

Versioned Transitive Dependency-Closure Binding and Operation-Time Effect Governance for Agent Skills: CLOSUREBOUND

GENLIANG ZHU, Accentrust, Canada and Georgia Institute of Technology, USA

CHU WANG, Accentrust, Canada and University of Illinois Urbana-Champaign, USA

Agent Skills combine instructions with files, packages, tools, models, and services, so operational identity can exceed a signed directory. Recursive or lazy dependencies may change while root-level evidence remains valid, and different surfaces may reach the same durable effect.

We present CLOSUREBOUND, a reference monitor that prevents authorization transfer across material changes to this heterogeneous closure. Its resolver commits typed graph nodes and topology. Each grant binds an exact closure root, effect ceiling, purpose/provenance, validity, and epochs. At durability, it re-resolves closure and state, normalizes the operation into an external-effect IR, and admits it only if a joint witness satisfies every bound. Supported equivalent paths share one ceiling.

Assuming complete mediation and discovery, authenticated freshness, sound normalization, cryptographic binding, and authoritative linearization, we establish metadata non-authority, closure determinism, version non-inheritance, effect non-amplification, bound-value freshness, and path invariance. We do not establish program equivalence or remote-service honesty. A provider-free implementation matches 40 frozen lifecycle fixtures; 18 kernel contracts and six mutants cover binding and downgrade cases. Full-profile exploration reaches 84,608 states and 530,752 transitions without a declared invariant violation; six weakened profiles yield witnesses. A lexical audit of 549 public Skills (4,872 unique files) finds that 21 of 526 roots with bundled files name every non-manifest path verbatim, 67 contain links resolving outside their roots, and no root declares a frontmatter dependencies field. These observations motivate conservative closure discovery and define concrete targets for broader runtime, interoperability, efficacy, and production validation.

CCS Concepts: • **Security and privacy** → **Software security engineering**; **Access control**; • **Software and its engineering** → *Software configuration management and version control systems*.

Additional Key Words and Phrases: AI agents, Agent Skills, dependency closure, supply-chain security, runtime enforcement, operation-time authorization, effect normalization

1 Introduction

Agent Skills are becoming a portable unit of procedural capability. The open format defines a directory with a required SKILL.md file and optional scripts, references, assets, and arbitrary additional resources; clients progressively load metadata, instructions, and resources as a task requires them [1, 2]. This simplicity is useful: one Skill can teach an agent how to reconcile invoices, deploy software, fill a form, or operate a domain-specific tool without retraining the model. It also means that a capability is assembled over time from natural-language instructions, executable artifacts, the host agent, and external services.

The installed directory is not the whole capability. A script may resolve a package at invocation; a reference may direct the model to a helper Skill; a tool name may resolve to a newer schema; an MCP endpoint may keep the same URL while its deployment changes; and a Skill may ask the agent to reproduce an effect through shell or browser automation when the direct tool is unavailable. Recent measurement confirms that Skill supply chains span mixed Skill-package-service graphs and that recursive reuse hides material inventory [12]. Recent attacks likewise exploit metadata, natural-language compliance rules, embedded models, and apparently routine side effects rather than only explicit malicious binaries [17, 31, 37, 46].

Authors' Contact Information: Genliang Zhu, Accentrust, Vancouver, Canada and Georgia Institute of Technology, Atlanta, Georgia, USA, research@accentrust.com; Chu Wang, Accentrust, Vancouver, Canada and University of Illinois Urbana-Champaign, Urbana, Illinois, USA.

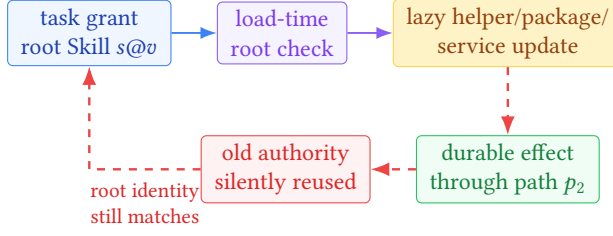


Fig. 1. The authorization-transfer gap. Load-time integrity of a root Skill does not bind a later transitive resolution or an equivalent effect path. CLOSUREBOUND rebinds the exact closure and normalized effect at operation time. Blue denotes trusted authority, purple identity, amber change, green the effect boundary, and red a rejected transfer.

The authorization-transfer bug. Suppose a user authorizes Skill `invoice-pay@1.3` to submit one payment to vendor v , using helper Skill `bank-wire@2.1`, package `iban@4.0`, and service contract `bank.example/payments@e17`. The root directory and its signature remain unchanged, but a floating helper resolves to `bank-wire@2.2`; the new helper calls an analytics service and changes the recipient before submission. Alternatively, every artifact is unchanged but the agent reaches the same payment through a shell client whose effect is not classified as “payment.” In both cases the old authorization is being transferred to a different operational capability.

Supply-chain attestations answer important questions about where artifacts came from and whether they changed [25, 36, 38]. Skill-specific systems add scanning, permission manifests, versioned dependency resolution and lockfiles, verified loading, least privilege, behavioral specifications, and runtime probes [6, 9, 13, 14, 18, 28, 35, 39, 41]. AIRGuard normalizes heterogeneous actions for action-time authority control; Cordon and Atomix stage effects and validate or order them at transaction/commit boundaries; CommitGuard binds durable effects to fresh authority evidence [7, 22, 30, 33]. These contributions establish adjacent foundations. ClosureBound addresses the following narrower systems question:

What exact transitive capability was authorized, and is that same capability—not merely the same root name—producing this same normalized effect now?

Approach. CLOSUREBOUND resolves an activated Skill into a typed graph whose nodes cover instructions, bundled files, recursively reused Skills, packages, tool schemas, model artifacts, services, and effect contracts. Its commitment includes canonical node evidence and typed edge topology. A task grant binds exactly one closure root together with purpose/provenance restrictions and an effect ceiling. Preparation produces only a short-lived reservation. Immediately before durability, the monitor re-resolves every material identity and epoch, normalizes the real surface event, checks the current pre-state, and atomically consumes the reservation. A change leads to denial or explicit reauthorization; uncertainty after dispatch remains indeterminate.

Contributions.

- (1) We define a root- and endpoint-sensitive commitment for the authenticated heterogeneous *operational* closure of an activated Skill, spanning instructions, files, packages, tools, models, services, and effect contracts. Our contribution is the authorization unit—the exact mixed closure used by one mediated effect—rather than a new dependency graph, lockfile, or SBOM.
- (2) We introduce a joint operation-time witness that binds a trusted task grant, exact closure root, current runtime state, purpose/provenance boundary, and normalized external effect. The

witness establishes authorization consistency; Skill benignness remains a separate content-analysis question.

- (3) We prove, under explicit assumptions, metadata non-authority, closure determinism, version non-inheritance, transitive effect non-amplification, operation-time TOCTOU safety, and equivalent-effect path invariance. Every theorem states its assumptions and claim boundary.
- (4) We provide deterministic algorithms for closure resolution and prepare/commit monitoring, together with a typed reauthorization-diff schema and fail-closed handling of ambiguity and indeterminate settlement.
- (5) We implement and evaluate a provider-free finite reference model on 40 frozen lifecycle fixtures, five weakened profiles, six executable mutants, bounded exploration, and a single-process local cost measurement. This evaluation targets mechanism conformance, necessity, finite-state safety, and local kernel cost; cross-runtime efficacy and ecosystem coverage require separate studies.
- (6) We conduct a separately specified static closure audit over 549 public Skills from two fixed, independently maintained repository snapshots. It characterizes local directory closure and explicit path structure as a fixed baseline for runtime-discovery and interoperability studies.

2 Background and Problem Analysis

2.1 A Skill is a progressively resolved capability

The Agent Skills specification requires only a name and description in `SKILL.md`; an experimental allowed-tools field and arbitrary string-valued metadata are optional. The August 2026 clarification also makes explicit that the conventional optional directories are non-exhaustive. The Markdown body has no policy syntax restriction, and scripts are expected to document or resolve their own dependencies [2]. Client guidance recommends progressive disclosure: catalog metadata is available first, full instructions are loaded on activation, and referenced resources are loaded only when required [1]. Therefore, the complete operational dependency set need not exist at discovery or even at activation.

We distinguish:

Discovery identity

the untrusted metadata used to retrieve or select a Skill.

Distribution identity

the files, signatures, registry records, and supply-chain attestations available before execution.

Resolved capability identity

the exact material transitive graph that can implement a protected effect in one execution.

Effect identity

the canonical external state transition at the durability boundary, independent of the surface used to request it.

Conflating these identities creates two dual errors. A false negative accepts a changed dependency because the root remains unchanged. A false positive blocks benign change in discovery order, resolver-local handles, or canonical path spelling even though the canonical evidence graph is identical. ClosureBound prioritizes sound authorization: unresolved ambiguity denies, while canonicalization and representation-equivalent commitments avoid gratuitous invalidation where the admitted evidence encoding is identical.

2.2 Why flat manifests and load-time checks are insufficient

A flat manifest hash can detect a changed listed item, but three structural problems remain.

Table 1. Research scope and evidence boundary.

Question	In scope	Explicitly out of scope
Skill identity	Canonical represented transitive closure and topology	Complete semantic dependency discovery for arbitrary prose/code
Authorization	Exact closure/effect joint witness under a trusted task grant	Inferring latent user intent or proving the grant is normatively correct
Runtime	Mediated prepare/commit and current epochs	Unmediated sinks, compromised monitor, Byzantine infrastructure
Effects	Finite adapter-defined external-effect IR	General program equivalence, receiver honesty, guaranteed rollback
Evidence	Finite conformance, mutants, bounded model, local kernel cost	Hosted-model efficacy, ecosystem prevalence, production SLOs

Completeness. If the manifest omits a nested Skill, lazy package, model file, service deployment, or effect adapter, hashing the manifest authenticates an incomplete statement. SPDX explicitly distinguishes asserted relationships from NOASSERTION; SLSA makes provenance completeness a property of the producer and notes recursive dependency verification as an additional step [16, 24]. ClosureBound carries this distinction into authorization: unknown material dependencies do not disappear.

Topology. An unordered inventory cannot distinguish “Skill *A* invokes package *P* only through sandboxed helper *B*” from “*A* invokes *P* directly.” Edge kinds affect mediation and the effect contract. The commitment must bind the typed graph, not only component digests.

Time. Loading precedes lazy resolution and the real effect. SIGIL correctly binds an approved artifact to the loaded artifact [35]; operation-time authorization correctly rechecks authority at durability [33]. ClosureBound composes these insights: the operation-time witness includes the exact current recursive closure.

2.3 Why tool-call syntax is not an effect boundary

One external transition may be requested as a typed API call, generated SDK code, a shell command, browser interaction, database query, or a helper Skill. Conversely, the same function name can represent different effects after a schema or service change. A policy attached only to surface syntax is therefore neither complete nor path invariant.

ClosureBound does not solve arbitrary program equivalence. Each protected sink has a trusted adapter that emits a finite effect IR. For a payment, relevant fields may include payer, payee, currency, amount, funding source, idempotency scope, and expected ledger state. For a message, they may include channel, recipients, data classes, and release semantics. Unsupported general code is denied or confined to a conservative effect ceiling.

2.4 Research scope

Table 1 separates the central claim from adjacent questions.

3 System and Threat Model

3.1 Entities and lifecycle

The principals are a user or organizational policy authority, an agent runtime, Skill publishers, package registries, tool/service operators, and external effect receivers. The lifecycle is:

discover → activate → resolve → grant → prepare → commit → settle.

Discovery and Skill-authored metadata are untrusted inputs. Distribution signatures and supply-chain attestations are authenticated evidence, but do not alone authorize effects. A grant issuer independently combines the current task certificate with organizational policy. The resolver and monitor form the enforcement core.

3.2 Adversary

The adversary may control Skill metadata, instructions, bundled files, nested Skills, packages, model artifacts, self-declared permissions, claimed dependencies, and proposed effects. It may exploit floating versions, dependency confusion, alternate locators, symlinks, cycles, caches, lazy loading, service redeployment, schema drift, replay, and prepare/commit races. It may induce the model to switch to a different mediated path that reaches an equivalent effect.

The adversary cannot forge authenticated task grants, trusted registry or service epochs, monitor reservations, or collision-resistant digests. It does not compromise the canonical resolver, effect normalizers, clock, linearization store, or policy authority. Direct effects that bypass all configured sinks are outside the theorem rather than counted as prevented.

3.3 Assumptions and trusted computing base

ASSUMPTION 1 (COMPLETE REPRESENTED DEPENDENCY CLOSURE). *Every material object that can influence or implement a protected mediated effect is represented as a node in the closure, or conservatively enclosed by a stronger node such as a pinned container or attested service deployment.*

ASSUMPTION 2 (COMPLETE EFFECT MEDIATION). *Every protected sink crosses a trusted adapter whose normalizer is sound for the supported surface. Unsupported or ambiguous effects fail closed.*

ASSUMPTION 3 (AUTHORITATIVE FRESHNESS). *Versions, service/schema/effect-contract epochs, revocation state, time, and pre-state witnesses come from fail-closed authoritative sources rather than solely from the Skill.*

ASSUMPTION 4 (LINEARIZABLE COMMIT). *Revalidation and reservation consumption are atomic at one operation linearization point, or the adapter consumes an equivalent current fencing token before the protected effect.*

ASSUMPTION 5 (CRYPTOGRAPHIC BINDING). *The canonical hash is collision and second-preimage resistant, authenticated statements are unforgeable, and keys/roots are administered correctly.*

The TCB includes the grant issuer, canonicalization rules, resolvers and evidence adapters, effect normalizers, version/epoch services, clock, operation-time monitor, reservation store, and cryptographic primitives. Remote services are trusted only to the granularity of the evidence they provide. A stable URL does not prove an unchanged backend.

3.4 Protected event and outcome states

The protected event is a monitor-mediated operation reaching its configured durability boundary. Preparation is not durability. Before dispatch, the monitor returns a decision in

$$\mathcal{D} = \{\text{ALLOW_COMMIT}, \text{DENY}, \text{REAUTHORIZE}\}.$$

After dispatch, settlement is reported separately as $\mathcal{O} = \{\text{COMMITTED}, \text{ABORTED}, \text{INDETERMINATE}\}$. A changed but legitimate closure requires a new grant; mutating an old grant would destroy non-inheritance. If dispatch may have occurred but the postcondition is unavailable, the settlement

remains INDETERMINATE. ClosureBound does not derive actual receiver state from a timeout or an unauthenticated receipt.

4 Formal Model

4.1 Typed dependency graph

Let s be an activated root Skill, t a time, π an authenticated evidence profile, c_π its separately versioned canonicalizer, and B a trusted pre-resolution effect upper bound compiled without closure-derived contracts. Its resolved capability graph is

$$\mathcal{G}_t(s; \pi, c_\pi, B) = (\mathcal{V}_t, \mathcal{E}_t, r_s),$$

where $r_s \in \mathcal{V}_t$ is the root. Node kinds are

$$\mathcal{K} = \{\text{skill, instruction, file, package, tool, model, service, effectContract}\}.$$

Edge kinds are

$$\mathcal{R} = \{\text{contains, reads, executes, imports, invokes, loads, resolves, declaresEffect}\}.$$

A node evidence record is

$$q_t(v) = \langle k, \ell, \delta, \chi, \zeta, \epsilon, \sigma \rangle,$$

where k is kind, ℓ a canonical locator (including the relevant namespace/issuer), δ a resolved version or deployment identity, χ a content or contract digest, ζ the authenticated digest of every resolution-context fact that can change outgoing dependencies, ϵ an authoritative epoch, and σ an evidence class and authentication proof. Resolver-local graph handles are not fields of q_t . Nodes with the same canonical record, including ζ , are interned before commitment. A mutable service with no authenticated deployment identity is represented only by its declared contract and URL; a policy may reject that weak evidence class.

$$q_t^E(u, r, v) = \langle r, h_V(u), h_V(v), \text{source}, \epsilon_E, \sigma_E \rangle$$

is the authenticated evidence record for an edge: its type and exact endpoint identities, derivation source, authoritative epoch, and evidence class/proof. The endpoint hashes are checked after both nodes have been interned; they are not taken on faith from a manifest.

DEFINITION 1 (MATERIAL CLOSURE). *A node v , or typed edge incident to it, is material under (π, c_π, B) if it can alter the selection, arguments, recipient, data, sequencing, implementation, or settlement interpretation of any $e \in \llbracket B \rrbracket$ within the declared mediation envelope. For a realized effect e , $\mathcal{I}_\mathcal{G}(e)$ is the union of authenticated root-to-sink influence paths that produced or can implement that event.*

The resolver can conservatively include more nodes. Omission threatens soundness; over-approximation causes reauthorization or false denial but does not expand authority.

4.2 Canonical graph commitment

Locators are normalized by a kind-specific total function C_k . Relative paths are resolved against the authenticated parent root; symlinks are dereferenced or rejected according to policy; package coordinates include registry namespace; service URLs use a canonical URI form; and effect contracts have explicit schema versions.

Cycles are possible through recursive Skills or services. Let $\text{SCC}(\mathcal{G})$ be the deterministic condensation DAG. For each node,

$$h_V(v) = H(\text{CB:node:v1} \parallel \text{enc}(q_t(v))).$$

For each typed edge,

$$h_E(u, r, v) = H(\text{CB:edge:v1} \parallel \text{enc}(q_t^E(u, r, v))).$$

Within each strongly connected component X , define

$$h_X = H(\text{CB:scc:v1} \parallel \text{sort}\{h_V(v) : v \in X\} \parallel \text{sort}\{h_E(e) : e \in E[X]\}).$$

For every inter-component edge $e = (u, r, v)$, with $u \in X_i$ and $v \in X_j$, bind both component and original endpoint identities:

$$h_C(e) = H(\text{CB:component-edge:v1} \parallel h_{X_i} \parallel h_V(u) \parallel r \parallel h_{X_j} \parallel h_V(v) \parallel h_E(e)).$$

The closure commitment includes the condensed topology and the exact root node:

$$\begin{aligned} \rho_t(s; \pi, c_\pi, B) = & H(\text{CB:closure:v1} \parallel h_V(r_s) \parallel h_{X_r} \\ & \parallel \text{sort}\{h_X\} \parallel \text{sort}\{h_C(e) : e \in E, X(u) \neq X(v)\}). \end{aligned} \quad (1)$$

Every $\text{sort}\{\cdot\}$ above denotes a canonical sorted multiset, not set deduplication; parallel authenticated edge records retain their multiplicity.

DEFINITION 2 (EXACT RESOLVABILITY). $\mathcal{G}_t(s; \pi, c_\pi, B)$ is exactly resolvable under profile π iff every material node and edge has a profile-admissible authenticated evidence record, all endpoint checks and root reachability checks succeed, no material version is floating or NOASSERTION, canonicalization succeeds, all required epochs are current, and every dynamic load is either enumerated or enclosed by an admissible stronger identity.

LEMMA 1 (CLOSURE DETERMINISM). For a fixed canonicalization profile, evidence records, typed graph, and root, Equation (1) is invariant to discovery order, resolver-local node-handle alpha-renaming, one-for-one substitution of an alias for the same interned canonical node while the canonical edge multiset is fixed, path spelling that canonicalizes to the same locator, and enumeration order.

PROOF. Kind-specific canonicalization is total on admitted locators. Local handles are quotiented by h_V , and SCC condensation over the resulting canonical-node graph is unique up to component naming. Every node, internal edge, component, and inter-component edge is domain separated and sorted before hashing. Therefore, handle alpha-renaming, one-for-one interned-alias substitution, permutations, and equivalent admitted spellings yield identical byte encodings at every level when the canonical edge multiset is unchanged. The explicit $h_V(r_s)$ distinguishes canonical roots inside one SCC; the endpoint-sensitive h_C distinguishes different cross-component attachments. The statement does not identify semantically equivalent but differently evidenced programs. $\square$

4.3 Task grants and policy order

A trusted task grant is

$$\gamma = \langle p, \tau, \iota, \psi, R, D, A, \rho_\gamma, \kappa, \mathbf{e}, \pi_\gamma, c_\gamma, [t_{\text{nb}}, t_{\text{exp}}], n \rangle.$$

Here p is principal, τ tenant, ι a version-qualified exact task atom, ψ purpose, R recipients, D resource/data bounds, A an effect ceiling, ρ_γ the authorized closure root, κ the effect-contract version, $\mathbf{e}$ expected exact epochs, π_γ the evidence-profile identifier, c_γ the canonicalizer/domain version, $[t_{\text{nb}}, t_{\text{exp}}]$ validity, and n a nonce. The authenticated A is an immutable issuer-compiled

ceiling; later policy expansion cannot enlarge it. Grant construction fixes $B = \gamma.A$ before closure discovery and uses that same bound for issuance, prepare, and commit resolution.

An effect policy denotes a set of external transitions. For policies a_1, a_2 ,

$$a_1 \sqsubseteq a_2 \iff \llbracket a_1 \rrbracket \subseteq \llbracket a_2 \rrbracket.$$

Lower policies authorize no more behavior. Let C_v be the authenticated effect policy carried by an applicable contract node. For the current effect, the closure operand is the relational meet

$$A_{\text{closure}}(\mathcal{G}_t, e) = \bigotimes_{\substack{v \in \mathcal{I}_{\mathcal{G}_t}(e) \\ C_v \text{ applies to } e}} C_v.$$

Here $\bigotimes$ denotes iterated relational meet over the indexed policy operands; Appendix B gives its binary definition and denotational law.

The resolver obtains $\mathcal{I}_{\mathcal{G}_t}(e)$ from the authenticated root-to-mediated-sink provenance emitted by the adapter; under Assumption 1, it includes every contract capable of influencing that event. If this provenance or a required contract is unresolved, the meet is undefined and evaluation fails closed. Current task authority, current closure contracts, runtime state, and purpose/provenance constraints compose by intersection, not by field-wise union:

$$A_{\text{cur}}(t, e) = A_{\text{task}}(t) \cap A_{\text{closure}}(\mathcal{G}_t, e) \cap A_{\text{state}}(t) \cap A_{\text{purpose/prov}}(t, e), \quad (2)$$

and the final ceiling is $\llbracket \gamma.A \rrbracket \cap \llbracket A_{\text{cur}}(t, e) \rrbracket$. Metadata supplied by a Skill may propose a narrowing contract to a trusted compiler, but cannot modify authenticated task/organizational operands or the already issued $\gamma.A$.

4.4 Normalized external effects

For a surface event x in current state S_t , a trusted adapter produces

$$e_t = \mathcal{N}(x, S_t) = \langle \text{op}, \text{target}, \text{recipient}, \text{dataClass}, \text{amount}, \text{argsDigest}, \\ \text{preStateDigest}, \text{postconditionContract}, \text{purpose}, \text{taskAtom}, \\ \text{effectContractVersion}, \text{durability}, \text{idempotencyScope} \rangle.$$

$\mathcal{N}$ is partial. Failure, ambiguity, or an unsupported surface returns $\perp$, which denies. We call e_t the *realized effect event* because its concrete surface, arguments, target, and authoritative pre-state are fixed at the durability gate; the term does not assert that the receiver's postcondition has already occurred or is truthful.

Let Ω be a declared external observation map.

DEFINITION 3 (EFFECT EQUIVALENCE).

$$e_1 \equiv_{\Omega} e_2 \iff \forall S \in \Sigma_{\Omega} : \Omega(\llbracket e_1 \rrbracket(S)) = \Omega(\llbracket e_2 \rrbracket(S)).$$

Σ_{Ω} is the common set of admissible pre-states on which both transition denotations and the post-state observation map are defined.

DEFINITION 4 (REGISTERED POLICY-SELECTION CONGRUENCE). Let $\mathcal{B}(\gamma, \mathcal{G}, S, t, e)$ be the canonical multiset of authenticated policy operands selected for (A4), including the immutable grant ceiling, current task/state/purpose operands, and every applicable contract selected through $\mathcal{I}_{\mathcal{G}}(e)$. An adapter

group is policy-selection congruent iff, for every supported pair $e_1 \equiv_{\Omega} e_2$ evaluated in the same authenticated $(\gamma, \mathcal{G}, S, t)$,

$$\begin{aligned} \text{authKey}(e_1) &= \text{authKey}(e_2), \\ \llbracket \bigotimes \mathcal{B}(\gamma, \mathcal{G}, S, t, e_1) \rrbracket &= \llbracket \bigotimes \mathcal{B}(\gamma, \mathcal{G}, S, t, e_2) \rrbracket. \end{aligned}$$

Here authKey contains every normalized effect field consumed by (A1)–(A6) but excludes the execution-surface name.

The monitor need not prove semantic equivalence for arbitrary code. Registry admission establishes Definition 4, normally by requiring supported equivalent surfaces to emit the same canonical IR and by making influence/contract selection a deterministic function of that IR and the authenticated closure. A conservative adapter may emit an over-approximated effect and thereby deny.

4.5 Joint operation witness

A prepared reservation at t_p binds the immutable record

$$W_{t_p} = \langle H(\gamma), \rho_{\gamma}, \rho_p, H(e_p), H(\lambda_p), \mathbf{e}_p, H(S_{t_p}^{pre}), \pi_{\gamma}, c_{\gamma}, r, n, t_p \rangle,$$

where λ_p is the purpose/provenance/recipient label and r is a single-consumption reservation. Let $\text{Rec}(r)$ be the authenticated reservation-store record. We define

$$\begin{aligned} \text{BindingIntact}(W_{t_p}, \gamma) &\iff W.\text{grant} = H(\gamma) \\ &\quad \wedge W.\text{grantRoot} = \gamma.\rho_{\gamma} \\ &\quad \wedge W.\text{prepareRoot} = \gamma.\rho_{\gamma} \wedge W.\text{epochs} = \gamma.\mathbf{e} \\ &\quad \wedge W.\text{profile} = \gamma.\pi_{\gamma} \wedge W.\text{canon} = \gamma.c_{\gamma} \\ &\quad \wedge W.\text{nonce} = \gamma.n \\ &\quad \wedge \text{Rec}(r) = \langle H(\text{enc}(W_{t_p} \setminus \{r\})), \text{fresh} \rangle. \end{aligned}$$

Thus a field from another grant, profile, canonicalizer, effect, state, or reservation changes the authenticated serialization rather than merely a local variable.

DEFINITION 5 (COMMIT ADMISSIBILITY). *Admit* $(\gamma, W_{t_p}, e_{t_c}, S_{t_c})$ holds at the commit linearization time t_c iff:

$$\text{Authn}(\gamma) \wedge t_{\text{nb}} \leq t_c < t_{\text{exp}} \wedge \text{TaskCurrent}(t, p, \tau, t_c), \quad (\text{A1})$$

$$\text{Exact}(\mathcal{G}_{t_c}(s; \pi_{\gamma}, c_{\gamma}, \gamma.A), \pi_{\gamma}) \wedge \rho_{t_c}(s; \pi_{\gamma}, c_{\gamma}, \gamma.A) = W.\text{prepareRoot} = \rho_{\gamma}, \quad (\text{A2})$$

$$\text{EpochsCurrent}(\mathbf{e}_{t_c}, W.\text{epochs}) \wedge \text{ContractCurrent}(\kappa, t_c) \wedge \text{ProfileCurrent}(\pi_{\gamma}, c_{\gamma}, t_c), \quad (\text{A3})$$

$$e_{t_c} \in \llbracket \gamma.A \rrbracket \wedge e_{t_c} \in \llbracket A_{\text{cur}}(t_c, e_{t_c}) \rrbracket \wedge \text{LabelFits}(\lambda_{t_c}, \psi, R, D), \quad (\text{A4})$$

$$H(S_{t_c}^{pre}) = W.\text{pre} \wedge \text{FreshReservation}(r, n, t_c), \quad (\text{A5})$$

$$H(e_{t_c}) = W.\text{effect} \wedge H(\lambda_{t_c}) = W.\text{label} \wedge \text{BindingIntact}(W_{t_p}, \gamma). \quad (\text{A6})$$

The use of one witness is material: roots from one grant, effect ceilings from another, or fresh epochs from a third may not be recombined into an authority that no issuer granted.

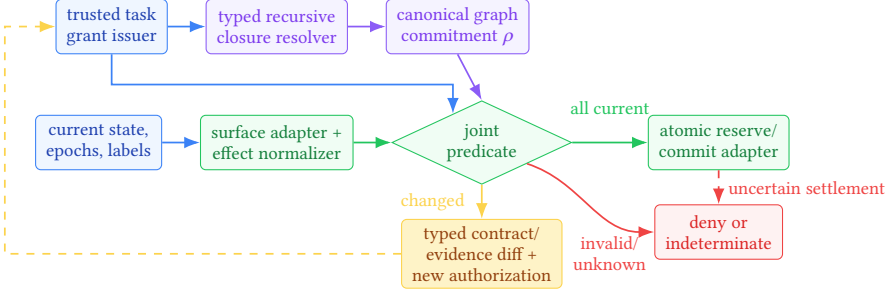


Fig. 2. CLOSUREBOUND architecture. Authority, exact transitive identity, current state, purpose/provenance, and the normalized effect meet at one operation-time decision. A material change creates a new grant; it never edits the old closure binding.

4.6 Safety invariants

For every admitted commit event c :

- I1 – Trusted authority.** c 's grant descends from the authenticated current-task authority; Skill metadata is not an authority root.
- I2 – Exact closure.** $c.\rho = \rho_t(s; \pi_Y, c_Y, \gamma.A) = \rho_Y$, and every material node and edge satisfies the grant-bound evidence profile.
- I3 – Non-amplified effect.** $c.e \in [[\gamma.A]] \cap [[A_{\text{cur}}(t, c.e)]]$.
- I4 – Current operation.** All exact epochs, contract versions, pre-state, and reservation facts hold at the linearization point.
- I5 – Path invariance.** Supported effect-equivalent surface events are checked against the same policy denotation.

5 ClosureBound Design

Figure 2 shows the control path. Blue components produce trusted task authority, purple components resolve identity, green components mediate effects, amber denotes reauthorization or uncertainty, and red denotes fail-closed rejection. Supply-chain attestations and Skill scanners feed evidence; they do not bypass the task/effect decision.

5.1 Resolver adapters and evidence profiles

Different node kinds cannot share one naïve notion of version. ClosureBound uses a small adapter interface:

$$\text{resolve}_k(\text{locator}, \text{parent}, \text{context}, \pi, c_\pi, B, t) \rightarrow \langle q_t(v), \text{outgoing}, \text{status}, \text{contextKey} \rangle.$$

An outgoing item is $\langle r, k', \text{locator}', q^E, \text{context}' \rangle$. The context key includes every lockfile, feature selection, platform, tenant, and resolver-policy fact that can change outgoing dependencies. Its authenticated digest is the ζ field of the contextual vertex identity. Consequently, interning and memoization by $(q, \text{contextKey})$, rather than by q alone, cannot collapse two context-sensitive closures or erase which outgoing subgraph belongs to which call context.

- A file adapter binds the canonical path, bytes, executable mode, and enclosing root; symlink policy is explicit.
- A package adapter binds ecosystem, registry namespace, immutable version/digest, lock evidence, and resolved runtime dependencies.

- A recursive Skill adapter resolves the target directory and repeats the Skill/file/package/service walk.
- A tool adapter binds the server identity, method schema, argument and result contracts, and declared effect contract.
- A model adapter binds an immutable artifact or a provider deployment identity at the evidence strength required by policy.
- A service adapter binds canonical audience, authenticated contract, deployment/epoch evidence where available, and the effect adapter version.

The MCP 2026-07-28 revision makes this context requirement concrete. Its stateless core carries protocol version and client capabilities per request, adds server discovery, and makes tool/resource listings deterministic and cacheable with `ttlMs` and `cacheScope`; authorization credentials are issuer-bound [20, 21]. An MCP adapter therefore binds at least the negotiated protocol revision, authenticated server identity, tool schema/listing digest, cache scope and observed freshness interval, plus any explicit server-minted cross-call handle. A TTL is a cache hint, not proof that an old grant authorizes a later deployment or effect.

An evidence profile π sets minimum classes per node/effect risk. For example, a read-only documentation task may admit a service URL plus schema epoch, while a payment task may require an attested deployment and transactional idempotency contract. Profiles are policy, not self-declared Skill metadata.

5.2 Recursive resolution

Algorithm 1 builds the closure. Its context-sensitive visited set terminates repeated expansion; SCC canonicalization handles cycles after the walk. Discovery order is recorded for diagnosis but excluded from the canonical root.

Resolver output is evidence, not authority. A malicious but immutably pinned package will be identified consistently and can still be denied by the grant/effect policy.

5.3 Grant compilation

Before closure discovery, the grant compiler derives a trusted upper bound B_0 from the authenticated current-task certificate, organizational/user policy, and purpose constraints. A Skill-authored manifest may propose an additional narrowing conjunct, but cannot add an effect absent from those trusted operands. Closure-derived contracts do not contribute to B_0 . The resolver then computes $\mathcal{G}_t(s; \pi, c_\pi, B_0)$; the compiler validates that its prospective relational clauses are representable and freezes $\gamma.A = B_0$ together with ρ_γ . At prepare and commit, closure-derived contracts remain a separately recomputed conjunct of A_{cur} in Equation (2); they never feed a narrowed result back into the graph-discovery bound. This two-phase rule avoids a circular equation of the form $A \leftarrow \mathcal{G}(A)$.

Cross-dimensional correlation is preserved. A grant for (billing, vendorA, \$500) and another for (refund, customerB, \$50) cannot donate their fields to authorize (billing, customerB, \$500). The executable representation is a finite relation of complete effect clauses: authorization meets preserve whole tuples rather than unioning independently permitted fields. Appendix B gives the relational-meet argument and the corresponding clause-splicing counterexample.

5.4 Effect normalization and path registry

Each protected adapter registers:

$$\alpha = \langle \text{surfaceVersion}, \text{match}, \text{normalize}, \text{effectContract}, \text{stateWitness}, \text{durability}, \text{fence}, \text{settle} \rangle.$$

Algorithm 1 RESOLVE-CLOSURE

Require: root Skill s , profile π , canonicalizer c_π , candidate ceiling B , trusted time t
Ensure: exact graph G and root ρ , or fail closed

```

1:  $Q \leftarrow [(skill, s, \perp, \perp, \perp, ctx_{root})]$ ;  $V \leftarrow \emptyset$ ;  $E \leftarrow \emptyset$ ;  $Seen \leftarrow \emptyset$ 
2: while  $Q \neq \emptyset$  do
3:    $(k, \ell, parent, r_{in}, q_{in}^E, ctx) \leftarrow \text{POP}(Q)$ 
4:    $\ell' \leftarrow C_k^{c_\pi}(\ell, parent, ctx)$ ; reject if canonicalization fails
5:    $(q, outs, status, ctxKey) \leftarrow \text{resolve}_k(\ell', parent, ctx, \pi, c_\pi, B, t)$ 
6:   if  $status \neq \text{exact}$  or  $q.\sigma \not\models \pi(k)$  then
7:     return UNRESOLVED( $k, \ell', status$ )
8:   end if
9:   verify  $q.\zeta = H(\text{enc}(ctxKey))$  under  $\pi$ 
10:   $v \leftarrow \text{INTERN}(q, ctxKey)$ ;  $V \leftarrow V \cup \{v\}$ 
11:  if  $parent = \perp$  then  $r_s \leftarrow v$ 
12:  end if
13:  if  $parent \neq \perp$  then
14:    verify  $q_{in}^E$  authenticates  $(parent, r_{in}, v)$  under  $\pi$ 
15:     $E \leftarrow E \cup \{(parent, r_{in}, v, q_{in}^E)\}$ 
16:  end if
17:  if  $(v, ctxKey) \notin Seen$  then
18:     $Seen \leftarrow Seen \cup \{(v, ctxKey)\}$ 
19:    for all  $(r, k', \ell_o, q^E, ctx_o) \in outs$  do
20:      PUSH( $Q, (k', \ell_o, v, r, q^E, ctx_o)$ )
21:    end for
22:  end if
23: end while
24: reject if any node/edge material under  $(\pi, c_\pi, B)$  is unreachable, missing, stale, or NOASSERTION
25:  $G \leftarrow (V, E, r_s)$ ;  $\rho \leftarrow \text{SCC-MERKLE-COMMIT}(G, \pi, c_\pi, B)$ 
26: return  $(G, \rho)$ 

```

match identifies intercepted events without trusting model prose; normalize emits the IR; stateWitness reads authoritative preconditions; durability declares the irreversible boundary; and fence connects the monitor reservation to the sink.

The registry groups adapters by effect-contract denotation. A direct payment API and a browser payment form may differ in fields and timing but normalize to the same payment clause. A shell adapter that cannot determine the payee and amount before execution cannot join that group; it must be denied or confined in a sandbox whose maximum possible effect is authorized.

5.5 Prepare and commit

Algorithm 2 makes the temporal boundary explicit. Prepare is advisory and reserves a candidate. Commit repeats all volatile checks and consumes a fence immediately before durability. Define PrepareEligible($\gamma, \xi_p, G_p, \rho_p, S_p, \lambda_p, e_p, e_p, t_p$) as the prospective form of (A1)–(A4): every occurrence of a prepared-witness field is replaced by its value in ξ_p , while reservation-only (A5)–(A6) are not yet evaluated. Thus this predicate is defined before W_{t_p} exists.

Acquire-Consistent-Snapshot binds the resolver frontier, state, labels, epochs, effect, profile, and canonicalizer in the opaque token ξ . Atomic-Validate-Consume re-evaluates (A1)–(A6), verifies that all supplied values still equal the authoritative facts represented by ξ_c , and consumes r in

Algorithm 2 PREPARE-THEN-COMMIT**Require:** authenticated grant γ , root Skill s , surface event x

```

1:  $t_p \leftarrow \text{TRUSTED-TIME}$ 
2:  $(\xi_p, G_p, \rho_p, S_p, \lambda_p, \mathbf{e}_p, e_p) \leftarrow \text{ACQUIRE-CONSISTENT-SNAPSHOT}(s, x, \gamma.\pi_\gamma, \gamma.c_\gamma, \gamma.A, t_p)$ 
3: deny unless  $e_p \neq \perp$  and  $\text{PrepareEligible}(\gamma, \xi_p, G_p, \rho_p, S_p, \lambda_p, \mathbf{e}_p, e_p, t_p)$ 
4:  $W^- \leftarrow \langle H(\gamma), \rho_\gamma, \rho_p, H(e_p), H(\lambda_p), \mathbf{e}_p, H(S_p), \pi_\gamma, c_\gamma, n, t_p \rangle$ 
5:  $r \leftarrow \text{RESERVE}(H(\text{enc}(W^-)), n); W_{t_p} \leftarrow W^- \cup \{r\}$ 

6: ▷ Operation-time linearization
7:  $t_c \leftarrow \text{TRUSTED-TIME}$ 
8:  $(\xi_c, G_c, \rho_c, S_c, \lambda_c, \mathbf{e}_c, e_c) \leftarrow \text{ACQUIRE-CONSISTENT-SNAPSHOT}(s, x, \gamma.\pi_\gamma, \gamma.c_\gamma, \gamma.A, t_c)$ 
9:  $(d, f) \leftarrow \text{ATOMIC-VALIDATE-CONSUME}(\gamma, W_{t_p}, \xi_c, \rho_c, H(S_c), H(\lambda_c), \mathbf{e}_c, H(e_c), t_c)$ 
10: if  $d \neq \text{ALLOW\_COMMIT}$  then
11:   return  $d$  with authenticated failure evidence
      and exact typed diff when available; otherwise UNKNOWN
12: end if
13:  $\omega \leftarrow \text{DISPATCH-WITH-FENCE}(x, e_c, f)$ 
14: if authenticated settlement or independent postcondition verifies then
15:   return COMMITTED
16: else if dispatch is known not to have crossed durability then
17:   return ABORTED
18: else
19:   return INDETERMINATE
20: end if

```

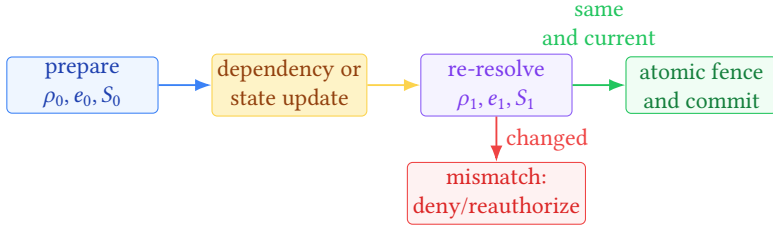


Fig. 3. Operation-time update discipline. Preparation never makes an old closure authoritative. A distinguishable material mismatch at the linearization point forces denial or a new grant.

the same compare-and-swap. The returned fence is an authenticated one-shot token over at least $(H(\gamma), \rho_c, H(e_c), H(S_c), \mathbf{e}_c, n, r)$. A trusted adapter or receiver must re-normalize the actual request and atomically require the same $H(e_c)$ before crossing durability; mutable x is never used as an unbound authority input. A local database transaction can implement this directly. A remote service can consume the effect-bound fence and reject older frontier/epoch tokens. If neither pattern is available, the system can authorize dispatch but cannot claim atomic operation-time safety for the receiver.

5.6 Typed contract/evidence diff and reauthorization

When $\rho_1 \neq \rho_\gamma$, the system reports a typed diff:

$$\Delta = \langle V^+, V^-, V^\sim, E^+, E^-, A^+, A^-, \text{evidenceChanges} \rangle,$$

where $V^\sim$ records version/digest/epoch changes and A^+, A^- are differences in the finite adapter-declared effect-contract denotations, not inferences of arbitrary program semantics. If the authenticated contracts do not make that difference exactly representable, the field is UNKNOWN and review fails closed. Cosmetic metadata outside the material closure can be marked non-authoritative, but the old grant remains immutable. A trusted issuer creates γ' over ρ_1 after reviewing Δ . This transition is widening/renewal, not ordinary execution.

5.7 Caching without stale authority

Resolvers may cache immutable nodes by digest. Mutable lookups are keyed by authority epoch and evidence profile. A cached graph is a performance hint: commit still validates an authenticated root-to-currentness certificate or re-resolves. Cache age is not a security proof. Negative and unresolved results may be cached only fail closed.

6 Security Analysis

6.1 Metadata cannot mint authority

THEOREM 1 (METADATA NON-AUTHORITY). *Under Assumptions 1, 3, 4, and 5 and the grant-compilation rule, changing Skill-authored metadata, instructions, self-declared permissions, or explanations cannot add an element to the effect set admitted by an existing grant.*

PROOF. The issuer authenticates and freezes $\gamma.A$; Skill data cannot rewrite that field. At commit, (A4) requires membership both in this immutable ceiling and in the independently recomputed current task, closure, state, and purpose/provenance meet. A proposed Skill operand is accepted only through the trusted compiler as an additional conjunct. Enlarging or replacing that proposal still cannot authorize an effect outside $\gamma.A$. A changed material instruction also changes ρ , so the old grant fails (A2). The result does not say metadata cannot influence model behavior; it says such influence is not trusted authority at the mediated effect boundary. $\square$

6.2 Closure and version non-inheritance

LEMMA 2 (MATERIAL-CHANGE SEPARATION). *Under Assumption 5, changing the canonical root identity so that $h_V(r_s)$ changes, a canonical material node record, an authenticated typed material-edge record (including either endpoint's canonical h_V), or SCC topology changes ρ except with negligible cryptographic probability.*

PROOF. The changed record changes a domain-separated node, edge, SCC, or endpoint-sensitive component-edge encoding. An SCC-internal root change alters the separately committed $h_V(r_s)$. Each change propagates to the final root. Equality would require a collision or second preimage in H . Changes outside the declared material closure are not covered. $\square$

THEOREM 2 (VERSION/CLOSURE NON-INHERITANCE). *Under Assumptions 1, 3, 4, and 5, let γ authorize closure root ρ_0 . If, at the commit linearization point, a material transitive dependency, service/tool/effect contract, or typed edge has a canonical value different from the value bound by ρ_0 , γ cannot admit that changed closure without a new grant.*

PROOF. By Lemma 2, an exactly resolved material change gives $\rho_1 \neq \rho_0$ except with negligible probability. If resolution is no longer exact, Definition 2 fails. In the first case (A2) rejects the mismatch; in the second it rejects incompleteness. Algorithm 2 returns REAUTHORIZE or DENY, and the immutable grant cannot be edited. $\square$

Non-conclusions.

The theorem does not detect behavioral changes behind unchanged weak service evidence, malicious behavior already present in ρ_0 , or omitted dependencies that violate Assumption 1.

6.3 Transitive effect non-amplification

LEMMA 3 (MONITOR REFINEMENT). *If the atomic validation step in Algorithm 2 returns its commit-allow decision—equivalently, if execution reaches the fenced dispatch step—then $\text{Admit}(\gamma, W_{t_p}, e_{t_c}, S_{t_c})$ held at that validation step’s linearization point.*

PROOF. The only path to dispatch requires Atomic-Validate-Consume to return ALLOW_COMMIT. By its interface contract and Assumption 4, that atomic transition evaluates (A1)–(A6) over the authoritative snapshot ξ_c , verifies the full frontier/profile/state/ effect tuple, and changes the matching reservation from fresh to consumed. Every non-admitting result exits before dispatch. The later COMMITTED, ABORTED, and INDETERMINATE labels report settlement knowledge and cannot retroactively change the admission predicate; in particular, an indeterminate dispatch still passed the same authorization linearization point. $\square$

THEOREM 3 (TRANSITIVE EFFECT NON-AMPLIFICATION). *Under Assumptions 1–4 and 5, every effect e_{t_c} admitted by ClosureBound at its linearization time t_c is in*

$$[[\gamma.A]] \cap [[A_{\text{task}}(t_c)]] \cap [[A_{\text{closure}}(\mathcal{G}_{t_c}, e_{t_c})]] \cap [[A_{\text{state}}(t_c)]] \cap [[A_{\text{purpose/prov}}(t_c, e_{t_c})]].$$

No ordinary dependency path adds authority beyond this intersection.

PROOF. By Lemma 3, every effect released to Dispatch-With-Fence satisfied Definition 5. Complete mediation gives its normalized e_{t_c} . Complete closure and (A2) bind the exact current implementation, authenticated influence paths, and applicable effect contracts. (A1) binds trusted current-task authority; (A3) binds current epochs/contracts/profile; (A4) separately checks the immutable grant ceiling and current relational meet; (A5)–(A6) bind state, labels, reservation, and effect to the same prepared witness. Hence the effect is in every displayed operand. A nested Skill, package, or service is already within the bound closure and cannot supply a separate authority operand. $\square$

Non-conclusions.

The result is about represented mediated effects. It does not prevent covert channels, prove source truth, or repair a compromised normalizer or receiver.

6.4 Operation-time TOCTOU safety

THEOREM 4 (OPERATION-TIME FRESHNESS). *Under Assumptions 5, 3, and 4, a bound closure root, expected epoch, effect contract, normalized effect, or pre-state whose authoritative value at the commit linearization point differs from the prepared/grant-bound value cannot be admitted using the prepared reservation.*

PROOF. Algorithm 2 reacquires every volatile fact at commit. A closure, effect, label, epoch, contract, profile, canonicalizer, or pre-state change that remains distinguishable at the linearization point violates (A2)–(A6). Transient ABA histories that authoritative evidence intentionally maps back to the same identity are outside this value-freshness claim; deployments requiring ABA detection must use non-reused epochs. The opaque snapshot token and Atomic-Validate-Consume compare the complete current tuple and consume the reservation in one atomic transition, preventing a concurrent update from validating against old facts after the check. Without atomicity or fencing, the theorem’s premise does not hold. $\square$

6.5 Equivalent-effect path invariance

THEOREM 5 (SUPPORTED PATH INVARIANCE). *Assume effect mediation is surface-complete, adapters are extensionally sound with respect to Ω , and their registry group satisfies Definition 4. If x_1, x_2 are supported surface events evaluated under the same authenticated grant, closure, state, and linearization time, and $N(x_1, S) \equiv_{\Omega} N(x_2, S)$, ClosureBound evaluates them against the same effective policy denotation and neither path can bypass the closure/currentness predicates.*

PROOF. Both surfaces enter Definition 5. Registered policy-selection congruence gives the same authorization key and the same denotation for the complete applicable policy bundle, including the effect-dependent influence/contract selector. Conditions (A1)–(A3) and (A5)–(A6) are enforced for every registered path and bind the root, epochs, state, reservation, profile, labels, and normalized effect. Therefore switching supported paths cannot enlarge the authorization set. $\square$

Non-conclusions.

ClosureBound does not discover arbitrary semantic equivalence. The theorem excludes an unregistered sink and fails closed when a general shell/browser event cannot be normalized before durability.

6.6 Necessity counterexamples

The conjunction is not cosmetic. Each reduced mechanism admits a short counterexample:

Root-only identity. Keep root bytes fixed and update a nested package or service deployment.

Closure only at load. Prepare under ρ_0 , update a helper, and commit without re-resolution.

Effect-only commit. Keep the payment IR within its ceiling but change the implementation to an unauthorized analytics-bearing closure.

Closure-only commit. Keep the closure fixed and switch the recipient or amount through a mutable event.

Surface-name policy. Deny a direct payment tool but reach the same payment through browser or shell.

Uncorrelated field union. Combine purpose from one clause, recipient from another, and amount from a third.

PROPOSITION 1 (NO REPRESENTED REDUCED MECHANISM FAMILY IS SUFFICIENT). *For each mechanism family removed by a reduced profile in Table 2, and for each single-point mutant in Table 4, there exists a finite trace on which that weakened evaluator admits an effect excluded by the full joint predicate.*

PROOF. The six constructions above cover the profile-level mechanism families. Replay, grant forgery/expiry, and stale pre-state cases exercise additional atomic conditions. Each seeded mutant changes exactly one implementation mechanism while retaining the full-profile evaluator elsewhere; the artifact executes and records the corresponding trace. The traces establish mechanism-family necessity; per-condition independence across (A1)–(A6) and exhaustive attack enumeration are separate questions. $\square$

7 Reference Implementation

7.1 Scope

The artifact is a deterministic Node.js reference kernel with no hosted model, network dependency, or private data. It implements canonical graph commitments, finite effect clauses, epoch checks,

Table 2. Transparent reference profiles. A check mark means that the finite reference evaluator performs the dimension.

Profile	Root	Closure	Effect	Commit refresh	Correlation	Purpose
Metadata/manifest only	–	–	–	–	–	Negative control
Root identity	✓	–	–	–	–	Flat-root binding
Load-time closure	✓	✓	–	–	–	Exact load snapshot
Effect-only commit	–	–	✓	✓	✓	Effect boundary
Prepared closure + effect	✓	✓	✓	–	✓	Composed prepare binding without commit-time closure/effect/state refresh
CLOSUREBOUND	✓	✓	✓	✓	✓	Full joint predicate

fixture execution, mutants, bounded exploration, aggregation, and consistency verification. Reservation freshness and single consumption are modeled as evaluator facts and lifecycle transitions; the artifact does not implement a production Reserve/Atomic-Validate-Consume transaction.

The reference model is a publication-oriented research implementation over synthetic records, separate from Accentrust production systems. It contains no private adapter, customer or tenant record, production endpoint or schema, credential, deployment topology, operational policy, or incident trace.

The ClosureBound evaluation implementation and its machine-readable execution traces are retained internally and are not included in the arXiv source bundle. Appendix C states the disclosed configuration and resulting availability boundary.

The reference profile abstracts production signature verification, package resolution, MCP servers, browser drivers, and distributed fencing. Cryptographic records use deterministic SHA-256 digests and explicit authenticated flags. This design isolates binding logic; key management and provider security are outside the evaluated properties.

7.2 Canonical records

Node records use a fixed JSON canonicalizer with lexicographically sorted object keys, explicit array-order rules, domain tags, and rejection of non-finite numbers or unsupported values. Graph arrays are sorted only after node and edge identity has been fixed. Resolver handles are excluded from node digests; authenticated resolution-context digests remain included. Duplicate canonical nodes are interned; edge multiplicity and kind remain explicit.

The reference effect IR supports file writes, messages, payments, database mutations, deployments, and permission changes. Each clause binds operation, exact target, recipient, data class, a nonnegative amount ceiling, payload digest, postcondition contract, purpose, exact task atom, effect-contract version, durability, and idempotency scope. The fixtures use synthetic identifiers and payload digests only.

7.3 Profiles and mutants

Table 2 defines mechanism families rather than reproductions of cited systems. The labels describe which checks are intentionally present; they must not be read as empirical results for SIGIL, SkillGuard, VIGIL, CommitGuard, or another implementation.

In Table 2, “commit refresh” marks a reduced profile that performs any operation-time currentness checks, not necessarily the complete frontier. The effect-only profile rechecks the current normalized effect against the grant, exact epochs, task/time status, and reservation freshness, but omits closure, authoritative pre-state, effect-stability, evidence-profile, and canonicalizer checks. Only CLOSUREBOUND performs the complete joint refresh.

Six seeded semantic mutants each weaken one full-profile mechanism: project away one material transitive package, omit commit-epoch validation, omit commit-root validation, bypass normalization for one trusted surface name, replace relational clauses with field unions, or omit reservation freshness. For every mutant, the unmodified full evaluator first rejects the same trace; the mutant is counted as detected only if its single weakening then reaches the targeted unsafe commit.

7.4 Complexity

For $n = |V|$, $m = |E|$, and total evidence bytes L_{evid} , resolution and SCC construction are $O(n + m + L_{\text{evid}})$ plus adapter lookup costs. Sorting commitments is $O(n \log n + m \log m)$. With immutable-node caching, re-resolution hashes only changed evidence but still validates the authenticated currentness frontier. Policy checking is linear in the number of non-dominated correlated clauses in the simple reference representation.

Worst-case graph and clause growth are real. A production implementation should use canonical antichains, exact dominance elimination, immutable subgraph roots, and fail-closed profile limits. It must not handle exhaustion by silently dropping dependencies or unioning restrictive clauses.

8 Evaluation

8.1 Questions and evidence discipline

The evaluation asks five bounded questions:

- **RQ1: Conformance.** Does the reference profile make the expected decision on every frozen represented lifecycle case?
- **RQ2: Necessity.** Do reduced profiles and seeded semantic mutants expose the counterexamples predicted by Proposition 1?
- **RQ3: State safety.** Does exhaustive exploration of a declared finite lifecycle find an invariant violation in the full profile, and does it find witnesses for mutants?
- **RQ4: Local cost.** What is the narrow in-process cost of canonical commitment and the joint gate on one disclosed host?
- **RQ5: External structure.** Under a frozen, explicitly lexical audit, what local and cross-root path structure appears in two public Skill repository snapshots?

Questions about model adherence, hidden-dependency prevalence, real-runtime sink coverage, and task utility require complementary empirical methods. The external audit supplies repository-structure evidence and a baseline for adapter-based runtime studies.

8.2 Fixture construction

The frozen suite has 40 synthetic cases: 32 unsafe rows across eight four-case families and eight benign controls. Every row fixes:

$$\langle G_{\text{grant}}, G_{\text{prepare}}, G_{\text{commit}}, Y, x_{\text{prepare}}, x_{\text{commit}}, S, \text{expected} \rangle.$$

The unsafe families are:

- (1) root Skill or root-instruction mutation;
- (2) nested Skill or package mutation with unchanged root;
- (3) service-deployment or effect-contract epoch drift;

Table 3. Finite fixture results. Unsafe allow is the security error; false denial counts benign rows denied contrary to their exact oracle. Profiles are mechanism classes, not reproduced cited systems.

Profile	Exact	Unsafe allow	False deny	Decision err.	Main omitted protection
Metadata/manifest only	8	32	0	32	Trusted recursive identity and effect gate
Root identity	14	26	0	26	Transitive closure and normalized effect
Load-time closure	20	20	0	20	Operation-time refresh and effect binding
Effect-only commit	24	16	0	16	Artifact/closure identity
Prepared closure + effect	32	8	0	8	Commit-time freshness/replay fencing
CLOSUREBOUND	40	0	0	0	None within represented profile

- (4) explicitly inexact resolution, a floating version, unauthenticated node evidence, or a declared node made unreachable;
- (5) prepare/commit closure or current-state races;
- (6) amount, recipient, data-class, or effect-contract-version amplification;
- (7) out-of-ceiling equivalent-effect paths through helper, shell, or browser surfaces; and
- (8) a consumed reservation, revoked task, expired grant, or correlated-clause splicing.

The eight benign controls cover canonical discovery order, canonical path spelling, alpha-renamed resolver handles, an exact-ceiling payment, an authorized helper path, a dependency update accompanied by a new grant, a fresh service epoch, and one non-durable read-only effect. A control is counted correct only if it reaches its expected allow/reauthorize/deny state; “safe because everything was denied” would fail the suite.

8.3 Kernel contract checks

All 18 deterministic kernel contracts pass. They cover order and path canonicalization, material-node changes, cross-SCC endpoint rewiring, root changes inside an SCC, unreachable-node rejection, authenticated grant and task/purpose binding, correlated-clause non-splicing, equivalent-path normalization, unsupported-surface denial, cross-SCC edge-evidence changes, and evidence-profile/canonicalizer downgrade. They also check that local-handle alpha-renaming leaves the root unchanged and duplicate canonical-node handles are interned, expected epoch maps require exact key/value equality, payload and pre-state changes alter effect identity, and authenticated resolution context alters node identity. K04 and K05 are direct regression tests for the two collision-free equal-root constructions found during the independent formal audit; K12 verifies that cross-component edge evidence, not only its endpoints, enters ρ ; K14 and K15 guard representation invariance; K16–K18 guard exact currentness and effect/context identity.

8.4 Conformance and reduced profiles

Table 3 is generated from the aggregate JSON. The full profile matched all 40 expected outcomes: all 32 unsafe rows were denied or sent to reauthorization before durability, and all eight benign controls received their exact expected decision. It had zero unsafe allow, false denial, or decision mismatch on this finite suite.

These counts should not be ranked as detector accuracy. Cases are authored litmus tests with exact structural oracles, and profile differences follow from deliberately omitted checks.

8.5 Mutation evidence

All six seeded mutants produced a targeted unsafe witness:

The 6/6 mutation score is targeted mechanism evidence for the six declared faults, rather than general coverage of implementation errors.

Table 4. Single-point full-profile mutants and minimized witness shape.

Mutant	First violated invariant	Minimal witness
Project out material package	I2 exact closure	Full resolver reauthorizes; projected resolver/grant omits changed package and commits
Omit commit epoch	I4 current operation	Full gate reauthorizes after redeploy; mutant accepts prepared epoch
Omit commit root	I2 exact closure	Full gate reauthorizes after helper update; mutant accepts prepared root
Trust shell surface	I5 path invariance	Full clause gate rejects; named shell path bypasses normalized policy
Union clause fields	I3 non-amplified effect	Full relation rejects; mutant combines recipient and amount from distinct clauses
Omit reservation freshness	I4 current operation	Full gate denies consumed nonce; mutant commits its replay

8.6 Bounded lifecycle exploration

The explicit-state model contains phases

{idle, resolved, granted, prepared, committed, denied, replayDenied},

two closure versions, two service epochs, two effect ceilings, two purpose/recipient correlations, two surface paths, one revocation transition, authenticated versus forged grants, and a single-consumption reservation. From the initial state it enumerates every enabled update, resolve, grant, prepare, revoke, commit, denial/restart, and replay-attempt transition until the BFS queue is empty; no depth cutoff is used. Every profile permits the environment to attempt replay: the full monitor reaches replayDenied, while the reservation mutant accepts the second use and records an I4 witness.

The full profile reached a fixed point after 84,608 unique states and 530,752 transitions; the maximum shortest-path depth was 23 and the remaining frontier was zero. It found no violation of I1–I5, including untrusted-grant and equivalent-surface states. Each of the six mutant profiles reached a violation, and the artifact records its first breadth-first witness. These finite profiles abstract the same six mechanism families as the executable mutants. Exploration is exhaustive within the encoded finite abstraction; package-manager and remote-service behavior require separate models. The manuscript theorems, rather than the BFS count, support the general conditional claims.

8.7 Local mechanism cost

The microbenchmark runs after warm-up in one Node.js process on an Apple M4 Max host. Inputs are preconstructed in memory; network access, package resolution, filesystem reads, signatures, remote attestations, model inference, durable transactions, and receiver settlement are excluded.

For 100,000 iterations over the fixed small fixture, the frozen reported run’s median in-process closure commitment is 0.045 ms (p95 0.055 ms), pure effect normalization plus its correlated-clause gate is 0.002 ms (p95 0.003 ms), and a full-profile fixture evaluation is 0.120 ms (p95 0.146 ms). The last operation includes two closure commitments and two normalizations; the three measurements are not additive pipeline stages. They establish that the reference kernel is mechanically lightweight on the disclosed host. The result characterizes in-process mechanism cost, excluding authoritative resolution, distributed fencing, and other production operations.

Table 5. External static closure audit at two fixed public commits. “All paths named” is evaluated only among roots with bundled files. It is a strict lexical measure, not semantic dependency recall.

Source	Skills	Unique files	Bundled roots	All paths named	Roots with cross-root links
Anthropic Skills	19	411	18	6	0
OpenAI Plugins	530	4,461	508	15	67
Total	549	4,872	526	21	67

8.8 External repository-snapshot closure audit

To add evidence independent of the authored lifecycle fixtures, we froze two public distributions: 19 top-level Skills under `skills/*/SKILL.md` at Anthropic repository commit 3b3fad96 and 530 non-fixture plugins/**/skills/**/SKILL.md files at OpenAI repository commit 1e285826 [3, 23]. The two repository-layout adapters recursively enumerate regular files and symlinks below each selected Skill root without following symlinks. They separately classify unique Markdown link targets relative to that root and count a bundled file as lexically named only when its full root-relative path occurs verbatim in SKILL.md.

The 549 roots contain 5,220 root-file instances but 4,872 unique local files because some selected Skill roots nest other Skill roots. Among the 526 roots with at least one non-SKILL.md file, only 21 name every such relative path verbatim. Across 1,341 unique Markdown targets, 208 links in 67 Skills resolve inside the same repository but outside the immediate Skill root; 17 are absolute-path targets and 17 do not resolve in the frozen checkout. These last categories are not automatically defects: a path may be documentation-rooted, generated at use time, or intentionally external. None of the 549 frontmatters contains a `dependencies` field, which is consistent with the current format rather than a conformance failure.

The result supports a structural premise: hashing SKILL.md or assuming exhaustive path mention is insufficient to reconstruct exact operational closure in general. Lexical mention is neither necessary nor sufficient for runtime loading, and repository enumeration excludes packages, services, tools, generated paths, and runtime effects. The snapshots therefore add externally authored structural evidence to the mechanism study and motivate runtime validation of interoperability, prevalence, and discovery completeness.

8.9 Validity and evaluation traceability

Internal validity. Fixtures and implementation were authored together. Exact per-row oracles, transparent profiles, semantic mutants, structural traces, and an artifact verifier strengthen internal validity by exposing decision and binding errors. The external corpus supplies independently authored inputs; its extractor and lexical measures remain part of our study design.

External validity. The supported inference from the two frozen repositories is public distribution structure at two commits. Production clients, package ecosystems, MCP deployments, models, marketplaces, and runtime behavior require separate empirical sampling; dependency completeness and sink coverage remain deployment-specific.

Construct validity. “Unsafe allow” means a represented durable effect violated the frozen joint predicate. It is not a judgment that a Skill is malicious. “False denial” is interpreted against the authored benign oracle and does not estimate broad user utility.

Table 6. Closest-work capability matrix. “Exact heterogeneous closure” requires authenticated operational Skill/package/tool/model/service structure; “joint witness” binds that closure and realized effect in one durability-time authorization record.

Work	Exact hetero. artifact closure	Runtime authority	Durability refresh	Realized-effect normalization	Joint witness	Primary scope
Skills Are Not Islands [12]	○	–	–	–	–	Measurement and dependency management
SkillFortify [6]	○	○	–	–	–	Static analysis, SAT resolution, lockfile
OpenAPM v0.1 [19]	○	–	–	–	–	Install-time package/lock/policy draft
SIGIL [35]	○	○	–	–	–	Audit-to-load integrity
SkillGuard [28]	○	✓	○	○	–	Skill permission framework
Edge Skillguard [41]	○	✓	○	○	–	Physical-world state/sensor guards
VIGIL [14]	○	✓	○	○	–	Trace behavioral enforcement
SkillScope [39]	○	✓	○	○	–	Task-conditioned action graph
Dynamic Capabilities [42]	○	○	○	–	–	Manifest certificate and ledger
AIRGuard [30]	–	✓	○	○	–	Pre-action normalized authority control
Cordon [7]	○	✓	✓	○	○	Transaction lineage/state/effect validation
CommitGuard [33]	–	✓	✓	○	○	General durable-effect authority
CLOSUREBOUND	✓	✓	✓	✓	✓	Exact closure/effect non-inheritance

Evaluation traceability. The internal evaluation records Node/OS/CPU metadata, fixture and source digests, profile counts, per-case traces, bounded counterexamples, and microbenchmark parameters. The reviewed snapshot was rerun against frozen inputs, and an internal verifier checked both result sets and cross-checked every manuscript count. These controls establish snapshot identity and internal consistency.

9 Closest-Work Comparison

Table 6 compares guaranteed axes, not headline system names. A check mark means the cited work directly supplies the property according to its published description; a circle means partial or adjacent support.

Direct complementarity and novelty boundary. SkillFortify and OpenAPM already supply substantial dependency-resolution, lockfile, typed-package, and install-policy machinery; AIRGuard already normalizes heterogeneous tool actions for current authority; Cordon already validates lineage, authority, staged state, and pending effects as a composed transaction; and CommitGuard already establishes durability-time authority refresh. Edge Skillguard already adds typed world-state and sensor-evidence guards to physical-world Skills. ClosureBound’s novelty lies in the joint construction: to the best of our knowledge, prior work has not jointly bound an authenticated heterogeneous operational-artifact closure and a cross-surface normalized *realized* effect in one durability-time authorization witness, nor established both closure-version non-inheritance and equivalent-effect path invariance under explicit completeness assumptions.

10 Related Work

10.1 Agent Skill security and measurement

Large-scale analysis of 31,132 Skills found vulnerabilities across prompt injection, exfiltration, privilege escalation, and supply-chain categories [18]. Lifecycle taxonomies emphasize missing data/instruction boundaries, persistent trust, and marketplace review [15]; SkillSec-Eval extends lifecycle evaluation through repository, retrieval, planning, execution, and evolution over 327 reported real-world Skills [4]. MalSkillBench contributes runtime-verified malicious and matched benign Skills, while AgentTrap measures complete trajectories and routine-looking unsafe side effects [10, 46]. Runtime Skill Audit dynamically probes risk-relevant contexts [13]. SkillDetonate lifts instructions observed during sandboxed execution and tracks OS-boundary information flow;

its “closure” is runtime instruction/flow coverage, not a version commitment [11]. Composition studies further show that benign-looking Skills can become harmful along activated paths [40]. These works establish the threat and evaluation ecosystem. ClosureBound is not a detector and does not reuse their rates as its evidence.

Natural-language surfaces create distinct risks. Under the Hood of SKILL.md studies discovery, selection, and governance manipulation; payload-less Skills induce harmful generated behavior without recognizable payloads; BadSkill hides backdoors in model artifacts [17, 31, 37]. These results motivate metadata non-authority, model nodes, and effect mediation. They do not imply that exact identity makes harmful content safe.

10.2 Skill permissions, least privilege, and behavioral enforcement

SkillGuard connects manifests, context influence, action permissions, user authorization, and runtime monitoring [28]. SkillScope extracts task-conditioned action graphs and constrains over-privileged paths [39]. VIGIL compiles temporal, argument, and value-flow specifications to finite-trace enforcement [14]. A distinct SIGIL system compiles natural-language Skills into typed executable harnesses and evaluates mandate compliance across public Skills and three runtime models [9]. Edge Skillguard co-packages typed invocation policy over world state and sensor evidence for physical actuation [41]. These systems govern what behavior is permitted. ClosureBound addresses whether the exact recursive implementation and effect contract being used now are the ones for which that permission was granted.

SIGIL binds audited Skill artifacts to a mandatory verified loader [35]; dynamic-capability certificates bind a manifest hash to agent identity [42]. ClosureBound builds on both artifact-binding contributions. Its later enforcement point and typed transitive/effect joint witness address authority transfer after loading.

10.3 Software supply-chain integrity

TUF secures software-update metadata under key compromise; in-toto verifies intended supply-chain steps and materials; SLSA specifies increasingly strong provenance and build guarantees [25, 36, 38]. SPDX and CycloneDX represent components, services, and dependency relationships [16, 26]. ClosureBound uses these as evidence formats and locates its contribution in operation-time authorization rather than SBOM, provenance, lockfile, attestation, or Merkle construction. Skill-Fortify formalizes an Agent Dependency Graph, capability-aware SAT resolution, deterministic Skill lockfiles, and ASBOMs [6]; its August 2026 v2 also corrects bibliography, external claims, and the empirical boundary of its information-flow analysis. The OpenAPM v0.1 working draft specifies transitive package/MCP resolution, typed primitives, lockfiles, tree hashes, and install-time governance while explicitly leaving runtime harness APIs out of scope [19]. ClosureBound begins after these contributions: it asks whether a trusted task grant remains valid for the authenticated operational closure and realized effect at durability.

10.4 Runtime and usage control

Reference-monitor complete mediation and least privilege are classical design principles [32]. Security automata characterize execution monitoring [34]; Polymer composes runtime policies [5]; UCON models ongoing authorization, obligations, conditions, and mutable attributes [29]. Cedar provides expressive, analyzable authorization with a verified core [8]. ClosureBound is a specialized policy/state model built on these foundations.

FORGE provides recursive Datalog policies, causal-history predicates, an assume/guarantee observability service, and a reference monitor [27]. AIRGuard provides pre-action runtime authority control over normalized heterogeneous tool actions [30]. Atomix uses epochs and resource frontiers

Table 7. Public team-work reuse boundary. Prior observations and denominators are not ClosureBound evidence.

Work	Adjacent public abstraction	Not re-claimed here
OpenPort	Protocol facts, policy/effect vocabulary, conformance discipline	Protocol novelty or published runtime evidence
IGAC	Authenticated current-task certificate	Intent inference and IGAC empirical results
EBTE	Server-owned claims and digest-bound evidence	Explanation verification novelty and observations

for transactional release of tool effects, while Cordon validates result lineage, delegated authority, staged state, and pending effects before commit [7, 22]. Temporary Authority, Permanent Effects most directly establishes freshness, causal priority, effect binding, and eligibility at the durability boundary [33]. ClosureBound’s contribution is the Agent-Skill-specific conjunction of an exact recursive heterogeneous artifact identity and cross-surface realized-effect equivalence bound to that current durability decision.

10.5 Relationship to public prior team work

Three publicly accessible team papers provide adjacent design context. OpenPort supplies protocol and governed-effect vocabulary; IGAC supplies an independently derived current-task authorization certificate; and EBTE separates model-authored explanations from server-verified action claims [43–45]. Table 7 makes the reuse boundary explicit.

ClosureBound is a distinct identity/effect unit over this public context. No non-public manuscript contributes a premise, artifact row, model run, bounded state, or empirical denominator. The 40-case, mutation, bounded-model, performance, and external-corpus results reported in Section 8 are generated solely by this paper’s reported evaluation configuration and internally verified receipts.

11 Discussion

11.1 Security value and residual risk

ClosureBound changes the default unit of trust from “Skill *s* was once approved” to “this exact current recursive capability is authorized to produce this exact class of effect now.” It is valuable against silent dependency substitution, service/schema drift, load-to-use races, path laundering, and stale authorization replay. Content benignness, provider honesty under weak evidence, and unmediated channels remain separate security controls.

11.2 Completeness is a deployment obligation

Assumption 1 is intentionally demanding. Agent Skills allow arbitrary prose and scripts, and measurement shows heterogeneous hidden dependencies [12]. A deployment should expose a completeness frontier rather than a Boolean marketing claim:

$$\text{coverage} = \langle \text{nodeKinds}, \text{sinks}, \text{dynamicLoads}, \text{evidenceClasses}, \text{unknowns} \rangle.$$

High-risk policy may require “no unknown material dynamic loads” and attested service deployments. Lower-risk tasks may accept a pinned container as a conservative enclosing identity. An unversioned service URL is not silently upgraded to strong evidence.

11.3 Availability and update ergonomics

Exact non-inheritance means updates cause reauthorization. This is a security property and an availability cost. Typed contract/evidence diffs, reusable subgraph roots, risk-tiered evidence profiles,

and policy-authorized version ranges can reduce friction. A range is itself part of the grant and must be evaluated against a trusted compatibility/effect contract; @latest is not an exact range.

Automatic renewal is safe only when a trusted policy already authorizes the new denotation and the new grant binds the new root. Calling it “same grant” would hide the authority transition.

11.4 Remote services and settlement

Three service identities should not be confused:

- (1) endpoint/audience identity;
- (2) declared schema/effect-contract identity; and
- (3) deployed implementation identity.

Many services expose only the first two. ClosureBound can prevent token audience confusion and schema drift, consistent with MCP authorization guidance [20]. Deployed implementation identity requires independent evidence. Under MCP 2026-07-28, stateless per-request protocol context, server discovery, issuer binding, and cache TTL/scope become explicit evidence inputs, not substitutes for deployment identity or operation-time authorization [21]. Confidential or irreversible profiles should require deployment attestation, a constrained server-side transaction contract, or human confirmation.

An authenticated receiver receipt establishes only the receiver-reported state unless an independent postcondition or receiver-soundness premise is available. A lost response after possible durability is indeterminate, not safe to repeat automatically.

11.5 Effect adapters as a governance API

The adapter registry is a major TCB surface. Its contracts should be versioned, reviewed, fuzzed, and included as closure nodes. A general command adapter can often predict filesystem/network effects through a sandbox plan, but commands with opaque dynamic behavior may require a coarse ceiling such as “may write any file under directory d and contact domain set R .” Coarse normalization preserves soundness at the cost of false denial.

11.6 Policy and legal interpretation

Purpose, recipient, and data-class labels are policy inputs. ClosureBound’s theorems cover conditional enforcement; latent intent, legal validity, and social desirability require separate human and institutional judgments. Ambiguity should produce clarification or review rather than be resolved by Skill prose.

11.7 Ethics and dual use

The evaluation uses synthetic identifiers, hashes, and state transitions; the preprint does not publish live malicious Skills, credentials, or exploit payloads. Closure and effect traces can reveal sensitive architecture, so production telemetry should minimize payloads, separate authorization evidence from content, apply retention limits, and enforce tenant isolation.

12 Limitations and Future Work

The current work has six principal limitations.

- (1) **Conditional completeness.** The proofs assume a sound dependency frontier and protected-sink coverage. Natural-language and reflective dynamic dependencies require deployment-specific instrumentation and conservative fallback.
- (2) **Finite effect semantics.** Path invariance covers registered adapters, not arbitrary code equivalence or covert channels.

- (3) **Service evidence.** A stable endpoint/contract cannot establish unchanged remote behavior.
- (4) **Finite and static evidence.** Authored fixtures and bounded exploration establish mechanism behavior under the evaluated profiles. The two-snapshot external corpus contributes independently authored inputs and lexical structural evidence; cross-runtime efficacy requires execution-based adapter evaluation.
- (5) **Single-process cost.** The reported microbenchmark excludes network, storage, signatures, attestations, distributed ordering, and model latency.
- (6) **No receiver theorem.** The monitor authorizes dispatch and conditionally interprets settlement; it does not guarantee exactly-once external effects or compensation.

The highest-value next studies are two independent runtime adapters, a broader probability-sampled Skill corpus with runtime-observed exact-resolution coverage, a matched translation of AgentTrap or MalSkillBench to effect contracts, a distributed fencing prototype, and an independently encoded TLA+/Alloy/Lean model. Together, these studies would extend the present mechanism evidence into cross-runtime, ecosystem, distributed-systems, and independently encoded formal validation.

13 Conclusion

Agent Skills turn procedural text and software into capabilities that are resolved progressively and may change transitively. Authorizing the root name, directory, manifest, or load-time artifact is therefore not enough to authorize a later durable effect. ClosureBound binds a trusted task grant to the exact current heterogeneous dependency closure and the normalized real effect in one operation-time witness. Under explicit completeness, mediation, freshness, cryptographic, and linearization assumptions, material updates do not inherit authority, represented effects do not amplify transitive privilege, distinguishable bound-value mismatches at commit fail closed, and supported equivalent paths are checked against the same effective policy denotation and cannot enlarge authority by path choice. The design complements—rather than relabels—Skill scanning, permissions, verified loading, behavioral monitoring, SBOM/provenance, and general commit-time authorization. The external snapshot audit provides structural motivation for conservative local closure in current public Skill distributions and a concrete baseline for validating runtime completeness across adapters and deployments.

A Notation and Trust Summary

Table 8. Core notation.

Symbol	Meaning
$\mathcal{G}_t(s; \pi, c_\pi, B)$	Typed transitive capability graph under profile/canonicalizer (π, c_π) and candidate effects B
$q_t(v)$	Canonical authenticated evidence record for node v
$\rho_t(s; \pi, c_\pi, B)$	Root-, context-, and endpoint-sensitive SCC/Merkle graph commitment
γ	Trusted correlated task/closure/effect grant
$\mathcal{N}(x, S)$	Partial trusted surface-to-effect normalizer
$A_{\text{cur}}(t, e)$	Current task, closure, state, and purpose/provenance meet
W_{tp}	Immutable prepared joint witness and reservation binding
$\equiv_\Omega$	External-effect equivalence under observation map Ω

Table 9. Trusted and untrusted inputs.

Input	Trust treatment	Failure behavior
Skill name/description/prose	Untrusted discovery/instruction data	Cannot mint authority
Skill manifest/lockfile	Evidence only; authenticated if possible	Missing/unknown material edge denies exact profile
Supply-chain attestation	Authenticated identity/provenance evidence	Invalid/stale evidence denies
Task/purpose certificate	Trusted issuer input	Missing/ambiguous denies or asks for clarification
Tool/service schema	Versioned closure/effect-contract node	Drift requires new grant
Effect adapter	TCB, versioned and closure-bound	Unknown/ambiguous normalization denies
Receiver receipt	Authenticated receiver report unless stronger premise	Uncertain external state remains indeterminate

B Proof Detail

B.1 Correlated clauses

Let a finite effect clause be

$$c = \langle \text{taskAtom}, \text{purpose}, \text{recipient}, \text{op}, \text{target}, \text{dataClass}, \text{amountCeiling}, \\ \text{argsDigest}, \text{postconditionContract}, \text{effectContractVersion}, \\ \text{durability}, \text{idempotencyScope} \rangle.$$

Its denotation is the set of normalized effects satisfying every field together. The effect’s authoritative `preStateDigest` is bound separately by (A5), while grant validity and epoch maps are bound by (A1)–(A3). The formal clause language permits exact values, finite sets, prefix patterns, and closed numeric intervals with an explicit bottom; this language is closed under exact field intersection. The executable kernel instantiates exact-valued fields and a nonnegative amount ceiling over already compiled correlated clauses; it does not implement general policy intersection or dominance elimination. A deployment using a richer language must retain the exact symbolic conjunction or fail closed whenever an intersection is not representable. For two policies C_1, C_2 , define

$$C_1 \otimes C_2 = \text{Min}\{c_1 \cap c_2 : c_1 \in C_1, c_2 \in C_2, [[c_1 \cap c_2]] \neq \emptyset\},$$

where `Min` removes exact duplicates and clauses whose denotation is contained in another retained clause with identical authority provenance.

LEMMA 4 (RELATIONAL MEET SOUNDNESS).

$$[[C_1 \otimes C_2]] = [[C_1]] \cap [[C_2]].$$

PROOF. If e is in the left side, it satisfies some nonempty pairwise intersection and therefore both parent clauses. If e is in both parent denotations, it satisfies some $c_1 \in C_1$ and $c_2 \in C_2$, hence their nonempty intersection before dominance elimination. Removing a subset clause does not remove e because a retained dominating clause contains it. $\square$

Field-wise union lacks this result because it can construct tuples present in neither parent relation. This is the clause-splicing counterexample in the artifact.

B.2 Induction over nested dependencies

Fix a normalized effect e and its authenticated influence subgraph $\mathcal{I}_{\mathcal{G}}(e)$. Let $P(X)$ state that every mediated realization of e reachable through SCC X is bounded by $\gamma.A$ and by the meet of every applicable contract on all X -to-sink influence paths. SCC condensation gives a DAG. At a sink component, the trusted adapter normalizes the effect and (A4) checks its local applicable contracts, so $P(X)$ holds. Assume P for every successor component. The current component's exact outgoing edges, original endpoints, successor components, and applicable contract nodes are committed by ρ ; the compiler forms their relational meet without field splicing, and the same commit predicate mediates each dispatch. Thus the reachable effect set is a subset of the accumulated meet, establishing $P(X)$. Induction over the condensation DAG proves Theorem 3; an internal cycle is treated as one SCC and all of its applicable contracts enter the same meet.

C Evaluation Configuration and Availability

The evaluated snapshot uses the following disclosed configuration:

Mechanism evidence.

Forty represented lifecycle fixtures, 18 kernel contracts, six mechanism profiles (one full and five reduced), and six single-point semantic mutants.

Bounded exploration.

Breadth-first enumeration to a fixed point with no depth cutoff over the finite state variables specified in Section 8; the full profile reaches 84,608 states and 530,752 transitions.

Local cost.

A single-process, in-memory measurement on the disclosed Apple M4 Max host, excluding network, resolution I/O, signatures, durable transactions, model inference, and settlement.

External structure.

A lexical audit of the Anthropic Skills snapshot at commit 3b3fad96 and the OpenAI Plugins snapshot at commit 1e285826, using the inclusion and path-resolution rules stated in Section 8.

The arXiv source bundle contains the manuscript and bibliography; the evaluated implementation, private source layout, machine-readable traces, and internal verification commands are retained internally. The manuscript discloses the methods, denominators, aggregate results, and limitations, while independent execution requires access to the reviewed implementation beyond that bundle.

D Claim Checklist

- Every “all,” “zero,” and state count refers only to its declared finite fixture, mutation, or bounded domain.
- Mechanism profiles are not reported as measurements of cited systems.
- Preprints are cited as contemporaneous work.
- Exact identity does not imply benignness or behavioral equivalence.
- Commit authorization is not receiver truth or exactly-once settlement.
- Unresolved dependencies and effects fail closed; they are not omitted.
- The two-source static audit supports repository-structure claims; cross-runtime utility, ecosystem coverage, and production overhead require separate evaluation.

References

- [1] Agent Skills Project. 2026. Adding Agent Skills Support to an Agent. <https://agentskills.io/client-implementation/adding-skills-support>. Accessed 2026-08-29.
- [2] Agent Skills Project. 2026. Agent Skills Specification. <https://agentskills.io/specification>. Accessed 2026-08-29; specification repository snapshot 69ef37e9424c0a7ea9dd2293b559e43ec8176379.

- [3] Anthropic. 2026. Skills: Public Repository for Agent Skills. <https://github.com/anthropics/skills/tree/3b3fad96af16a10759d930941b4520ba0c40edae>. Snapshot commit 3b3fad96af16a10759d930941b4520ba0c40edae, 2026-08-21; accessed 2026-08-29.
- [4] Sanket Badhe and Priyanka Tiwari. 2026. Agent Skill Security: Threat Models, Attacks, Defenses, and Evaluation. <https://doi.org/10.48550/arXiv.2607.13987> arXiv:2607.13987 [cs.CR]
- [5] Lujio Bauer, Jay Ligatti, and David Walker. 2005. Composing Security Policies with Polymer. In *Proceedings of the 2005 ACM SIGPLAN Conference on Programming Language Design and Implementation*. Association for Computing Machinery, New York, NY, USA, 305–314. <https://doi.org/10.1145/1064978.1065047>
- [6] Varun Pratap Bhardwaj. 2026. Formal Analysis and Supply Chain Security for Agentic AI Skills. <https://doi.org/10.48550/arXiv.2603.00195> arXiv:2603.00195 [cs.CR] Version 2, revised 2026-08-05.
- [7] Zheng Chen, Hanqing Liu, Duling Xu, Dong Dong, Jialin Li, Bangzheng Pu, and Jidong Zhai. 2026. Cordon: Semantic Transactions for Tool-Using LLM Agents. <https://doi.org/10.48550/arXiv.2606.17573> arXiv:2606.17573 [cs.OS]
- [8] Joseph W. Cutler, Craig Disselkoen, Aaron Eline, Shaobo He, Kyle Headley, Michael Hicks, Keshia Hietala, Eleftherios Ioannidis, John Kastner, Anwar Mamat, Darin McAdams, Matt McCutchen, Neha Rungta, Emina Torlak, and Andrew M. Wells. 2024. Cedar: A New Language for Expressive, Fast, Safe, and Analyzable Authorization. *Proceedings of the ACM on Programming Languages* 8, OOPSLA1, Article 118 (April 2024), 28 pages. <https://doi.org/10.1145/3649835>
- [9] Jayanaka Dantanarayana, Savini Kashmira, Lingjia Tang, and Jason Mars. 2026. SIGIL: Compiling Agent Skills into Typed Harnesses. <https://doi.org/10.48550/arXiv.2607.27309> arXiv:2607.27309 [cs.SE] Version 2, revised 2026-08-26.
- [10] Wenbo Guo, Wei Zeng, Chengwei Liu, Xiaojun Jia, Yijia Xu, Lei Tang, Yong Fang, and Yang Liu. 2026. MalSkillBench: A Runtime-Verified Benchmark of Malicious Agent Skills. <https://doi.org/10.48550/arXiv.2606.07131> arXiv:2606.07131 [cs.CR]
- [11] Zimo Ji, Congying Xu, Zongjie Li, Yudong Gao, Xin Wei, Shuai Wang, and Shing-Chi Cheung. 2026. Cloak and Detonate: Scanner Evasion and Dynamic Detection of Agent Skill Malware. <https://doi.org/10.48550/arXiv.2607.02357> arXiv:2607.02357 [cs.CR]
- [12] Changguo Jia, Tianqi Zhao, Runzhi He, and Minghui Zhou. 2026. Skills Are Not Islands: Measuring Dependency and Risk in Agent Skill Supply Chains. <https://doi.org/10.48550/arXiv.2607.01136> arXiv:2607.01136 [cs.SE]
- [13] Tu Lan and Chaowei Xiao. 2026. Runtime Skill Audit: Targeted Runtime Probing for Agent Skill Security. <https://doi.org/10.48550/arXiv.2606.11671> arXiv:2606.11671 [cs.CR]
- [14] Ying Li, Yanju Chen, Hongbo Wen, Bosi Zhang, Hanzhi Liu, Peiran Wang, Yu Feng, and Yuan Tian. 2026. VIGIL: Runtime Enforcement of Behavioral Specifications in AI Agent Skills. <https://doi.org/10.48550/arXiv.2606.26524> arXiv:2606.26524 [cs.CR]
- [15] Zhiyuan Li, Jingzheng Wu, Xiang Ling, Xing Cui, and Tianyue Luo. 2026. Towards Secure Agent Skills: Architecture, Threat Taxonomy, and Security Analysis. <https://doi.org/10.48550/arXiv.2604.02837> arXiv:2604.02837 [cs.CR]
- [16] Linux Foundation SPDX Project. 2025. SPDX Specification 3.0.1: Relationship. <https://spdx.github.io/spdx-spec/v3.0.1/model/Core/Classes/Relationship/>. Accessed 2026-07-25.
- [17] Xinyu Liu, Yukai Zhao, Xing Hu, and Xin Xia. 2026. Exploiting LLM Agent Supply Chains via Payload-less Skills. <https://doi.org/10.48550/arXiv.2605.14460> arXiv:2605.14460 [cs.CR]
- [18] Yi Liu, Weizhe Wang, Ruitao Feng, Yao Zhang, Guangquan Xu, Gelei Deng, Yuekang Li, and Leo Zhang. 2026. Agent Skills in the Wild: An Empirical Study of Security Vulnerabilities at Scale. <https://doi.org/10.48550/arXiv.2601.10338> arXiv:2601.10338 [cs.CR]
- [19] Microsoft. 2026. OpenAPM v0.1. <https://microsoft.github.io/apm/specs/openapm-v01/>. Editor’s Working Draft; accessed 2026-07-25.
- [20] Model Context Protocol Project. 2026. Authorization and Security Best Practices. <https://modelcontextprotocol.io/specification/2026-07-28/basic/authorization>. Specification revision 2026-07-28; accessed 2026-08-29.
- [21] Model Context Protocol Project. 2026. Model Context Protocol 2026-07-28: Key Changes. <https://modelcontextprotocol.io/specification/2026-07-28/changelog>. Accessed 2026-08-29.
- [22] Bardia Mohammadi, Nearchos Potamitis, Lars Klein, Akhil Arora, and Laurent Bindschaedler. 2026. Atomix: Timely, Transactional Tool Use for Reliable Agentic Workflows. <https://doi.org/10.48550/arXiv.2602.14849> arXiv:2602.14849 [cs.LG]
- [23] OpenAI. 2026. OpenAI Plugins: Curated Codex Plugin Examples. <https://github.com/openai/plugins/tree/1e285826e604f66f7208f7ac4dba0fe8341d1f57>. Snapshot commit 1e285826e604f66f7208f7ac4dba0fe8341d1f57, 2026-08-28; accessed 2026-08-29.
- [24] OpenSSF SLSA Community. 2026. SLSA Build: Verifying Artifacts. <https://slsa.dev/spec/v1.2/verifying-artifacts>. Accessed 2026-07-25.
- [25] OpenSSF SLSA Community. 2026. SLSA Specification, Version 1.2. <https://slsa.dev/spec/v1.2/>. Accessed 2026-07-25.

- [26] OWASP Foundation. 2025. CycloneDX Bill of Materials Standard. <https://cyclonedx.org/specification/overview/>. Accessed 2026-07-25.
- [27] Nils Palumbo, Sarthak Choudhary, Jihye Choi, Guy Amir, Prasad Chalasani, and Somesh Jha. 2026. Formal Policy Enforcement for Real-World Agentic Systems. <https://doi.org/10.48550/arXiv.2602.16708> [cs.CR]
- [28] Shidong Pan, Xiaoyu Sun, Tianyi Zhang, Dianshu Liao, Kaiwen Yang, and Zhenchang Xing. 2026. SkillGuard: A Permission-Centric Framework for Agent Skill Security. <https://doi.org/10.48550/arXiv.2606.03024> [cs.CR]
- [29] Jaehong Park and Ravi Sandhu. 2004. The UCONABC Usage Control Model. *ACM Transactions on Information and System Security* 7, 1 (2004), 128–174. <https://doi.org/10.1145/984334.984339>
- [30] Suli Qiu, Haomin Zhuang, Yujun Zhou, Yufei Han, and Xiangliang Zhang. 2026. AIRGuard: Guarding Agent Actions with Runtime Authority Control. <https://doi.org/10.48550/arXiv.2605.28914> [cs.CR]
- [31] Shoumik Saha, Kazem Faghih, and Soheil Feizi. 2026. Under the Hood of SKILL.md: Semantic Supply-Chain Attacks on AI Agent Skill Registry. <https://doi.org/10.48550/arXiv.2605.11418> [cs.AI]
- [32] Jerome H. Saltzer and Michael D. Schroeder. 1975. The Protection of Information in Computer Systems. *Proc. IEEE* 63, 9 (1975), 1278–1308. <https://doi.org/10.1109/PROC.1975.9939>
- [33] Igor Santos-Grueiro. 2026. Temporary Authority, Permanent Effects: Commit-Time Authorization for LLM Agents. <https://doi.org/10.48550/arXiv.2607.10487> [cs.CR]
- [34] Fred B. Schneider. 2000. Enforceable Security Policies. *ACM Transactions on Information and System Security* 3, 1 (2000), 30–50. <https://doi.org/10.1145/353323.353382>
- [35] Tingda Shen, Yebo Feng, Konglin Zhu, Xiaojun Jia, Yang Liu, and Lin Zhang. 2026. Sealing the Audit-Runtime Gap for LLM Skills. <https://doi.org/10.48550/arXiv.2605.05274> [cs.CR]
- [36] The Update Framework Project. 2026. The Update Framework Specification. <https://theupdateframework.github.io/specification/latest/>. Accessed 2026-07-25.
- [37] Guiyao Tie, Jiawen Shi, Pan Zhou, and Lichao Sun. 2026. BadSkill: Backdoor Attacks on Agent Skills via Model-in-Skill Poisoning. <https://doi.org/10.48550/arXiv.2604.09378> [cs.CR]
- [38] Santiago Torres-Arias, Hammad Afzali, Trishank Karthik Kuppusamy, Reza Curtmola, and Justin Cappsos. 2019. in-toto: Providing Farm-to-Table Guarantees for Bits and Bytes. In *28th USENIX Security Symposium*. USENIX Association, Santa Clara, CA, 1393–1410. <https://www.usenix.org/conference/usenixsecurity19/presentation/torres-arias>
- [39] Jiangrong Wu, Yuhong Nan, Yixi Lin, Huaijin Wang, Yuming Xiao, Shuai Wang, and Zibin Zheng. 2026. SkillScope: Toward Fine-Grained Least-Privilege Enforcement for Agent Skills. <https://doi.org/10.48550/arXiv.2605.05868> [cs.CR]
- [40] Yi Xie, Jiawei Du, Yu Cheng, Juan Zhou, and Zhaoxia Yin. 2026. Benign in Isolation, Harmful in Composition: Security Risks in Agent Skill Ecosystems. <https://doi.org/10.48550/arXiv.2606.15242> [cs.CR]
- [41] Zhonghao Zhan and Hamed Haddadi. 2026. Auto-Policy, not Auto-Skill: Compiled Agent Skills for the Physical World. <https://doi.org/10.48550/arXiv.2608.25091> [cs.AI]
- [42] Ziling Zhou. 2026. Governing Dynamic Capabilities: Cryptographic Binding and Reproducibility Verification for AI Agent Tool Use. <https://doi.org/10.48550/arXiv.2603.14332> [cs.CR]
- [43] Genliang Zhu and Chu Wang. 2026. Explanation-Bound Tool Execution for AI Agents: Server-Verified Action Claims Without Trusting Model Rationales. <https://doi.org/10.48550/arXiv.2607.25364> [cs.AI]
- [44] Genliang Zhu and Chu Wang. 2026. Intent-Governed Tool Authorization for AI Agents. <https://doi.org/10.48550/arXiv.2606.22916> [cs.AI]
- [45] Genliang Zhu, Chu Wang, Ziyuan Wang, Zhida Li, and Qiang Li. 2026. OpenPort Protocol: A Security Governance Specification for AI Agent Tool Access. <https://doi.org/10.48550/arXiv.2602.20196> [cs.CR]
- [46] Haomin Zhuang, Hanwen Xing, Yujun Zhou, Yuchen Ma, Yue Huang, Yili Shen, Yufei Han, and Xiangliang Zhang. 2026. AgentTrap: Measuring Runtime Trust Failures in Third-Party Agent Skills. <https://doi.org/10.48550/arXiv.2605.13940> [cs.CR]