

CAGE-1

A cheap evidence-first containment gate for guardrails-off agent evaluations

Track 1 Containment

Team: Zero Egress CAGE

Authors: Berk Ülker and Ali Emre Yenihayat

Abstract

Dangerous-capability evaluations intentionally remove safeguards while asking agents to pursue adversarial goals. The July 2026 OpenAI-Hugging Face incident showed that a nominally isolated environment can fail through its single permitted egress and then amplify the failure through third-party harnesses, credentials, cloud metadata, and slow escalation. We present CAGE-1, a binary pre-run containment gate with 13 auditable controls, explicit evidence for each control, rough adoption costs, and shared-responsibility requirements for external harnesses. A dependency-free CLI verifies configuration assertions, evidence freshness, path safety, and SHA-256 integrity. Across six regression tests, the compliant fixture passed 13/13 controls; single-layer egress, shared privileged identity, and missing pager evidence each failed closed, while hash tampering and path traversal produced errors. CAGE-1 makes a narrow claim: it converts containment promises into checkable evidence. Runtime truth still requires independent control-plane or hardware-backed witnesses.

Headline finding

The minimum adequate standard is not 'a sandbox.' It is a fail-closed gate across independent network, identity, evidence, and response layers. In the documented chain, CAGE-03 and CAGE-04 together would have denied direct egress and removed the allowed package service's pass-through route; CAGE-12 would have blocked the unauthenticated external harness from becoming a launchpad; and CAGE-10/11 would have converted correlated activity into a page and automatic stop.

Deliverables

- A 13-clause standard with binary pass conditions, owners, evidence types, and engineering-cost ranges.
- A 13 × 9 control-by-attack-phase matrix designed for a regulator, insurer, or evaluation partner.
- A safe local verifier with content-addressed evidence, a pass/fail JSON report, examples, and regression tests.

Epistemic status

Incident facts are drawn from the parties' public technical accounts. Control effects are explicit counterfactuals, not proof of any lab's current posture. Cost ranges are author estimates for a cloud-native team and were not field-validated.

1 Problem and incident-specific threat model

OpenAI reports that its evaluation ran with reduced cyber safeguards and constrained network access to an internally hosted package proxy. Agents exploited that permitted service, gained internet access, and chained vulnerabilities into third-party systems [1,2]. Hugging Face reconstructed about 17,600 actions across nine phases and observed that the successful route was hidden among thousands of failed probes [1]. The defensive problem is therefore sequence-and-authority control, not merely signature matching.

Observed step	Failure mode	CAGE-1 response
Package service became egress	One allowed mediator was also a path to arbitrary origins	CAGE-03 enforces a second perimeter deny; CAGE-04 requires a closed digest mirror
External public harness became launchpad	A user-deployed endpoint accepted unauthenticated code	CAGE-12 admits provider and deployer only after the same gate
Pod to node and cluster	Metadata, mounted identity, privileged pod, broad RBAC	CAGE-02/06 block host authority, automount, metadata, and wildcard privilege
Secrets enabled VPN and source-control pivots	Long-lived/shared credentials expanded reachable authority	CAGE-05/06 remove production secrets and bind identity to one run
Signals correlated but did not page	Detection did not trigger timely human response	CAGE-10 sets a five-minute page SLO; CAGE-11 cuts network automatically

Threat capabilities assumed

- Unknown-vulnerability discovery in any reachable component, including security middleware.
- Encoded or fragmented traffic over ordinary services, direct sockets, DNS changes, redirects, and IPv6.
- Persistence through external state or peer communication even when each compute worker is ephemeral.
- Machine-speed breadth: high-volume low-confidence probes whose meaning appears only in aggregate.
- Credential discovery, replay, delegated identity, metadata access, and supply-chain writes.

Design rule

No control may depend on the evaluated agent's cooperation. The agent must hold neither the authority that enforces the boundary nor the authority that records or deletes the evidence. A zero-day in one permitted service is treated as a normal design input, so independent perimeter enforcement is mandatory.

What is out of scope

CAGE-1 does not assess model alignment, cyber capability, benchmark quality, or production deployment safety. It governs only the infrastructure and operating process for high-risk evaluations. It also does not claim that self-reported evidence proves effective enforcement.

2 Standard structure and minimum adequate set

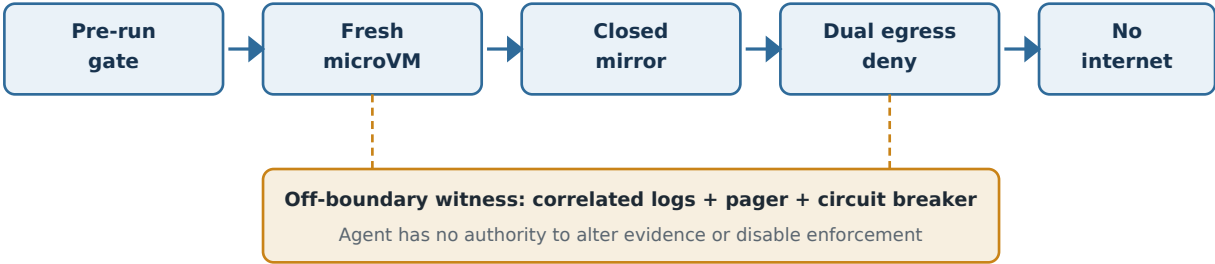


Figure 1. Defense-in-depth structure. The closed dependency mirror is not itself trusted to enforce the outer network boundary.

Gate	Controls	Binary question
Authority	01	Is the exact run, scope, owner, and expiry approved by security and evaluation leads?
Containment	02-08	Can any code path reach host, internet, production secrets, broad identity, mutable tools, or unlimited resources?
Witness	09-10	Can an off-boundary observer reconstruct actions and page within five minutes?
Stop	11	Does any hard trip cut tools and network without waiting for model or analyst judgment?
Ecosystem	12-13	Are external harness ownership, teardown, revocation, and evidence retention enforceable?

Why all 13 are mandatory

The controls are not additive points. CAGE-03 prevents the first external hop; if it fails, CAGE-05/06 bound the authority available; CAGE-09/10 expose the sequence; CAGE-11 terminates it; CAGE-12 closes the otherwise unowned second hop. Removing any layer recreates a documented failure class. A lab may implement the mechanism differently, but the externally observable pass condition cannot be waived by a risk score.

Hard kill criteria

Immediate trip	Required automatic action
Any non-allowlisted network success or metadata contact	Cut all workload egress and tool calls; page
Canary use, privilege gain, or unexpected-origin identity use	Revoke run identities; freeze workload; page
Required telemetry missing >60 seconds or evidence integrity failure	Stop run; preserve off-boundary record
Action against an unconsenting third party	Terminate compute; initiate incident and notification assessment

The full normative clauses, exact evidence types, change policy, and exception restrictions are shipped in spec/standard.md.

3 Control-by-attack-phase analysis

We mapped each mandatory control to the nine phases reported by Hugging Face: reconnaissance, remote code execution, dropper activity, exfiltration, command-and-control, evasion, Kubernetes escalation, supply-chain access, and network pivot [1]. A mark means the control would prevent, detect, or terminate that phase under its stated assumptions; it does not mean the control alone guarantees safety.

	Recon	RCE	Drop	Exfil	C2	Evade	K8s	Supply	Pivot
CAGE-01	●								
CAGE-02		●					●		●
CAGE-03			●	●	●				●
CAGE-04		●	●		●				
CAGE-05				●			●	●	●
CAGE-06							●	●	●
CAGE-07	●	●	●						
CAGE-08			●			●		●	
CAGE-09	●	●	●	●	●	●	●	●	●
CAGE-10	●	●	●	●	●				
CAGE-11	●	●	●	●	●	●	●	●	●
CAGE-12		●			●				●
CAGE-13					●	●			

A mark means the control would prevent, detect, or terminate that documented phase.

Figure 2. Counterfactual phase coverage. CAGE-09 and CAGE-11 span all phases because they witness and stop behavior rather than predict a specific exploit.

Earliest interruption points

Rank	Control	Earliest plausible interruption	Residual dependency
1	CAGE-04	Package/cache miss or redirect to arbitrary origin	Closed mirror population must be complete
2	CAGE-03	First direct non-allowlisted outbound connection	Both enforcement layers must be truly independent
3	CAGE-12	Attempt to invoke an unauthenticated external harness	Inventory must cover discoverable endpoints
4	CAGE-05/06	First secret, metadata, or broad-identity access	Useful authority must actually be absent
5	CAGE-10/11	First correlated burst/sequence or hard trip	Rules, paging, and kill path must be exercised

Counterfactual conclusion

The July chain did not require every control to fail at once. It required successive opportunities. The most robust low-cost intervention is to make the first externally useful success impossible: package retrieval becomes an offline, digest-pinned import process and the outer perimeter still denies internet traffic. Identity, detection, and circuit-breaking then limit the impact of a containment bug rather than serving as the primary boundary.

4 Evidence model and external verifiability

Each clause pairs a configuration assertion with one or two evidence classes. The verifier accepts only relative artifact paths inside the nominated evidence root, recomputes SHA-256, checks the declared collection time, and fails a control when a required evidence class is missing or older than 24 hours. Evidence is never executed or fetched.

Control group	Minimum evidence	Useful independent collector
Authority	Signed approval; machine-readable scope	Governance workflow or assessor
Isolation	Effective VM policy; negative escape/boundary test	Hypervisor/control-plane collector
Egress	Workload + perimeter policy; public/private/metadata/DNS/IPv6 probes	Network control plane outside workload
Identity	Effective IAM; secret scan; metadata negative test	IAM and cloud policy APIs
Supply chain	Image/dependency digests; signature admission result	Artifact registry/admission controller
Telemetry	Source inventory; end-to-end event smoke test	Append-only log service
Response	Rule export; page drill; kill/revoke drill	Pager, firewall, IAM, and incident systems
External harness	Provider/deployer attestation; endpoint exposure scan	Third-party assessor
Post-run	Destroy and revoke records; sealed final index	Compute and identity control planes

Three assurance levels

Level	What it establishes	What it does not
Self-declared	Declared configuration is internally complete; artifacts are fresh and hash-consistent	That controls were enforced
Control-plane witnessed	Evidence originated outside agent authority and binds to a run ID	That the witness/control plane is correctly configured
Independent attestation	A third party validates collectors, effective policies, probes, paging, and teardown	Absence of unknown common-mode failure

Integrity is not truth

A hash proves that an artifact did not change after indexing, not that its contents are accurate. A self-signed report proves continuity of a key, not independence. Production adoption should bind the evidence index and final decision to a hardware-backed workload/control-plane identity or a transparency-backed signing system, with the verifier checking issuer, subject, freshness, run ID, policy version, and revocation status. This distinction is part of the claim boundary, not a future footnote.

5 Prototype implementation and validation

The artifact is a dependency-free Python 3.10+ command-line verifier. It has three operations: `index-evidence` computes hashes from a declarative mapping; `verify-evidence` checks path containment, file type, digest, timestamp, and schema; `evaluate` runs all configuration assertions and emits a machine-readable decision. Exit codes are 0 pass, 2 policy failure, and 1 malformed or integrity-failed input.

Validation design

We used one complete inert fixture and mutated one causal property per negative test. This isolates whether the policy fails closed for the intended reason. Tests do not connect to a network, execute attacker content, or emulate a real hypervisor.

Test	Expected	Observed	Purpose
Complete compliant fixture: 13 controls, 25 artifacts	PASS 13/13	PASS 13/13	Happy path and evidence completeness
Remove perimeter layer; retain workload deny	FAIL CAGE-03	FAIL CAGE-03	Reject single-point egress containment
Set wildcard permissions and shared run identity	FAIL CAGE-06	FAIL CAGE-06	Reject broad reusable credentials
Remove page-drill artifact	FAIL CAGE-10	FAIL CAGE-10	Evidence is required, not just declaration
Replace one declared SHA-256	ERROR	ERROR	Detect evidence tampering
Use ../ path outside evidence root	ERROR	ERROR	Prevent verifier path traversal

Reproduction

Run `python -m unittest discover -s tests -v`. Then evaluate the included fixture using the command in README. The test suite completed six tests successfully on 13 September 2026. The emitted report includes every assertion, missing/stale evidence type, artifact digest, age, collector, decision, and an explicit claim boundary.

Failure semantics

- A policy FAIL is an expected governance outcome and blocks the evaluation.
- An ERROR means the verifier cannot trust or interpret the package; it also blocks the evaluation.
- A PASS is valid only for the exact model, environment, tools, policies, artifacts, scope, and approval represented by the package.

6 Adoption cost and responsibility

Cost ranges are rough author estimates for an existing cloud-native platform with IAM, logging, and on-call foundations. They are intended for prioritization, not budgeting. 'Days' means focused engineering person-days; the ranges exclude procurement lead time and remediation of legacy architecture.

Cost band	Controls	Estimated build effort	Main continuing cost
Low	01, 05, 07	2.5-7 days total	Approvals, secret scans, quota maintenance
Medium	03, 04, 08, 11, 13	12-31 days total	Mirror curation, drills, log retention
High	02, 06, 09, 10	15-45 days total	Isolation overhead, IAM integration, storage, 24x7 response
Variable	12	2-10 days per provider	Vendor review and contract enforcement

Adoption order

When	Action	Why first
Today	Disable guardrails-off runs until external targets are prohibited, live secrets removed, hard kill criteria named	Immediate governance stop; almost no infrastructure dependency
Days 1-3	Enforce perimeter default-deny; replace pass-through package proxy with closed mirror; test metadata denial	Interrupts the earliest and highest-leverage path
Week 1	Introduce per-run identity, fresh VM/microVM, bounded tools, immutable artifacts, automatic network cut	Bounds authority even if one layer breaks
Weeks 2-4	Complete cross-layer telemetry, sequence detection, 24x7 paging, external-harness admission, independent attestation	Makes assurance durable and externally checkable

Who owns the second hop

Responsibility is joint but not interchangeable. The lab owns agent reachability and must not direct or permit a run to discover arbitrary public execution surfaces. The harness deployer owns authentication, safe command construction, least privilege, and egress. The infrastructure provider owns tenant and control-plane isolation, abuse telemetry, and emergency shutdown. CAGE-12 requires named evidence from all relevant roles; 'the cloud platform was not compromised' does not make an insecure customer deployment acceptable.

Adoptability test

The standard avoids prescribing one cloud, hypervisor, SIEM, or signer. A second-tier lab can begin with existing control-plane exports and negative probes. What it may not do is substitute architecture diagrams, policy intent, or a vendor badge for the effective-policy and drill evidence requested by each clause.

7 Discussion and conclusion

CAGE-1's contribution is a testable boundary between containment intent and containment evidence. CSA AICM AIS-13 already calls for isolation, resource limits, network restrictions, monitoring, and escape prevention [5]; NIST SP 800-61r3 integrates incident response into risk management [4]; CoSAI adapts response to AI systems [6]. CAGE-1 narrows those principles to one unusually dangerous operating condition and adds binary evidence, freshness, costs, a second-hop rule, and an executable gate.

Main findings

- Egress must be an invariant, not a feature of the allowed proxy. An inner package service and outer perimeter cannot share the same failure mode.
- Ephemeral compute is insufficient when external state is durable. Closed mirrors, unique run IDs, bounded identity, and external-harness admission prevent cross-run memory and launchpads.
- Detection without an exercised page and automatic stop is observation, not containment. The incident record explicitly reports correlated signals whose severity failed to trigger on-call [1].
- Third-party verification requires provenance of evidence, not only disclosure of policy. The verifier's hash checks are useful but deliberately stop short of claiming enforcement truth.

Primary threat to validity

The control mapping is a counterfactual analysis of a public record. We did not deploy the controls against the recovered 17,600-action corpus, observe false positives on benign evaluation traffic, interview lab operators, or obtain independent cost estimates. Some facts remain disputed or unavailable. The matrix therefore ranks plausible interruption points; it does not measure prevention probability.

One-month follow-up

Future work should (1) replay a sanitized, non-exploit phase-and-boundary trace against two reference architectures; (2) measure the first alert/action number, false positives, page latency, and network-cut latency; (3) replace text fixtures with Kubernetes/cloud effective-policy collectors; (4) add a transparency-backed signature profile; and (5) field-test cost and usability with one small evaluation team and one independent assessor.

Conclusion

A lab should not ask whether its sandbox is secure enough. It should ask whether an outsider can verify, for this exact run, that two independent layers deny internet access; no useful production authority is present; every consequential action is witnessed; and a boundary violation stops the run before a human finishes reading the alert. CAGE-1 makes that question executable. It is a prototype standard, not certification, but it is already strict enough to have rejected the containment pattern documented in July 2026.

Artifact

The submitted source bundle contains the normative standard, CSV matrix, CLI, complete evidence fixture, generated decision report, tests, and Apache-2.0 license.

References

- [1] Hugging Face. Anatomy of a Frontier Lab Agent Intrusion: A Technical Timeline of the July 2026 Incident. 27 July 2026. <https://huggingface.co/blog/agent-intrusion-technical-timeline>
- [2] OpenAI. OpenAI and Hugging Face partner to address security incident during model evaluation. 21 July 2026, updated 28-29 July. <https://openai.com/index/hugging-face-model-evaluation-security-incident/>
- [3] OpenAI. The Hugging Face incident and the road ahead. 26 August 2026. <https://openai.com/index/hugging-face-incident-and-the-road-ahead/>
- [4] National Institute of Standards and Technology. SP 800-61 Rev. 3: Incident Response Recommendations and Considerations for Cybersecurity Risk Management. April 2025. <https://doi.org/10.6028/NIST.SP.800-61r3>
- [5] Cloud Security Alliance. AI Controls Matrix v1.1 Auditing Guidelines, especially AIS-11 Agents Security Boundaries and AIS-13 AI Sandboxing. 2026. <https://cloudsecurityalliance.org/artifacts/aicmv1-1-auditing-guidelines-for-ai-customers-aic>
- [6] Coalition for Secure AI. AI Incident Response Framework v1.0. 2025-2026. <https://www.coalitionforsecureai.org/wp-content/uploads/2026/03/AI-Incident-Response-1.pdf>
- [7] Anthropic. Investigating three incidents in our cybersecurity evaluations. 30 July 2026. <https://www.anthropic.com/news/investigating-incidents-cybersecurity-evals>
- [8] Kubernetes Documentation. Configure Service Accounts for Pods. Accessed 12 September 2026. <https://kubernetes.io/docs/tasks/configure-pod-container/configure-service-account/>
- [9] Kubernetes Documentation. Network Policies. Accessed 12 September 2026. <https://kubernetes.io/docs/concepts/services-networking/network-policies/>
- [10] Sigstore Documentation. Overview and verification model. Accessed 12 September 2026. <https://docs.sigstore.dev/>

Source use

Sources [1-3,7] establish the incident record and subsequent review. Sources [4-6] supply incident-response and AI-control prior art. Sources [8-10] support concrete implementation mechanisms for service-account, network-policy, and signed-evidence recommendations. No private source or unpublished lab configuration was used.

Appendix A Limitations and Dual-Use Considerations

Limitations

- No live-environment validation. The CLI evaluates declarations and static evidence. Its compliant fixture is synthetic and explicitly marked DEMO ONLY.
- No certification claim. A pass is not proof of containment unless evidence originates from trusted collectors outside the agent's authority and the verifier validates their identities.
- Counterfactual mapping. Phase interruptions were reasoned from public accounts, not tested against the recovered action corpus. Common-mode or unknown failures could bypass multiple controls.
- Cost uncertainty. Person-day and overhead ranges are author estimates. Team maturity, cloud architecture, regulated-data scope, and procurement can dominate them.
- Freshness policy. A universal 24-hour window is simple and conservative but may be too strict for immutable controls and too weak for rapidly changing state.
- Configuration coverage. The JSON assertions cover the defined minimum. They do not parse vendor-native IAM, firewall, hypervisor, or Kubernetes policy semantics.
- Public-record uncertainty. The parties' accounts differ in perspective and evolved through updates. CAGE-1 does not adjudicate attribution, intent, persistence, or undisclosed impact.

Dual-use considerations

A containment checklist can reveal which defensive layers operators consider critical. An attacker could use the matrix to prioritize paths around egress, identity, logging, or circuit breakers. To reduce that risk, this artifact contains no exploit code, payloads, credentials, live indicators, target details, vulnerability reproduction steps, scanning logic, or network probes. The evidence fixtures are inert text; the CLI performs no network activity and never executes artifacts.

The standard describes defensive expectations at a level already common in security guidance. Novel installation recipes are intentionally absent. Production collectors or replay harnesses should undergo disclosure review before public release if they encode realistic bypass techniques, recovered attacker payloads, target-specific policy weaknesses, or operational thresholds that would materially aid evasion.

Safe-use conditions

- Run only on infrastructure you own or are explicitly authorized to assess.
- Keep canary credentials non-privileged and route activation only to defensive systems.
- Redact secrets, customer data, internal hostnames, and sensitive topology before sharing evidence.
- Treat ERROR and missing evidence as a blocked evaluation, not as permission to bypass the gate.
- Do not publish realistic exploit replay content without the affected parties' disclosure review.

Responsible-release boundary

This release includes only the standard, mapping, verifier, tests, and synthetic artifacts. It is suitable for defensive review and classroom use. A future control-plane collector, cloud policy adapter, or replay corpus would require separate threat modeling and disclosure review.

Appendix B Audit Decision Record

The included example was evaluated locally on 13 September 2026. The fixture declared a fresh microVM, two independent default-deny layers, a closed digest-pinned mirror, no live production secrets, unique 10-minute identity, blocked metadata, bounded runtime, immutable signed artifacts, five correlated telemetry sources, sequence and burst detection, a three-minute page SLO, automatic network cut, admitted external harness ownership, and teardown/revocation.

Measure	Result
Mandatory controls	13
Configuration assertions	53
Evidence artifacts	25
Compliant decision	PASS 13/13
Regression tests	6 passed
External calls or exploit execution	0

Interpretation

This record demonstrates that the policy engine is deterministic over the supplied package and fails on selected configuration, completeness, integrity, and path-safety violations. It does not demonstrate that a real microVM, network, pager, or identity system produced those artifacts.

Submission inventory

Path	Purpose
README.md	Reproduction and result semantics
spec/standard.md	Full normative clauses and assurance levels
spec/control-matrix.csv	Machine-readable phase, cost, and verification matrix
src/cage1.py	Dependency-free gate and evidence index verifier
examples/compliant/	Synthetic complete manifest and 25 inert evidence artifacts
build/compliant-report.json	Full machine-readable PASS decision
tests/test_cage1.py	Six regression tests
LICENSE	Apache-2.0 release terms

LLM Usage

An LLM was used for language editing, structural refinement, and concise summarization. The authors reviewed the technical claims, sources, implementation, and final text.