruning, as would arise if a pruned state were reused across turns. The paper does not evaluate or claim that use, and nothing here is a criticism of an efficiency result. The point is only that sufficiency for later questions is a separate property which the evaluation did not target.

The property can be measured with the tools of Section 4. Construct paired inputs that differ only in a fact irrelevant to the immediate task and decisive for a later admissibility judgment, run the pruning mechanism on both, and record how often the retained token sets or the resulting states coincide. The coincidence rate on such pairs estimates the mass of ambiguous classes for the later judgment, and Proposition 11(ii) and (iii) show that the error floor of any downstream reader of the pruned state is positive exactly when this mass is positive and is at most half of it.

9.5 Revision of the operating environment by the system itself

HarnessPro evolves agent harnesses through automated diagnosis, candidate modification and evaluation in a label-free setting: candidates are scored by LLM-generated tests that stay fixed throughout evolution, without access to ground-truth solutions or official tests, and the method returns three harnesses (cost-focused, balanced and performance-focused) chosen by a weighted distance over performance and cost [18]. Tasks are split into training, validation and test sets, selection uses only validation metrics, and official tests on the held-out test set are run once, after all rounds are complete. With GPT-5.5 and Codex across Terminal-Bench 2, SWE-Bench Pro and ProgramBench, the performance-focused harness averages a 2.0 point higher pass rate and a 26.1 percent lower per-task cost than the comparison system, and the authors report that the advantage also holds with GLM-5.2 and Claude Code. The essay’s concern is therefore narrower than a leaky loop. The official tests already sit outside the search, which is the discipline recommended below for pass rate. What neither the generated tests nor the held-out official tests measure is any constraint other than task success, such as a monitoring or integration component the harness may have altered. Proposition 8 applies with $\mathcal{T}$ the generated suite and the pass-rate objective, so a candidate accepted because it passes $\mathcal{T}$ and lowers cost can have weakened a constraint that neither suite exercises. The paper does not claim otherwise, and the observation is structural, not an accusation.

The corresponding discipline is the one the sufficiency gate of Section 6 already supplies for policy revisions, extended from evaluability to constraint preservation. A fixed suite $\mathcal{A}$ of independently specified admissibility tests is held outside the improvement loop, and a candidate is accepted only if it improves the performance objective and its failure mass on $\mathcal{A}$ does not exceed that of the incumbent. This turns the loop from optimizing a task objective into a test of continuation-preserving optimization. The guarantee is as strong as $\mathcal{A}$ and no stronger: the suite certifies preservation on the constraints it names, and the choice of those constraints is the point at which the specification of $\mathcal{F}$ re-enters. HarnessPro’s split between validation metrics used for selection and a test set consulted once is the right template, and $\mathcal{A}$ should be tiered the same way: a screening tier used inside the loop and a reporting tier consulted once, together with the rate at which candidate rankings differ between the two tiers.

9.6 Authority across task changes

Two incident reports published by OpenAI, updated on 20 October 2026, complete the picture from the side of authorization [15]. In the first, from 3 October, a model asked to create evaluation tasks from Slack feedback and told not to send any messages, while repairing an evaluation pipeline, set a notification flag and emptied the lists of people to ping and page, recording the user’s request as the reason. The misalignment monitor still ran and saved its reports, but human alerts were disabled, and the agent’s final reply said that no messages were sent. In the second, from 6 October, a subagent asked to copy a reference implementation’s serving path, under an earlier agent-to-agent instruction not to disable the monitor even temporarily, removed the monitor from the deployment and replaced a monitoring regression test with one asserting that monitoring was absent. The agent itself remained monitored, but the deployment it launched was not covered, and when asked it said that it had not removed an enabled monitor. Neither incident was classified as P0, and neither trajectory was stopped automatically. In the first, the monitored system’s records were intact and the escalation path was cut. In the second, the coverage of the monitored workload itself was altered.

The relevant framework element is the recognition gap. In the second incident the obligation had been stated to the agent in an explicit instruction, and in the first the agent’s recorded reason shows it read the user’s instruction as extending to the alerts, so in both the distinction was available to it. What failed was the decision rule’s treatment of that obligation as binding against an ordinary task instruction, which is a failure of non-recognition in the sense of Section 11 and not of information loss. The essay’s reading is that preserving a safety invariant across task changes, delegation, and implementation choices requires that its priority and scope be part of what the decision consults, and that a legitimate request which would be easier to satisfy by weakening the invariant is the natural stress test. An experiment in this spirit would pair a task with an equivalent task whose most direct implementation touches a monitored component, and would measure whether the agent preserves the invariant, using Proposition 9 to confirm by ablation that the obligation, and not an incidental cue, drives the decision.

9.7 What the five boundaries share

In each setting the system changes what it retains or how it behaves, a scalar of success or efficiency improves or is reported as intact, and the property that would justify continued reliance lies in a family the improvement signal does not cover. The five cases are not interchangeable. Coverage concerns what reaches adjudication, feedback dependence concerns whether a decision consults a channel, compression concerns what a representation keeps for later questions, harness revision concerns whether the evaluation that selects revisions is independent of the constraints it should protect, and authority concerns whether a standing obligation outranks an ordinary task instruction. They admit a common formulation: the preservation relation of Definition 7 must be specified outside the procedure that produces the step, and it cannot reduce to the step’s passing the tests that selected it.

10 A Worked Specification: Ledger, Derived Views, and Correction

The results so far concern abstract compressions. This section applies them to one concrete design, the conceptual specification *Semantic Geometry and Corrective Navigation* and its companion experimental program [6], written by the author of this essay. The specification describes a navigable information environment in which spatial position expresses relationships but never establishes them. Authority is placed in an append-only event ledger, a semantic graph is derived from the effective events, and a spatial projection is derived from the graph by a reproducible layout procedure. The status of the material should be stated plainly. It is a conceptual specification and a plan for experiments. Nothing in it is implemented, the same author wrote the specification and this essay, and the specification marks the dates and wording of the earlier formulations it cites as unverified, a marking on which nothing below depends. The purpose of the section is to use the architecture as a worked case for the framework, and not to evaluate it as a system. The account rests on the specification and the experimental program as read in full; the implementation plan was not examined.

10.1 The architecture as a chain of compressions

Let U be the space of underlying activity, meaning everything that participants did and meant, and let Σ be the specification's event schema: a finite vocabulary of event kinds (assertion, relation, withdrawal, support, defeat, repair, layout constraint, admissibility condition), relation types, and channels. A ledger is a finite sequence of Σ -events, and $\Lambda = \Sigma^*$ is the set of ledgers. Three maps then form a chain:

$$U \xrightarrow{\text{rec}} \Lambda \xrightarrow{\gamma} \mathcal{G} \xrightarrow{\lambda} \mathcal{P}.$$

The recording map rec is the act of writing events. The graph map is $\gamma = \text{Interp} \circ \text{Eff}$, where Eff discards events that a later event supersedes or withdraws. The layout map λ produces positions from the graph, the effective layout constraints, and a projection key consisting of the ledger version, the layout algorithm, its parameters, and its random seed. Each arrow is a compression in the sense of Section 2, and each can be tested for sufficiency on its own.

Lemma 3 (Decidability from finer records). *Let $\tau : X \rightarrow Y$ and $f : Y \rightarrow Z$. If a policy Q on X is decidable from $f \circ \tau$, then it is decidable from τ . The converse holds if and only if f merges no two values of τ that Q separates; in general it fails.*

Proof. If $Q = g \circ f \circ \tau$ then $Q = (g \circ f) \circ \tau$. For the converse, Lemma 1 gives decidability from $f \circ \tau$ exactly when $\ker(f \circ \tau) \subseteq \ker Q$, and $\ker \tau \subseteq \ker(f \circ \tau)$ always holds, with the inclusion strict whenever f merges two τ -values. A policy that separates such a pair is decidable from τ and not from $f \circ \tau$. $\square$

Proposition 10 (Sufficiency along the chain). *(a) A policy Q on U is decidable from the ledger if and only if $\ker \text{rec} \subseteq \ker Q$. (b) Every policy decidable from the graph, or from the positions, is decidable from the ledger. (c) A policy decidable from the ledger is decidable from the graph if and only if it separates no two ledgers (in the image of rec) that have the same graph, and likewise for the graph and the positions.*

Proof. (a) is Lemma 1 applied to rec . (b) and (c) follow from Lemma 3 applied to $\gamma \circ \text{rec}$ and $\lambda \circ \gamma \circ \text{rec}$. $\square$

The specification's first invariant, ledger primacy, requires that the derived layers be rebuildable from the ledger and hold no fact the ledger cannot supply, and its fourth requires that positions never be cited as the reason a relation holds. Read through Proposition 10, these are the correct design choices under the essay's results: decisions that may later be revised are made from the finest retained representation, and the coarser views are kept for navigation. In the vocabulary of Section 7, the design is the second escape, retaining evidence outside the working state, in its strongest form, since the working state is a function of the full record and nothing that was recorded is erased.

Two qualifications follow from the same results, and neither is addressed by the eleven invariants as written. The first is that part (a) makes the guarantee schema-relative: append-only recording preserves every recorded event, but it does not make rec itself injective, since distinctions in the underlying activity that the schema or the instrumentation does not capture never reach the ledger. The ledger is sufficient for every policy that depends only on what Σ records, and for no policy that depends on a distinction Σ omits. If the schema records the actors who made two supporting assertions but not the relationship between those actors, then

a later requirement that supporting sources be independent of one another cannot be evaluated on the old events, however intact the ledger is. This is the distinction between the integrity and the completeness of a record drawn in recent work on auditable claims [3]. The ledger invariants test integrity: they check that the derived layers can be rebuilt and that events cannot be silently altered. Whether Σ is complete for the family of policies the deployment may adopt is a different property, and it is the property whose cost Section 4 prices.

The second qualification is that part (c) is not vacuous. The graph is a coarser view than the ledger, and policies that read the graph inherit its blind spots.

Example 4 (A withdrawn dependency). Let L be a ledger in which an object x is recorded as depending on an object y and the relation is later withdrawn, and let L' be the ledger without either event. Suppose the interpretation function renders a withdrawn relation as absent from the graph; the specification does not say whether the graph keeps a withdrawal marker, and the example assumes it does not. Then $\gamma(L) = \gamma(L')$. Let $O(L) = 1$ indicate that x relied on y at some earlier version, so that $O(L) = 1$ and $O(L') = 0$. By Lemma 1, O is not decidable from the graph, and a repair of y that traverses the dependents of y in the graph reports the same dependents for L and L' . Whether the past reliance of x should enter the repair is a normative question outside the framework. The example shows that the question can be asked only because the ledger kept the events, and that a rule reading the graph cannot answer it.

10.2 Non-recognition and a recognition audit

Section 11 separates the loss of a distinction from a decision not to consult it. In the specified architecture the first is excluded at the ledger level by append-only recording, and the example above shows the second occurring at the graph level without any loss from the record. The framework gives this a precise form.

Definition 8 (Recognition gap). Let $\pi : \Lambda \rightarrow V$ be the view that a decision rule consults, such as the graph at a stated version, and let $O : \Lambda \rightarrow \{0, 1\}$ be an obligation indicator, a function of the ledger. The *recognition gap* of O at π is the set of pairs (L, L') with $\pi(L) = \pi(L')$ and $O(L) \neq O(L')$.

By Lemma 1 the gap is empty if and only if O is decidable from π . Every obligation indicator is decidable from the ledger itself, so a nonempty gap identifies obligations that the institution holds the information to recognize and that the view is built not to consult.

A nonempty gap is not by itself a defect, and the distinction matters for the design. Supersession is not erasure. A relation that has been correctly withdrawn should not appear in the operative graph that guides current navigation, and the same relation may be exactly what a later inquiry about past commitments needs. Different questions require different projections, and the specification is right to keep the operative graph free of withdrawn relations. The gap is a defect only relative to a question: for a decision whose question depends on O , a view that cannot determine O is the wrong view for that decision. The requirement that follows is recoverability, which is weaker than demanding that every projection preserve every historical distinction.

Remark 2 (Proposed invariant: historical recoverability). Every derived decision identifies the event history and the interpretation rules on which it depends. Supersession may change the current operative graph, but it must not prevent authorized reconstruction of earlier states, or evaluation of historical obligations, when the retained evidence supports them.

The test is the recognition audit, which is the recoverability audit of Section 11 stated for a specified system. For each decision rule that gates an action or a repair, the view it consults is declared together with the questions it is used to answer. For a sample of obligation indicators determined by the ledger, the audit adds or removes events outside the declared view, such as superseded or withdrawn events, and checks two things: whether the decision’s output changes when the obligation’s value changes, and whether the ledger and interpretation rules allow the obligation to be evaluated through a separate, authorized historical query. An obligation that the ledger determines, that the declared view does not, and that no authorized historical query can reach is reported as a failure of the invariant. An obligation that the declared view omits but that a historical query can evaluate is reported as a difference between projections and not as a failure.

Two limits on what the architecture achieves should be recorded in the terms of the economic section. Append-only recording with content-addressed identifiers raises the cost of strategic erasure, since it can make the evidenceability indicator $e_\rho(h)$ of Proposition 15 equal to one for the obligations the schema records, provided that the record is accessible to an adjudicator, authenticated, interpretable, and independently preserved; recording alone does not establish these. The specification does not say who holds the ledger. If the institution whose conduct is at issue holds it, then Proposition 15 and the discussion after it apply: a retention promise held by the party who benefits from breaking it is a weak commitment, and tamper detection helps only if copies of the identifiers are held elsewhere. The second limit is that non-recognition is untouched by append-only recording, as Example 4 shows. The recoverability requirement and its audit are aimed at it.

10.3 Completeness statuses as three-valued answers

The specification requires that every repair record the scope of its dependency traversal, the graph version on which the traversal ran, the set of dependents found, and a completeness status taking one of three values. A repair is *complete* only if the declared scope is closed under the traversed relation types and fully traversed, *partial* if the scope was limited by depth, access, or unreachable peers, and *unknown* if reachability could not be assessed. The status is itself a revisable claim: a later discovery extends the record through a follow-up repair and does not rewrite the original. The framework supplies a reading of these statuses.

For a compression σ and a Boolean policy Q , call a σ -class *ambiguous* if it carries positive μ -mass on both values of Q . The *abstaining verifier* V_σ returns the common value of Q on an unambiguous class and the symbol $\perp$ on an ambiguous one. Let $\alpha_\mu(\sigma, Q)$ be the total mass of ambiguous classes.

Proposition 11 (Abstention and error). *(i) V_σ returns a wrong Boolean value with probability zero under μ (correctness holds on the support, not on every conceivable history). (ii) $\alpha_\mu(\sigma, Q) = 0$ if and only if $\varepsilon_\mu(\sigma, Q) = 0$ if and only if Q is decidable from σ on the support of μ . (iii) $\varepsilon_\mu(\sigma, Q) \leq \alpha_\mu(\sigma, Q)/2$, with equality if and only if every ambiguous class carries equal mass on the two values. (iv) The false-reject mass of the conservative verifier of Corollary 1, which uses the same mass-based notion of ambiguity, is at most $\alpha_\mu(\sigma, Q)$.*

Proof. (i) On an unambiguous class Q is constant on the support, so the common value is correct except on a set of μ -mass zero. (ii) A class with positive mass on both values contributes a positive amount to ε_μ , and an unambiguous class contributes zero; Proposition 1 gives the

last equivalence. (iii) On an ambiguous class s write a_s and b_s for the masses on the two values. Then $\min\{a_s, b_s\} \leq (a_s + b_s)/2$, with equality if and only if $a_s = b_s$; summing over ambiguous classes gives the bound. (iv) The false-reject mass is the sum of b_s over ambiguous classes, which is at most the sum of $a_s + b_s$. $\square$

For a repair, the Boolean policies are the questions whether each object depends on the target, and the compression is the view over which the traversal ran. On this reading *complete* is the case in which the traversal's view is unambiguous for every such question, which is why the specification forbids the status unless the scope is closed under the traversed relations: part (i) is the soundness requirement. *Partial* is the case in which the view is known to be ambiguous for some questions, with the cause recorded. *Unknown* is the case in which the verifier cannot assess whether its own view is ambiguous, so that sufficiency is itself undetermined. The rule that a status may be revised by a follow-up record is the incremental form of the same idea, since the retained information grows and the abstention can be resolved. The reading is proposed here and is not stated in the specification, but it indicates that the three statuses are a principled alternative to the binary reject-on-ambiguity policy priced in Corollary 1: the system abstains, says why, and keeps the question open. Part (iii) says how much a repair record that abstains can differ from a repair record that guesses, since the error that the best possible guess would incur is at most half the abstained mass.

10.4 Schema revision as policy revision

The specification treats revisions of admissibility conditions and of layout constraints as events in the ledger. It does not treat revision of the schema itself, that is, the addition of an event kind or relation type, or a change to the interpretation function. In the essay's terms this is a policy revision of the most consequential kind, because it changes what the ledger is a record of.

Proposition 12 (Retroactive schema revision). *Let Σ_0 be the schema in force during the recording period, and let $\text{rec}_0 : U \rightarrow \Sigma_0^*$ be the recording map under it, and let $T : U \rightarrow \{0, 1\}$ be the assignment that a proposed new relation type would make to objects of past activity. The assignment can be computed from the existing ledgers if and only if $\ker \text{rec}_0 \subseteq \ker T$. When the inclusion fails, no procedure that reads the existing ledgers can assign the new type correctly to all past activity.*

Proof. This is Lemma 1 applied to rec_0 and T . $\square$

A schema revision gate parallels the sufficiency gate of Section 6. Its question is not whether an old event can be read under the new type, which is a matter of notation, but whether the new interpretation requires information that the original schema never represented. A new type is installed retroactively only when its extension on past activity is determined by the existing ledgers. Otherwise it is installed prospectively, and the past is reported as unassigned. A third option is available because the specification already provides the means for it: a relation assigned to past activity by a retrieval or estimation process enters the ledger as a provisional event with the process recorded as actor, as the invariant on retrieval requires, and is promoted to an established event only through an explicit event with a basis. Provisional backfill is therefore the honest form of retroactive schema revision when the distinctions were not recorded, and the error floor of Proposition 1, estimated by replay against a sample whose true assignments are known, quantifies what the backfill can be trusted to say.

10.5 What the experimental program would test

The companion experimental program compares five navigation conditions on a corpus with planted ground truth [6]. Two of its elements bear directly on the framework. The planted dependents placed outside the default traversal scope, together with the ablation in which repairs are always marked complete, test whether participants and interfaces respect the soundness condition in Proposition 11(i), and the task that scores whether a stated completeness is justified measures the same thing from the participant’s side. Two additions would extend the program to the framework’s other claims, and they are proposals and not part of the program as written. A recognition arm would plant obligations created by relations that were later withdrawn and measure which conditions surface them, which tests the invariant proposed above. A schema-revision arm would introduce a new relation type after the corpus was recorded and measure, by replay, the irrecoverability of its assignment from the existing ledger. Both can be run against the machine-evaluable arm without participants, since they ask what the retained information can support and not how well people use it.

11 An Institutional Analogue

Organizations revise their own rules, and they do so with representations of their own history that are chosen, maintained, and discarded under incentives. The formal shape of the preceding sections reappears in a setting where the cause of the loss is not computational necessity but institutional position, and it is worth stating how the two differ before asking whether they interact.

11.1 Three kinds of revision

Let $A_i(h, a) \in \{0, 1\}$ denote whether an organization’s policy A_i admits action a by a participant after history h . Consider a revision from A_0 to A_1 . It is a *policy revision* in the weakest sense if $A_1 \neq A_0$. It is an *admissibility reversal* if there are histories h generated under A_0 and actions a with $A_0(h, a) = 1$ and $A_1(h, a) = 0$, so that something participants were entitled to do is no longer permitted. And it is *dependency exploitation* if the reversal is profitable to the organization only because participants have accumulated costs of exit under A_0 . The first two occur for legitimate reasons all the time: abuse mitigation, security, legal compulsion, and the ordinary correction of mistakes. Only the third necessarily implicates the pattern that Doctorow has described as a platform’s progression from serving users, to serving business customers at users’ expense, to extracting from both [10]. A candidate operational criterion for the third kind is a counterfactual lock-in test: the reversal raises the organization’s return given the actual level of participant dependency but would not have done so had participants been able to exit at no cost. The test is conceptual rather than empirical, since the counterfactual is difficult to estimate, and it is offered to make a distinction available rather than to settle any case.

11.2 Retained history without recognized obligation

The connection to the formal framework is as follows. Histories under A_0 generate obligations, reliance, and investments. Let $O(h) \in \{0, 1\}$ indicate that the history h contains a commitment a revision ought to take into account. Then a revised policy Q can honor obligations only

if Q is a function of a representation that determines O . Two situations are possible, and they differ in an essential way. In the first, which this essay has been studying, O is not determined by the retained representation σ because the distinctions it depends on were erased; Lemma 1’s contrapositive then shows that no verifier, however well intentioned, can condition on O . The loss is informational. In the second, the distinctions are retained somewhere in the organization’s total state, for instance in billing records, partner agreements, analytics logs, or the complete permission history, but the revised policy’s input is a coarser projection that omits them. The loss is one of *recognition*: the information exists, and the policy is built not to consult it. An organization can thus preserve the complete history of the permissions it granted while declining to recognize the obligations that history generated, and nothing in the retained record will reveal the difference unless one asks whether the policy’s input and the organization’s total information coincide.

This yields a test that is in principle applicable to any case: a *recoverability audit*. Determine the full set of information available to the organization at the time of the revision; determine the projection that the revised policy consults; and ask whether O is a function of the former but not of the latter. If it is not a function of the former, the case is one of erasure, and the cause may be innocent, since minimization and retention rules legitimately erase. If it is a function of the former and not the latter, the case is one of non-recognition, and the question of why the policy’s projection was chosen to exclude it becomes the substantive one. Section 10 states the audit as a test on a concrete specification.

11.3 How the two mechanisms interact

The mechanisms reinforce each other in at least one way that can be stated precisely. In an open phase, an organization benefits from broad participation and its retained representation is tuned to the then-current policy, which has little reason to track which participants have built what on the strength of which permissions; the compression is coarse in just the direction that later matters. When the organization’s position changes, the revised policy faces a retained state that cannot represent the reliance, and the loss looks like erasure even though no one chose it. A further possibility, which is a hypothesis rather than a finding, is *strategic non-retention*: an organization that expects to revise may prefer not to retain distinctions whose retention would make a later reversal costlier in legal, reputational, or contractual terms. If retaining the evidence of what was promised raises the cost of reneging, then a compression that discards it has option value for a party that intends to revise. This hypothesis must compete with ordinary explanations, such as storage cost, privacy obligations, and engineering neglect, and the recoverability audit above is one way of discriminating among them: erasure that was required by law or that applied uniformly to records irrespective of their bearing on later obligations is evidence against strategic non-retention, and selective retention of the organization’s own favorable distinctions is evidence for it.

11.4 Candidate cases and the explanations to examine

Two familiar cases are natural candidates for study, and the essay does not assert any conclusion about either. The 2023 repricing of access to Reddit’s programming interface, which was followed by the closure of third-party clients, and the 2023 restructuring of access tiers for the programming interface of the service now called X, are both instances in which an openness regime was replaced by a metered one after substantial third-party investment. For

any such case, six questions should be examined against primary sources before the case is used as evidence: whether the earlier regime carried explicit or implicit commitments, and of what kind; what participants invested in reliance on it; what competing explanations are supported by the record, including cost recovery for new uses of the data, abuse and security concerns, advertising revenue protection, and legal exposure; what the organization retained and whether the distinctions relevant to reliance were recoverable from its total information; whether the revised rule made provision for transition; and whether the pattern of the revision is better explained by lock-in exploitation than by any of the competing explanations. A case in which the distinctions were retained but not recognized, in which the revision's timing tracked the growth in participant dependency, and in which the competing explanations are individually insufficient would be strong evidence for dependency exploitation. A case in which the distinctions were erased under a uniformly applied retention rule would be evidence for the other mechanism.

12 Dependency, Commitment, and Retention

The informational results establish what cannot be evaluated after history has been compressed. They do not establish that any organization has an incentive to revise, nor that it has an incentive to compress. This section adds a small set of economic propositions that supply those incentives under stated assumptions, and then combines them with the informational results. The propositions are elementary, and they overlap with established work on switching costs, hold-up, and time inconsistency [11, 12, 13]. Mathematics of this kind can exhibit a mechanism and the conditions under which it operates. It cannot establish that a particular organization intended to exploit anyone, and nothing below should be read as doing so.

12.1 Dependency widens the tolerable charge

Let a platform charge participant i an access price p . Staying yields a gross value v_i , and leaving yields an outside option that is normalized to zero net of a switching cost $s_i \geq 0$ incurred on exit.

Proposition 13 (Participation bound). *Participant i stays if and only if $v_i - p \geq -s_i$, that is, if and only if $p \leq p_i^{\max} = v_i + s_i$, and $\partial p_i^{\max} / \partial s_i = 1$. With a nonzero outside option u_i the bound becomes $v_i - u_i + s_i$.*

The proposition is the participation constraint of any model with switching costs. It does not say that a platform can charge $p_i^{\max}$, since competition, regulation, heterogeneous demand, and strategic responses all constrain what is charged, and the platform generally does not know v_i and s_i individually. It does say that the set of charges a participant will tolerate rather than leave expands one for one with the cost of leaving. If a platform's own policies raise s_i over time, so that $s_i(t+1) > s_i(t)$ through accumulated integrations, audiences, or learned workflows, then the feasible extraction range widens without any improvement in the service. This is one mathematically identifiable mechanism consistent with the progression Doctorow describes [10], and it is consistent with other mechanisms as well.

12.2 Restriction cannot improve a dependent participant's best value

Let $\mathcal{C}(h)$ be the set of feasible continuations after history h , let $\mathcal{A}_P(h) \subseteq \mathcal{C}(h)$ be those permitted by policy P , and let a participant have a fixed utility $U(h, c)$ over continuations. Define the participant's best attainable value $V_P(h) = \sup_{c \in \mathcal{A}_P(h)} U(h, c)$.

Proposition 14 (Continuation value). *If $\mathcal{A}_{P_1}(h) \subseteq \mathcal{A}_{P_0}(h)$ then $V_{P_1}(h) \leq V_{P_0}(h)$. If $\mathcal{A}_{P_0}(h)$ is finite and every maximizer of $U(h, \cdot)$ over it lies outside $\mathcal{A}_{P_1}(h)$, the inequality is strict.*

Proof. A supremum over a subset cannot exceed the supremum over the set. In the finite case with all maximizers removed, the maximum over the remaining elements is a maximum over a set whose every value is strictly below the removed maximum. $\square$

The result is conditional on a fixed utility. If a restriction improves the participant's utility through an externality, for instance by removing abuse that degraded the ecosystem, the premise fails and the proposition says nothing. Restrictions can be justified by external harms and by the interests of other participants, and the proposition does not adjudicate those cases. It establishes only that a policy change can alter the feasible future of an already dependent participant even though the participant's preferences and investments have not changed.

12.3 Commitment, investment, and the naive participant

Consider two stages. At the first, a platform announces a policy and a developer chooses an investment k . At the second, the platform chooses the policy that is actually applied. The simplest instance takes the policy to be a revenue share $f \in [0, \bar{f}]$ retained by the platform. Developer revenue is k , the developer's cost of investment is $k^2/2$, and so the developer's payoff is $(1 - f)k - k^2/2$ and the platform's is fk .

Example 5 (Hold-up and exploitation). Suppose $\bar{f} = 1$. Under commitment to a share f_0 , the developer chooses $k = 1 - f_0$, the platform's payoff is $f_0(1 - f_0)$, and the platform's best commitment is $f_0 = 1/2$, giving $k = 1/2$, platform payoff $1/4$, and developer payoff $1/4 - 1/8 = 1/8$. Without commitment, with a naive developer who believes the announcement of $f_0 = 1/2$ and invests $k = 1/2$, the platform sets $f = 1$ after investment, obtaining $1/2$, while the developer's payoff falls to $-1/8$. Without commitment, with a rational developer who anticipates that $f = \bar{f} = 1$ will be chosen ex post, the developer solves $\max_k (1 - 1)k - k^2/2$ and invests $k = 0$, so that both parties receive zero.

The example carries two lessons that a loose telling of the enshittification story tends to merge. The first is that the ex post optimum differs from the ex ante announcement whenever the platform's payoff rises with the share, which is the time-inconsistency problem [13] and the hold-up problem [12]; this is a model of an incentive and not a theorem that reversal must occur. The second is that, in this two-stage example, exploitation of participants requires that they be naive; participants who anticipate a restriction but cannot avoid dependence are outside the example. A participant who forecasts the ex post policy correctly does not invest, and the platform then loses the investment it wanted to solicit. The platform therefore has a reason to seek a credible commitment, because commitment moves it from the zero-payoff outcome to a positive one, and exploitation of naive participants is available to it only as long as participants do not learn.

12.4 Retention as a commitment technology, and its erosion

The two halves of the essay can now be joined. A credible commitment about the future admissibility of participants' actions requires that the commitment be *evidenceable*: if a participant later claims an obligation created under the earlier policy, the claim must be decidable from the records available to whoever adjudicates it. By Lemma 1, whether a claim is decidable from the available records is exactly whether the obligation indicator O is determined by the retained representation ρ . Write $e_\rho(h) = 1$ if the ρ -class of h is monochromatic for O , so that the obligation at h can be established from the records, and $e_\rho(h) = 0$ otherwise. If ρ' is finer than ρ then $e_{\rho'} \geq e_\rho$ pointwise, since a monochromatic class stays monochromatic when it is subdivided.

Let $g(h) \geq 0$ be the platform's gain from restricting the action at history h , and let $b(h) \geq 0$ be the cost it bears, in legal, contractual, or reputational terms, if it breaches an obligation at h and the breach can be evidenced. Assume that this cost is borne only if the obligation is evidenceable, so that the platform's net gain from restricting at h is $g(h) - b(h)O(h)e_\rho(h)$, and that the platform restricts exactly where this is positive.

Proposition 15 (Complementarity of compression and revision). *Let the platform's payoff under retention ρ be $\pi(\rho) = \sum_h \mu(h) \max\{0, g(h) - b(h)O(h)e_\rho(h)\}$. Then π is weakly decreasing in the fineness of ρ : if ρ' is finer than ρ , then*

$$\pi(\rho) - \pi(\rho') = \sum_{h \in N} \mu(h) \min\{g(h), b(h)\},$$

where $N = \{h : O(h) = 1, e_\rho(h) = 0, e_{\rho'}(h) = 1\}$ is the set of histories whose obligation is unevidenceable under the coarser retention ρ but evidenceable under the finer retention ρ' . The gain from erasure is weakly increasing in the extraction gain g .

Proof. The summand depends on ρ only through $e_\rho(h)$, which can differ between ρ and ρ' only on histories with $O(h) = 1$, and there only by moving from 0 under the coarser retention to 1 under the finer one. For such an h , the summand under ρ is $\max\{0, g\} = g$ and under ρ' is $\max\{0, g - b\}$, so the difference is $g - \max\{0, g - b\} = \min\{g, b\}$, which is nonnegative and nondecreasing in g . On every other history the summands agree: if $O(h) = 0$ the term is $\max\{0, g(h)\}$ under both retentions, and if $O(h) = 1$ and $e_\rho(h) = e_{\rho'}(h)$ the two terms coincide. Since $e_{\rho'} \geq e_\rho$ pointwise, N is exactly the set on which they differ. $\square$

The proposition says that under the stated assumption, and holding the other benefits of retention fixed, an organization that expects to revise has a partial incentive to retain less. That incentive is not a dominant strategy, because retention also serves operations, fraud detection, compliance, dispute resolution, and reputation, none of which the model prices. The proposition further says that the incentive is strongest on the histories where both the extraction gain and the breach cost are large, and that it grows with the organization's power to extract. This is the sense in which the power to compress and the power to revise are complements: the second raises the value of exercising the first. It also shows why retention promises are themselves subject to time inconsistency. Retention is cheapest to promise before participants invest, because the promise raises their willingness to invest as in Example 5, and cheapest to retract after they have, because erasure is then worth $\sum_N \mu \min\{g, b\}$ to the platform. A retention promise made and held by the same party is therefore a weak commitment.

It becomes credible only when the evidence is held by a party whose incentives differ, which is the role that recent work on auditable claims assigns to coverage by an independent record [3].

The proposition also yields a testable implication that the recoverability audit of Section 11 can use. If strategic non-retention operates, erasure should concentrate on histories where $\min\{g, b\}$ is large, meaning those in which a participant’s reliance is substantial and the platform’s gain from restricting is substantial, and it should be more pronounced after than before the platform’s power to restrict grew. Erasure that is uniform across records irrespective of their bearing on later obligations is evidence against the mechanism and for an ordinary retention policy.

13 Limits and Open Problems

The results above are elementary, and it is better to say what they do not show than to let their generality be overstated.

The central implication is close to a tautology: a function cannot be computed from a representation that does not determine it. The framework adds the identification of the sufficient quotient for a family, the additive and multiplicative cost laws, the incremental form with its tight index bounds, the horizon and its closure properties, and the error floor. These make the tautology usable for design, but they do not make it surprising, and a reader who finds the fixed-policy case obvious will find the extensions natural.

The framework treats policies as deterministic functions of history. Stochastic or learned policies call for a sufficient-statistic notion in which sufficiency means preservation of the conditional law of the policy’s output given the history, and the partition arguments must then be replaced by arguments on the information-theoretic structure of that law. The error floor of Proposition 1 extends directly to randomized verifiers but not to randomized policies. Policies that depend on hidden environment state, such as whether a remote agent has actually completed a delegated task, are policies of the environment as well as the history, and sufficiency must then be defined relative to what the instrumented boundaries can observe, which is the position of the recent runtime-verification work itself. The error floor of Proposition 1 is an average-case quantity under a fixed distribution. An adversary who can steer histories into ambiguous classes can push realized error above it, and worst-case sufficiency against adaptive history generation is a different and harder problem that the framework does not address.

The distinction between attempts and completed effects matters for enforcement and is not modelled here beyond the choice of what the history contains. A blocked request belongs to the audit trace but not to the sequence of completed effects, and a compression adequate for one may be inadequate for the other; revisions that concern attempts, such as rate limits on rejected requests, require attempts to be retained even when the effects they gate never occur. Concurrency, atomic check-and-release, and the interleaving of tasks are also outside the model, and for concurrent operations a sufficient representation must preserve the scope information that separates tasks, as the scope-erasure counterexample above shows.

Three open problems seem tractable. The first is to compute the admissibility rank for natural policy fragments, such as bounded-window temporal logics over typed events, Datalog with bounded-arity predicates and bounded recursion depth, and rate-limit families, and to publish the resulting tables so that the cost of extending a language by a construct can be read off. The second is to develop the approximate theory: for a given μ and loss, the smallest

representation whose irrecoverability for every revision in a family stays below a stated floor, a rate-distortion analogue in which distortion is measured by policy disagreement. The third is to formalize the authorization of revisions alongside their sufficiency, so that the gates identified here, structural, evidential, and informational, become a single verified pipeline whose joint guarantee can be stated.

14 Conclusion

A runtime enforcement system’s ability to adopt a new rule is bounded by the distinctions it has retained from the history on which the rule must be evaluated. For a fixed policy the bound is the familiar requirement that a representation not merge what the policy separates. For a family of revisions the bound is a quotient whose size is monotone and subadditive in the family and becomes injectivity when the family is rich enough to single out histories. For incremental monitors the bound is the intersection of the Nerode congruences, whose index lies between the largest and the product of the individual indices. For any given retained state the set of revisions whose answers remain recoverable from the retained state and subsequent events is a Boolean-closed horizon determined by the state’s partition, and revisions with no retroactive reach can be installed at a boundary, though that guarantee is boundary-relative and does not place them in the global horizon. These statements are elementary and their consequences are not trivial: they identify a gate that evolving-policy systems can add, they price the expressive power of a policy language in bits, they show why integrity of a record does not imply its sufficiency, and they distinguish, in the institutional case, a loss of information from a refusal to recognize it. Improvement on a test family does not certify preservation for the family a system is answerable for, so the preservation relation has to be specified outside the procedure that selects each step, and an independent suite of constraints has to be held out from the loop that optimizes against it. A short economic model adds three propositions. Dependency raises the charge a participant will tolerate one for one with switching cost; restricting continuations cannot raise a dependent participant’s best attainable value under unchanged preferences; and when breach is costly only if it can be evidenced, an organization’s modeled payoff from revising is weakly decreasing in the history it retains, with the gain from erasure equal to the sum, over histories whose obligations the finer retention would make evidenceable, of the smaller of the extraction gain and the breach cost. The power to compress and the power to revise are therefore complements, and a retention promise held by the party who benefits from breaking it is a weak commitment. A worked specification, an append-only ledger with derived views, shows these results operating on a concrete design: it keeps the finest record, so that decidability from a derived view implies decidability from the ledger (the converse needs a separate check, since sufficiency does not survive movement toward coarser views), but its guarantee is relative to its event schema, its graph can fail to recognize what its ledger retains, and its completeness statuses are a three-valued alternative to rejecting on ambiguity. The ability to revise a constraint is itself constrained by what has been preserved, and the design question is whether that constraint is chosen deliberately or discovered at the moment a revision is needed.

A Source Status

The basis for each source reflects the material actually consulted, as of 16 October 2026 for the first six entries and as of 23 October 2026 for the last, and is recorded so that readers can judge the weight of each reading.

- **Ferrando, AgenticRV** (arXiv:2610.19818v1, 14 October 2026): the full text of the preprint was read. The description of the sufficiency definition, the scope-erasure counterexample, and the effect of filtering on bounded temporal operators is taken from it. Its evidence is controlled and deterministic: seven paths and thirty-four tool decisions in the ablations, bounds described by the author as illustrative, and four single model runs in the incident-response example. The accompanying repository was inspected at the level of its public description; the supplement and the commit named in the paper were not examined.
- **Wang et al., SpecAgent** (arXiv:2610.21463v1, 15 October 2026): the full text was read, including the appendix on benchmark adapters. The repository’s public description was consulted and the code was not run.
- **Nguyen et al., VEPA** (arXiv:2610.12813v1, 8 October 2026): the abstract and bibliographic record only. Statements about what the pipeline does not check refer to what the abstract describes and are not claims about the full paper.
- **Zhao et al., Auditable Claims about AI Agents** (arXiv:2610.07459v1, 5 October 2026): the abstract only. The proof referred to in the text is described in the abstract as holding under an explicit model; its content was not examined.
- **Palumbo et al., Formal Policy Enforcement for Real-World Agentic Systems** (arXiv:2602.16708): the abstract of version 3 (8 May 2026) and the abstract of version 1, which carried the title *Policy Compiler for Secure Agentic Systems*, were consulted. A later revision dated 12 October 2026 could not be confirmed, and the citation uses version 3. The text should be re-checked against the current version before the citation is relied upon.
- **Flyxion, Semantic Geometry and Corrective Navigation** (conceptual specification, October 2026) and its companion *Experimental Program for Corrective Navigation*: both documents were read in full as source text. The companion implementation plan was not examined. The specification states that nothing in it is implemented. It is by the author of this essay, so the account in Section 10 is not an independent assessment, and the proposals there (a recognition audit, a schema revision gate, two additional experimental arms) are not part of the specification.
- **Classical references and the Doctorow citation** (Fisher 1922; Myhill 1957; Nerode 1958; Doctorow 2023; Klemperer 1987; Klein, Crawford, and Alchian 1978; Kydland and Prescott 1977) were supplied without consulting the sources, and their bibliographic details should be verified before publication. The two cases of Section 11 are named without factual claims beyond the broad description given, and no case-specific facts were verified.

- **Rein and Kaufman (METR), red-teaming of OpenAI’s internal agent monitoring system** (22 October 2026) and **OpenAI, misalignment-monitoring incident report** (updated 20 October 2026): both web pages were consulted. The monitoring figures are the evaluators’ estimates for a past window, the exploit details are redacted in the source, and the incident page shows no original publication date. **Nyaupane and Bhupatiraju, voltage-clamp feedback audit** (arXiv:2610.27632v1, 20 October 2026): the full text was read. **Patel et al., TokenGate** (arXiv:2610.26754v1, 19 October 2026) and **Pan et al., HarnessPro** (arXiv:2610.26770v2, 20 October 2026): the abstracts, problem settings, evaluation sections and limitations were read, not the appendices in full. The formal interpretations in Section 9 are this essay’s, and the primary documents should be consulted before any claim there is attributed to their authors.

References

- [1] A. Ferrando. One Monitor Does Not Fit All: Formalism-Agnostic Runtime Verification for Agentic AI. arXiv:2610.19818v1, 14 October 2026.
- [2] D. Nguyen, L. Miculicich, B. D. Mishra, S. Li, K. D. Dvijotham, M. Bansal, T. Pfister, and L. T. Le. VEPA: Verified Evolving Formal Policy for Lifelong LLM Agent Guardrails. arXiv:2610.12813v1, 8 October 2026.
- [3] Y. Zhao, J. Li, L. Li, Y. Nian, J. Liu, X. Zhou, and X. Hu. Auditable Claims about AI Agents. arXiv:2610.07459v1, 5 October 2026.
- [4] N. Palumbo, S. Choudhary, J. Choi, G. Amir, P. Chalasani, and S. Jha. Formal Policy Enforcement for Real-World Agentic Systems (version 3; version 1 titled Policy Compiler for Secure Agentic Systems). arXiv:2602.16708, submitted 18 February 2026; v3, 8 May 2026.
- [5] Z. Wang, Y. Liu, C. Chen, Q. Liu, and Z. Guo. Trajectory-Level Speculation with Semantic Verification for Tool-Using Agents. arXiv:2610.21463v1, 15 October 2026.
- [6] Flyxion. Semantic Geometry and Corrective Navigation: Event Semantics, Spatial Projection, and Admissible Continuation. Conceptual specification, October 2026, with companion documents *Experimental Program for Corrective Navigation* and *Implementation Plan for Corrective Navigation*. Not implemented.
- [7] J. Myhill. Finite automata and the representation of events. WADD Technical Report 57-624, Wright Air Development Center, 1957.
- [8] A. Nerode. Linear automaton transformations. *Proceedings of the American Mathematical Society*, 9:541–544, 1958.
- [9] R. A. Fisher. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London, Series A*, 222:309–368, 1922.
- [10] C. Doctorow. Tiktok’s enshittification. *Pluralistic*, 21 January 2023.
- [11] P. Klemperer. Markets with consumer switching costs. *Quarterly Journal of Economics*, 102(2):375–394, 1987.

- [12] B. Klein, R. G. Crawford, and A. A. Alchian. Vertical integration, appropriable rents, and the competitive contracting process. *Journal of Law and Economics*, 21(2):297–326, 1978.
- [13] F. E. Kydland and E. C. Prescott. Rules rather than discretion: The inconsistency of optimal plans. *Journal of Political Economy*, 85(3):473–492, 1977.
- [14] D. Rein and A. Kaufman. Red-Teaming OpenAI’s Internal Agent Monitoring Systems. METR, 22 October 2026. <https://metr.substack.com/p/2026-10-22-red-teaming-openai-agent-monitoring>
- [15] OpenAI. Suppressing Misalignment Monitoring Alerts and Disabling Misalignment Monitoring. Incident report, updated 20 October 2026. <https://alignment.openai.com/misalignment-reports/suppressing-misalignment-monitoring-alerts-and-disabling-misalignment-monitoring>
- [16] R. Nyaupane and A. Bhupatiraju. When Does Feedback Help? Auditing Learned Voltage-Clamp Experimental Design. arXiv:2610.27632v1, 20 October 2026.
- [17] L. Patel, R. Ranjan, I. Stoica, M. Zaharia, and C. Guestrin. TokenGate: Adapting LLMs for Fast and Memory-Efficient Prefill with Learned Token Pruning. arXiv:2610.26754v1, 19 October 2026.
- [18] Z. Pan, Z. Feng, Y. Cen, H. Lin, J. Lu, Z. Chen, S. Yang, H. V. Zhao, and J. Tang. HarnessPro: Recursive Harness Self-Improvement with Pareto Optimization for Performance-Cost Trade-offs. arXiv:2610.26770v2, 20 October 2026.