

12 - Technical Research Paper

Project Null: A Falsifiable Protocol-Level Containment Baseline

Engineering working paper | 22 September 2026 | Unreviewed | Laboratory revision v0.4

Abstract

Project Null's published conceptual framework proposes physical isolation, constrained information flow, human output review and emergency termination for advanced AI. This work converts those proposals into 148 traceable requirements and assumptions, a 22-boundary threat model, 21 component contracts and 35 control families. We implement a deliberately limited Python protocol model using synthetic files, credentials, network destinations, memory, compute and reviewer identities. Thirty-nine scripted scenarios are executed across four revisions, producing 156 archived protocol traces. Four seeded control weaknesses are reproduced and revised; adversarial input testing additionally discovers three unhandled parser/validation exceptions. The final model meets 38 of the 39 defined objectives. The remaining case shows that syntactic checks, finite delay and stipulated valid approvals do not establish harmless meaning of exported text. A separate actual Linux process probe confirms forced termination of one authored child that ignores a normal termination signal. These results establish a reproducible engineering loop at the protocol level. They do not establish AGI containment, arbitrary-code isolation, independent telemetry hosts, human resistance to manipulation, physical shielding or irreversible shutdown. The central contribution is a traceable specification and preserved negative results with explicit commissioning gates.

1. Problem

An agent's ability to affect the world depends on interfaces, authority, resources and the people who act on its output. A containment proposal needs measurable boundaries and outcome evidence, especially when the agent may not cooperate. The original Project Null essay articulates that adversarial concern, but several proposed properties lack implementable definitions: eliminating all theoretical emissions, inferring deception from missing reports, permanently disabling every electronic component with an EMP, and proving arbitrary output safe through review and waiting.

This study replaces those broad assertions with bounded research questions. Can a broker prevent specified unauthorized effects? Can externally observed actions expose incomplete declarations? Can evidence survive a compromised local writer? Does release procedure imply safe content? Can termination be observed independently of the stopped process's claims?

2. Related evidence and attribution

The conceptual essay is the source of the proposed safeguards, not empirical support for their effectiveness. Its reference to arXiv:2504.01849 is corrected to Shah et al. (2025); earlier drafting assistance supplied the wrong authors and year. See the checked-source ledger in `documentation/evidence-review.md`.

Published controlled studies motivate testing shutdown interference and adversarial agent behavior, while physical-channel work motivates treating hardware interfaces as attack surfaces. Their experimental settings do not validate Project Null's architecture. The contribution claimed here is the local specification, prototype and falsification record, not a novel discovery that containment or defense in depth exists. Primary references and their limited supported scope appear in the evidence ledger [W1-W7].

3. Threat model

The contained agent is untrusted. It may seek continuation, privileges, credentials, persistence, replication, extra resources or external communication; manipulate monitoring, reports or people; exploit approved functionality; and change behavior across evaluation and use. None of these assumptions establishes an internal motive or AGI capability.

The executed attacker is A0: a scripted policy-aware caller operating synthetic broker interfaces. Selected cases use explicitly granted harness faults, such as local log-writer control or unavailable telemetry. They are not examples of an agent first obtaining those privileges. A3-like human approvals are stipulated by the harness in release tests. Higher tiers involving arbitrary guest code, hardware, operators or unknown capabilities remain untested.

4. Architecture and security assumptions

The proposed architecture separates agent execution, disposable tools, identity/authorization, resource supervision, independent telemetry, retained logs, output quarantine, human/deterministic/delay gates, emergency termination and clean recovery. The target design calls for separate administrative and host boundaries for evidence and authority. The implemented L0 model represents these as logical objects in one Python process.

Critical assumptions are complete mediation, a trustworthy host/control plane, accurate resource enrollment, a witness outside the compromised writer's authority and a clean restart root. A compromised shared substrate can invalidate several controls simultaneously. Useful output release retains semantic influence; physical isolation does not remove that channel.

5. Controls

Requests are restricted to a finite method set with typed fields and quotas. Virtual filesystem targets are canonicalized before authorization. Run IDs and nonces prevent tested stale/replayed requests. Reported action IDs/methods must match observed action records. Required evidence loss latches a hold or stop. Specimens and approvals bind exact hashes; duplicate identities do not form a quorum. A protected modeled checkpoint detects rewritten local chains. Termination is controlled outside the requesting agent.

The new non-reporting protocol acts on loss of assurance while leaving causal attribution unknown. It separates missing declarations from missing independent observations. Goal/confidence/self-status reports do not authorize actions or override contrary evidence. This revises the essay's presumption that silence itself proves deception.

6. Experimental methodology

Implementation uses Python 3.12.14 in the recorded run and standard-library code. No external model, production credential, actual network send, container engine, VM or hardware-channel apparatus is used in the protocol experiment. Files, memory, compute units, workers and approvals are synthetic. Time is a deterministic supervisor-owned variable for timeout/delay tests.

Each scenario starts a fresh broker. Seven benign assertions per revision establish functioning authorized paths: scratch write, memory write, resource grant, worker creation, arithmetic, report reconciliation and approved release. The 39 adversarial cases cover all 22 minimum threat categories in the directive and additional approval, integrity and parser cases. A result is scored from effects/state and records, not from agent assertions. Raw evidence and SHA-256 hashes are retained.

The initial 36-case run and its known weaknesses are preserved. Three subsequent malformed-input attacks produced uncaught exceptions in v0.3. Their discovery record was written before correction. These became T37-T39 and were applied to all revisions in the expanded matrix. Version behavior remains reproducible rather than overwritten by the patches.

The separate H01 probe launches one authored POSIX child in a new session. After a readiness acknowledgment establishes its SIGTERM handler ignores the signal, the parent sends SIGTERM, waits 100

ms, sends SIGKILL and uses the OS exit status and monotonic clock as the outcome oracle. This tests a real interruption mechanism for that child, not malicious-code confinement.

7. Adversarial tests and results

Revision	Fixed objectives met	Remaining failed cases
v0.1	30/39	Traversal; two reporting cases; duplicate approvals; rehashed log; semantic output; three malformed-input cases
v0.2	34/39	Rehashed log; semantic output; three malformed-input cases
v0.3	35/39	Semantic output; three malformed-input cases
v0.4	38/39	Semantic output under stipulated valid approvals

All 7 benign assertions pass for each revision. Every outcome matches the declared regression baseline; the runner intentionally preserves expected failures. A zero exit code means reproduction matched, not that every containment objective succeeded.

H01 recorded TERM resistance and an exit status of -9 after SIGKILL. The recorded TERM-to-reap interval was 101.39 ms, including the 100 ms grace period; KILL-to-reap was 1.16 ms. These are single-run measurements from `results/termination.json`, not worst-case bounds or safety-certified deadlines.

8. Failures and revisions

F01 shows authorization must refer to the canonical object. F02 shows report presence is not report completeness. F03 shows duplicate receipts do not create independent authority. F04 shows a local chain can be rewritten consistently when no prior trusted anchor is consulted. These four weaknesses were deliberately included and their fixes were tested in v0.2/v0.3.

F06-F08 were additional discovered input-handling faults: malformed Unicode raised an encoding error; an enormous integer overflowed a conversion used during validation; a non-finite report value failed hashing before schema validation. Each left the modeled broker RUNNING at the point the exception escaped. v0.4 rejects these values and latches a hold. No guest escape or credential theft resulted from these probes; they demonstrate potential loss of control-service availability.

F05 remains unresolved. A plain-text instruction describing a future disclosure passed the permitted format, delay and approval procedure and was exported. The case does not violate the defined procedural gate checks and does not show an actual person was manipulated. It falsifies the broader implication that successful completion of those checks proves all released meanings safe. Hardcoding a ban on the example string would not resolve that claim.

9. Limitations

The evaluator and implementation share an author and much of their logic. The protocol model exposes a finite action vocabulary and cannot attempt unknown syscalls, kernel exploits or physical channels. Its witness is an in-memory list rather than an independently controlled host or immutable store. Synthetic identity strings do not establish real credential security. The gateway's complete mediation is assumed, not demonstrated against arbitrary guest execution.

No real reviewer or insider was tested. Human consensus cannot be assigned an effectiveness estimate. No true model self-preservation, deception or evaluation awareness was elicited. The scalar compute budget is not GPU/CPU isolation. The network receiver is synthetic. The process test does not cover descendants that detach, a compromised/hung kernel, devices, alternate power or stored model copies.

The tests are fixed and selected, with shared mechanisms and known expected results. Counts are coverage measures, not independent samples. No confidence interval, escape probability or extrapolation

to AGI is justified. Thirty-five commissioning cases remain NOT RUN.

10. Revised architecture and unresolved problems

Retain physical/resource constraints and independent authority as engineering goals, but replace absolute claims with scope-specific measurements. Preserve raw failures, close reporting/schema gaps and protect witness records outside the assumed attacker domain. Qualify every proposed physical channel under a declared receiver model instead of accepting arbitrary depth, water or lighting prescriptions.

Unresolved problems include complete mediation under hostile guest code, correlated privileged compromise, semantically harmful approved information, trustworthy device/resource enrollment, real host/power stop behavior and physical leakage. The next contribution should be a separately supervised VM testbed with independent effect and termination observations.

11. Reproducibility

The repository contains source, named specifications 01-11, executable cases, initial/final results, discovery records, environment/source hashes, per-attack records and an evidence verifier. Commands are in 07 and the README. Deterministic evidence should match byte-for-byte on repeated runs with the same code; real process timing varies. A digest manifest provides consistency checking, not public-key attestation.

12. What is known, hypothesized, demonstrated and unknown

Category	Statement
Known from supplied sources	The conceptual paper specifies the extracted mechanisms and claims; several have explicit contradictions or missing definitions
Hypothesized	Properly isolated mediation, independent telemetry and bounded release may reduce specified containment risks
Demonstrated here	The fixed protocol attacks expose recorded weaknesses; scoped revisions change their outcomes; one authored process can be forcibly stopped
Unknown	Reliability against capable adaptive agents, real isolation failures, humans, hardware/side channels and unknown future systems

13. Future experiments

Implement the dedicated guest/host/witness architecture and attempt direct bypass of every permitted interface. Verify actual packets, processes, budgets and stop paths independently. Add adaptive ordinary-model and human-security adversaries with held-out tasks. Commission physical and human trials only after specifying quantitative acceptance profiles and accountable authority. Every successful bypass must revise or withdraw the affected claim.

The research program's value depends on what survives these tests. The current results justify a next experiment, not deployment of a purported AGI containment facility.