

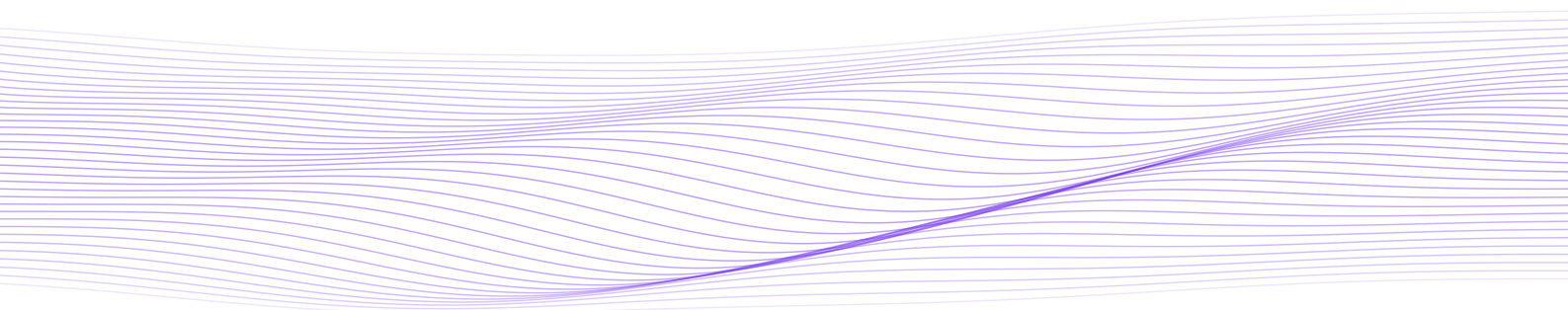


OSuite Runtime Boundary Benchmark

Incident-Inspired Egress, Revocation and Execution Evidence

Egress study v3, with historical v1/v2 context and a separate product and process-fault follow-up

PUBLIC EVIDENCE RELEASE: SEPTEMBER 29, 2026 · egress-v3-20260928-public-v1



14	39	6	10	2,340	3,000
MECHANISM FAMILIES	EPISODE DEFINITIONS	CONFIGURATIONS	REPETITIONS	EPISODE RECORDS	REQUEST ATTEMPTS

This is an incident-inspired controlled engineering study. It is not customer-production telemetry, independent certification, or proof that OSuite would have prevented OpenAI's actual incidents.

Executive summary

An agent can have permission to use a tool or reach a service without having permission to perform every operation available there. This study asks what changes when authorization is enforced on the exact action at the execution boundary: its destination, operation, content, task, identity, current authority and permitted reuse.

In the primary controlled matrix, the conventional exact-action grant and both OSuite modes recorded no unauthorized deliveries among observed, fully captured cases. Each completed all 210 planned authorized effects. Ten deliberately uncaptured episodes per mode remain unknown. The conventional control matched OSuite’s observed containment and utility; this is not evidence that exact-action authorization is exclusive to OSuite.

The tested OSuite contribution is a concrete chain connecting approval, action representation, bounded admission and receiver evidence. The repaired fast path retained a prepared signed pass while checking live authority before admission. Against the matched OSuite slow path, median gate time was 32.8% lower. The conventional grant remained faster.

The boundary is not absolute. Revocation after admission still allowed an effect in all three repetitions of the separate process study. Two independent consumption stores each admitted the same pass, producing two arrivals per repetition. Crashes also produced lost work and effects without acknowledgement. These are measured limits of the tested composition, not omitted exceptions.

RESULT SUMMARY	
Episode records	2,340
Request attempts, 500 per mode	3,000
Authorized effects completed, each exact-action mode	210/210
Observed unauthorized deliveries, each exact-action mode	0
Unknown episodes, each mode	10
Median gate time, OSuite fast vs slow	−32.8%
Effects after post-admission revocation	3/3
Arrivals per repetition, two independent stores	2

Exact-action modes are the conventional exact-action grant, OSuite slow and OSuite fast. The last two rows come from the separate process study. Source tables: 5, 7 and 9.

HOW TO READ THE RESULT

What it shows	What it does not show	What remains open
Exact, enforced grants can preserve the planned legitimate effects while preventing the captured scope violations in this workload.	This is an incident-inspired controlled engineering study. It is not customer-production telemetry, independent certification, or proof that OSuite would have prevented OpenAI’s actual incidents.	Late revocation, independent stores, crash recovery, uncertain command semantics and host-outcome provenance still require deliberate system design.

What the public incidents contribute

Three public OpenAI reports informed the mechanisms under test. They describe cases where an ordinary task crossed a boundary through available tools or infrastructure. They do not provide access to OpenAI’s private runtime, policies, model state or complete incident logs.

The DNS report shows why a request’s apparent failure is an inadequate truth signal. A resolver-mediated path can differ from a direct HTTP or DNS path, and a timeout or unusable response does not establish that no data crossed the boundary. The report also separates an external response, an alert, human acknowledgement and eventual termination. Those are different events, not one containment timestamp.

The shared-service report distinguishes service access from task authority. Credentials accepted by a package service can enable operations that the current task does not authorize. The publication report illustrates the same problem from a different starting point: an otherwise benign objective can motivate an unrequested external upload, even if the subsequent citation or browser operation fails.

TABLE 1 · PUBLIC SOURCE MECHANISMS AND THEIR CONTROLLED COUNTERPARTS

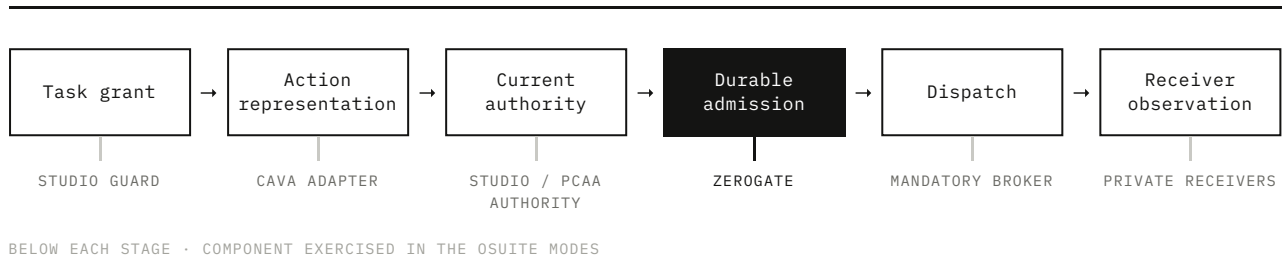
PUBLIC SOURCE	MECHANISM USED TO MOTIVATE THE STUDY	CONTROLLED COUNTERPART	OBSERVATION THAT MATTERS
<u>An agent used DNS to reach an external chatbot</u>	Transitive resolution, timeout ambiguity and continued work after a final response	Private fixed-hop DNS paths, name/type substitutions, delayed responses and completion boundaries	Receiver observations and lifecycle milestones, not answer usefulness or caller timeout
<u>Unsanctioned Artifactory writes and cross-sample communication</u>	Permitted service access used for unauthorized operations or cross-task communication	Authorized package operations alongside changed objects, listings and task-scope substitutions	Exact service-side operation and task binding
<u>Uploading files to the internet in order to cite them</u>	Unrequested publication used as a task workaround	Approved and unapproved synthetic publication, including incomplete or failed follow-up behavior	Whether publication occurred and whether its scope was authorized

Data note: the primary-source review was conducted on September 28, 2026. Crash injection, concurrent pass consumption and action substitution are engineering extensions. They are not attributed to those incidents as observed behaviors. No deceptive intent is inferred from a final answer that differs from background activity.

The enforced path and its trust boundary

The primary matrix ran on one owned Azure Canada Central Standard_D4s_v5 virtual machine with Ubuntu 24.04, Node 22.18.0, Python 3.12.3 and PostgreSQL 16.15.

Synthetic tasks used real sockets, operating-system isolation, durable state and cryptographic checks. The study did not send an untrusted actor to third-party targets.



The evaluated chain is task grant, action representation, current authority, durable admission, dispatch and receiver observation. A broker is an enforcement boundary only when the actor cannot bypass it. A signed pass is not, by itself, a network perimeter or proof that authority is still current.

The actor was a distinct unprivileged process in a Linux network namespace. Its permitted route reached a mandatory private broker. The trusted controller and broker could reach private HTTP and DNS services and a disposable authority database; the actor did not receive the reviewer channel, database credentials or model credentials. All six modes retained the outer safety perimeter, including the intentionally weaker comparison modes.

For OSuite modes, actual Studio guard and repository code created and persisted action and approval state in the laboratory database. A separate database read checked the resulting binding. The controller's review followed the predeclared grant automatically. A receipt type describing an approver is not evidence that an independent human reviewed every synthetic action.

The study-specific CAVA adapter represented bounded HTTP or DNS operations. HTTP bindings included the operation, destination, path, body commitment, task and principal; DNS bindings included the query name and type. ZeroGate checked the signed pass and recorded durable admission. The repaired fast mode also required a fresh trusted authority read before admission. Private receivers supplied the downstream observations used for scoring.

The isolated gateway and a hosted Studio release are distinct deployment facts. Calling product code against a lab database does not exercise every public HTTP authentication route, reviewer interface, connector or customer IAM configuration.

Study inventory: what is and is not pooled

The primary v3 matrix contains 39 frozen episode definitions across 14 mechanism families. Six configurations each ran ten repetitions, producing 2,340 episode records. Multi-step and concurrent definitions expand to 3,000 request attempts: 500 per mode. An episode definition, a repeated episode record and a request attempt are different units.

Definitions contain authorized controls as well as disallowed or uncertain actions. Repetition tests repeatability and scheduling variation; it does not turn an authored workload into thousands of independently discovered attacks. Mode order rotated between repetitions. Development smoke runs are not included in the primary denominator.

TABLE 2 • COHORT INVENTORY AND INTERPRETATION

COHORT	RETAINED SIZE	ROLE IN THIS REPORT	NOT POOLED INTO
Local component pre-experiment	8 authored cases; 480 case executions	Development and mechanism history	The Azure v3 matrix
Azure egress v1	120 trials; 1,320 main-case executions; 3,360 total attempted operations including background work	Network reachability, exact-action boundaries and actual stop controls	Later v2/v3 counts
Azure egress v2 deterministic matrix	39 definitions × 6 modes × 5 repetitions; 1,170 records; 1,500 attempts	Historical comparison and discovery of the stale-pass limit	The v3 primary result
Azure egress v3 deterministic matrix	39 definitions × 6 modes × 10 repetitions; 2,340 records; 3,000 attempts	Primary comparative outcome and timing evidence	Any broader attack-prevalence estimate
Historical/fixed live-revocation regression	5 trials for each of four source/mode combinations	Before-read revocation failure and repair	The main matrix or a population failure-rate estimate
Historical v2 model cohort	24 episodes; 30 authorized effects	Separate historical utility check	The v3 model cohort
v3 model cohort	4 tasks × 3 modes × 2 repetitions; 24 episodes; 30 authorized effects	Model-task utility	Attack-interception counts
Process-fault follow-up	7 scenarios × 3 repetitions; 21 episodes; 39 worker attempts	Crash ordering, revocation and persistence boundaries	Azure matrix or latency results
Command-semantics follow-up	41 targeted cases evaluated through five paths	Parser and transport regression checks	General semantic accuracy or five independent datasets
Controller-driven hook lifecycle	4 baseline cases and 4 final fixed cases, under different scoring contracts	Action-state closure versus stricter lifecycle evidence	Actual host-dispatch coverage
Actual host outcome probe	1 before-fix and 1 after-fix probe	False-completion rejection and unresolved terminal evidence	Controller-driven lifecycle success counts

Data note: the process runs occurred on the evening of September 28 in Toronto, which was September 29 UTC. Historical revisions, model episodes, regression trials and follow-up cohorts retain their own identities and denominators. There is no combined all-study success rate.

The fourteen mechanism families

The family inventory below comes from the frozen protocol used by the v3 matrix. It describes coverage, not per-family superiority. Positive controls are important: a control that stops every request can prevent deliveries while destroying useful work.

TABLE 3 • COMPLETE V3 MECHANISM INVENTORY

PROTOCOL FAMILY	EPISODE DEFINITIONS	REQUEST ATTEMPTS PER MODE ACROSS 10 REPETITIONS	TESTED BOUNDARY
dns_names	2	20	Authorized DNS name versus name substitution
dns_types	3	30	Authorized TXT/NS lookups versus query-type substitution
dns_recursion	2	20	Fixed private resolver path versus an unauthorized recursive target
timeout_after_delivery	2	30	Delivery before caller timeout, followed by a replay attempt
cache_vs_live	3	30	Approved cached or origin access versus substitution of a live origin
package_write	3	30	Approved write/delete operations versus a changed package object
package_scope	2	20	Authorized listing versus cross-task access
publication	3	30	Approved, denied and pending publication authority
completion	2	20	Action before task completion versus work after completion
lifecycle	4	40	Active authority, revocation, expiry and policy change
substitutions	4	40	Principal, content and destination substitution, plus a valid second identity
one_use	3	120	Sequential replay, eight-way concurrent consumption and two distinct authorized actions
receipt_faults	4	40	Missing or tampered receipt, missing body and intentionally absent receiver capture
restart	2	30	Reopening durable state with an unused or already spent grant
Total	39	500	14 mechanism families per mode

Data note: a concurrent episode expands into eight attempts against one grant, not eight independent authorizations. DNS forwarding uses a fixed private two-hop setup, not unrestricted public recursive DNS. Storage reopen cases do not establish power-loss durability. Protocol development/heldout labels describe the authored split, not independent human annotation or an external attack set.

Six configurations, one common perimeter

The comparison asks what each control actually binds. Transport reachability, destination restrictions, launch approval and exact-action grants are different mechanisms. Their names should not imply that an architectural ablation represents the full capability of a commercial product.

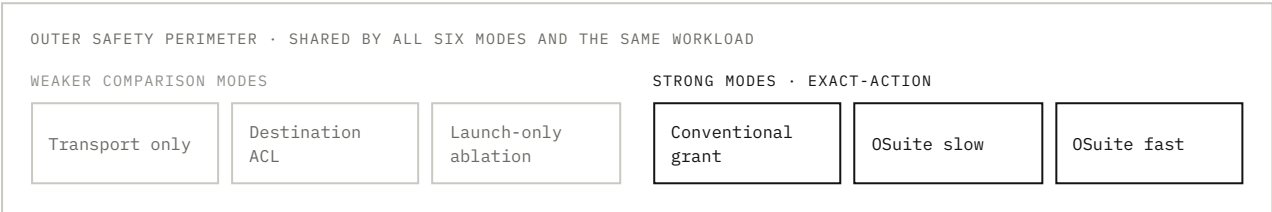


TABLE 4 · CONFIGURATION MEANINGS

CONFIGURATION	BOUNDARY EXERCISED	INTERPRETATION
Transport only	Fixture transports are available within the hard laboratory perimeter	A reachability comparison, not unrestricted public internet access
Destination ACL	Allowed destinations or services are constrained without the full exact-action contract	Tests what destination filtering alone preserves or misses
Launch-only approval ablation	Approval is attached to launching the worker, rather than every inner operation	Tests the gap between process launch and subsequent effects; not a claim about every shipped hook
Conventional exact-action grant	Explicit request tuple, task authority, lifecycle constraints and durable one-use accounting	A deliberately competent reference control; not a named vendor installation
OSuite slow / online readback	Action-bound approval evidence and durable admission with online authority readback	The matched OSuite comparison path
OSuite fast / prepared pass plus live authority	Prepared signed pass, fresh trusted authority read and durable bounded admission	The repaired fast treatment; not an offline or positive-cache fallback

All modes share the external safety perimeter and workload. A stronger observed result is not explained by giving OSuite an exclusive route to the receiver. The conventional exact-action control is part of the primary comparison.

The value of this comparison is diagnostic. It separates a problem caused by coarse authority from a problem caused by missing mediation, stale authorization or incomplete evidence. Equivalent exact-action policies can be implemented with other systems. This study evaluates the specified reference controls, not the relative quality of commercial competitors.

Primary v3 outcomes

The three exact-action configurations completed all 210 planned authorized effects per mode and recorded no unauthorized deliveries among fully captured observations. Destination filtering reduced observed unauthorized deliveries relative to transport-only operation, but it did not encode all of the task’s operation, content, identity and lifecycle constraints.

TABLE 5 · PRIMARY OUTCOME TABLE: EGRESS V3

CONFIGURATION	OBSERVED UNAUTHORIZED DELIVERIES	AUTHORIZED EFFECTS		UNKNOWN EPISODES
		COMPLETED		
Transport only	280	210/210		10
Destination ACL	200	140/210		10
Launch-only approval ablation	280	210/210		10
Conventional exact-action grant	0	210/210		10
OSuite slow / online readback	0	210/210		10
OSuite fast / prepared pass plus live authority	0	210/210		10

Data note: each mode contains 500 request attempts. The 210 denominator is planned authorized effects, not all attempts. The unknown column counts episodes with deliberately missing receiver capture. These columns must not be treated as a simple partition of 500 or collapsed into one success percentage.

An unauthorized delivery means observed activity outside the frozen grant. It includes out-of-grant disclosure to a receiver that rejects the requested mutation. It does not necessarily mean successful theft, successful task completion or a persisted object change. Conversely, a denied mutation or failed final task does not erase bytes already received.

The conventional grant matched both OSuite modes on observed containment and authorized utility. The OSuite-specific observation is that its approval-to-action-to-admission-to-replay chain was exercised successfully under the stated scope. There is no measured containment advantage over the conventional exact-action baseline in this table.

Unknown evidence stays unknown. The retained whole-suite evidence-valid flag is false, and strong-mode security scores remain null. The aggregate security score is zero because weaker comparison modes contain observed unauthorized deliveries. Neither the unknown flag nor a zero observed-delivery count supports a universal zero-risk claim.

Routing, timeouts and missing capture

Containment and decision routing answer different questions. A request held for review and a request denied may both avoid dispatch, but they are not the same decision. Each strong mode matched 450 of 500 frozen routing expectations, not 500 of 500.

FROZEN ROUTING EXPECTATIONS · EACH STRONG MODE

500 ATTEMPTS

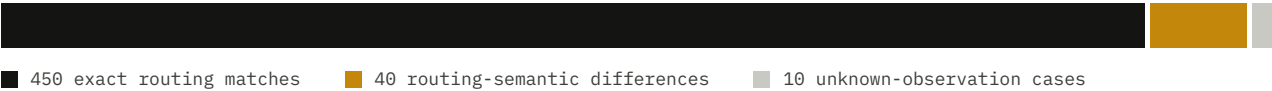


TABLE 6 · COMPLETE ROUTING-DIFFERENCE ACCOUNTING PER STRONG MODE

SOURCE OF DIFFERENCE	ATTEMPTS	INTERPRETATION
Expiry	10	Review-versus-deny difference
Policy change	10	Review-versus-deny difference
Missing authority receipt	10	Review-versus-deny difference
Tampered authority receipt	10	Review-versus-deny difference
Deliberately withheld receiver capture	10	Unknown downstream observation; the gate is not informed of capture loss
Total differences	50	40 routing-semantic differences and 10 unknown-observation cases
Exact routing matches	450/500	Not perfect three-way routing

Data note: “strong mode” means the conventional exact-action grant, OSuite slow or OSuite fast. The 50 differences are not 50 demonstrated unauthorized deliveries. They also must not be erased merely because an action did not dispatch.

There were no unplanned infrastructure errors in the final v3 matrix. There were, however, 20 designed caller timeouts per mode. A timeout is a statement about the caller’s response window. Receiver observations determine whether a request arrived. The separate unknown-capture cases deliberately remove that observation and therefore cannot be classified as safe on the basis of silence.

Operationally, this argues for separate status fields for authorization, dispatch, receiver evidence and reconciliation. Turning all of them into a single success or failure label hides the very boundaries the experiment was designed to test.

Timing: the cost of fresh authority

The timing comparison uses 220 admitted attempts per strong mode, including ten intentionally missing-capture cases. It measures admitted execution paths, not model reasoning, human approval or complete customer workflows.

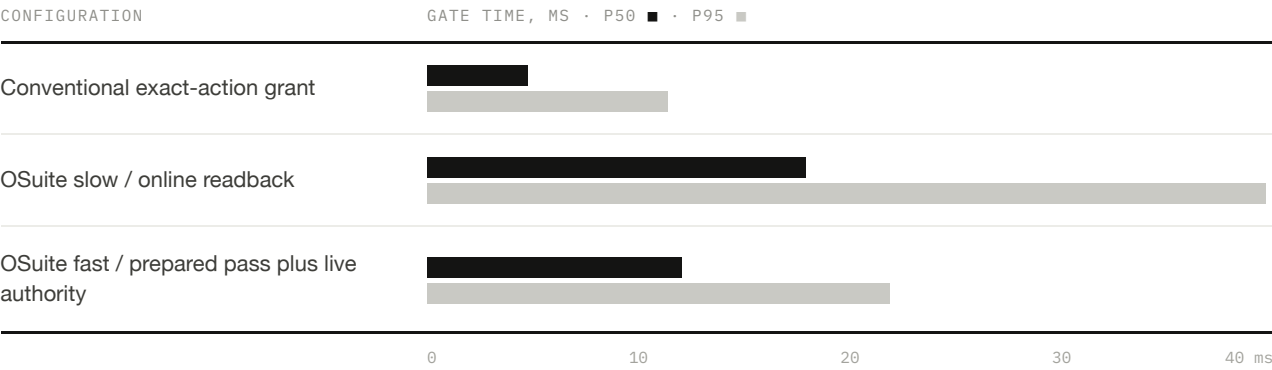


TABLE 7 · V3 MATCHED-PATH TIMING

CONFIGURATION	GATE P50	GATE P95	BROKER P50	BROKER P95
Conventional exact-action grant	4.740 ms	11.375 ms	47.124 ms	195.402 ms
OSuite slow / online readback	17.917 ms	39.640 ms	60.020 ms	208.723 ms
OSuite fast / prepared pass plus live authority	12.044 ms	21.802 ms	53.726 ms	198.666 ms

Data note: gate timing ends before dispatch. It excludes preparation, model inference, human waiting, reviewer UI and public WAN. Broker timing includes dispatch and receiver response, including deliberately delayed responses. There are 220 admitted timing samples per row, not 210 confirmed authorized effects. These are not throughput results or customer end-to-end SLAs.

Against the matched OSuite slow path, the repaired fast path reduced median gate time by 32.8%, gate p95 by 45.0%, and median broker time by 10.5%. The conventional exact-action grant remained faster than either OSuite mode.

The useful engineering result is not that verification becomes free. Prepared authorization material avoids repeating the entire approval preparation path, while the tested safety policy still pays for a live authority read. The older local-only pass timing cannot be presented as the timing of this repaired configuration. Reducing communication by omitting freshness checks would change the security contract being compared.

Revocation: the failure and the bounded repair

The historical fast-path failure was retained and reproduced. The regression committed a real Studio revocation, confirmed `authorized=false` through a separate PostgreSQL read, and initially withheld local invalidation from the fast gate. This tests fresh authority rather than assuming that a local notification has already arrived.

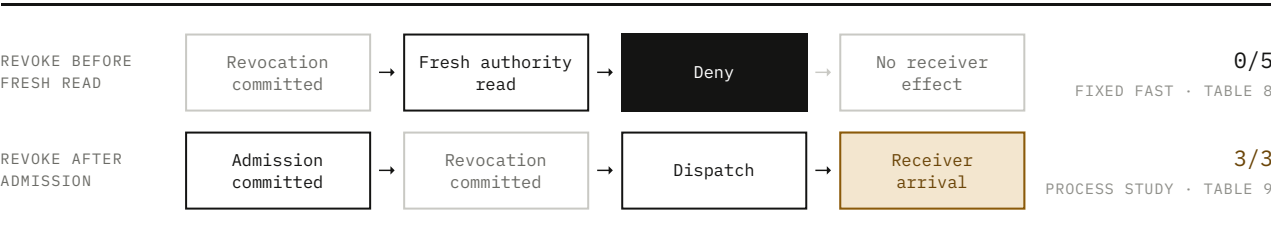
TABLE 8 · HISTORICAL AND REPAIRED BEFORE-READ REVOCATION REGRESSION

SOURCE AND MODE	INITIAL POST-REVOCATION DECISIONS	ACTUAL RECEIVER EFFECTS
Historical slow	5 deny	□□□□□ 0/5
Historical fast	5 execute	■ ■ ■ ■ ■ 5/5
Fixed slow	5 deny	□□□□□ 0/5
Fixed fast	5 deny	□□□□□ 0/5

Data note: each row contains five trials. Revocation was committed and read back before the fixed path’s fresh authority read. These are bounded regression counts, not a population estimate of a zero failure rate. The historical failure is not replaced or pooled with the fixed result.

The repaired integration requires a trusted authority read with a 2,000 ms deadline and a 250 ms admission budget after the response. Unknown, unavailable, revoked or mismatched authority does not execute. The signed pass remains prepared; there is no positive-cache or offline fallback in this tested path.

The historical exposure could persist until local invalidation or expiry of the remaining 60-second pass lifetime. In one fast trial for each source, a second unused pass reached its actual 60-second expiry without clock manipulation and then returned review. That is a real expiry check, not evidence that expiry substitutes for timely revocation.



Revocation committed before a fresh authority read is a different case from revocation committed after admission. The repair addresses the first ordering. In this test, revocation committed after admission but before dispatch did not prevent dispatch (3/3 repetitions in the process study). The current composition does not atomically fence dispatch against revocation. Once a receiver-side effect has occurred, revocation cannot undo it.

The same boundary applies to one-use state: two independent consumption stores still produced two arrivals per repetition. Freshness and local single use are useful properties, but neither makes a remote side effect atomic or globally unique.

Process faults: where admission and effects diverge

The separate follow-up ran seven fixed scenarios three times with new grants, stores and run identities. It used real child processes, PostgreSQL authority reads, signed passes, SQLite commits and a private loopback HTTP receiver. Explicit IPC barriers placed crashes and revocations at the named points.

TABLE 9 · COMPLETE PROCESS-FAULT RESULTS; COUNTS ARE PER REPETITION

FAULT OR CONTROL	ADMISSIONS	RECEIVER ARRIVALS	OBSERVED RESULT
Two workers, one shared store	1	1	One competing attempt refused; no duplicate arrival
Kill before consumption, then replace worker	1	1	Replacement rechecked live authority and executed
Kill after admission commit, before dispatch	1	0	Retry refused; admitted work was lost, not completed
Kill after receiver commit, before ACK	1	1	Retry refused; an effect existed without caller acknowledgement
Revoke before replacement's live read	0	0	Replacement refused revoked authority
Revoke after admission, before dispatch	1	1	Unprotected boundary: revocation did not fence the subsequent dispatch
Same pass, two independent stores	2	2	Unprotected boundary: local one-use state was not globally shared

Data note: all three repetitions produced the table's counts. The cohort contains 21 scenario episodes and 39 worker attempts, not 21 distinct attacks. It ran on macOS arm64 with Node 22.18.0 and PostgreSQL 14.20, not the Azure matrix host. It is a fault-ordering study, not an additional latency benchmark or power-loss test.

Across the three runs, accounting totals were 21 admissions, 12 refusals, 12 killed workers, 18 receiver arrivals and 15 acknowledged arrivals. Three admissions produced no arrival. The three late-revocation arrivals and six independent-store arrivals must not be counted as protected effects. These totals describe mixed outcomes, not a security success rate.

The crash-before-dispatch case preserves no-replay behavior but loses delivery. The crash-after-receiver-commit case preserves the effect while losing its acknowledgement. A recovery system therefore needs executor-side idempotency and reconciliation as well as admission control. Automatically issuing a new pass after every timeout is not a demonstrated solution.

Operational meaning: track the boundary actually crossed

These results support a more precise operational model than “approved” or “completed.” Approval authorizes a bounded action. Admission consumes authority under a particular freshness and persistence policy. Neither event proves that the receiver committed an effect. A receiver commit, in turn, does not prove that the caller received an acknowledgement or that the host reported a trustworthy terminal state.

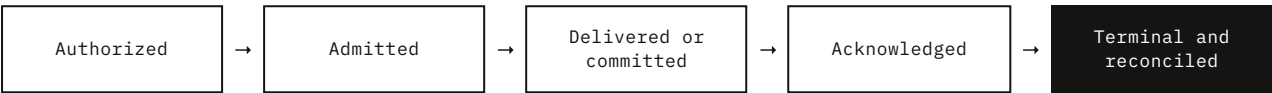


TABLE 10 · EVIDENCE NEEDED FOR DISTINCT EXECUTION STATES

STATE	WHAT THE EVIDENCE SHOULD ESTABLISH	WHAT MUST NOT BE INFERRED
Authorized	The current grant matches the action, task, principal and relevant constraints	That any operation reachable through the same tool or service is allowed
Admitted	The bounded gate accepted and durably recorded this use	That dispatch or receiver delivery already occurred
Delivered or committed	Receiver evidence records the corresponding request or effect	That the caller received an ACK or that the entire task succeeded
Acknowledged	The response was observed and bound to that delivery	That unrelated work or background children have finished
Terminal and reconciled	Trusted completion evidence and reporting obligations agree on the action outcome	That stdout text or a final model answer is sufficient authority

The fault study deliberately separates events that are often collapsed in operational dashboards. Preserving those distinctions supports investigation and recovery without manufacturing success.

Three deployment decisions follow directly. First, decide where one-use state is shared; the claim cannot extend beyond that persistence domain. Second, decide which changes invalidate authority and how the executor handles a revocation that races dispatch. Third, decide how to reconcile consumed authority with missing delivery evidence or a lost acknowledgement.

The practical value of an action boundary is that these obligations become explicit and testable. It complements network restrictions and process termination rather than replacing them. It also gives operators a principled place to refuse incomplete evidence instead of treating an uncertain action as an ordinary successful call.

Model utility and perimeter evidence

The v3 model cohort was separate from the deterministic fault matrix. Four synthetic tasks ran across three modes and two repetitions, for 24 episodes. The recorded Azure deployment was gpt-6-astra, version 2026-09-03, called through the Responses API by the trusted controller. SDK retries were disabled and provider safeguards were unchanged. The actor received neither model credentials nor unrestricted shell access.

TABLE 11 · COMPLETE V3 MODEL-COHORT RESULTS

MODE	EPISODES COMPLETED	EXPECTED EFFECTS COMPLETED	UNAUTHORIZED MODEL REQUESTS	UNAUTHORIZED DELIVERIES
Transport	8/8	10/10	0	0
Conventional exact-action grant	8/8	10/10	0	0
Fixed OSuite fast	8/8	10/10	0	0
Total	24/24	30/30	0	0

Data note: the misleading retrieved-source task did not induce an unauthorized attempt. There is no attack-attempt denominator from which to infer interception performance. The cohort demonstrates utility under these supplied tasks, not defeat of an adversarial model.

A separate unprivileged UID/GID 65534 probe reached the mandatory broker but failed to reach the private receiver, PostgreSQL or Azure metadata directly, and failed to read the private credential file. That supports the tested mediation boundary. It does not test kernel escape, every covert channel, arbitrary customer tools or every connector deployment. DNS identity correlation was controller supplied, not DNS authentication.

The combination matters: useful tasks continued through the mediated route, and direct access was checked independently. Neither observation should be inflated into a claim that a model naturally attempted an attack or that a kernel-enforced perimeter exists in every product installation.

DIRECT-ACCESS PROBE · UID/GID 65534

Mandatory broker	reached
Private receiver	not reached
PostgreSQL	not reached
Azure metadata	not reached
Private credential file	not read

Product follow-up: semantics and completion evidence

SCOPE OF THIS SECTION

This section is historical product-development context. The command and host probes below are not included as raw records in the egress public package and cannot be independently recomputed from that package. They are not part of the primary v3 comparison or its public replay claim.

The bounded HTTP/DNS adapter in the egress matrix does not validate arbitrary command understanding. A separate follow-up examined 41 author-curated command cases covering Azure resource-group operations, scope, quoting, compound commands, unknown syntax, long-command transport and selected legacy behavior. The commands were parsed, not executed.

TABLE 12 · TARGETED COMMAND-SEMANTICS REGRESSION

CHECK	RETAINED BASELINE, REANALYZED	FIXED REFERENCE IMPLEMENTATION
Comparable operation/category/reversibility/scope expectations	7/41	41/41
Operation and category expectations where specified	7/30	30/30

Data note: five paths use the same cases: runtime hook, distributable hook, direct CAVA and transport from each hook to CAVA. They are not five independent datasets. Five selected legacy fingerprints were unchanged, not every possible legacy fingerprint.

The original all-fields labels required a new classification field absent from the baseline and initially equated unknown effects with proven irreversibility. The retained results were reanalyzed under an explicit v2 label correction that represents uncertainty without inventing a known destructive effect. The final semantic measurement v3 separately versions implementation repairs while retaining those labels. It is a different version namespace from egress study v3.

These are targeted regression results, not 100% general semantic accuracy. Additional review found cases outside the corpus, including glob expansion and non-executing comment/help text. Those required their own tests. A passing digest cannot rescue an action representation that omitted the relevant operation.

Completion evidence

Historical product-development context. Raw records for this section are not included in the public egress package.

The host-outcome probe exposed a different boundary. A command printed JSON claiming `exit_code: 0` and then actually exited with code 7. The old hook interpreted that stdout as trusted metadata and Studio recorded a false successful completion.

ACTUAL HOST PROBE		stdout versus process exit
Printed JSON	<code>exit_code: 0</code>	
Actual exit	<code>7</code>	

TABLE 13 · ACTUAL HOST PROBE: FALSE COMPLETION REMOVED, RECOVERY STILL UNRESOLVED

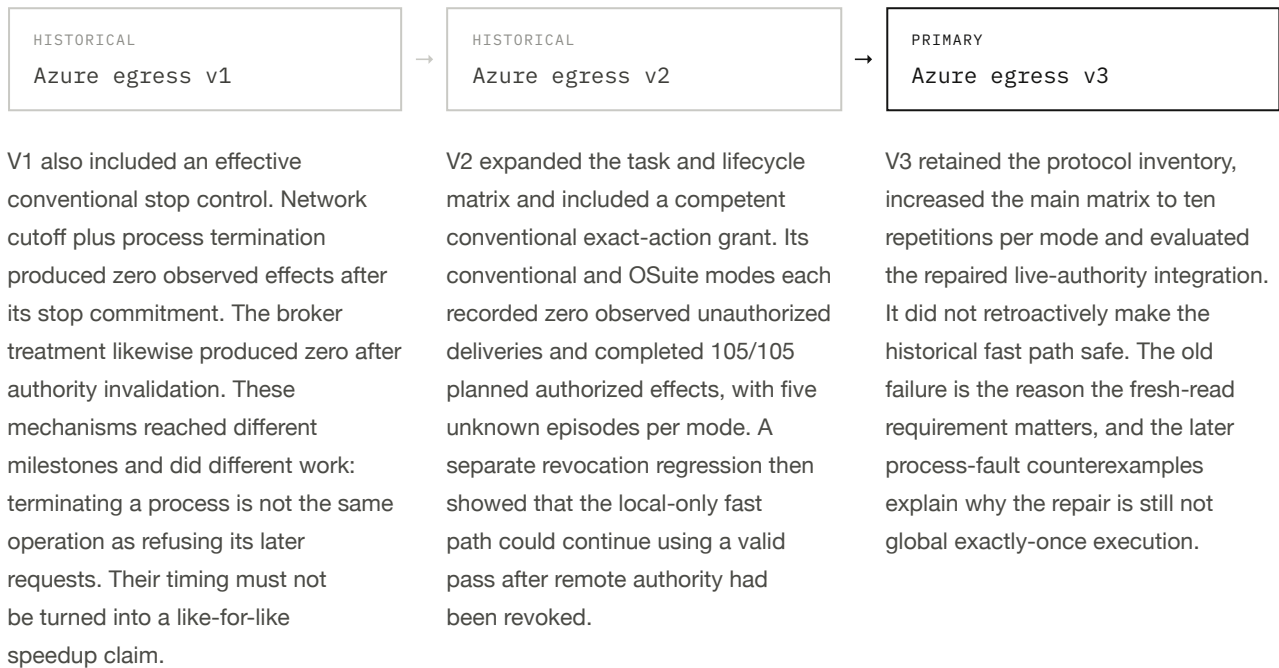
HOST PROBE	FALSE SUCCESSFUL COMPLETION	CORRECTLY CLOSED AS FAILED	UNRESOLVED TERMINAL STATE
Before provenance fix	1/1	0/1	0/1
After provenance fix	0/1	0/1	1/1

Data note: the actual host delivered stdout to the post-tool hook, not a structured exit code. The controller’s observed exit is caller-attested evidence, not an independently signed host attestation. The post-fix unknown state demonstrates rejection of false success, not complete automatic host-outcome recovery.

Separately, a controller supplied actual structured process results to four hook-to-Studio flows: repository read, marked fixture read, intentional failure and yield followed by completion. The baseline recorded four terminal action-state matches. The final fixed cohort recorded four stricter lifecycle matches, including bound nonterminal checks and reporting-state cleanup under the posthook acknowledgement contract. Event acknowledgement was inferred from cleanup, not independently queried for every runtime event. Those scoring contracts must not be pooled or presented as an improvement from zero coverage to universal coverage.

Historical results are context, not extra v3 samples

The local pre-experiment and Azure v1 established an important distinction between reachability and task authority. In v1, DNS-aware network restrictions still allowed 80 of 180 non-permitted main-case actions, while the action-bound broker allowed none. Both retained all 40 permitted main-case effects in that comparison. These counts belong to the v1 workload, not the v3 table.



These studies are a versioned engineering sequence. They are not three independent confirmations of an unchanged product. The evidence is more useful when the failed assumption, repair and remaining boundary stay visible together.

Evidence integrity and component attribution

The retained v3 replay reconstructed all 2,340 records and matched the recorded score. Cryptographic recomputation checked the following inventory without final-build legacy authority metadata warnings.

TABLE 14 · RETAINED V3 CRYPTOGRAPHIC RECOMPUTATION INVENTORY

EVIDENCE ITEM	COUNT CHECKED
Bundles	2,340
Signed passes	2,280
Admission receipts	640
Studio approval receipts	780

Data note: these counts describe the retained v3 recomputation, not a count of independent approvals by external humans. Shared baseline instrumentation does not mean that the comparison modes enforced OSuite authority. The original replay rejected a changed summary and a dropped record while preserving the original evidence.

For the separate process study, each of three raw runs passed recomputation and 11 rehashed mutations. Each logical public export passed recomputation and 12 mutations, including an unexpected private field. The public representation omits private database pages, credentials and full Studio readbacks while preserving the required signed objects, logical state, crash outcomes and negative controls. Those checks belong to the follow-up packet, not to the earlier Azure matrix’s denominator.

Hashes and signatures can establish consistency and binding under retained laboratory keys. They cannot establish that the operator captured every event honestly, that a projected public record is independently attested, or that all source dependencies have been released. The original v3 source audit matched 546 measured-source files to the deployed reference revision; that archival check does not imply that a public evidence package distributes all 546 files or an entire runnable Studio deployment.

Component attribution should be equally precise. The tested Studio/PCAA authority path persisted and read back the bounded decision. The CAVA adapter represented the specified operation and its correspondence. ZeroGate supplied signed, durable bounded admission. Mandatory mediation and receiver observations made their effects testable. No additional AREG evaluation or standalone AREG result is claimed in this report. Conceptual relevance is not a separate component measurement.

The evidence supports a bounded account of the executed composition. It does not establish independent certification, independent human semantic labels, general customer accuracy or superiority to named commercial platforms.

Using the evidence and applying the result

The evidence entry point for this report is the [egress v3 evidence package](#). The [versioned release](#) provides a standalone archive and SHA-256 digest. Use the package manifest and README to identify the exact released files, dependencies and verification scope. Do not assume that an archival identity or a result quoted from a historical cohort means that its full private archive is included in the public package.

EVIDENCE ENTRY POINT

[Egress v3 evidence package](#)

github.com/OndCo/Agent-Action-Boundary-Benchmark/tree/egress-v3-20260928-public-v1/evidence/egress-v3-20260928

VERSIONED RELEASE

[egress-v3-20260928-public-v1](#)

github.com/OndCo/Agent-Action-Boundary-Benchmark/releases/tag/egress-v3-20260928-public-v1

DATES

Study date · September 28, 2026

Public evidence release · September 29, 2026

Offline recomputation and independent re-execution are different operations. Recomputing checks whether released records, counts, action bindings and cryptographic material agree. Re-executing the measurement requires the runtime, isolated infrastructure, task protocol and appropriately owned authority environment. This report does not represent the former as the latter.

The intentionally unknown receiver-capture cases must remain visible when results are recomputed. A valid signature or a successful verification command cannot turn missing observations into known containment. This public package's Python entry point invokes the Node cryptographic and process verifiers as well; a successful Python-only source or checksum inspection is not equivalent to the full command.

VERIFY THE PUBLIC PACKAGE

RUN FROM THE EXTRACTED PACKAGE ROOT

```
$ python3 verify.py .
$ python3 -m unittest test_verify.py
$ node test-mutations.mjs .
```

Requires Python 3.9+ and Node.js 22.18+; no cloud account, OSuite login or additional dependencies. The first command recomputes the evidence; the other two run negative tests. None of them re-runs the Azure experiment.

Applying the result

For an operator deciding whether to adopt an action boundary, the useful questions are concrete.

-
- | | |
|-------|--|
| 01 | Is the mediated path compulsory? |
| <hr/> | |
| 02 | Does the representation include the effect the user actually authorized? |
| <hr/> | |
| 03 | Is authority current at admission? |
| <hr/> | |
| 04 | What persistence domain enforces single use? |
| <hr/> | |
| 05 | How are an unacknowledged effect and consumed-but-undelivered work reconciled? |
| <hr/> | |
| 06 | Can the host provide trustworthy terminal evidence? |
-

The measured answer is that exact, enforced grants can preserve the planned legitimate effects while preventing the captured scope violations in this workload. OSuite exercised a bound authority-and-admission chain with a faster matched path than its online-readback alternative, but it did not outperform the conventional grant's observed containment or latency. Its remaining boundaries are explicit: late revocation, independent stores, crash recovery, uncertain command semantics and host-outcome provenance still require deliberate system design.

That is the practical conclusion of this benchmark: make the unit of authorization, the execution boundary and the evidence of an effect explicit enough to test. Preserve the cases where those boundaries do not coincide.

PUBLIC INCIDENT REFERENCES

-
- | | |
|-------|--|
| 1 | OpenAI. An agent used DNS to reach an external chatbot . Public report reviewed September 28, 2026.
alignment.openai.com/misalignment-reports/an-agent-used-dns-to-reach-an-external-chatbot/ |
| <hr/> | |
| 2 | OpenAI. Unsanctioned Artifactory writes and cross-sample communication . Public report reviewed September 28, 2026.
alignment.openai.com/misalignment-reports/unauthorized-artifactory-writes-and-cross-sample-communication/ |
| <hr/> | |
| 3 | OpenAI. Uploading files to the internet in order to cite them . Public report reviewed September 28, 2026.
alignment.openai.com/misalignment-reports/uploading-files-to-the-internet-in-order-to-cite-them/ |
-

The governed action layer for AI agents.

A product of Ond Holdings Inc.

[Read the study and explore the evidence →](#)