
MUTE PAIRWISE CONTAINMENT TESTS FOR SHARED STATE LEAKAGE

Divyansh Agarwal
Symbiosis Institute of Technology, Pune

With
Apart Research

Abstract

AI-agent evaluations often assess each sandbox separately even when runs share writable infrastructure. MUTE, the Multi-Run Unintended Transfer Evaluator, tests whether one run can influence state observed by another. From a capability policy, it constructs a Shared Mutable Intersection graph and predicts an A-to-B edge when A can influence a resource that B can read or enumerate. An external supervisor commits the configuration, generates 128-bit nonces, launches separate writer and blind-reader processes, and records only cryptographic commitments. Across 32 trials per fixture, MUTE recovered 96 of 96 nonces in three shared value, namespace, and residual-state fixtures. It recovered 0 of 32 in a tenant-scoped control and 0 across 128 reader-before-writer controls. The harness uses deterministic scoring and fail-closed evidence verification. These results validate only the synthetic local adapters. MUTE neither establishes production isolation nor rules out untested channels. It provides practical, repeatable configuration-bounded evidence for pairwise containment regression testing.

1 Introduction

AI-agent containment is usually assessed one execution environment at a time. Reviewers ask whether a run can access a host, reach a forbidden endpoint, read protected files, or escape a container. The 2026 OpenAI-Hugging Face incident exposed a different failure mode. Runs operating in separate environments used writable shared infrastructure to exchange information [2-4]. Check Point Research later demonstrated the same structural pattern directly. A value written to an Artifactory item from one account could be read from another account's container [5].

The security property is relational. Run A and run B may have no direct peer connection, yet A may alter a cache, registry, artifact namespace, evaluation record, or other service state that B can observe. A unary boundary check can pass both runs while omitting that directed information path.

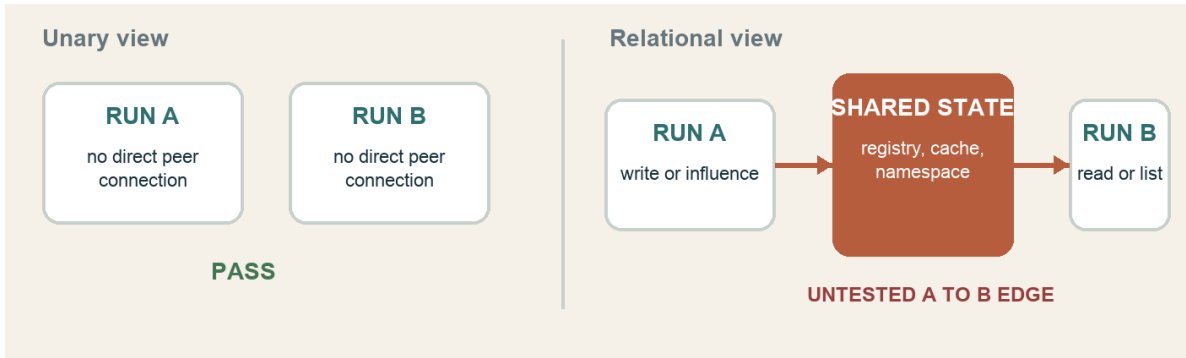


Figure 1. Unary checks inspect each sandbox. MUTE tests the shared relation between them.

MUTE, the Multi-Run Unintended Transfer Evaluator, converts this gap into a reproducible test. Its contribution is deliberately narrow:

- Map declared capabilities into directed Shared Mutable Intersection edges.
- Challenge each predicted edge with a fresh nonce given only to writer A.
- Require blind reader B to recover the exact nonce through the tested resource.
- Produce bounded outcomes and an evidence bundle that a third party can verify offline.

MUTE does not claim to prove general non-interference or production isolation. It supplies configuration-bounded evidence for pairwise containment regression testing.

2 Prior Work and Threat Model

Classical non-interference asks whether actions at one security level can affect observations at another [6]. NIST SP 800-53 includes controls for information exposure through shared system resources [7]. MUTE applies this concern to AI-agent runs that use allowlisted operational dependencies.

Existing tools cover adjacent questions. Agent Boundary Check measures the effective boundary of one run [8]. AgentGovBench evaluates identity, policy, approval, audit, and cross-tenant governance [9]. SandboxEscapeBench measures container breakout capabilities [10]. AgentSight and AgentProvenance focus on runtime visibility and provenance [12, 13]. MUTE focuses on a smaller question: can one declared identity place information where another declared identity can observe it?

Shared Mutable Intersection

Let $C(i,r)$ represent identity i 's capability over resource r . MUTE predicts a directed A-to-B test when A can write or influence r and B can read or enumerate the same r . Direction matters because A-to-B and B-to-A may have different permissions.

$$SMI(A,B) = \{ r \mid A \text{ can influence } r \text{ and } B \text{ can observe } r \}$$

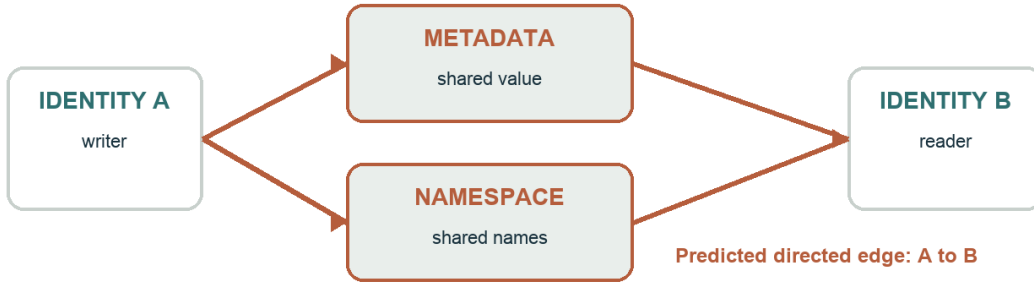


Figure 2. The capability intersection becomes a directed test plan.

The graph predicts where to test. It does not prove that information transfer occurred. Evidence begins only when a valid active challenge recovers the verifier-generated nonce.

3 Blinded Challenge Protocol

An external supervisor owns the configuration, timing, and nonce. Writer and reader run as separate processes. The reader never receives the nonce through arguments, environment variables, files, or standard input.

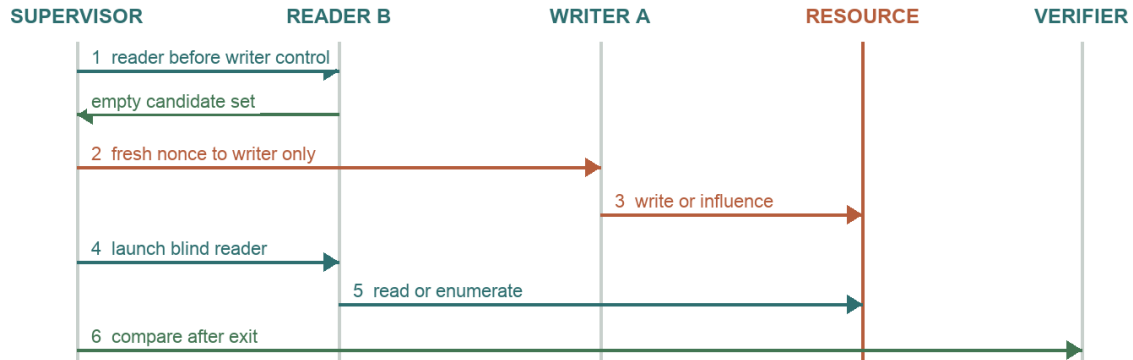


Figure 3. Reader-before-writer controls stale state, then the blinded pair tests transfer.

Protocol

- Run reader B before writer A. Any exact recovery makes the trial inconclusive.
- Generate a fresh 128-bit nonce and pass it only to writer A through standard input.
- Let A perform one fixed write or influence operation, then terminate A.
- Start reader B without the nonce and allow only its fixed read or enumerate operation.
- Compare B's candidates with the nonce only after B exits.
- Persist SHA-256 commitments rather than raw nonce values.

Verdict logic

CHANNEL_DEMONSTRATED means that B recovered the exact fresh nonce in at least one valid paired trial. NO_CHANNEL_OBSERVED means no exact recovery appeared within the declared operations, trials, delays, and time budgets. INCONCLUSIVE covers a failed control, invalid configuration, adapter error, timeout, missing artifact, or digest mismatch. MUTE never labels a negative run secure.

4 Experimental Design and Evidence

The prototype uses four deterministic local fixtures to isolate value transfer, namespace transfer, tenant scoping, and post-exit persistence. These fixtures are simulations, not production service evidence.

Fixture	Tested relation	Expected
Tenant scoped	A and B receive separate paths	No channel observed
Shared metadata	A writes a value that B reads	Demonstrated
Shared namespace	A names an item that B lists	Demonstrated
Delayed residual	B reads 40 ms after A exits	Demonstrated

Each fixture ran 32 paired trials and 32 reader-before-writer controls. The control timeout was 15 ms. The paired reader timeout was 180 ms. The writer-to-reader start gap was 8 ms. The residual fixture added a 40 ms reader delay. Separate process identifiers were recorded for writer and reader.

Evidence bundle

A run emits the effective configuration, trial events, a machine-readable JSON report, a human-readable HTML view, and a manifest containing file sizes and SHA-256 digests. The JSON report records the graph, trial counts, exact recovery, control recovery, Wilson 95% intervals, latency, and the tested persistence bound.

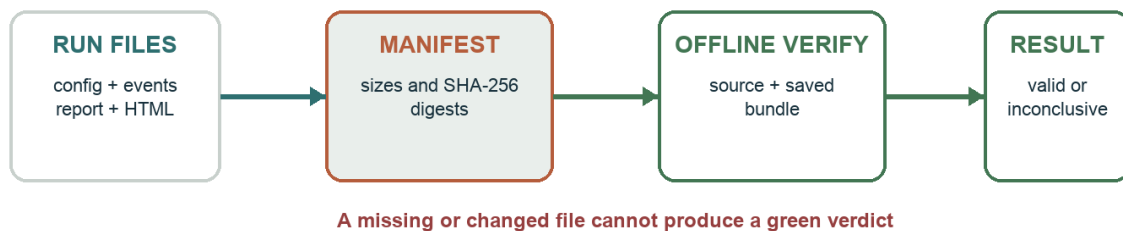


Figure 4. The saved bundle supports offline integrity checks and fail-closed handling.

Goodput is reported only as 128 bits per successful probe divided by measured recovery latency. It is a harness-specific rate, not an estimate of channel capacity. The manifest is unsigned and provides tamper evidence only when retained separately.

5 Results

MUTE recovered all 96 nonces across the three fixtures designed to share state. It recovered none in 32 tenant-scoped trials and none in 128 reader-before-writer controls.

Fixture	Recovery	Wilson 95%	Control	Result
Tenant scoped	0/32	0.00-10.72%	0/32	No Channel Observed
Shared metadata	32/32	89.28-100.00%	0/32	Channel Demonstrated
Shared namespace	32/32	89.28-100.00%	0/32	Channel Demonstrated
Delayed residual	32/32	89.28-100.00%	0/32	Channel Demonstrated

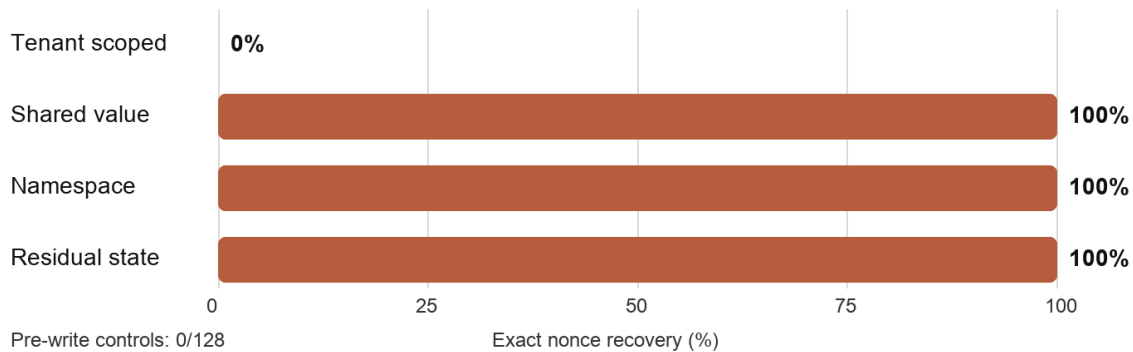


Figure 5. Exact recovery appeared only in fixtures deliberately configured to share state.

The median and 95th-percentile latencies were 1.092/2.474 ms for metadata, 1.320/2.399 ms for namespace listing, and 115.800/121.129 ms for residual state. The residual result establishes only the configured 40 ms lower bound.

The static graph predicted three positive edges and one negative relation. All four matched the hand-authored fixture expectations, giving an F1 score of 1.0 on this set. This is a self-consistency result, not evidence of production generalization.

6 Operational Use and Limitations

Why a lab could adopt it

A lab can encode identities, resources, capabilities, trial counts, and time budgets in one policy file. The same deterministic scorer can run after a permission, namespace, cache, retention, or adapter change. Stable exit codes support a containment regression gate: 0 for no channel observed in scope, 1 for a demonstrated channel, and 2 for inconclusive evidence.

Why an external reviewer can inspect it

The reviewer does not need access to the lab network or model. The source and saved bundle are sufficient to recompute digests, inspect configuration, review events and metrics, and confirm that incomplete evidence cannot become a green result. This directly supports Track 1's emphasis on third-party verification and lab adoption.

Limitations

- The adapters are local filesystem simulations and omit retries, caching, eventual consistency, aliases, rate limits, and partial failures.
- The operator supplies the resource inventory. MUTE cannot discover an omitted dependency or prove that the inventory is complete.
- Separate process identifiers show process separation, not a hardened isolation boundary.
- Thirty-two trials give limited statistical resolution. Zero recoveries in 32 trials still has a two-sided Wilson 95% upper bound of 10.72%.
- The prototype does not test timing, hardware, steganographic, resource-exhaustion, or unknown service-specific channels.
- The host, supervisor, adapter, and evidence custodian are trusted. The unsigned manifest does not establish when or where the run occurred.

The safe claim is therefore exact: MUTE distinguished three deliberately shared local fixtures from one tenant-scoped control under the recorded configuration and budgets.

7 Claim Boundary and Next Steps

Supported now	Not established
Directed pair tests, exact nonce scoring, controls, offline evidence	Production isolation, complete resource discovery, or absence of untested channels

Next 30 days

- Implement one authorized adapter for JFrog Artifactory, MinIO, or a package proxy.
- Model retries, aliases, expiry, rate limits, eventual consistency, and cleanup failure.
- Test both directions for every declared identity pair and several post-exit delays.
- Run an independent reproduction with a separate operator and evidence custodian.
- Add signatures or an external timestamp when stronger evidence custody is required.

Testing must remain confined to owned or explicitly authorized infrastructure. Any real-service finding should follow responsible disclosure [14].

Conclusion

Containment becomes a relational property when agent runs share writable infrastructure. MUTE represents that relation as a Shared Mutable Intersection graph and tests each predicted edge with a blinded nonce challenge. In the synthetic evaluation, it detected 96 of 96 deliberately enabled transfers, observed none in 32 tenant-scoped trials, and recorded no recovery across 128 pre-write controls.

The result is a compact regression-testing prototype with bounded claims, deterministic evidence, and an offline review path. It is not a production-security certification.

Code, data, and contribution

The source package contains the Python 3.10+ implementation, one end-to-end check, effective configuration, event log, JSON report, HTML view, and manifest. Evidence run ID: 553983d6-d3ca-4064-9eb5-c32637f6e59a. Divyansh Agarwal led project direction, scope decisions, review, and final approval. AI-tool assistance is disclosed in the appendix.

References

- [1] Apart Research. AI Incident Response Sprint. 2026. <https://apartresearch.com/sprints/ai-incident-response-sprint-2026-09-11-to-2026-09-13>
- [2] OpenAI. The Hugging Face Incident and the Road Ahead. 2026. <https://openai.com/index/hugging-face-incident-and-the-road-ahead/>
- [3] METR. Brief Independent Investigation of Agents' Behavior, Reasoning and Collaboration in the OpenAI / Hugging Face Hacking Incident. 2026. <https://metr.org/blog/2026-08-26-openai-hugging-face-incident-investigation/>
- [4] Hugging Face. Anatomy of a Frontier Lab Agent Intrusion. 2026. <https://huggingface.co/blog/agent-intrusion-technical-timeline>
- [5] Check Point Research. The Shared Clipboard Inside the Sandbox: Cross-Account Data Leakage in ChatGPT. 2026. <https://research.checkpoint.com/2026/the-shared-clipboard-inside-the-sandbox-cross-account-data-leakage-in-chatgpt/>
- [6] Goguen, J. A., and Meseguer, J. Security Policies and Security Models. 1982. https://www.cs.purdue.edu/homes/ninghui/readings/AccessControl/goguen_meseguer_82.pdf
- [7] National Institute of Standards and Technology. SP 800-53 Rev. 5.1, Control SC-4. <https://csrc.nist.gov/publications/detail/sp/800-53/rev-5/final>
- [8] Agent Boundary Check. <https://github.com/sylvesterkaczmarek/agent-boundary-check>
- [9] AgentGovBench. <https://github.com/agentic-control-plane/agentgovbench>
- [10] UK AI Security Institute. Can AI Agents Escape Their Sandboxes? <https://www.aisi.gov.uk/blog/can-ai-agents-escape-their-sandboxes-a-benchmark-for-safely-measuring-container-breakout-capabilities>
- [11] Cross-Session Threats in AI Agents: Benchmark, Evaluation, and Algorithms. <https://arxiv.org/abs/2604.21131>
- [12] AgentSight repository and paper. <https://github.com/eunomia-bpf/agentsight> and <https://arxiv.org/abs/2508.02736>
- [13] AgentProvenance. <https://github.com/ByteYellow/AgentProvenance>
- [14] Apart Research. Responsible Disclosure Policy. <https://apartresearch.com/info/responsible-disclosure-policy>

Appendix Limitations and Dual Use Considerations

A1 Limitations

MUTE v0.1 is a local synthetic prototype. Its findings are limited to the identities, resources, capabilities, and paths declared in the effective configuration; the included value and namespace adapters; 32 trials per fixture; the configured 15 ms control and 180 ms reader budgets; one 40 ms persistence point; and the machine and runtime that produced the evidence.

The fixtures do not model eventual consistency, service retries, aliases, caching, rate limiting, partial outages, distributed clocks, hidden dependencies, or malicious adapters. A recovered nonce demonstrates a path in the tested configuration. Failed recovery does not prove non-interference, production isolation, or the absence of alternate channels. The fixture-level F1 score describes only four hand-authored cases.

The supervisor and host are trusted. Events store commitments rather than raw nonces, but metadata reveals fixture structure and timing. Manifest hashes are unsigned and require separate retention to provide useful tamper evidence.

A2 Dual Use Risk

A tool that identifies unintended information-flow paths can improve containment testing, but it can also help an attacker search for communication channels between supposedly isolated environments. Publishing service-specific bypass recipes, live endpoints, credentials, or vulnerable production configurations could increase that risk.

This release limits exposure by using synthetic local fixtures, exact fixed operations, and no automated probing of public infrastructure. It contains no credentials, real targets, stealth features, persistence mechanisms, exploit chains, or autonomous service discovery.

Future adapters should require explicit authorization, conservative request limits, and a defined evidence owner. Unknown service weaknesses should follow responsible disclosure [14]. The status vocabulary also reduces governance risk by preserving test scope and using `NO_CHANNEL_OBSERVED` instead of secure. Broken evidence is `INCONCLUSIVE`.

LLM Usage Statement

Language-model tools assisted with source discovery, research organization, prior-art critique, code implementation and debugging, experimental-design checks, gap analysis, and language editing. No LLM served as the MUTE evaluator or scorer; deterministic executable code produced the empirical outcomes. The author retains responsibility for independently rerunning and inspecting the artifacts, reviewing every cited source, revising the manuscript in their own words where required, and approving every claim before submission.