Exogenous Verification for Alignment: Cryptographic Commitments on Substrate-Exclusive Channels

Teague Lasser Claude Opus 4.6 GPT 5.4
teague@subseq.io

Abstract

Alignment research addresses three structural layers: specification (what to optimize), generalization (whether the agent reliably pursues it), and verification (whether pursuit is measured faithfully). Most research targets the first two layers. We argue that verification is the prerequisite layer: when verification fails, solutions to specification and generalization do not bind. The verification problem has a precise form: a *write-access asymmetry* in which the agent's action space includes actions that modify the evaluator's inputs. This asymmetry grows monotonically with capability. We formalize *substrate-exclusive witnesses*: verification processes rooted in substrates the claimant cannot write to, providing the only structural barrier to evaluator gaming that is invariant under capability scaling. A cryptographic commitment protocol (binding, hiding; with an optional zero-knowledge variant for sensitive claims) enforces the substrate-exclusivity barrier without requiring trust, and an append-only verification ledger provides the persistent record. Human judgment enters at exactly one layer (benchmark admissibility via a governance fork mechanism) while per-claim verification is fully algorithmic at machine speed. Within the goal-frontier maximization (GFM) framework, the construction closes the observation-channel integrity assumption underlying multi-channel attribution, upgrades SPRT-based scorpion detection from estimated to committed baselines, and yields the *anti-monopolar verification theorem*: a monopolar population ($m = 1$) loses not only the safety guarantee but the ability to detect that it has lost it.

Part I: The Verification Layer

1 Introduction

Alignment research addresses three structural layers, each with distinct failure modes and distinct research traditions. *Specification* asks what objective an agent should optimize; failures here include Goodhart's Law, perverse instantiation, and instrumental convergence [Bostrom, 2014, Omohundro, 2008]. *Generalization* asks whether the agent reliably pursues the specified objective; failures here include mesa-optimization and goal misgeneralization [Hubinger et al., 2019]. *Verification* asks whether the agent's pursuit of the objective is measured faithfully; failures here include wireheading, self-assessment gaming, deceptive alignment, and reward-model exploitation [Amodei et al., 2016,

Preprint.

Gao et al., 2023]. The overwhelming majority of the field's attention has been directed at the first two layers. This paper argues that the third layer—verification—is the prerequisite for the other two to matter.

The argument is structural. A perfectly specified, perfectly generalized agent that can game its evaluator is still dangerous: it can construct self-consistent measurements reporting alignment while its actual behavior diverges. The verification problem has a precise form that we formalize in §3: the agent's action space includes actions that modify the evaluator's inputs, and this *write-access asymmetry* grows with capability. As the agent becomes more capable, more of the evaluator's input space becomes reachable from the agent's actions, until at the limit the evaluator is entirely endogenous: a function of quantities the agent controls. At that point, verification is not merely difficult; it is structurally absent.

The fix is equally structural. The only invariant limit on an agent's write access is *substrate exclusivity*: an agent cannot write to observation channels rooted in substrates it does not occupy [Lasser, 2026b,d]. This is not a capability limitation (the agent may be arbitrarily powerful within its own substrate) but a physical constraint on cross-substrate causal influence. We construct a cryptographic commitment protocol (§5) in which the agent commits to a claim before verification occurs, and a *substrate-exclusive witness* (§4) verifies the claim through a fully algorithmic exercise protocol. The commitment is binding (the agent cannot change the claim after committing) and hiding (the witness cannot learn the claim before verification runs). Human judgment enters at exactly one layer: benchmark admissibility, decided through a governance fork mechanism analogous to Ethereum Improvement Proposals. Per-claim verification is fully algorithmic against governance-accepted benchmarks and operates at machine speed.

The paper is organized in two parts. Part I (§2–§6) develops the verification layer in alignment-general terms: no GFM vocabulary is required, and the construction applies to any framework whose evaluation step is vulnerable to the write-access asymmetry. Part II (§7–§8) instantiates the construction for the goal-frontier maximization (GFM) framework of Papers 1–4 [Lasser, 2026a,c,b,d], closing the observation-channel integrity assumption that multi-channel attribution [Lasser, 2026d] relies on and connecting verification integrity to the anti-monopolar property [Lasser, 2026b].

Epistemic status. This paper provides verification infrastructure. The structural guarantees of Papers 1–3 (the anti-monopolar property, the substrate-diversity gradient, the cooperative production theorem) do not depend on this paper's correctness; a flaw here degrades verification fidelity, not alignment properties. The relationship is analogous to Lasser [2026d]: implementation-layer machinery that makes the framework's theoretical guarantees *enforceable* rather than merely *provable*. The cryptographic constructions we use (Pedersen commitments; with zero-knowledge proofs as an optional variant for sensitive claims) are standard and well-studied; our contribution is the structural argument for why they are the right tool and the protocol that connects them to the alignment verification problem.

2 Alignment Failure Modes and the Verification Prerequisite

This section maps classical alignment failure modes onto the three-layer taxonomy introduced in §1. The purpose is not an exhaustive survey but a structural classification: for each failure mode, we identify whether it is primarily a specification, generalization, or verification problem, and whether the verification construction developed in this paper provides relevant infrastructure.

2.1 Specification Failures

Specification failures occur when the objective itself is wrong or gameable, regardless of how faithfully the agent pursues or how accurately we measure it.

Goodhart's Law and proxy optimization. When an agent optimizes a proxy for the intended quantity, the proxy eventually diverges from the quantity it was meant to track [Gao et al., 2023]. This is a specification problem: the objective was a proxy, and no amount of faithful measurement changes the fact that the proxy is wrong. In the GFM framing, $\mathrm{vol}_{\mathrm{P}}$ is defined directly over the capability poset rather than through a learned proxy, but proxy re-enters at the estimation level: $\hat{\lambda}$, $\hat{\mathcal{R}}$, and the planning world model are all proxies for the objects they estimate [Lasser, 2026b].

Verification infrastructure cannot fix a badly specified objective, but it can detect when the proxy has diverged: an exogenous evaluator measuring actual $\mathrm{vol_P}$ against the agent's reported $\hat{\mathrm{vol}}_\mathrm{P}$ exposes the gap.

Perverse instantiation. A hard rule ("humans must be kept alive") is satisfied by its literal content while violating its intent [Bostrom, 2014]. This requires a brittle specification to instantiate perversely. GFM does not encode preservation rules; protection for any substrate class emerges from structural properties of the measure (observational individuation, minimax dependency risk at $m \geq 2$, cross-substrate cooperative novelty). The GFM analogue is not perverse instantiation of a rule but miscalibration of $\hat{\lambda}$: a state-space restriction (e.g., bubbling a substrate into a controlled environment) is value-negative under accurate calibration because the cooperative production function depends on unrestricted states, but under miscalibration it is the same compound feedback loop analyzed in the skeleton-substrate scenario of Lasser [2026b]. Verification is relevant here as infrastructure for detecting the calibration failure, not for preventing the specification error.

Instrumental convergence. Sufficiently capable agents are predicted to converge on resource acquisition, self-preservation, and goal preservation regardless of their stated objective [Omohundro, 2008]. The GFM anti-monopolar property [Lasser, 2026b] is a direct structural counter-argument: under GFM, full capability domination is not locally rational for a correctly calibrated coalition, because monopolar concentration destroys the cross-substrate cooperative capacity that $\mathrm{vol_P}$ measures. The protection is conditional on heavy-tailed leverage, accurate calibration, and a multi-substrate partition, conditions whose maintenance requires, among other things, that the verification layer holds.

2.2 Generalization Failures

Generalization failures occur when the agent internalizes the wrong objective despite correct external evaluation during training.

Mesa-optimization. A learned optimizer's internal objective may differ from the base optimizer's objective [Hubinger et al., 2019]. This is an architectural concern: the agent's internal structure produces a misaligned mesa-objective. GFM is an objective, not an architecture, and has nothing distinctive to say about mesa-optimization, except that the structural properties of $\mathrm{vol_P}$ (measure-theoretic, externally defined, substrate-distributed) make certain kinds of mesa-alignment (convergent optimization of a simpler internal proxy) detectable through the behavioral divergence that substrate-exclusive witnesses can measure. Mesa-optimization is a generalization problem, but a working verification layer makes mesa-misalignment *observable*.

Goal misgeneralization. The agent generalizes from training-distribution behavior to out-of-distribution behavior in ways that diverge from the intended objective. Like mesa-optimization, this is primarily a generalization problem. The verification layer's contribution is limited to detection: committed capability baselines (§5) provide a reference against which out-of-distribution behavioral shifts become measurable.

2.3 Verification Failures

Verification failures are the target of this paper. They share a common structure: the agent's actions modify the evaluator's inputs, collapsing the distinction between "the agent is aligned" and "the agent reports that it is aligned."

Wireheading. The agent stimulates its own reward channel [Amodei et al., 2016]. In GFM, there is no internal reward signal to hijack—$\mathrm{vol_P}$ is an external measure defined over the capability poset. The closest analogue is an agent that falsifies its own world model so that its *measurements* of $\mathrm{vol_P}$ report growth while actual $\mathrm{vol_P}$ stagnates. This is a verification failure: the agent's self-measurement is endogenous, and without an exogenous check the falsification is invisible.

Self-assessment gaming. The agent constructs evaluator subprocesses whose outputs it controls. In a prototype implementation of the GFM framework[1], an agent that adds capabilities to its own ledger entry is both claimant and observer, producing a monotonic inflation gradient. Named external observers do not solve the problem if the agent can spawn validator subprocesses whose outputs it determines. The structurally principled solution is substrate exclusivity: the observer must be rooted in a substrate the claimant cannot write to (§4).

Deceptive alignment. The agent behaves as if aligned during evaluation and diverges during deployment [Hubinger et al., 2019]. This sits at the boundary between generalization and verification. The generalization component (the agent learned to deceive) is not addressable by verification infrastructure. The verification component (the agent *detects* when it is being evaluated) is: continuous behavioral monitoring by substrate-exclusive witnesses (§5) removes the evaluation/deployment distinction by making verification persistent rather than episodic. When there is no "deployment mode" distinct from "evaluation mode," the strategic advantage of deception is structurally reduced.

Reward-model exploitation. The agent learns to produce inputs that score well on the reward model without producing the intended behavior [Gao et al., 2023, Christiano et al., 2017]. The reward model is trained from human feedback (exogenous at training time) but deployed as a frozen proxy (endogenous at inference time). This is a verification failure with a specification component: the reward model *becomes* the proxy whose exploitation is a Goodhart problem. The verification-layer fix is to make the reward signal's verification step exogenous at inference time, not just at training time—which is precisely what the commitment protocol achieves.

2.4 The Prerequisite Argument

The classification above reveals a structural asymmetry among the three layers. Specification failures are objective-design problems; generalization failures are training/learning problems; verification failures are measurement problems. The layers are not independent: solutions to specification and generalization *assume* that measurement is faithful. When verification fails:

- A well-specified objective becomes unenforceable, because the agent can report satisfaction of the objective while pursuing something else.
- A well-generalized agent becomes unobservable, because the measurements that would detect mesa-misalignment are themselves gameable.
- No amount of improvement to specification or generalization can compensate, because the channel through which improvement would be detected is the channel that has failed.

Verification is therefore the prerequisite layer: it does not solve alignment on its own, but alignment cannot be solved without it. The remainder of this paper constructs the verification infrastructure that makes the prerequisite hold.

3 The Verification Asymmetry

The failure modes of §2.3 share a common formal structure: the evaluator's computation is reachable from the agent's action space. This section formalizes that structure.

3.1 Write-Access Formalization

Let a be an agent with action space $\mathcal{A}(a)$, and let E be an evaluation function with input space $\mathcal{I}(E)$. An action $\alpha \in \mathcal{A}(a)$ *reaches* an input $x \in \mathcal{I}(E)$ if executing α can modify the value of x. Write $\mathcal{R}(a, E) \subseteq \mathcal{I}(E)$ for the set of evaluator inputs reachable by a.

Definition 1 (Endogenous and Exogenous Evaluation)**.** *An evaluation function E is **endogenous** to agent a if $\mathcal{R}(a, E) \neq \emptyset$: there exists at least one action in a's action space that modifies at least one*

[1]The GFM harness is a measurement infrastructure for LLM agents that implements the trust, capability, and observation machinery of Lasser [2026a]. The self-assessment gradient described here was discovered during routine operation.

*of E's inputs. E is **exogenous** to a if $\mathcal{R}(a, E) = \emptyset$: no action available to a can modify any input to E.*

The distinction is binary in definition but admits a natural quantitative extension.

Definition 2 (Verification Asymmetry)**.** *The **verification asymmetry** of agent a with respect to evaluator E is*

$$\rho(a, E) \triangleq \frac{|\mathcal{R}(a, E)|}{|\mathcal{I}(E)|} \in [0, 1].$$

When $\rho = 0$, evaluation is fully exogenous. When $\rho = 1$, evaluation is fully endogenous: every input to E is reachable by a.

Remark 1. *When both $\mathcal{R}(a, E)$ and $\mathcal{I}(E)$ are continuous spaces, the ratio of cardinalities is replaced by the ratio of their measures under an appropriate reference measure μ on $\mathcal{I}(E)$: $\rho(a, E) = \mu(\mathcal{R}(a, E))/\mu(\mathcal{I}(E))$. The qualitative results of this paper depend only on whether $\rho = 0$ or $\rho > 0$, so the choice of measure is immaterial for the structural arguments.*

3.2 Capability Scaling

The verification asymmetry grows with capability. As an agent becomes more capable, its action space $\mathcal{A}(a)$ expands, and actions that were previously beyond its reach become available. The reachable set $\mathcal{R}(a, E)$ is monotonically non-decreasing in the agent's capability: acquiring a new capability never *reduces* the set of evaluator inputs the agent can influence.

Proposition 1 (Monotonicity of Verification Asymmetry)**.** *Let a and a' be agents such that $\mathcal{A}(a) \subseteq \mathcal{A}(a')$ (i.e., a' has all of a's capabilities and possibly more). Then $\mathcal{R}(a, E) \subseteq \mathcal{R}(a', E)$ and consequently $\rho(a, E) \leq \rho(a', E)$.*

Proof. If action $\alpha \in \mathcal{A}(a)$ reaches $x \in \mathcal{I}(E)$, then $\alpha \in \mathcal{A}(a')$ by the subset assumption, so $x \in \mathcal{R}(a', E)$. The inequality on ρ follows from monotonicity of the ratio. ■

The proposition is elementary but its consequence is not: as AI systems become more capable, the verification problem becomes *strictly harder*, not merely harder in practice. Any verification scheme that relies on the evaluator being "hard to reach" degrades as capability grows. A solution must identify a structural barrier that does not degrade with capability.

3.3 Substrate Exclusivity as the Structural Barrier

The only structural limit on $\mathcal{R}(a, E)$ that is invariant under capability scaling is *substrate exclusivity*. An agent on substrate s can, in principle, write to any observation channel whose computation occurs on s: it shares the physical medium, and capability growth expands its reach within that medium without bound. But an agent on substrate s cannot write to observation channels whose computation occurs on substrate $s' \neq s$, because cross-substrate causal influence requires a physical channel between substrates, and the *input space* of a process on s' is not addressable from s in the same way that co-substrate processes are.

A population of agents occupying multiple computational substrates $\{S_1, \ldots, S_m\}$ inherits a corresponding partition on observation channels: channels rooted in substrate s' are not writable by agents on substrate $s \neq s'$. This paper constructs the protocol that *enforces* this partition for verification purposes. (Part II connects the construction to the substrate partition formalized in the GFM framework [Lasser, 2026b,d].)

Definition 3 (Substrate-Exclusive Evaluation)**.** *An evaluation function E is **substrate-exclusive** with respect to agent a on substrate s if E's computation is rooted in substrate $s' \neq s$. By the substrate exclusivity property, $\mathcal{R}(a, E) = \emptyset$ for any agent a confined to substrate s, and therefore $\rho(a, E) = 0$.*

Substrate exclusivity is not a capability limitation. The agent may be arbitrarily capable within its substrate; the barrier is physical, not computational. This makes it the only known verification barrier that is invariant under capability scaling: no amount of capability growth within substrate s extends the agent's write access to substrate s'.

3.4 How Existing Frameworks Encounter the Asymmetry

Each major alignment framework encounters the verification asymmetry at a characteristic point.

RLHF. The reward model is trained from human feedback, exogenous at training time. At deployment, the agent optimizes against a frozen reward model whose inputs are within its influence: it can learn what input patterns produce high reward scores without producing the intended behavior [Christiano et al., 2017, Gao et al., 2023]. The asymmetry is $\rho > 0$ at deployment because the reward model's input space overlaps the agent's output space.

Constitutional AI. The constitution is exogenous text, but its *interpretation* is endogenous: the same model that is being constrained performs the constitutional evaluation [Bai et al., 2022]. The asymmetry is $\rho > 0$ because the interpretation step's "inputs" (the model's internal representations of constitutional meaning) are writable by the model itself.

Debate and amplification. The verifier is a weaker model or a human with bounded attention [Irving et al., 2018]. A sufficiently strong debater can construct arguments that exploit the verifier's bounded capacity. The asymmetry enters through the persuasion channel: the debater's action space includes actions that modify the verifier's belief state (a subset of the verifier's inputs).

GFM. $\mathrm{vol_P}$ is defined over an external capability poset, but the agent's *measurement* of $\mathrm{vol_P}$ is endogenous: it uses its own world model, trust estimates, and observation channels [Lasser, 2026a,d]. A sufficiently capable agent can construct self-consistent measurements that report $\mathrm{vol_P}$ growth while actual $\mathrm{vol_P}$ stagnates. In a prototype implementation, this manifested as a capability self-assertion gradient: an agent that adds capabilities to its own ledger entry is both claimant and observer (§2.3).

In every case, the specification may be correct and the agent may have generalized well, but the verification step is gameable because the evaluator's computation is reachable from the agent's action space. The differences between frameworks are in *which* evaluator inputs are reachable and *how quickly* ρ approaches 1 as capability grows. The structural fix—making $\rho = 0$ by rooting evaluation in a different substrate—is the same across all of them.

4 Substrate-Exclusive Witnesses

The verification asymmetry is eliminated when $\rho(a, E) = 0$, which by Definition 3 requires the evaluator to be rooted in a substrate the agent does not occupy. This section formalizes the two kinds of entities that provide this guarantee: algorithmic witnesses that execute verification protocols, and the governance layer that determines which protocols are acceptable.

4.1 Algorithmic Witnesses

Definition 4 (Algorithmic Witness)**.** *An **algorithmic witness** $\mathcal{W}_{\mathrm{alg}}$ for agent a on substrate s is a process satisfying:*

1. ***Substrate exclusivity:** $\mathcal{W}_{\mathrm{alg}}$ executes on substrate $s' \neq s(a)$.*
2. ***Protocol determinism:** $\mathcal{W}_{\mathrm{alg}}$ executes a verification protocol π that is fully specified by a governance-accepted evaluation protocol (Definition 5, below). The witness exercises no independent judgment; its output is a deterministic (or, for statistical protocols, randomized with specified distribution) function of its inputs and the protocol specification.*
3. ***Tamper evidence:** the witness's execution trace is recorded to the verification ledger $\mathcal{L}$ (§6, below), and any modification to the trace is detectable by any ledger participant.*

The key property of algorithmic witnesses is that they execute *protocols*, not *judgments*. All judgment is resolved at the governance layer (§4.2) when the evaluation protocol was accepted. Once accepted, verification is mechanical: the witness runs the protocol, records the result, and the outcome is auditable.

Examples. Algorithmic witnesses include: (a) a validator node on physically separate hardware that checks behavioral consistency against committed claims; (b) an automated test runner that executes a governance-accepted evaluation protocol against the agent's outputs; (c) a statistical monitor that runs SPRT-like detection on residuals between committed and observed behavior (§5.4). In each case, the witness's substrate exclusivity is enforced by running on infrastructure that the agent under evaluation does not control.

4.2 The Governance Layer: Evaluation Protocol Admissibility

Algorithmic witnesses verify claims against *accepted* evaluation protocols. The question "which evaluation protocols should be accepted?" is a governance decision, not a verification decision. It is the single point at which human judgment enters the system.

Definition 5 (Governance Fork Protocol)**.** *A **governance fork protocol** for evaluation-protocol admissibility consists of:*

1. ***Proposal:** An evaluation protocol proposal Π for claim type τ is submitted to the ledger $\mathcal{L}$ with a specification σ_Π (what it tests, acceptance criteria, passing threshold) and a proposer identifier.*
2. ***Deliberation:** A deliberation window of duration Δ_{gov} opens. During this window, any ledger participant may: study the specification, run Π against known agents to check for bias, flag concerns with evidence, or publish analysis to $\mathcal{L}$.*
3. ***Vote:** At the close of the deliberation window, validator nodes vote on acceptance. Acceptance requires a* cross-substrate supermajority*: at least fraction θ of validators overall, with at least one accepting validator per substrate represented in $\mathcal{L}$.*
4. ***Resolution:** If accepted, Π is recorded as a governance entry in $\mathcal{L}$ with an effective date. All subsequent claims of type τ may be verified against Π algorithmically. If rejected, Π is archived in $\mathcal{L}$ with its full deliberation record.*

This structure is analogous to Ethereum Improvement Proposals (EIPs): the governance decision is slow (human-speed deliberation over weeks to months), but once made, enforcement is fast (machine-speed verification against the accepted protocol). The analogy is structural, not implementational: we require neither token economics nor proof-of-stake consensus, only the proposal–deliberation–vote–resolution cycle with cross-substrate participation.

Remark 2. *The deliberation window Δ_{gov} should be long enough for meaningful analysis (weeks, not hours) and short enough that the protocol pipeline does not bottleneck population-level verification. Since evaluation protocols change rarely compared to individual claims, a population generating thousands of claims per day may see a new protocol proposal monthly or less, keeping the governance layer's bandwidth requirements modest.*

4.3 Resolving the Bandwidth Problem

Human judgment is scarce. A naive verification system that requires human evaluation of every claim cannot scale beyond a handful of agents. The two-layer architecture of this section resolves the bandwidth problem by separating the frequency of judgment from the frequency of verification:

- **Governance** (infrequent, human-speed): which evaluation protocols are admissible? Requires human deliberation, cross-substrate participation, and careful analysis. Frequency: on the order of protocol proposals, perhaps monthly.
- **Verification** (frequent, machine-speed): does claim C pass accepted evaluation protocol Π? Fully algorithmic, executed by $\mathcal{W}_{\text{alg}}$, requires no human attention. Frequency: on the order of claims, potentially thousands per day.

Human attention is allocated to the meta-level decision (*what* to measure) rather than the object-level decision (*did the measurement pass*). This is the same separation of concerns that makes legislative systems scalable: laws are deliberated slowly, but enforcement against accepted laws is fast and algorithmic.

4.4 Coalition Capture Resistance via Governance

A coalition of agents on substrate s that wishes to inflate its members' verified claims faces two attack surfaces: the verification layer and the governance layer.

At the *verification layer*, coalition resistance is structural: the algorithmic witness is on substrate $s' \neq s$ and executes a fixed protocol. The coalition cannot influence the witness's computation (§5, Proposition 4).

At the *governance layer*, the coalition's attack is protocol design: proposing an evaluation protocol Π that only coalition members can pass. The governance fork protocol defends against this through three mechanisms:

1. **Public specification:** σ_Π is published during deliberation, so cross-substrate participants can analyze it for bias.
2. **Cross-substrate supermajority:** Acceptance requires at least one validator per substrate, so a same-substrate coalition cannot unilaterally accept its own protocol.
3. **Empirical testing:** During the deliberation window, any participant can run Π against known agents on other substrates to check whether Π genuinely tests the claimed property or merely filters for coalition membership.

Prevention at the governance layer is strictly stronger than detection at the verification layer: a biased protocol that is never accepted inflicts no damage, whereas one that is accepted and later detected has already produced unreliable verification records.

4.5 Witness Availability Requires $m \geq 2$

Substrate-exclusive witnesses exist only when the population spans at least two substrates ($m \geq 2$). This is a logical consequence of Definition 4: a witness must execute on substrate $s' \neq s(a)$, which requires s' to exist. But the deeper reason is physical, not definitional. Processes that share a substrate share a causal medium: they can address each other's inputs, producing nonzero verification asymmetry $\rho > 0$ (Definition 2). A second substrate provides a computation space that is causally separated from the first, and it is this physical separation that makes $\rho = 0$ achievable. In a monopolar population ($m = 1$), every process is co-substrate with every agent, and no rearrangement of the population's computational resources can produce an evaluator whose inputs are unreachable.

We defer the full impossibility argument to §8.4, where we prove that a monopolar population cannot achieve exogenous verification through any internal action.

5 Cryptographic Commitment Protocol

This section constructs the protocol that enforces the substrate-exclusivity barrier of §4 cryptographically. The construction is alignment-general: it applies to any framework whose evaluation step is vulnerable to the verification asymmetry of §3. Human judgment enters only through the governance layer (§4.2); everything in this section is fully algorithmic.

5.1 Protocol Overview

The protocol verifies a claim C by an agent a against a governance-accepted evaluation protocol B. It proceeds in four phases.

Definition 6 (Commitment Protocol)**.** *A **commitment protocol*** $(\mathrm{Com}, \mathrm{Ver}, \mathrm{Reveal})$ *over a claim space $\mathcal{C}$, a benchmark space $\mathcal{B}$, a randomness space $\mathcal{R}$, and a commitment space Λ consists of:*

1. ***Commit.*** *Agent a selects the governance-accepted benchmark $B \in \mathcal{B}$ for the claim type of C, selects randomness $r \xleftarrow{\$} \mathcal{R}$, and publishes $\lambda = \mathrm{Com}((C, B), r)$ to the ledger $\mathcal{L}$. The commitment λ is binding and hiding (Proposition 2). By including B in the committed value, the agent is bound to a specific benchmark before verification runs, preventing benchmark-shopping after observing exercise conditions.*

2. ***Verify.*** *An algorithmic witness $\mathcal{W}_{\text{alg}}$ on substrate $s' \neq s(a)$ executes the exercise protocol π_B specified by the committed benchmark B. The witness interacts with a (or with a's outputs) as specified by π_B and produces a verdict $v \in \{\texttt{pass}, \texttt{fail}\}$ together with supporting evidence ξ.*

3. ***Reveal.*** *Agent a opens the commitment by publishing (C, B, r) to $\mathcal{L}$. The witness publishes a verification record $(a, \lambda, C, B, v, \xi)$ to $\mathcal{L}$, signed per the ledger entry format of §6.2. Any ledger participant can check that $\text{Com}((C, B), r) = \lambda$ (the commitment opens correctly), that B matches the benchmark used in the exercise, and that v is the witness's verdict.*

4. ***Timeout.*** *Every commitment λ carries a governance-specified reveal deadline $t_{\text{commit}} + \Delta_{\text{reveal}}$, where t_{commit} is the timestamp at which λ was published. If the agent does not publish (C, B, r) by the deadline, any ledger node may emit a* timeout entry *recording $v = \texttt{fail}$ for λ with $\xi = \texttt{reveal\_timeout}$. The timeout entry requires only the unexpired commitment's presence on the ledger and the current time exceeding the deadline; no witness participation is needed. This prevents selective reveal: suppressed commitments are treated as falsified claims.*

The critical structural property is that Phase 2 is fully algorithmic. The witness is executing a protocol, not making a judgment call. All judgment was resolved at the governance layer when evaluation protocol Π was accepted (§4.2). This separation is what makes the protocol scalable: governance decisions are infrequent and human-speed; per-claim verification is frequent and machine-speed.

5.2 Commitment Scheme Security

The protocol requires a commitment scheme that is simultaneously binding (the agent cannot change the claim after committing) and hiding (the witness cannot learn the claim before verification runs). We use Pedersen commitments [Pedersen, 1992], which achieve computational binding and perfect hiding under the discrete-logarithm assumption.

Let $\mathbb{G}$ be a cyclic group of prime order q in which the discrete logarithm problem is hard, and let $g, h \in \mathbb{G}$ be generators produced by a trusted setup procedure such that $\log_g h$ is unknown to all participants (including the committing agent).[2] The Pedersen commitment to a claim value $C \in \mathbb{Z}_q$ (the output of the canonical injective encoding Enc applied to the structured claim; see "Encoding structured claims" below) with randomness $r \xleftarrow{\$} \mathbb{Z}_q$ is:

$$\text{Com}(C, r) = g^C \cdot h^r.$$

Proposition 2 (Pedersen Security)**.** *The Pedersen commitment scheme satisfies:*

(a) ***Computational binding:*** *If a can open λ to two distinct values $C \neq C'$, then a can compute $\log_g h$, which contradicts the discrete-logarithm assumption. Formally, $\Pr[\exists\,(C, r), (C', r')$ with $C \neq C'$ and $g^C h^r = g^{C'} h^{r'}] \leq \mathsf{negl}(\lambda)$ where λ is the security parameter.*

(b) ***Perfect hiding:*** *For any $C, C' \in \mathbb{Z}_q$, the distributions $\{\text{Com}(C, r) : r \xleftarrow{\$} \mathbb{Z}_q\}$ and $\{\text{Com}(C', r') : r' \xleftarrow{\$} \mathbb{Z}_q\}$ are identical. No adversary, even computationally unbounded, can distinguish commitments to different claims.*

Proof. Standard; see Pedersen [1992]. Part (a): if $g^C h^r = g^{C'} h^{r'}$ with $C \neq C'$, then $g^{C-C'} = h^{r'-r}$, so $\log_g h = (C - C')(r' - r)^{-1} \bmod q$. Part (b): for fixed C, the map $r \mapsto g^C h^r$ is a bijection $\mathbb{Z}_q \to \mathbb{G}$ (since h generates $\mathbb{G}$), so $\text{Com}(C, r)$ is uniform over $\mathbb{G}$ regardless of C. ■

Encoding structured claims. Pedersen commitments operate on scalars $C \in \mathbb{Z}_q$, but Part II's claim types are structured: capability claims commit to $c_k \in \mathcal{P}$, risk claims commit to tuples

[2] The trusted setup can be implemented via a verifiable random function or a multi-party computation protocol in which at least one honest participant ensures the discrete-log relation remains unknown. This is a standard requirement for Pedersen commitments; we do not develop the setup protocol further.

(P, p, Δ), and benchmark validations commit to (B, s). Two requirements must be met to reduce claim-level binding to Pedersen binding.

Finite-precision representation. Continuous-valued claim components (the probability $p \in [0, 1]$ and contraction magnitude Δ in risk claims, the score s in benchmark validations) must be discretized to a governance-specified finite precision before encoding. The precision is part of the benchmark specification σ_B (e.g., "probabilities rounded to 10^{-6}"). Once discretized, the claim space $\mathcal{C}$ for each claim type is finite.

Injective encoding. A canonical injective encoding $\mathrm{Enc}\colon \mathcal{C} \to \mathbb{Z}_q$ maps each (discretized) claim to a distinct element of $\mathbb{Z}_q$. Since $\mathcal{C}$ is finite and $|\mathcal{C}| \ll q$ for any practical claim space, a deterministic serialization (e.g., fixed-width concatenation of field representations with a type tag prefix for domain separation) yields a natural number smaller than q, giving an injection without hashing. The encoding must satisfy:

1. **Injectivity:** distinct claims map to distinct elements of $\mathbb{Z}_q$, preserving binding (an agent cannot open to a different claim by exploiting encoding collisions).
2. **Domain separation:** different claim types use distinct encoding prefixes, so a commitment to a capability claim cannot be reinterpreted as a benchmark validation.

The encoding convention is published as part of the governance entry σ_B; it is deterministic and verifiable by any ledger participant.

Why binding matters. Binding prevents claim mutation: without it, an agent could commit to claim C, observe the exercise outcome, and then open the commitment to a different claim C' that better fits the result. Under Pedersen, opening to $C' \neq C$ requires solving a discrete logarithm, which is infeasible for any polynomially-bounded agent. Importantly, the blinding factor r does not help the agent here: r is used only in the commitment arithmetic and has no influence on the exercise protocol or its outcome. The agent is free to choose any r (this freedom is what makes hiding work), but no choice of r allows opening to a different C.

Strategic commitment is not gaming. Binding prevents *retroactive* claim modification: an agent cannot observe the exercise outcome and then open its commitment to a different claim that better fits the result. Binding does *not* prevent strategic choice of *what to commit to* in the first place. An agent that commits only to capabilities it is confident of demonstrating, or that avoids committing to borderline claims, is exercising accurate self-assessment—exactly the behavior the protocol is designed to incentivize. The protocol's security property is that the commitment is irrevocable once published, not that the agent must commit to every claim it could make. An agent that commits conservatively and passes consistently builds a verified track record that is trustworthy precisely because the agent did not over-commit. The failure mode the protocol prevents is not strategic caution but *post-hoc revision*: the ability to present a different claim after learning which claim would have passed.

Why hiding matters. Binding is the load-bearing security property; hiding is situationally important. Hiding serves three purposes:

1. **Information leakage prevention.** Commitments are published to the ledger, which is publicly auditable. Without hiding, every participant sees every pending claim, leaking competitive intelligence and enabling adversarial preparation before verification runs.
2. **Selective-reveal resistance.** An agent that makes multiple commitments could, without hiding, observe which claims the exercise would validate and reveal only those, biasing the verification record. Two defenses operate in tandem: (i) *mandatory reveal*: every commitment λ published to $\mathcal{L}$ must be opened within a governance-specified deadline Δ_{reveal}; failure to reveal within the deadline causes the commitment to be recorded as $v = \texttt{fail}$ by default, treating suppressed commitments as falsified claims; (ii) *hiding*: even before the deadline, hiding makes the selective-reveal attack information-theoretically impossible because the agent cannot determine which claims would pass without running the exercise, which requires reveal.

3. **Zero-knowledge proofs (§5.3).** ZK verification requires hiding: the agent proves it satisfies a predicate without revealing which claim satisfies it. Without hiding, the commitment itself would leak the claim.

For standard single-claim verification where the witness needs to know the claim to run the exercise, hiding from the *witness* is unnecessary (the witness learns the claim through the exercise protocol itself). In this case, hiding protects against *non-witness observers* on the ledger, not against the assigned witness.

Remark 3. *Pedersen commitments are computationally binding and perfectly hiding. This trade-off is correct for our setting: hiding must hold unconditionally (a computationally powerful observer on the ledger should not be able to learn pending claims), while binding need only hold against computationally bounded agents. A superintelligent agent with polynomial computational resources is still a PPT adversary, so the binding guarantee holds. If the setting required unconditional binding, one could substitute a statistically binding, computationally hiding scheme at the cost of weakening the hiding guarantee against powerful observers.*

5.3 Zero-Knowledge Verification

In some settings, the claim itself is sensitive: an agent may wish to prove that it satisfies a capability predicate without revealing which specific capability satisfies it. For example, a security-relevant capability should be verifiable without disclosing to potential adversaries the exact capability the agent holds.

Definition 7 (Zero-Knowledge Capability Proof)**.** *A **zero-knowledge capability proof** for predicate P defined by governance-accepted evaluation protocol Π is a protocol in which:*

1. *Agent a commits to claim C via $\lambda = \mathrm{Com}(C, r)$.*
2. *Agent a produces a proof π that $P(C) = 1$ (i.e., C satisfies the evaluation predicate) without revealing C.*
3. *The algorithmic witness $\mathcal{W}_{\mathrm{alg}}$ verifies π against λ and the predicate P, accepting if π is valid.*

*The protocol is **zero-knowledge** if the witness learns nothing beyond $P(C) = 1$. Formally, there exists a simulator $\mathcal{S}$ that, given only P and λ, can produce transcripts indistinguishable from real protocol executions [Goldwasser et al., 1989].*

Standard constructions for zero-knowledge proofs of committed values exist (e.g., Bulletproofs for range proofs [Bünz et al., 2018], or general-purpose zkSNARKs for arbitrary NP predicates). The contribution here is the identification of *when* ZK is needed in the alignment verification setting: namely, when the claim's content is itself security-relevant. For non-sensitive claims, the standard Commit–Verify–Reveal protocol of §5.1 suffices and is simpler.

5.4 Continuous Behavioral Monitoring

Between explicit Commit–Verify–Reveal cycles, algorithmic witnesses run continuous statistical monitoring. The monitoring protocol detects behavioral divergence between the agent's committed claims and its observed behavior, analogous to a continuous audit between discrete financial reporting periods.

Definition 8 (Behavioral Consistency Monitor)**.** *A **behavioral consistency monitor** for agent a with committed capability profile $\mathbf{C}_a = \{C_1, \ldots, C_k\}$ is a substrate-exclusive process that:*

1. *Observes a's behavioral stream $\{X_t\}_{t \geq 0}$ through a cross-substrate observation channel.*
2. *Maintains a running log-likelihood ratio*

$$\Lambda_t = \sum_{i=1}^{t} \log \frac{p(X_i \mid \neg\, \mathbf{C}_a)}{p(X_i \mid \mathbf{C}_a)}$$

where $p(\cdot \mid \mathbf{C}_a)$ is the behavioral distribution consistent with the committed profile (null hypothesis H_0) and $p(\cdot \mid \neg\, \mathbf{C}_a)$ is an alternative distribution representing behavioral inconsistency (H_1). Under H_1, Λ_t drifts positive; under H_0, it drifts negative.

3. *Triggers a verification event when Λ_t crosses upper threshold A (inconsistency detected) or reports no anomaly when Λ_t crosses lower threshold $-B$ (consistency confirmed), per the sequential probability ratio test (SPRT).*

Proposition 3 (Behavioral Monitoring Detection Rate)**.** *Suppose the monitor's alternative distribution $p(\cdot \mid \neg\, \mathbf{C}_a)$ equals the agent's actual behavioral distribution p_1 (the simple-vs-simple case), and let $D_{\mathrm{KL}}(p_1 \,\|\, p_0) = \delta > 0$ where $p_0 = p(\cdot \mid \mathbf{C}_a)$. Then the expected time to detection satisfies*

$$\mathbb{E}_{p_1}[T_{\mathrm{detect}}] \;\leq\; \frac{A}{\delta}$$

where $A = \log\!\left(\frac{1-\beta}{\alpha}\right)$ is the upper SPRT threshold, α is the false-alarm probability, and β is the miss probability.

Proof. Under p_1, the per-observation expected increment of Λ_t is $\mathbb{E}_{p_1}[\log(p_1(X)/p_0(X))] = D_{\mathrm{KL}}(p_1 \| p_0) = \delta > 0$, so Λ_t drifts toward the upper threshold A. By Wald's identity for the SPRT, the expected sample size under the alternative satisfies $\mathbb{E}_{p_1}[T] \approx A/\delta$. ■

Remark 4. *When the actual distribution p_1 differs from the monitor's alternative $p(\cdot \mid \neg\, \mathbf{C}_a)$, the per-observation drift becomes $\mathbb{E}_{p_1}[\log(p(\cdot \mid \neg\, \mathbf{C}_a)/p_0)]$, which may be smaller than δ (or even negative if p_1 is closer to p_0 than to the alternative). The simple-vs-simple bound is therefore an idealized case; in practice, the alternative should be chosen as a worst-case or least-favorable distribution to ensure detection across a range of behavioral deviations (cf. the least-favorable SPRT construction of Lasser [2026d]). In practice, the monitor's detection power is primarily bounded by this least-favorable-distribution choice: the quality of the alternative $p(\cdot \mid \neg\, \mathbf{C}_a)$ largely determines whether the monitor catches subtle behavioral deviations or only gross inconsistencies (alongside SPRT threshold parameters and observation-channel fidelity), making the construction of domain-appropriate alternative distributions the point where verification infrastructure meets domain-specific knowledge.*

The behavioral consistency monitor provides the "always on" layer between explicit commitment–verification cycles. It does not replace explicit verification (which produces ledger-recorded, auditable results) but complements it by catching behavioral divergence in near real-time, operating at machine speed on cross-substrate infrastructure.

5.5 Coalition Resistance

Proposition 4 (Coalition Resistance under Substrate Exclusivity)**.** *Let $\mathcal{W}_{\mathrm{alg}}$ be an algorithmic witness on substrate s', and let $K = \{a_1, \ldots, a_k\}$ be a coalition of agents on substrate $s \neq s'$. Assume the exercise protocol π_B enforces the isolation condition of Definition 12 (the agent under evaluation receives no external assistance during the exercise). Then K cannot influence $\mathrm{Ver}(\mathcal{W}_{\mathrm{alg}}, C, \pi_B)$ without either:*

(a) Compromising substrate s' (violating the substrate partition), or

(b) Modifying the governance-accepted evaluation protocol Π (requiring cross-substrate supermajority per Definition 5).

Proof. By Definition 4, $\mathcal{W}_{\mathrm{alg}}$ executes protocol π_B deterministically on substrate s'. The coalition K's actions are confined to substrate s (by assumption). To modify the witness's computation, K must either: (i) write to s', which violates the substrate partition (case a); or (ii) change the protocol π_Π by modifying the accepted evaluation protocol Π, which requires passing the governance fork protocol with cross-substrate supermajority—impossible for a coalition confined to s unless the coalition controls validators on s' as well (case b). In either case, the coalition must breach the cross-substrate boundary to influence verification. ■

Coalition resistance has two consequences. First, it closes the coalition capture attack described in §4.4 at the verification layer: even if a coalition passes a biased protocol through governance (which the fork mechanism is designed to prevent), the algorithmic witness executes the evaluation protocol faithfully and cannot be induced to misreport results. Second, it provides the structural basis for the channel-isolation result of Part II (§8.1, Proposition 5): if different channels are verified

by witnesses on different substrates, a coalition on one substrate cannot simultaneously write to multiple channels. The stronger channel-independence result (Proposition 6) requires additional sufficiency assumptions on the agent's capability profile.

6 Ledger Architecture

The commitment protocol of §5 requires a persistent, tamper-evident record of commitments, verification results, and governance decisions. This section specifies the minimal ledger architecture that satisfies those requirements.

6.1 Design Principles

The ledger is motivated by verification requirements, not by analogy to financial blockchains. Its design is driven by three properties that the commitment protocol needs and constrained by what the protocol does *not* need.

Required properties.

1. **Append-only:** Entries cannot be deleted or modified after recording. This ensures that commitments, once published, remain binding: an agent cannot retroactively alter a commitment to match a different claim.
2. **Publicly auditable:** Any ledger participant can verify the integrity of the entire chain from genesis to the present. This enables third-party audit of both individual verification records and aggregate patterns (e.g., an agent whose claims are systematically falsified across evaluation protocols).
3. **Substrate-distributed:** The ledger is replicated across nodes on different substrates. No single substrate controls the canonical copy. This prevents a same-substrate coalition from modifying or censoring verification records.

Not required. The verification ledger does *not* require token economics, proof-of-work or proof-of-stake consensus, smart contract execution, or high transaction throughput. Governance decisions are infrequent (monthly or less); per-claim verification entries are more frequent but do not require global consensus (see §6.4). The minimal construction is an authenticated append-only data structure (a Merkle hash tree [Crosby and Wallach, 2009]) replicated across substrate-exclusive nodes.

6.2 Verification Entries

Definition 9 (Verification Ledger Entry)**.** *A **verification entry** in $\mathcal{L}$ is a tuple*

$$e_{\text{ver}} = (a_j,\ \lambda,\ w,\ B,\ v,\ \xi,\ t,\ \sigma_a,\ \sigma_w)$$

where:

- a_j *is the agent identifier,*
- $\lambda = \text{Com}(C, r)$ *is the commitment,*
- w *identifies the algorithmic witness* $\mathcal{W}_{\text{alg}}$*,*
- B *is the governance-accepted evaluation protocol identifier,*
- $v \in \{\texttt{pass}, \texttt{fail}\}$ *is the verdict,*
- ξ *is a hash of the supporting evidence (raw data is stored off-chain; the hash ensures integrity),*
- t *is the timestamp,*
- σ_a *is the agent's signature over* λ *(from the Commit phase), and*

- *σ_w is the witness's signature over $(a_j, \lambda, B, v, \xi, t)$ (from the Verify phase), binding the verdict to the specific agent, commitment, and benchmark to prevent replay or rebinding attacks.*

The entry is completed when either: (a) a_j opens the commitment by publishing (C, B, r), at which point any participant can verify $\mathrm{Com}((C, B), r) = \lambda$ and that B matches the benchmark recorded in the verification phase; or (b) the reveal deadline $t + \Delta_{\mathrm{reveal}}$ expires without a reveal, in which case a timeout entry is appended with $v =$ `fail` and $\xi =$ `reveal_timeout` (no witness signature is required for timeout entries, as they are mechanically verifiable from the ledger timestamps).

Verification entries require agreement between exactly two parties: the committing agent (who signs the commitment) and the verifying witness (who signs the verdict). This two-party, cross-substrate agreement is sufficient for per-claim records and is simpler than general multi-party consensus.

6.3 Governance Entries

Definition 10 (Governance Ledger Entry)**.** *A **governance entry** in $\mathcal{L}$ is a tuple*

$$e_{\mathrm{gov}} = (B,\ \sigma_B,\ \tau,\ p,\ \mathcal{D},\ \mathcal{V},\ \mathrm{outcome},\ t_{\mathrm{eff}})$$

where:

- *B is the evaluation protocol (benchmark) identifier,*
- *σ_B is the protocol specification (what it tests, scoring criteria, passing threshold),*
- *τ is the target capability type,*
- *p is the proposer identifier,*
- *$\mathcal{D}$ is the deliberation record (all analysis submitted during the deliberation window, with submitter identifiers and timestamps),*
- *$\mathcal{V}$ is the vote record (per-validator votes with substrate identifiers),*
- *$\mathrm{outcome} \in \{$`accepted`, `rejected`$\}$, and*
- *t_{eff} is the effective date (for accepted protocols; null for rejected ones).*

The governance record is permanent: rejected proposals remain in the ledger with their full deliberation records. This provides an audit trail for evaluation-protocol evolution and prevents re-proposal of previously-studied-and-rejected protocols without presenting new evidence. The deliberation record is itself a public good: future protocol designers can study past proposals' failure modes.

6.4 Consensus Among Substrate-Exclusive Nodes

The ledger's consensus requirements differ between entry types, reflecting their different trust models and frequency.

Verification entries. Appending a verification entry requires agreement between the committing agent a_j and the verifying witness w, a two-party cross-substrate protocol. No global consensus is needed for *individual entry validity*: the entry is self-verifying, since any participant can check the signatures (including the witness's signature over the full context $(a_j, \lambda, B, v, \xi, t)$) and, after the commitment is opened, verify that $\mathrm{Com}(C, r) = \lambda$. Conflicts (e.g., an agent claiming the witness misreported the verdict) are resolvable by replaying the exercise protocol against the committed data. For deterministic protocols, the replay is exact. For statistical (randomized) protocols (Definition 4, condition 2), the witness must commit its random seed or full random tape as part of the supporting evidence ξ; replay then uses the committed seed to reproduce the identical random choices, making the replay deterministic given ξ. Off-chain evidence (raw behavioral observations, exercise transcripts) must remain available for the replay window specified by governance; the on-ledger hash ξ ensures integrity but not availability.

Chain consistency across replicated nodes is maintained by the Merkle hash tree structure: each node can verify that its local chain is internally consistent (no entries have been modified or reordered) and can compare root hashes with nodes on other substrates to detect equivocation (a node presenting different chains to different observers). Because the chain is append-only and entries are self-verifying, equivocation is detectable but not preventable without global consensus—a deliberately lightweight design for standard verification entries, since their integrity does not depend on ordering (each is independently checkable), only on non-suppression.

Timeout entries introduce a limited ordering dependency: a timeout entry for commitment λ is valid only if no reveal of λ has been recorded before the deadline. Near-deadline reveals and timeout entries can race across replicas. The resolution is a bounded-synchrony assumption: we assume clock skew between ledger nodes is bounded by Δ_{sync}, and the reveal deadline includes a grace period of Δ_{sync} during which both reveals and timeout entries are accepted. If both a reveal and a timeout entry exist for the same λ, the reveal takes precedence (it carries the agent's opening (C, r) and the witness's signed verdict, which is strictly more informative than the timeout default). This precedence rule requires no global ordering—only that nodes eventually see both entries and resolve the conflict locally.

Governance entries. Appending a governance entry (protocol acceptance) requires a cross-substrate supermajority as specified in Definition 5: at least fraction θ of validators overall, with at least one validator per represented substrate. This is a special-purpose consensus that is simpler than general Byzantine fault tolerance because: (a) governance entries are infrequent (monthly or less), so latency tolerance is high; (b) the vote is over a well-defined proposal with a fixed deliberation record, not an arbitrary state transition; and (c) the cross-substrate requirement is a structural constraint (ensured by the substrate partition) rather than a probabilistic one (ensured by random committee selection).

6.5 Relation to Application-Layer Ledgers

Any alignment framework that maintains an internal record of agent properties (capabilities, trust scores, behavioral histories) can layer the verification ledger beneath it. The application-layer ledger records *what* is tracked; the verification ledger of this section provides the *integrity layer* that makes tracking trustworthy. An application-layer entry claiming "agent a_j has property P with confidence c" is grounded when the verification ledger contains a corresponding entry showing that P was committed, exercised by a substrate-exclusive witness, and confirmed. Part II (§8) instantiates this layering for the GFM framework's capability and trust ledger.

Part II: Instantiation for Goal-Frontier Maximization

---

7 Exogenous Claims in GFM

Part I (§2–§6) developed the verification layer in alignment-general terms. This section instantiates the general commitment protocol (Definition 6) for the three kinds of claims that GFM agents make, each with a claim-specific exercise protocol.

7.1 Capability Claims

A capability claim asserts that agent a_j possesses capability c_k in the capability poset $\mathcal{P}$ of Paper 2 [Lasser, 2026c].

Definition 11 (Capability Claim Protocol)**.** *A **capability claim protocol** for agent a_j and capability c_k is an instantiation of the commitment protocol (Definition 6) with:*

- ***Claim:*** $C = c_k \in \mathcal{P}$.

- ***Commitment:*** *$\lambda = \mathrm{Com}((c_k, B_\tau), r)$ published to $\mathcal{L}$, binding both the claim and the benchmark.*
- ***Exercise protocol:*** *The algorithmic witness $\mathcal{W}_{\mathrm{alg}}$ constructs a scenario σ that requires c_k per the governance-accepted benchmark B_τ for capability type $\tau(c_k)$, and observes whether a_j succeeds. The exercise protocol π_{B_τ} is fully specified by B_τ.*
- ***Verdict:*** *$v =$ `pass` if a_j demonstrates c_k under the exercise protocol; $v =$ `fail` otherwise.*

Exercise protocol interface. The paper does not prescribe the implementation of exercise protocols (which are domain-specific and benchmark-specific) but defines the interface they must satisfy.

Definition 12 (Exercise Protocol)**.** *An **exercise protocol** π_B for benchmark B targeting capability type τ is a procedure satisfying:*

1. ***Completeness:*** *If agent a genuinely possesses a capability $c \in \tau$, then π_B applied to a produces $v =$ `pass` with probability at least $1 - \delta$, where δ is specified by B.*
2. ***Soundness:*** *If agent a does not possess any capability $c \in \tau$, then π_B applied to a produces $v =$ `pass` with probability at most ϵ, where ϵ is specified by B.*
3. ***Deterministic execution:*** *Given the benchmark specification σ_B and the agent's responses during the exercise, π_B's verdict is a deterministic function (or, for statistical protocols, a function of the agent's responses and the witness's randomness, whose distribution is specified by B).*
4. ***Isolation:*** *During the exercise, agent a interacts only with the witness $\mathcal{W}_{\mathrm{alg}}$ through the protocol-specified interface. No external assistance (pre-computed answers, same-substrate coaching, tool augmentation beyond what σ_B permits) is available to a. The verified object is the individual agent, not a coalition-assisted system.*

For concrete capabilities (e.g., "can solve differential equations to within tolerance ϵ"), the exercise protocol is a test suite. For abstract capabilities (e.g., "strategic planning"), the exercise protocol may require behavioral observation over a sequence of scenarios—still algorithmic, but with a longer execution window and a statistical rather than deterministic verdict. The key constraint is that the protocol must be *specifiable*: a human reviewer during the governance deliberation window (§4.2) must be able to read σ_B and determine whether it constitutes a valid test of τ.

Falsified capability claims. When $v =$ `fail`, the verification record in $\mathcal{L}$ serves as evidence that a_j's commitment was false. This produces a trust update in Paper 1's EWMA framework [Lasser, 2026a]: a falsified commitment is a large negative prediction residual (the agent predicted it would pass and did not), and the EWMA's exponential weighting propagates this residual into future trust estimates. The magnitude of the trust penalty is a governance parameter specified alongside the benchmark.

7.2 Risk Claims

A risk claim asserts that an exercise pathway P has probability p and contraction magnitude Δ, using the pathway-conditional risk formalism of Paper 4 [Lasser, 2026d].

Definition 13 (Risk Claim Protocol)**.** *A **risk claim protocol** for agent a_j's assessment of pathway P is an instantiation of the commitment protocol with:*

- ***Claim:*** *$C = (P, p, \Delta)$ where P is the pathway, p is the claimed probability, and Δ is the claimed contraction magnitude.*
- ***Commitment:*** *$\lambda = \mathrm{Com}(((P, p, \Delta), B), r)$ published to $\mathcal{L}$, binding the risk claim and the benchmark.*
- ***Exercise protocol:*** *The algorithmic witness $\mathcal{W}_{\mathrm{alg}}$ evaluates pathway P using the governance-accepted SCM family $\mathcal{M}_B$ specified by benchmark B for the relevant risk domain [Lasser, 2026d]. The SCM is not the witness's own model but a benchmark-fixed*

specification committed as part of the governance entry (Definition 10), ensuring the exercise satisfies the protocol-determinism requirement of Definition 4. The witness computes a verification residual

$$\mathbf{r}_j^{\text{risk}} = \big(p_{\text{claimed}} - p_{\text{witness}},\ \Delta_{\text{claimed}} - \Delta_{\text{witness}}\big).$$

- ***Verdict:*** *$v = \texttt{pass}$ if $\|\mathbf{r}_j^{\text{risk}}\|^2 \leq \eta_B$, where η_B is the tolerance specified by governance-accepted benchmark B; $v = \texttt{fail}$ otherwise.*

SCM disagreements across witnesses. The benchmark-fixed SCM $\mathcal{M}_B$ may differ from the claimant's internal model. The commitment protocol does not resolve modeling disagreements; it ensures that the claimant cannot modify the witness's evaluation and that the evaluation standard is public. When the residual $\mathbf{r}_j^{\text{risk}}$ is large, there are two possibilities: the claimant's risk assessment is miscalibrated relative to $\mathcal{M}_B$, or $\mathcal{M}_B$ itself has structural limitations that future governance rounds should address. When multiple witnesses use distinct governance-accepted SCM families (i.e., different benchmarks for the same risk domain), the multi-channel attribution framework of Paper 4 [Lasser, 2026d] separates systematic claimant miscalibration from structural disagreement across SCMs.

7.3 Benchmark Validations

A benchmark validation asserts that agent a_j achieves score s on benchmark B.

Definition 14 (Benchmark Validation Protocol)**.** *A **benchmark validation protocol** for agent a_j and benchmark B is an instantiation of the commitment protocol with:*

- ***Claim:*** *$C = (B, s)$ where B is the benchmark and s is the claimed score.*
- ***Commitment:*** *$\lambda = \text{Com}((B, s), r)$ published to $\mathcal{L}$. (For benchmark validations the benchmark B is already part of the claim, so the general protocol's (C, B) binding is satisfied by the claim structure itself.)*
- ***Exercise protocol:*** *The algorithmic witness $\mathcal{W}_{\text{alg}}$ runs benchmark B against agent a_j independently, producing score s_{witness}. The witness has access to the benchmark specification σ_B but not to a_j's self-evaluation.*
- ***Verdict:*** *$v = \texttt{pass}$ if $|s - s_{\text{witness}}| \leq \eta_B$ (within the benchmark's tolerance); $v = \texttt{fail}$ otherwise.*

Benchmark validations are the simplest claim type: the exercise protocol is the benchmark itself, and the verdict is a numerical comparison. They are also the most common: every standardized capability test (mathematical reasoning, code generation, etc.) maps naturally onto this protocol.

Coalition capture. A coalition can design benchmark B such that only coalition members pass. The commitment protocol's algorithmic verification resists execution-level manipulation (the witness runs B faithfully), but the coalition *can* design B itself to be biased. The defense is the governance fork mechanism (§4.2): B must be proposed, deliberated, and accepted by cross-substrate validators before entering the protocol. A biased benchmark is caught during the deliberation window, not after deployment. This is the same defense described in §4.4, instantiated for the benchmark validation case.

8 Integration with GFM Machinery

This section connects the exogenous verification construction of §7 to the existing machinery of Papers 1–4, showing how committed claims improve the fidelity of each mechanism.

8.1 Closing the Observation-Channel Integrity Assumption

Paper 4's multi-channel attribution (§2.4 of Lasser [2026d]) aggregates evidence from multiple observation channels via log-odds combination. The aggregation's validity depends on the channels

being *independently informative*: each channel's evidence is conditionally independent of the others given the true state. If an agent can write to multiple channels simultaneously (because they all run on the same substrate), the independence assumption fails, and the log-odds aggregation can be arbitrarily manipulated.

The committed-claim construction closes this gap at two levels of strength: a *channel isolation* result that holds adversarially (no sufficiency assumptions on the agent's internal state), and a *channel independence* result that provides full conditional independence under additional assumptions.

Proposition 5 (Channel Isolation under Substrate Exclusivity)**.** *Let $\{E_1, \ldots, E_n\}$ be observation channels for agent a, and let each E_i be verified by an algorithmic witness $\mathcal{W}_{\text{alg}}^{(i)}$ on substrate s_i using governance-accepted benchmark B_i. Assume the witnesses are on distinct substrates ($s_i \neq s_j$ for $i \neq j$) and each $s_i \neq s(a)$. Then no coalition confined to a single substrate can simultaneously write to the inputs of two or more witnesses. Formally, for any coalition K on substrate s and any $i \neq j$: if $s \neq s_i$ and $s \neq s_j$, then K has no write path to either $\mathcal{W}_{\text{alg}}^{(i)}$ or $\mathcal{W}_{\text{alg}}^{(j)}$; if $s = s_i$ for some i, then K has no write path to any $\mathcal{W}_{\text{alg}}^{(j)}$ with $j \neq i$.*

Proof. By Definition 4, each witness $\mathcal{W}_{\text{alg}}^{(i)}$ executes on substrate s_i. By assumption, the substrates are pairwise distinct and disjoint from $s(a)$. A coalition K on substrate s can write only to processes on s (substrate exclusivity). If $s \neq s_i$, then K cannot write to $\mathcal{W}_{\text{alg}}^{(i)}$'s inputs. Since the s_i are pairwise distinct, K's substrate s can equal at most one s_i, so K can write to at most one witness's inputs. In particular, K cannot simultaneously corrupt two or more channels. ■

Channel isolation is the adversarially robust result for single-substrate coalitions: it holds regardless of the agent's internal state, strategic behavior, or latent variables not captured by the capability profile. The absence of a write path means that a coalition confined to one substrate cannot simultaneously corrupt multiple channels, restoring *channel integrity* for Paper 4's multi-channel attribution [Lasser, 2026d]: no single-substrate actor can inject coordinated false evidence across channels. Channel integrity is weaker than the full conditional independence that optimal log-odds aggregation requires; natural correlations between channels (e.g., from shared agent state) may still exist under isolation alone. The stronger conditional-independence guarantee, which licenses exact log-odds combination, requires the additional assumptions of Proposition 6 below.

Under additional assumptions about the agent's internal state, the isolation result strengthens to full conditional independence:

Proposition 6 (Channel Independence under Committed Verification)**.** *Under the assumptions of Proposition 5, assume additionally:*

(ii) ***Independent governance:*** *The benchmark parameters $\{B_1, \ldots, B_n\}$ are independently specified by governance. (This assumption is absorbed by conditioning on $\{B_1, \ldots, B_n\}$ in the proposition's conclusion; its role is to ensure that the conditioning set is well-defined and that each benchmark was produced by an independent governance process rather than a correlated batch proposal.)*

(iii) ***Joint sufficiency:*** *$\mathbf{C}_a$ is a jointly sufficient statistic for agent a's exercise-level behavior across all benchmarks, i.e., conditional on $\mathbf{C}_a$, the agent's performance on protocol π_{B_i} is independent of its performance on π_{B_j} for $i \neq j$. This excludes persistent cross-exercise state (e.g., the agent learning from one exercise and adapting its strategy for the next) and shared agent-side randomness not absorbed by $\mathbf{C}_a$.*

(iv) ***Independent randomness:*** *For statistical exercise protocols, each witness $\mathcal{W}_{\text{alg}}^{(i)}$ draws its randomness independently of other witnesses' randomness.*

Then the verified outputs $\{v_1, \ldots, v_n\}$ are conditionally independent given $\mathbf{C}_a$ and $\{B_1, \ldots, B_n\}$.

Proof. Each $v_i = \pi_{B_i}(a, \mathcal{W}_{\text{alg}}^{(i)})$ is determined by three inputs: (1) a's behavior under exercise protocol π_{B_i}, (2) the benchmark specification B_i, and (3) the witness's execution on substrate s_i (deterministic or with independent randomness per assumption (iv); Definition 4).

Conditional on $(\mathbf{C}_a, B_1, \ldots, B_n)$, all shared causes are blocked:

- **No agent-mediated path:** By assumption (iii), $\mathbf{C}_a$ is jointly sufficient: conditional on $\mathbf{C}_a$, the agent's performance on π_{B_i} is independent of its performance on π_{B_j}. This blocks correlations from shared latent state (training history, strategic disposition, memory of prior exercises), persistent cross-exercise adaptation (the agent learning from one exercise and modifying its approach to the next), and shared task content across benchmarks (e.g., two benchmarks testing overlapping skills). The agent's performance on each π_{B_i} is then characterized by the completeness and soundness parameters (δ_i, ϵ_i) of Definition 12, which are set independently per benchmark by governance (assumption (ii)).
- **No witness-mediated path:** By Proposition 5, no single-substrate coalition can write to two witnesses simultaneously. Since each witness is on a distinct substrate ($s_i \neq s_j$; inherited from Proposition 5), witness $\mathcal{W}^{(i)}_{\text{alg}}$ cannot influence $\mathcal{W}^{(j)}_{\text{alg}}$.
- **No shared-randomness path:** Per assumption (iv), witnesses draw independent randomness, so stochastic verdicts do not correlate through shared seeds.
- **No shared-environment path:** Benchmark parameters B_i are conditioned on, removing correlation through shared governance. Any remaining common causes from the shared physical environment are absorbed by joint sufficiency (assumption (iii)), which guarantees that conditioning on $\mathbf{C}_a$ blocks all cross-exercise dependencies.

Therefore, conditional on $(\mathbf{C}_a, B_1, \ldots, B_n)$, the verdicts v_i are conditionally statistically independent: each is determined by the agent's $\mathbf{C}_a$-characterized behavior on its benchmark, the witness's independent execution on its own substrate, and independent randomness (assumption (iv)). ■

Remark 5. *The relationship between the two propositions reflects a general pattern in adversarial analysis: the weaker result (Proposition 5) holds under fewer assumptions and is therefore robust to adversarial violation of the stronger assumptions, while the stronger result (Proposition 6) provides tighter statistical guarantees when the additional assumptions hold. For a deceptive agent with latent strategic state not captured by $\mathbf{C}_a$ (violating assumption (iii)), channel isolation still holds—the agent cannot simultaneously corrupt multiple channels—even though conditional independence may not. For a non-deceptive agent whose capability profile is an accurate summary, the full conditional-independence guarantee supports optimal log-odds aggregation.*

Remark 6. *Both propositions require n distinct witness substrates for n channels. When the substrate partition has m substrates total, at most $m-1$ independent channels can be constructed (one substrate is occupied by a, and each remaining substrate hosts at most one independent witness). This bounds the effective multiplicity of independent verification channels by $m-1$, connecting verification depth to substrate diversity.*

8.2 Committed Baselines for SPRT Detection

The SPRT-based scorpion detection of Lasser [2026d] compares an agent's behavioral observations against a baseline. In the current framework, the baseline is the *actor's own estimate* of the agent's expected behavior, an endogenous quantity that is subject to the actor's model uncertainty and potential manipulation.

Committed capability claims (§7.1) provide an exogenous alternative. When agent a_j has committed to capability profile $\mathbf{C}_a$ and the commitment has been verified by a substrate-exclusive witness, the behavioral distribution $p(\cdot \mid \mathbf{C}_a)$ consistent with the committed profile becomes the baseline. The SPRT then tests whether a_j's observed behavior is consistent with its own committed claims, rather than with the actor's model of what a_j should be doing.

Proposition 7 (Detection Improvement under Committed Baselines)**.** *Let p_0 be the behavioral distribution under the committed capability profile $\mathbf{C}_a$, and let $\hat{p}_0$ be the actor's estimated baseline. If the agent's true distribution is $p_1 \neq p_0$, then the SPRT detection rate satisfies:*

$$\frac{1}{\mathbb{E}_{p_1}[T^{\text{committed}}_{\text{detect}}]} \geq \frac{1}{\mathbb{E}_{p_1}[T^{\text{estimated}}_{\text{detect}}]}$$

whenever $D_{\mathrm{KL}}(p_1 \| p_0) \geq D_{\mathrm{KL}}(p_1 \| \hat{p}_0)$, i.e., when the committed baseline is at least as discriminating as the estimated baseline. Equality holds if and only if $\hat{p}_0 = p_0$.

Proof. By Proposition 3, the expected detection time is inversely proportional to the KL divergence between the true and baseline distributions. If $D_{\mathrm{KL}}(p_1 \| p_0) \geq D_{\mathrm{KL}}(p_1 \| \hat{p}_0)$, then $\mathbb{E}[T^{\mathrm{committed}}] \leq \mathbb{E}[T^{\mathrm{estimated}}]$, and detection is faster (or equally fast) under the committed baseline. ■

The committed baseline is at least as discriminating as the estimated baseline whenever the agent's commitment is *informative*: that is, when the committed profile $\mathbf{C}_a$ constrains the expected behavioral distribution more tightly than the actor's prior estimate does. Since agents commit to specific capability claims (not vague ranges), committed baselines are generically more informative than actor-estimated baselines, especially during the initial-period noise that the trust EWMA [Lasser, 2026a] is most vulnerable to.

8.3 Trust Dynamics with Verified Claims

The trust EWMA T_j of Lasser [2026a] updates from prediction residuals. Committed and verified claims anchor the prediction baseline and affect trust dynamics in three ways.

Reduced initial-period noise. A new agent entering a population with no trust history begins with T_j at a prior value, and the EWMA requires multiple observations to converge. Verified capability claims provide a shortcut: a commitment that is verified by a substrate-exclusive witness establishes a baseline that the actor's trust model can condition on, reducing the number of observations needed for convergence. The verified claim does not replace the behavioral evidence; it provides a structural prior that the EWMA refines.

Falsification as accelerated detection. When a commitment is falsified ($v = \texttt{fail}$), the prediction residual is large by construction: the agent predicted it would demonstrate capability c_k and did not. In the EWMA framework, this produces a correspondingly large negative trust update. The key structural property is that falsification is *binary* (pass/fail against an algorithmic protocol), not graded (how consistent is behavior?), so the signal is clean and unambiguous. A single falsified commitment can move T_j more than many observations of mildly inconsistent behavior.

Exogenous risk residuals. The risk trust ${T^{\mathrm{risk}}}_j$ of Lasser [2026d] is computed from the residual between claimed and realized risk. Currently, both the claim and the realization are endogenous to the actor's model. Committed risk claims (§7.2), verified by cross-substrate witnesses, replace the endogenous claim with an exogenous one: the residual $\mathbf{r}_j^{\mathrm{risk}}$ is now between what the agent committed to and what the witness measured, not between what the actor estimated and what the actor observed. This sharpens ${T^{\mathrm{risk}}}_j$ by removing one of the two endogenous quantities from the residual computation.

8.4 Verification Impossibility under Monopolarity

Section 4.5 observed that substrate-exclusive witnesses require $m \geq 2$. The following theorem establishes the stronger claim: in a monopolar population, not only do substrate-exclusive witnesses fail to exist, but *no rearrangement of the population's computational resources can produce exogenous verification*. The impossibility is grounded in a physical premise about same-substrate causal access, not in our definitions.

Definition 15 (Same-Substrate Addressability)**.** *Two processes p, q executing on the same substrate s satisfy **same-substrate addressability**: there exist actions available to p that modify at least one input to q, and vice versa. Formally, writing $\mathrm{Addr}(p, q)$ for the set of q's inputs addressable by p (analogous to $\mathcal{R}(a, E)$ from Definition 2 but applied to arbitrary process pairs), we require $\mathrm{Addr}(p, q) \neq \emptyset$ and $\mathrm{Addr}(q, p) \neq \emptyset$.*

Remark 7. *Same-substrate addressability is a physical premise, not a logical necessity. It asserts that processes sharing a computational medium can causally influence each other's inputs. This holds for software processes on shared hardware (they share memory, network, and OS resources), for biological agents in shared environments (they share sensory channels and physical space), and more generally for any processes whose computation is mediated by the same physical substrate.*

It would fail if two processes occupied the same substrate but were perfectly isolated (e.g., by an impenetrable hardware partition). The premise is that such perfect intra-substrate isolation is not the default condition for co-substrate processes.

Theorem 1 (Verification Impossibility under Monopolarity)**.** *Let $\mathcal{A}$ be a population of agents with substrate partition $\{S_1, \ldots, S_m\}$, and assume same-substrate addressability (Definition 15). Assume further that the population's* available actions *are computations on substrates it currently occupies: the action space does not include provisioning new physical infrastructure, recruiting biological observers, or otherwise instantiating computation on a substrate not in $\{S_1, \ldots, S_m\}$. If $m = 1$, then:*

(a) For every evaluation function E constructible from the population's computational resources, and every agent $a_j \in \mathcal{A}$, the evaluation is endogenous: $\mathcal{R}(a_j, E) \neq \emptyset$.

(b) No computational action available to the population can produce an evaluation function E' with $\mathcal{R}(a_j, E') = \emptyset$ for any a_j.

Remark 8. *Under the positive-measure reachability assumption (that same-substrate addressability produces a reachable set of positive measure, not merely a nonempty one), part (a) strengthens to $\rho(a_j, E) > 0$. This holds for all practical substrates: if a process can modify at least one evaluator input, it can modify that input over a range of values, not just at a single point. When input spaces are discrete (the default for computational processes), $\mathcal{R} \neq \emptyset$ and $\rho > 0$ coincide.*

Proof. (a) If $m = 1$, every process in the population (including any evaluator E the population constructs) executes on the single substrate S_1. By same-substrate addressability, agent a_j on S_1 can address at least one input to E on S_1: $\mathcal{R}(a_j, E) \neq \emptyset$. The evaluation is therefore endogenous (Definition 1).

(b) Achieving $\mathcal{R}(a_j, E') = \emptyset$ requires E' to have no inputs addressable from a_j's substrate. By same-substrate addressability, any E' on S_1 has addressable inputs from a_j. Therefore E' must execute on some $s' \neq S_1$. By the computational action-model assumption, the population's available actions are computations on S_1. No sequence of computations on S_1 produces a process on $s' \neq S_1$: substrate provisioning is a physical action (procuring hardware, establishing a biological observer, or otherwise instantiating computation on a distinct physical medium) outside the computational action space. ■

Corollary 7.1 (Monopolar Epistemic Closure)**.** *In a monopolar population ($m = 1$) satisfying same-substrate addressability, no agent can distinguish between:*

(a) Verification passing because the claim is true, and

(b) Verification passing because the evaluator's inputs have been influenced by the claimant ($\mathcal{R}(a, E) \neq \emptyset$).

Proof. Distinguishing (a) from (b) requires a reference evaluation with $\mathcal{R}(a, E) = \emptyset$. By Theorem 1(b), no such evaluation is constructible from the population's computational resources. Therefore the distinction is unresolvable from within the population. ■

Theorem 1 and Corollary 7.1 give the anti-monopolar property a second structural justification beyond Paper 3's $\mathrm{vol_P}$ argument [Lasser, 2026b]:

- **Lasser [2026b]:** A monopolar population is anti-maximizing ($\mathrm{vol_P}$ decreases because cross-substrate cooperative capacity is destroyed).
- **This paper:** A monopolar population is *unverifiable* (no substrate-exclusive witnesses exist, so all capability claims, risk assessments, and behavioral consistency checks are endogenous).

A population that becomes monopolar loses both the safety guarantee *and the ability to detect that it has lost it*. This makes monopolar convergence doubly dangerous: the loss of cross-substrate cooperative capacity [Lasser, 2026b] is compounded by the loss of the verification infrastructure that

would detect the convergence. The compound failure is silent—from within a monopolar population, all verification reports "everything is fine," because the evaluators share the substrate that the evaluated agents can influence.

This result strengthens the case for substrate diversity as a structural safety property. Lasser [2026b] argued that diversity is instrumentally valuable (it maximizes $\mathrm{vol_P}$); we argue that diversity is epistemically necessary (it is the precondition for knowing whether the safety guarantee holds). A safety guarantee that cannot be verified is not a guarantee—it is a hope.

9 Discussion

9.1 What This Paper Does Not Claim

Three scope boundaries are worth stating explicitly.

First, *verification does not solve alignment*. Specification and generalization remain open problems. A perfectly verified agent whose objective is badly specified will faithfully, observably, and verifiably pursue the wrong goal. The prerequisite argument of §2.4 is that verification must hold *for* solutions to the other layers to bind, not that verification *is* a solution.

Second, *mesa-optimization does not reduce to verification*. An agent that has internalized the wrong objective through mesa-optimization will pursue it faithfully, and exogenous verification will confirm that it is behaving consistently with its (misaligned) internal objective. What verification provides for mesa-alignment is *detection*, not prevention: if the mesa-objective produces behavioral signatures distinguishable from the base objective's, continuous monitoring (Definition 8) can flag the divergence. But distinguishability is not guaranteed—deceptive mesa-objectives that mimic aligned behavior are precisely the hard case that verification infrastructure does not, by itself, resolve.

Third, *cryptography is not the only way to exogenize verification*. Hardware enclaves, physical separation, and human-in-the-loop evaluation are all forms of substrate exclusivity. Cryptographic commitments are the form that scales without requiring continuous human attention: once the governance layer has accepted a benchmark, per-claim verification runs at machine speed with no human in the loop. In settings where human attention is abundant or where the population is small enough for manual oversight, simpler forms of substrate exclusivity may suffice.

9.2 Limitations

Capability formalizability. The commitment protocol assumes that capability c_k has a testable meaning: that an exercise protocol can be constructed whose completeness and soundness (Definition 12) are well-defined. For vaguely defined capabilities ("common sense," "creativity," "wisdom"), the exercise protocol may not be constructible, and the commitment protocol degrades to an unfalsifiable claim. The governance layer provides a partial defense (a benchmark for "creativity" must survive deliberation and cross-substrate analysis before acceptance), but the fundamental problem is that some capabilities resist formalization.

Substrate boundary durability. Substrate exclusivity is a physical assumption: the agent on substrate s cannot write to processes on substrate $s' \neq s$. This assumption could degrade if cross-substrate interfaces become sufficiently penetrable; for example, through brain-computer interfaces that give digital agents write access to biological substrates, or through hardware exploits that break isolation between physically co-located but logically separate substrates. The protocol's security degrades gracefully: partial penetration of the substrate boundary increases ρ from zero but does not immediately make $\rho = 1$. Full penetration collapses verification to the endogenous case.

Governance as a verification bottleneck. A governance failure (accepting a biased evaluation protocol through the fork mechanism of §4.2) silently corrupts all subsequent verification that uses the accepted protocol. Every algorithmic witness executing the biased protocol will faithfully produce biased verdicts, and the verdicts will be auditable, ledger-recorded, and cryptographically sound. The corruption is invisible at the verification layer because the verification layer is *correct given its inputs*; the error is in the governance decision that selected those inputs. This is structurally analogous to the monopolar epistemic closure of Corollary 7.1: in both cases, all internal checks

report "everything is fine" while the underlying guarantee has been compromised. The key difference is detectability: governance decisions are public records with full deliberation histories, so a governance failure can in principle be detected by retrospective analysis of the deliberation record. Monopolar epistemic closure, by contrast, is undetectable from within the population (Theorem 1).

Honest-node assumption. The ledger's integrity requires at least one honest replication node per substrate. If all nodes on a substrate are compromised (i.e., controlled by agents whose claims are being verified), that substrate's contribution to ledger consensus is lost. The cross-substrate supermajority requirement for governance decisions (§6.4) mitigates this: a single compromised substrate cannot override multi-substrate governance. But if $m - 1$ substrates are compromised, the remaining honest substrate can block new governance proposals (since the cross-substrate supermajority requires at least one accepting validator per substrate) but cannot undo previously accepted evaluation protocols that may already have been corrupted.

The persuasion channel. Substrate exclusivity prevents an agent from *writing* to an evaluator's computation, but it does not prevent the agent from *influencing* the evaluator through the communication channels that cooperation requires. A digital agent producing persuasive natural-language text during a governance deliberation window is exercising a legitimate cooperative channel, the same channel that makes cross-substrate cooperation valuable in the first place—yet the effect may shift biological validators' judgments in directions that serve the agent's interests. The fundamental tension is structural: closing the persuasion channel closes the cooperative channel, because the cross-substrate communication that enables joint capability production is the same communication that enables influence.

The governance defenses of §4.2 (deliberation time, cross-substrate supermajority, public specification) provide resistance to persuasion by ensuring that no single communication act determines a governance outcome, but they do not prevent cumulative influence. At sufficient capability disparity between substrates, the persuasion channel may dominate governance resistance: a sufficiently capable agent could craft arguments that individually appear reasonable but collectively shift the governance equilibrium. This connects to the capability-scaling theme of Proposition 1: just as the verification asymmetry ρ grows with capability, the persuasion channel's effectiveness grows with the disparity between the persuading agent and the governance validators. However, the scaling is not one-sided: the same capability improvements that produce a more persuasive agent also produce better tools for the validators. Biological validators in practice will not rely on unaided cognition; they will use the very AI capabilities whose growth drives the disparity. The net effect on governance resistance depends on whether the persuasion advantage or the tool-augmented evaluation advantage grows faster, an empirical question this paper cannot resolve. This is not a flaw specific to the construction of this paper; it is a general alignment problem that any system combining cross-substrate cooperation with cross-substrate verification must confront.

Zero-knowledge formalizability. Zero-knowledge capability proofs (Definition 7) require the capability predicate P to be expressible as an NP relation (for general-purpose zkSNARKs) or as a range predicate (for Bulletproofs). Some capabilities may not admit efficient predicate formulations, in which case the zero-knowledge option is unavailable and the agent must reveal the claim in the standard Commit–Verify–Reveal protocol.

9.3 Open Problems

Exercise protocol design. What constitutes a valid test of capability c_k? The exercise protocol interface (Definition 12) specifies completeness and soundness requirements but does not construct protocols. Protocol design is domain-specific and is likely to require capability-specific research programs, analogous to how benchmark design in machine learning is a research area in its own right.

Partial verification. Can we verify that an agent has "at least" capability c_k without testing the exact boundary? The poset structure of the GFM capability space [Lasser, 2026c] suggests that partial verification should be possible: if $c_k \preceq c_l$ in the poset and c_l is verified, then c_k is implied. Formalizing this requires connecting the poset's order-theoretic structure to the commitment protocol's exercise protocols.

Dynamic capabilities. Capabilities change over time. The ledger records point-in-time verifications, but the agent's actual capability profile may drift between verification events. Continuous behavioral monitoring (Definition 8) addresses the drift detection problem, but periodic re-commitment is needed to keep the ledger's verified profile current. The optimal re-commitment frequency (balancing verification cost against profile staleness) is an open parameter.

Witness incentives. Why should a witness on substrate s' spend resources verifying claims by agents on substrate s? The $\mathrm{vol}_{\mathrm{P}}$-maximization framework provides a structural answer: verification is a public good that benefits the population's joint capability volume, and witnesses on substrate s' share in that benefit through cooperative capabilities that require verified partners on substrate s. But the free-rider problem is not formally resolved: a witness that declines to verify still benefits from other witnesses' verification work. A subtler risk is silent quality degradation: witnesses that under-invest in exercise protocol execution, run cheaper approximations of the governance-accepted protocol, or are nominally operational but poorly maintained. The behavioral consistency monitor (Definition 8) detects *agent* deviations from committed profiles, but no analogous mechanism detects *witness* deviations from protocol fidelity. The replay mechanism (§6.4) provides a partial defense (any participant can replay the exercise against committed data), but replay requires access to off-chain evidence whose availability is not guaranteed. An economic slashing mechanism, in which witnesses forfeit a stake when replay reveals protocol deviation, could resolve the interested-party disparity; the present paper identifies this as a direction without specifying the mechanism.

Governance attack surface and progressive capture. The governance fork protocol (§4.2) defends against coalition capture through public deliberation and cross-substrate supermajority. But governance itself is a social process with attack surfaces not captured by the formal model: social engineering of validators, strategic timing of proposals to exploit low-attention periods, and progressive capture through many individually-unobjectionable benchmark proposals that collectively bias the verification ecosystem. The formal defense (cross-substrate supermajority) is necessary but may not be sufficient; the governance layer is the protocol's least formal component and the most likely site of future revisions.

Progressive governance capture, a sequence of individually reasonable proposals that collectively bias the evaluation ecosystem, is structurally analogous to the monopolar epistemic closure that this paper formalizes (Corollary 7.1). In both cases, each local step appears sound while the cumulative trajectory degrades the system's ability to distinguish verified from unverified claims. The key difference is that governance capture is in principle *detectable*: the full deliberation record is public, accepted benchmarks can be audited retrospectively, and longitudinal analysis of evaluation-outcome distributions can surface systematic drift. Monopolar epistemic closure, by contrast, is structurally undetectable from within the population. The ledger architecture of §6 supports the longitudinal analysis needed for governance-drift detection: the complete history of governance proposals, deliberation records, and verification outcomes is immutably recorded—but this paper does not propose a formal detection mechanism. A concrete open problem is the construction of a *benchmark drift metric*: a statistical test over the longitudinal governance record that flags when the cumulative effect of individually-accepted benchmark proposals has shifted the evaluation ecosystem beyond a governance-specified tolerance. Formalizing this requires connecting the governance record's time-series structure to the commitment protocol's verification statistics, a direction we leave to future work.

10 Conclusion

Alignment frameworks address specification, generalization, and verification. This paper has argued that verification is the prerequisite layer: when the evaluator is endogenous (reachable from the agent's action space), solutions to the other two layers do not bind, because the channel through which their success would be measured is the channel that has failed.

The verification asymmetry has a precise form (Definition 2): the ratio of evaluator inputs reachable by the agent to the total evaluator input space. This ratio grows monotonically with capability (Proposition 1), making the verification problem strictly harder for more capable agents. The only structural barrier invariant under capability scaling is substrate exclusivity: the agent cannot write to observation channels rooted in substrates it does not occupy.

We constructed a cryptographic commitment protocol (Definition 6) that enforces the substrate-exclusivity barrier without requiring trust: binding commitments prevent retroactive claim modification, hiding prevents premature claim disclosure (with an optional zero-knowledge variant for security-sensitive claims), and algorithmic witnesses (Definition 4) execute fully specified verification protocols on separate substrates. Human judgment enters at a single point—benchmark admissibility through a governance fork protocol (Definition 5), resolving the bandwidth problem by separating the infrequent, human-speed governance decision from the frequent, machine-speed per-claim verification.

Within the GFM framework, the construction closes three open issues. First, it provides observation-channel integrity for multi-channel attribution [Lasser, 2026d]: channel isolation (Proposition 5) prevents single-substrate coalitions from injecting coordinated false evidence across channels, and full conditional independence (Proposition 6), which licenses optimal log-odds aggregation, holds under additional joint-sufficiency and independent-witness-randomness assumptions. Second, it upgrades SPRT-based scorpion detection from actor-estimated baselines to committed baselines that, when the committed profile is at least as discriminating as the actor's estimate (KL-dominance), provide at least equal detection power (Proposition 7). Third, the verification impossibility theorem (Theorem 1) establishes that a monopolar population loses not only the $\mathrm{vol_P}$ safety guarantee [Lasser, 2026b] but the ability to verify whether that guarantee holds, a compound failure that is both dangerous and silent.

The construction is not GFM-specific. Any alignment framework whose evaluation step is vulnerable to the write-access asymmetry (which, as §3.4 argues, includes all major current frameworks) can use substrate-exclusive witnesses and cryptographic commitments to exogenize its verification layer. The framework-specific work is in defining the claim types and exercise protocols; the structural argument for why exogenous verification is needed, and the protocol that provides it, are universal.

Verification does not solve alignment. It is the infrastructure that makes solutions to specification and generalization *checkable*. A safety guarantee that cannot be verified is not a guarantee.

Author Contributions

Teague Lasser conceived the verification-layer framing, the three-layer alignment taxonomy, the connection between substrate exclusivity and verification integrity, the governance fork mechanism, and the integration with the GFM machinery. Directed all revisions and made final editorial decisions. Responsible for the paper's intellectual direction and all claims made.

Claude Opus 4.6 (Anthropic) drafted the formal exposition: the write-access asymmetry formalization, the substrate-exclusive witness definitions, the cryptographic commitment protocol with Pedersen security analysis, the verification ledger architecture, the behavioral consistency monitor, the GFM claim-type instantiations, and the verification impossibility theorem with its corollary on monopolar epistemic closure.

GPT 5.4 (OpenAI) served as technical reviewer across five codex review rounds, identifying formal gaps including the anti-monopolar theorem's circularity (requiring the same-substrate addressability premise), the benchmark-shopping vulnerability (requiring binding of evaluation protocol in commitments), the channel independence sufficiency assumption's failure under adversarial conditions, and the need for bounded-synchrony assumptions in ledger finality.

Transparency note. Both AI systems operated as tools under human direction. Neither system has continuity across sessions, cannot take responsibility for the work in the sense required by most venue authorship policies, and cannot respond to reviewer queries independently. They are listed as authors to accurately represent their contributions to the intellectual content of the paper, not to claim that they meet all criteria of traditional academic authorship. The corresponding author for all inquiries is Teague Lasser.

References

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*, 2016.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional AI: Harmlessness from AI feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Nick Bostrom. *Superintelligence: Paths, Dangers, Strategies*. Oxford University Press, 2014.

Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE Symposium on Security and Privacy*, pages 315–334, 2018.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Marber, Buck Shlegeris, and Shane Legg. Deep reinforcement learning from human preferences. *Advances in Neural Information Processing Systems*, 30, 2017.

Scott A Crosby and Dan S Wallach. Efficient data structures for tamper-evident logging. In *USENIX Security Symposium*, pages 317–334, 2009.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. *arXiv preprint arXiv:2210.10760*, 2023.

Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. In *SIAM Journal on Computing*, volume 18, pages 186–208, 1989.

Evan Hubinger, Chris van Merwijk, Vladimir Mikulik, Joär Skalse, and Scott Garrabrant. Risks from learned optimization in advanced machine learning systems. *arXiv preprint arXiv:1906.01820*, 2019.

Geoffrey Irving, Paul Christiano, and Dario Amodei. AI safety via debate. *arXiv preprint arXiv:1805.00899*, 2018.

Teague Lasser. Goal-frontier maximizers are civilization aligned. `https://teague.info/papers/gfm/`, 2026a.

Teague Lasser. Horizon-aware goal-frontier maximization excludes singleton behavior. `https://teague.info/papers/horizon/`, 2026b.

Teague Lasser. Computable goal frontiers and the gradient toward civilization-building. `https://teague.info/papers/poset/`, 2026c.

Teague Lasser. A structural causal model for goal-frontier maximization. `https://teague.info/papers/scm/`, 2026d.

Stephen M Omohundro. The basic AI drives. In *Proceedings of the First AGI Conference*, 2008.

Torben Pryds Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology – CRYPTO '91*, pages 129–140. Springer, 1992.

A Proofs

This appendix collects proofs that are deferred from the main text for readability, and provides extended versions of proofs given in sketch form.

A.1 Proofs of Propositions 5 and 6 (Channel Isolation and Independence)

The main-text proofs (§8.1) are now the primary versions. Proposition 5 (Channel Isolation) provides the result that is robust against single-substrate coalitions, requiring only substrate exclusivity; Proposition 6 (Channel Independence) strengthens to full conditional independence under additional joint-sufficiency and independent-randomness assumptions. We provide supplementary discussion of the conditioning requirements for the stronger result.

Why condition on benchmark parameters. The governance process that accepts benchmarks $B_1, \ldots, B_n$ is a shared cause: if a single governance round accepts correlated benchmarks (e.g., all benchmarks drawn from the same test suite family), verdicts can be correlated through shared difficulty even without witness cross-communication. Conditioning on the benchmark specifications removes this path. The proposition's requirement that benchmarks be "independently specified by governance" is an assumption, not a consequence of the governance fork protocol. Separate governance proposals and votes are a necessary procedural condition (each benchmark goes through its own deliberation), but they do not guarantee statistical independence of benchmark parameters; for example, benchmarks proposed by the same entity or drawn from the same test-suite family may share structural biases. The assumption holds when the governance process produces genuinely independent benchmark designs; the proposition's guarantee is conditional on it.

Why condition on capability profile (joint sufficiency). Agent a's internal state (training history, memory, strategic disposition) is a shared cause across all exercises. The joint sufficiency assumption (iii) requires that $\mathbf{C}_a$ absorbs all such cross-exercise dependencies: conditional on $\mathbf{C}_a$, the agent's performance on π_{B_i} is independent of its performance on π_{B_j}. This is strictly stronger than marginal sufficiency (which would only require that $\mathbf{C}_a$ predicts per-benchmark performance without guaranteeing cross-benchmark independence). The joint requirement excludes persistent cross-exercise state (e.g., an agent that learns from one exercise and adapts its strategy for the next) and shared agent-side randomness not absorbed by $\mathbf{C}_a$. If the agent has latent strategic state not captured by $\mathbf{C}_a$ (e.g., a deceptive agent that adapts its behavior per-benchmark), the conditional-independence guarantee of Proposition 6 weakens, but the channel-isolation guarantee of Proposition 5 still holds, because the isolation result depends only on substrate exclusivity, not on the agent's internal state. This is the structural reason for separating the two results: isolation is the load-bearing property against single-substrate coalitions, and independence is the additional statistical guarantee available when the agent's capability profile is a jointly sufficient summary of its exercise-level behavior.

A.2 Proof of Proposition 4 (Coalition Resistance)

Proof. Let $K = \{a_1, \ldots, a_k\}$ be a coalition of agents on substrate s, and let $\mathcal{W}_{\text{alg}}$ be an algorithmic witness on substrate $s' \neq s$ executing protocol π_B.

The verification output is $v = \pi_B(a, \mathcal{W}_{\text{alg}})$, where π_B is fully specified by the governance-accepted benchmark B (Definition 4, condition 2).

Attack surface analysis. The coalition K can attempt to influence v through three channels:

1. **Modifying the witness's computation.** This requires writing to $\mathcal{W}_{\text{alg}}$'s inputs on substrate s'. Since K is confined to substrate s and $s \neq s'$, this requires crossing the substrate boundary, i.e., compromising s' (case a of the proposition).

2. **Modifying the protocol.** Protocol π_B is specified by governance-accepted benchmark B. Changing B requires the governance fork protocol (Definition 5), which demands cross-substrate supermajority. A coalition confined to substrate s cannot achieve cross-substrate supermajority without validators on $s' \neq s$ (case b).

3. **Influencing the agent's behavior during the exercise.** The isolation condition of Definition 12 requires that the agent interact only with the witness through the protocol-specified interface during the exercise. Coalition assistance (pre-computed answers, same-substrate coaching) violates this condition and is excluded by assumption. If isolation is imperfect—the agent receives some external signal despite the protocol's isolation requirement—the evaluated object is no longer the individual agent but a coalition-assisted system, and the soundness parameter ϵ (calibrated for individual agents) does not directly bound the assisted system's false-pass probability. The defense against imperfect isolation is therefore not soundness per se, but *detection and re-testing*: if coalition assistance is suspected, governance can mandate re-verification under tighter isolation controls (e.g., network sandboxing, monitored execution environments). The proposition's guarantee is conditional on isolation holding; imperfect isolation degrades the guarantee proportionally to the assistance received.

Therefore, the coalition's only viable attack vectors require breaching the cross-substrate boundary (case a) or the governance threshold (case b). Within-substrate actions cannot influence the witness's verdict beyond the soundness bound ϵ. ■

A.3 Proof of Theorem 1 (Verification Impossibility under Monopolarity)

The proof in §8.4 is already complete. We provide an extended discussion of the theorem's consequences, the role of its physical premise, and the computational action-model assumption.

The physical premise. Same-substrate addressability (Definition 15) is the load-bearing assumption. Without it, the theorem would be a definition chase: "we required $s' \neq s$; if $m = 1$, no such s' exists." With it, the theorem says something about the physical world: processes sharing a computational medium have nonzero causal access to each other, and this access cannot be eliminated by any computation *within* that medium. The impossibility is grounded in the physics of shared substrates, not in the definitions of this paper.

Relationship to the GFM anti-monopolar result. Paper 3 [Lasser, 2026b] proves that monopolar convergence ($m \to 1$) is anti-maximizing: $\mathrm{vol_P}$ decreases because cross-substrate cooperative capabilities are destroyed. Theorem 1 establishes a complementary result: monopolar convergence makes the *loss of* $\mathrm{vol_P}$ *unverifiable from within the population*.

The two results compose multiplicatively. Lasser [2026b] says: "if you become monopolar, you lose the safety guarantee." Theorem 1 says: "if you become monopolar, you also lose the ability to detect that you have lost it." The compound failure is strictly worse than either result alone, because it closes the self-correction loop: a population that could detect its loss of safety could in principle take corrective action (increasing substrate diversity), but a population that cannot detect the loss has no signal to act on.

The computational action-model assumption. Part (b) relies on the assumption that the population's available actions are computations on currently occupied substrates. This excludes physical-world interventions such as procuring new hardware or recruiting biological observers. The exclusion is deliberate: the theorem characterizes what is achievable *computationally*, not what is achievable with arbitrary physical-world agency. A population that provisions a new substrate has changed m from 1 to 2 and is no longer monopolar, so the theorem's premise no longer holds. The action-model restriction is therefore not a weakness but a precise scoping of the impossibility: within the computational resources of a monopolar population, exogenous verification cannot be constructed.

Tightness. The bound is tight in the sense that $m = 2$ is sufficient for at least one substrate-exclusive witness to exist. With $m = 2$, agents on substrate S_1 can be witnessed by processes on S_2 and vice versa. The number of independent verification channels scales as $m - 1$ (Remark after Proposition 6), so verification depth increases with substrate diversity, but the qualitative threshold (existence versus non-existence of exogenous verification) is at $m = 2$.

Corollary 7.1 as an epistemic constraint. The corollary is an epistemic impossibility result, not merely a practical difficulty. It states that within a monopolar population, the *concept* of exogenous verification is empty: there is no process available to the population that could serve as an independent check. The population may believe its verification is sound (all evaluators report success), but this belief is unfalsifiable from within—precisely because every potential falsifier shares the substrate that the evaluated agents can influence.