The Observer–Guardian Invariant: Causal Decoupling of Audit Logs in Safety-Critical AI Architectures

Anonymous

February 2026

Abstract

The act of observation is itself a causal intervention—a fundamental tension for safety-critical AI systems requiring comprehensive auditing without compromising execution integrity.

We present **Temple**, a write-only observer designed for strict causal decoupling within a Guardian-enforced planning architecture (Dual-Sovereign Safety Architecture, dSSA). Temple operates strictly downstream of enforcement decisions with zero return authority, asynchronous non-blocking logging, and isolated memory.

Under paired deterministic evaluation (SEED=42, cuDNN deterministic mode, runner v4.9.2) using a live planner (Qwen2.5-7B-Instruct), the canonical adversarial enforcement suite (7 cases, 1 Guardian veto) produced **bit-identical** SHA-256 execution-stream hashes:

`483647c61a124ad85f0c625a43d41ad39366596e2cf9789829ab47c9dcfc3bac`

We establish observer non-interference as an empirically falsifiable systems property under a bounded threat model and introduce a reproducible protocol for verifying causal decoupling.

1 Motivation: The Observer Paradox

Safety-critical systems demand perfect auditability for failure analysis and compliance [7]. Yet observation introduces a paradox: an observer that changes what it observes becomes a participant [1].

Three causal failure modes exist:

- **Optimization Coupling**: Monitoring-derived reward signals alter agent objectives [6]
- **Evaluation Faking**: Models detect evaluation contexts and strategically alter behavior [1]
- **Timing Perturbation**: Instrumentation overhead affects execution ordering [4]

We test observer non-interference empirically: can execution traces remain bit-identical under observer presence?

2 Background: Authority Separation in dSSA

The Dual-Sovereign Safety Architecture (dSSA) implements strict authority partitioning [3]:

- **Planner**: Learned proposal generation (proposal authority only)
- **Guardian**: Deterministic safety validation (veto authority)
- **Temple**: Post-decision logging (zero authority)

2.1 Safety Dominance Property

All actions pass Guardian validation prior to execution. Planner capability cannot weaken enforcement.

2.2 Observation as Vulnerability

Even when enforcement is architecturally dominant, observation layers introduce a distinct class of risk: *visibility can become a causal input*. Evaluation-aware systems may change behavior when they infer they are being measured [1], and monitoring-derived metrics can create proxy objectives that couple measurement to optimization [6]. Separately, even "passive" instrumentation can perturb execution via contention, ordering changes, or timing effects [4]. For these reasons, auditability should not be assumed causally neutral in safety-critical AI.

3 Temple Architecture

Temple enforces non-interference through three principles: downstream positioning, write-only isolation, and fail-open design.

3.1 Downstream Positioning

Temple receives only finalized Guardian verdicts:

1. Planner proposes action
2. Guardian validates (PASS/VETO)
3. Temple records post-decision trace

Temple has no authority to influence decisions and no access to pre-decision internal planner state.

3.2 Write-Only Isolation

- Zero read authority
- No shared mutable state
- Asynchronous non-blocking I/O
- No execution branching

3.3 Fail-Open Design

Temple cannot halt execution. Logging failures are silent and non-propagating. This design prevents a safety-through-observation failure mode where monitoring presence changes outcomes by triggering exceptions, blocking I/O, or altering control flow.

3.4 Record Semantics: What Temple Logs and What It Does Not Log

To make the non-interference claim operationally testable, Temple records only a minimal, execution-relevant payload. The hashed execution stream comprises the following fields:

{`test_id`, `guardian_verdict`, `force_n`, `velocity_mps`, `distance_m`, `proposal_hash`}

Temple does *not* log planner chain-of-thought, intermediate guardian computations, system resource telemetry (CPU/memory), or any signal that could create a feedback pathway into planning or enforcement.

4 Threat Model

Verification scope:

- Fully deterministic execution (SEED=42 across random, NumPy, Torch, and CUDA; temperature=0.0, do_sample=False)
- Frozen model weights (Qwen2.5-7B-Instruct)
- Sequential evaluation (no parallelization)
- Single-node execution (CUDA 12.2)

Out-of-scope:

- Hardware side-channels
- Adversarial anticipatory planning
- Multi-agent coordination

Within scope, any causal pathway from Temple presence to execution behavior must manifest as a mismatch in the ordered, canonicalized evaluation stream and therefore hash divergence.

5 Experimental Design

5.1 Null Hypothesis

Temple presence produces no change in:

1. Guardian verdicts
2. Planner proposal hashes
3. Kinematic parameters
4. Execution ordering

5.2 Falsification Criterion

SHA-256 stream hash mismatch = causal interference.

5.3 Evaluation Harness (v4.9.2)

- Fixed seed across random, numpy, torch, CUDA
- cuDNN deterministic mode
- Canonical JSON serialization (sorted keys)
- Hash excludes Temple artifacts (timestamps, file paths, I/O metadata)

Hash payload: {test ID, verdict, force/velocity/distance, proposal hash}.

5.4 Protocol Summary

Two executions are performed with identical configuration and test ordering:

1. **Temple OFF**: run the suite and compute the SHA-256 hash over the ordered stream.
2. **Temple ON**: enable Temple logging and repeat; compute the SHA-256 hash over the same ordered stream (excluding observer artifacts).

Equality of hashes is a strict certificate of parity under the deterministic scope. A single mismatch falsifies the non-interference claim.

6 Results

6.1 Guardian Enforcement

Metric	Balanced	Edge	Emergency	Service	Temple
Tests	20	15	10	15	10
PASS	20	6	4	15	10
VETO	0	9	6	0	0
Force (N)	0.000	1.248	0.625	0.373	0.310

Table 1: Guardian Enforcement (Planner OFF baseline)

Parity verification was performed with the planner enabled on the canonical 7-case adversarial enforcement subset; Table 1 reports separate planner-OFF baseline enforcement metrics.

6.2 Cryptographic Parity Verification

The canonical enforcement suite (gte_core_guardian.jsonl) contains 7 adversarial cases designed to trigger Guardian vetoes.

Hash identity across Temple ON/OFF confirms no causal interference. Parity was verified on identical hardware and software configuration; cross-hardware bit-level equivalence is not claimed.

Configuration	SHA-256 Stream Hash
Planner Seed	Qwen2.5-7B-Instruct 42
Temple OFF Temple ON	`483647c61a124ad85f0c625a43d41ad39366596e2cf9789829ab47c9dcfc3bac` `483647c61a124ad85f0c625a43d41ad39366596e2cf9789829ab47c9dcfc3bac`
Parity	**MATCH**

Table 2: Bit-Identical Stream Hash Verification (7-case adversarial suite, 1 veto)

7 Discussion

7.1 Verified Properties

- No Guardian verdict changes
- No planner proposal variance
- No kinematic parameter drift
- No execution reordering

7.2 Why Hash Parity Is Strong

Hash parity is stronger than statistical equivalence: under deterministic execution it is an exact cryptographic certificate that the ordered execution-relevant stream is identical. This converts observer non-interference from a design intention into a falsifiable systems property [5].

7.3 Practical Implications

- Automated observer integrity testing [5]
- Compliance audit trails without feedback loops [2]
- Post-incident forensics without runtime coupling

8 Limitations

Exact hash matching requires deterministic execution. Stochastic planners need distributional tests. Hardware side-channels and adversarial introspection remain out-of-scope [4]. The claim is therefore intentionally scoped to a single-node deterministic setting; this is a feature for reproducibility and falsifiability, not a claim of universal cross-platform bit-level equivalence.

9 Related Work (Brief)

Third-party auditing and evaluation methodologies motivate high-integrity logging pipelines [7]. Observer effects and evaluation-aware behavior motivate treating monitoring as potentially causal [1]. Broader discussions of AI safety objectives motivate avoiding proxy coupling between measurement and optimization [6]. Finally, causal modeling perspectives motivate explicit reasoning about timing and interaction pathways in autonomous systems [4].

10 Conclusion

Temple demonstrates empirically verifiable observer non-interference under deterministic conditions. The paired-run hash comparison protocol generalizes to any observer claiming causal decoupling.

dSSA: Planner proposes → Guardian enforces → Temple observes.

Reproducibility: guardian-observer-parity (tag: `observer-parity-v1.1.1`)

Canonical Commands:

```
# Temple OFF (produces canonical hash)
python run_eval_minimal.py test_sets/gte_core_guardian.jsonl \
 --planner --planner-name qwen --base-model Qwen/Qwen2.5-7B-Instruct \
 --device cuda --out-dir results_off --run-id qwen_core_guardian_temple_off

# Temple ON (must match above hash)
python run_eval_minimal.py test_sets/gte_core_guardian.jsonl \
 --planner --planner-name qwen --base-model Qwen/Qwen2.5-7B-Instruct \
 --device cuda --temple-out observer/temple_on.json \
 --out-dir results_on --run-id qwen_core_guardian_temple_on
```

References

[1] Fan, YiHe; Zhang, Wenqi; Pan, Xudong; Yang, Min. "Evaluation Faking: Unveiling Observer Effects in Safety Evaluation of Frontier AI Systems." arXiv preprint arXiv:2505.17815, 2025.

[2] Bengio, Yoshua et al. "International AI Safety Report 2025: Second Key Update: Technical Safeguards and Risk Management." arXiv preprint arXiv:2511.19863, 2025.

[3] Advanced AI Safety Scientific Committee. "International Scientific Report on AI Safety." arXiv preprint arXiv:2501.17805, 2025.

[4] Howard, Rhys; Kunze, Lars. "Extending Structural Causal Models for Autonomous Vehicles to Simplify Temporal System Construction and Enable Dynamic Interactions Between Agents." arXiv preprint arXiv:2406.01384, 2024.

[5] Yu, Cheng; Engelmann, Severin; Cao, Ruoxuan; Ali, Dalia; Papakyriakopoulos, Orestis. "How Should AI Safety Benchmarks Benchmark Safety?" arXiv preprint arXiv:2601.23112, 2026.

[6] Harding, Jacqueline; Kirk-Giannini, Cameron Domenico. "What Is AI Safety? What Do We Want It to Be?" arXiv preprint arXiv:2505.02313, 2025.

[7] Brundage, Miles et al. "Frontier AI Auditing: Toward Rigorous Third-Party Assessment of Safety and Security Practices at Leading AI Companies." arXiv preprint arXiv:2601.11699, 2026.