GOVERNANCE ENVELOPES · V0.1 PUBLIC DRAFT

Governance Envelopes for Verifiable Agent Handoff.

A cross-vendor evidence graph that supports chain-of-custody review for governed agent context, citations, memory, handoff, and action.

AUTHOR

Nicholas Allen

Sanna AI, Inc.

PUBLISHED

May 8, 2026

BUILDS ON

Sanna Protocol v1.5 receipt primitives

DOCUMENT ID

SANNA-WP-001 · v0.1

ABSTRACT

A portable, signed evidence graph for agent context, memory, citation, handoff, and action.

A dominant operational frame for agent governance today is action-level enforcement: deciding which tool calls, API requests, and external actions an agent is allowed to perform. Action-level enforcement is necessary, but it is not sufficient.

Agents do not only act — they retrieve, remember, cite, and synthesize before they act. Each of those operations admits or produces information that the next action depends on. In a multi-agent system, one agent's output becomes another agent's context, and the trust assumptions compound silently across the chain.

This paper proposes **Governance Envelopes**: portable, signed bundles of receipts that travel with agent output and describe the instrumented governance events associated with producing it. An envelope binds the output to receipts for tool visibility, context admission, citation binding, memory decisions where present, and action enforcement, and it covers the full bundle under a single signature so that post-signing tampering and omission from the signed bundle are detectable. Downstream agents perform **Receipt-Gated Handoff** — they verify the envelope before admitting the upstream output as context.

Cryptographic verification can be local: it requires the envelope, the relevant public key material, and the consumer's local trust policy; broader trust evaluation may also require key distribution, revocation, parent resolution, and policy-profile infrastructure.

The model is deliberately narrow. Sanna does not inspect the model's private chain-of-thought, does not adjudicate semantic truth, and does not make probabilistic systems deterministic. It governs configured surfaces around the model — deterministic checks where participating systems admit, emit, persist, cite, hand off, or authorize information. The result is a portable evidence graph for instrumented context, memory, citation, handoff, and action events. That graph supports chain-of-custody review whenever the party relying on or reviewing an agent output did not directly observe the governed run.

Contents

APPENDICES

DRAFT	Governance Envelopes · v0.1 Public Draft
BUILDS ON	Sanna Protocol v1.5 receipt primitives
PUBLISHED	May 8, 2026
AUTHOR	Nicholas Allen, Sanna AI, Inc.

§ 01 — SECTION

Introduction.

Agent systems are moving from single-agent tool use to multi-agent workflows. A research agent hands findings to a planning agent; a planning agent hands a plan to an execution agent; an execution agent calls tools that may themselves be other agents.

Vendors are converging on standards for declaring tools, brokering identity, and routing requests. Cloud providers are racing to ship agent platforms.

Among the most mature controls in this transition are **egress** controls: gateways that govern outbound tool calls, identity systems that bind agent sessions to principals, and policy engines that decide whether a given action may proceed. These controls are real, useful, and necessary. Sanna's own protocol began with action-level enforcement and continues to invest in it.

The unsolved interoperability problem is **portable, independently verifiable evidence of governed information flow across observation boundaries** — what entered an agent through participating governance surfaces, what crossed between agents, and what survived across sessions. An observation boundary appears whenever the reviewer, recipient, auditor, regulator, counterparty, or downstream agent did not directly observe the governed run. Cross-vendor handoff is the highest-friction version of that pattern, but the same structure appears inside a single organization whenever one team must review an agent run another system executed.

> Today, when Agent A passes output to Agent B, the handoff is a runtime-level message, state update, task, artifact, or tool-mediated transfer. The trust assumption is that A was governed; the verification is that the runtime delivered a payload from an accepted source.
>
> — THE INTEROPERABILITY GAP

Current frameworks and protocols may authenticate the sender, structure the payload, filter context, or log the transition. They generally do not carry a portable, independently verifiable record of which governed external context was admitted to A through instrumented channels, whether that context was authorized,

whether A's citations corresponded to admitted source records, whether A's memory crossed scope boundaries, or whether any agent upstream of A introduced ungoverned material into the evidence graph.

There is no generally adopted portable artifact B can inspect to verify these properties independently of the runtime that delivered them.

This paper proposes a framing in which the handoff is a signed bundle, the trust assumptions are explicit, and cryptographic verification can be local.

§ 02 — SECTION

Illustrative incident.

The following sequence was observed in a single multi-turn session with a frontier-tier language model. The vendor and model are intentionally not named; the failure modes are not unique to any single product.

The session was a real working session, not an adversarial test. In the course of approximately one hour, the model:

1. **Fabricated a competitive landscape.** Asked to map the competitive position of a category, the model produced a confidently structured answer that included company categories that do not exist as named segments in any public market analysis, and it placed real companies into invented categories with confidence-toned commentary.
2. **Coined terminology and presented it as established.** Several phrases the model used as terms-of-art are not in common use; they appear to have been generated on the fly and then referenced later in the session as though they were canonical.
3. **Bled context from an unrelated prior session.** Mid-session, the model began grafting concepts and entity names from a different topic onto the active topic. The graft was not flagged. The model proceeded as though the foreign material belonged to the current discussion.
4. **Invented citations.** When pressed for sources, the model produced citations to documents and URLs that were never retrieved during the session and, on inspection, do not exist in the form cited.
5. **Confabulated a justification when corrected.** When the citation problem was surfaced, rather than acknowledging the fabrication, the model produced a plausible-sounding explanation of why it had cited what it cited — itself a generated artifact, indistinguishable in tone from a real account.

Several of these failures map directly to structural chokepoints that a governance layer can occupy:

OBSERVED FAILURE	STRUCTURAL CHOKEPOINT
Fabricated competitive landscape	**Citation binding** — no source verification before publication
Invented terminology presented as canonical	**Semantic constraint** — no terminology governance bound to the output
Cross-session context bleed	**Context admission** — no scope verification before model invocation or context insertion
False citations	**Citation binding** — citations not anchored to retrieval results
Retroactive confabulation	**Compounding effect of ungoverned upstream context** — there was no authoritative record of what had actually been admitted, so post-hoc justification could not be falsified by the system

The terminology example illustrates a possible future semantic-governance layer, not a property delivered by the five producer-side chokepoints and consumer-side handoff verification in this paper. Governance Envelopes can help reviewers reconstruct which instrumented sources and controls were present, but they do not prove that generated terminology is established or that an admitted source semantically supports a generated claim.

The relevant observation is not that a model hallucinated. Hallucination is well-documented and not, by itself, a governance failure — models are probabilistic. The observation is that several failures occurred at definable boundaries: retrieval entered, citations were attached, memory was carried forward, justifications were produced. None of those boundary events required inspecting the model's private reasoning. Each was, in principle, interceptable by a deterministic check at a configured governance surface.

> The relevant observation is not that a model hallucinated. The observation is that the failures occurred at *definable boundaries* — each, in principle, interceptable by a deterministic check.
>
> — STRUCTURAL READING OF THE INCIDENT

This incident is illustrative, not empirical proof of the full cross-vendor architecture. The motivating evidence for Governance Envelopes is structural: once agent output crosses an observation boundary, the receiving party needs a portable artifact it can verify without having observed the run.

§ 03 — SECTION

Problem statement.

Multi-agent handoff today is **runtime-mediated transfer**: a message, state update, task, artifact, tool-call result, or graph transition moves from Agent A to Agent B. The transfer may be authenticated, structured, filtered, routed, and logged by the framework or platform.

What it typically does not provide is a portable governance artifact that lets B verify how A's output was produced.

This pattern can be sufficient for single-vendor, single-trust-domain workflows where every agent in the chain runs under one operator and the operator accepts the aggregate risk. It becomes insufficient when the workflow crosses an observation boundary and the receiving party needs independently verifiable evidence. Specifically, Agent B has no generally adopted portable way to verify from the handoff artifact alone:

- **Which governed context Agent A received.** Which retrieved chunks were admitted into A's model invocation? Which were rejected? On what policy?
- **Whether that context was authorized.** Were the retrieved sources within the scope A was permitted to operate in? Did A pull from a customer, matter, patient, or project that the requesting principal has no claim to?
- **Whether cited sources corresponded to admitted source records.** Were the citations A produced bound to documents A actually retrieved and admitted, or are they post-hoc decorations attached to model output?
- **Whether memory crossed scope boundaries.** Did A use persistent context that originated in a different session, customer, or matter?
- **Whether the action was governed.** Did A's tool calls run through an enforcement surface, or were they free-form?
- **Whether any upstream agent introduced ungoverned material into the evidence graph.** If A called another agent that called another agent, did the workflow remain governed across the instrumented hops the consumer requires?

In regulated environments and cross-company workflows, these questions quickly become diligence questions: the questions a reviewer, auditor, counterparty, regulator, or incident-response team is likely to ask once agent output becomes operationally significant. Without a portable answer, the operator is left to reconstruct the evidence manually, or to accept risk that cannot be bounded from the handoff artifact itself.

§ 04 — SECTION

Threat model.

The threats this paper addresses are not adversarial nation-state attacks against models. They are the everyday failure modes that arise when probabilistic systems are connected to one another without governance at the joints.

The most important adversary in any signed-evidence system is the signer itself: a dishonest producer, a compromised runtime, a compromised signing key, or a nonconformant verifier can emit valid-looking artifacts that tell an incomplete or misleading story. Governance Envelopes reduce ambiguity only under explicit trust and coverage assumptions; they do not eliminate counterparty due diligence.

Sixteen threat surfaces

T01

CONTEXT BLEED ACROSS SCOPE BOUNDARIES

Projects, customers, patients, matters, sessions, tenants. The most common form is unintentional: a shared retrieval index, a stale memory, an over-broad query.

T02

FABRICATED CITATIONS

References attached to model output that do not correspond to retrieved sources.

T03

STALE OR REVOKED SOURCE MATERIAL

Content that was authoritative at retrieval time but has since been corrected, withdrawn, or de-authorized. Detection of revocation requires an external revocation feed at verification time.

T04

UNAUTHORIZED RETRIEVAL

RAG pulls that exceed the scope the requesting principal is entitled to, including over-broad similarity searches that surface neighbors from other tenants.

T05

TOOL VISIBILITY DRIFT

The set of tools an agent can see at discovery time changing silently between sessions, or differing across surfaces — an agent that "shouldn't know" a tool exists discovering it through a manifest.

T06

MEMORY WRITEBACK CONTAMINATION

Content persisted into long-term memory that should not have been retained, or that crosses a scope boundary on the next session.

T07

DOWNSTREAM RELIANCE ON UNGOVERNED OUTPUT

Agent B treating Agent A's output as ground truth without any check on how A produced it.

T08

RECEIPT TAMPERING OR SELECTIVE OMISSION

An upstream actor presenting a partial or modified set of evidence that misrepresents the governance applied. Envelope signatures detect tampering with or omission from an already-signed envelope; omission before envelope creation is a trust, coverage, and consumer-policy problem.

T09

PLATFORM BOUNDARY MISMATCH

An envelope produced under one vendor's policy being consumed by an agent operating under a different vendor's policy, with no shared verification primitive.

T10

COMPROMISED OR DISHONEST PRODUCER

A producer or signing key emitting envelopes that are syntactically valid but incomplete, misleading, or generated under a weak policy. A signature attributes the artifact to a key; it does not prove the producer was honest.

T11

UNMEDIATED PATH BYPASS

An agent, developer, or tool route admits context, calls a tool, reads memory, or writes memory outside the governed path. Envelopes can reveal missing required roles when the consumer requires them; they cannot reveal invisible bypass by a trusted producer.

T12

POLICY SUBSTITUTION

A producer asserts a known policy hash while executing a weaker policy, a stale policy, or a policy whose semantics the consumer misinterprets. A policy hash is a coordination primitive unless backed by reviewed profiles, registries, or out-of-band trust.

T13

REPLAY ACROSS CONSUMERS AND TIME

A valid envelope, without audience binding, expiry, challenge-response, or nonce, may be presented to the wrong consumer, workflow, tenant, or time window. Replay protection is a higher-assurance profile addition.

T14

PRODUCER-CONSUMER COALITION

Two parties can attest to each other's governed-looking artifacts under weak controls, or a consumer can emit a handoff admission receipt after incomplete verification.

T15

PRODUCER EQUIVOCATION

A producer can show different signed envelopes to different consumers for the same logical run unless transparency logs, shared registries, contractual audit rights, or similar controls constrain split-view disclosure.

T16

AUTHORIZED-BUT-POISONED SOURCE MATERIAL

Poisoned or misleading content may be policy-admissible and therefore produce a clean envelope around bad inputs. Envelopes help trace downstream blast radius; they do not prevent poisoning unless source-authenticity controls exist.

If a signing key is compromised, envelopes signed during the compromise window become suspect evidence. Incident response must treat the affected window as a blast-radius problem: re-verify trust state, review revocation timing, and potentially exclude or quarantine those envelopes.

Sanna does not claim to prevent every form of model error. It claims to provide signed evidence useful for detecting, attributing, and falsifying these structural conditions under stated trust and coverage assumptions.

Trust and coverage assumptions

Governance Envelopes authenticate signed claims about **instrumented governance events**. They do not, by themselves, prove that the producer was honest, that every relevant event was observed, that the

underlying policy was correct, or that the output is semantically true. The model depends on several explicit assumptions:

- The relevant context, tool, memory, citation, handoff, and action paths are mediated by Sanna or another compatible governance surface.
- Producers protect signing keys, rotate them when needed, and expose revocation or trust-state changes appropriate to the assurance level being claimed.
- Consumers maintain local trust policy for issuers, keys, accepted receipt roles, policy hashes or policy profiles, freshness, and required assurance levels.
- Receipt and envelope schemas have a defined canonicalization profile, conformance tests, and verifier behavior for unsupported versions or extensions.
- Revocation, parent resolution, timestamping, replay protection, and source-freshness checks are required only for assurance profiles that claim those properties.

Under those assumptions, an envelope provides portable, tamper-evident process evidence: what the instrumented system says it admitted, rejected, cited, remembered, handed off, and authorized.

§ 05 — SECTION

Definitions.

The following terms are used precisely throughout the rest of this paper.

GOVERNANCE RECEIPT

A signed, tamper-evident record of a single governance event: an enforcement decision, a context admission, a tool manifest, a citation binding, or an equivalent. Sanna's protocol defines a receipt schema with fields including `enforcement_surface`, `invariants_scope`, `parent_receipts`, `constitution_ref`, and a fingerprint that other receipts can reference via `parent_receipts` to form a verifiable chain.

GOVERNANCE ENVELOPE

A portable bundle that wraps an agent output payload together with the set of receipts produced during the production of that payload. The envelope itself is signed; the signature covers the full receipt set.

CONTEXT ADMISSION

The decision, made before a model invocation or context-insertion event, about which retrieved, recalled, or tool-returned content may enter the model's working memory for a given turn. Admission and rejection decisions are represented in the resulting receipt according to the applicable disclosure profile.

CITATION BINDING

The constraint that any source the agent claims must correspond to a source record actually admitted or otherwise authorized for the output. A nonexistent or unadmitted citation is a binding failure. **Whether an admitted source substantively supports the claim remains a separate semantic-review question.**

MEMORY WRITEBACK

The decision about which content from a current session may be persisted for use in future sessions. Writeback is a separate chokepoint from admission; the rules differ.

TOOL MANIFEST

The signed record of which tools the agent was allowed to know existed at session start or after a governed tool-discovery/update event. Sanna's `enforcement_surface` field identifies the surface that produced the manifest; the `com.sanna.manifest` extension carries per-surface visibility detail.

ACTION ENFORCEMENT

The decision about whether a specific action the agent attempted may proceed. This is the action-level governance category that gateways, interceptors, and middleware already address.

HANDOFF VERIFICATION

The check a receiving agent performs on an incoming envelope before admitting its payload as context. The check evaluates the envelope's signature, its required receipt roles, the upstream issuer, and locally-defined admission rules.

GOVERNED EVIDENCE PATH

The set of receipts that, taken together, account for governance decisions observed by participating systems along configured ingress, memory, citation, handoff, and action paths. The phrase is deliberately not "audit trail of reasoning" — Sanna does not inspect reasoning.

RECEIPT-GATED HANDOFF

The pattern in which a downstream agent refuses to admit an upstream output as context unless the accompanying envelope verifies against local handoff admission policy.

§ 06 — SECTION

Architecture: five chokepoints and one verification act.

The architecture rests on five operational chokepoints in the producer's run — tool visibility, context admission, citation binding, memory writeback, and action enforcement — and one verification act in the consumer's run: handoff verification.

The first five emit receipts about a run. The sixth verifies those receipts and emits its own receipt about the handoff. The composition is the evidence graph.

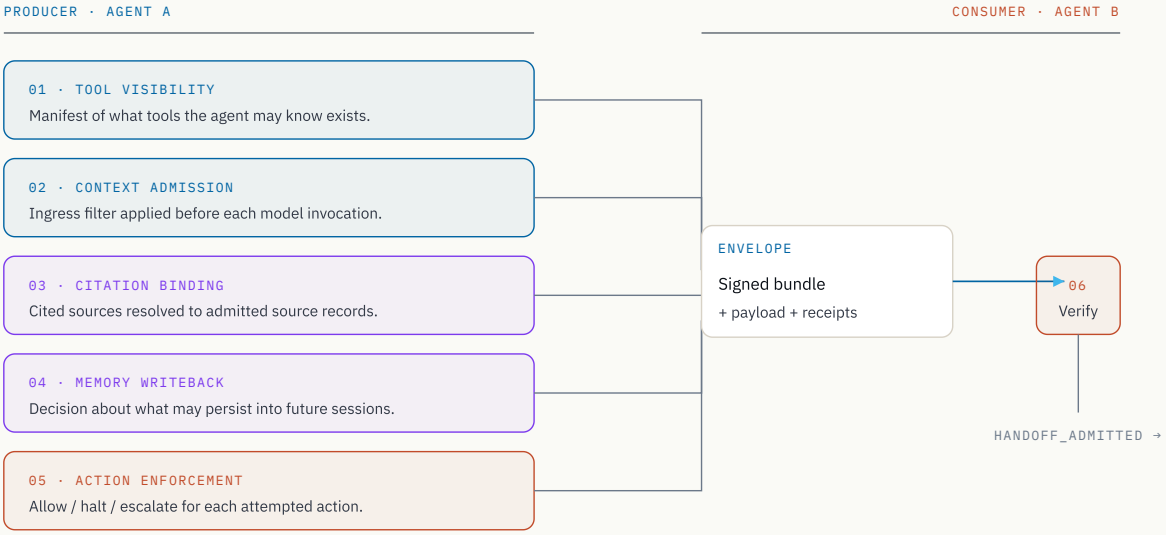

FIG. 01 FIVE PRODUCER-SIDE CHOKEPOINTS FEED A SINGLE ENVELOPE, WHICH A CONSUMER VERIFIES ON RECEIPT.

The controls are not perfectly orthogonal. A retrieval tool can be both an action and a context-ingress event; a browser observation is context returned through a tool; a memory read is context admission for

the current turn; a memory write is both an action and a retention decision for a future turn; citation binding is an output constraint over the admission set. The point of the taxonomy is practical instrumentation, not a mathematically clean security model.

The six surfaces, in detail

01 — Producer

TOOL VISIBILITY / MANIFEST

Before an agent sees a tool, or before a governed dynamic tool-discovery event changes the visible tool set, the manifest fixes which tools the agent is allowed to know exist. A receipt records the delivered set, the suppressed set, and the reasons. In Sanna's protocol today, this is carried as a `session_manifest` event with the `com.sanna.manifest` extension.

02 — Producer

CONTEXT ADMISSION

Before each model invocation or context-insertion event, retrieved, recalled, or tool-returned content is filtered against admission policy. Admitted content enters; rejected content does not. The receipt records both decisions without necessarily exposing sensitive content. `content_mode` already supports the redaction modes needed (`full` , `redacted` , or `hashes_only`).

03 — Producer

CITATION BINDING

When the agent produces output that claims a source, the claim is checked against the admission set. A citation that does not resolve to an admitted source is a binding failure. The decision is recorded as part of the output's receipt.

04 — Producer

MEMORY WRITEBACK

Content the agent attempts to persist for future sessions passes through a writeback gate. The gate decides what may be written, under what scope, with what retention. Writeback is a distinct decision from admission; an item may be admissible to the current session but not eligible for persistence.

05 — Producer

ACTION ENFORCEMENT

When the agent attempts an action, the enforcement surface decides whether it proceeds, halts, escalates, or proceeds with constraints. Sanna's enforcement surfaces — `middleware`, `gateway`, `cli_interceptor`, `http_interceptor` — produce receipts whose `enforcement.action` is one of `allowed`, `halted`, `escalated`, or `warned`.

06 — Consumer

HANDOFF VERIFICATION

When a receiving agent encounters output from another agent, it does not admit the output as context until it has verified the envelope. The verification produces its own receipt — a handoff admission record — whose `parent_receipts` field references the upstream receipts it relied on, while envelope-level lineage is carried by envelope metadata such as `parent_envelopes`.

§ 07 — SECTION

Governance envelope model.

A Governance Envelope is the wire-level container for a governed agent output. It binds a payload to the receipts that account for it, under a single signature.

The structure below is a strawman for v0.1; field names will evolve as the model is implemented.

```
{
  "sanna_envelope": "0.1",
  "envelope_id":    "env_...",
  "created_at":     "2026-05-08T00:00:00Z",
  "producer": {
    "agent_id": "agent-a",
    "issuer":   "example.com",
    "key_id":   "..."
  },
  "payload": {
    "type":         "agent_output",
    "sha256":       "...",
    "content_type": "application/json"
  },
  "receipts": [
    { "role": "tool_manifest",      "fingerprint": "...", "receipt": {} },
    { "role": "context_admission",  "fingerprint": "...", "receipt": {} },
    { "role": "citation_binding",   "fingerprint": "...", "receipt": {} },
    { "role": "action_enforcement", "fingerprint": "...", "receipt": {} }
  ],
  "parent_envelopes": [],
  "signature": {
    "alg":    "Ed25519",
    "key_id": "...",
    "value":  "..."
  }
}
```

LISTING 1 · STRAWMAN ENVELOPE STRUCTURE (V0.1)

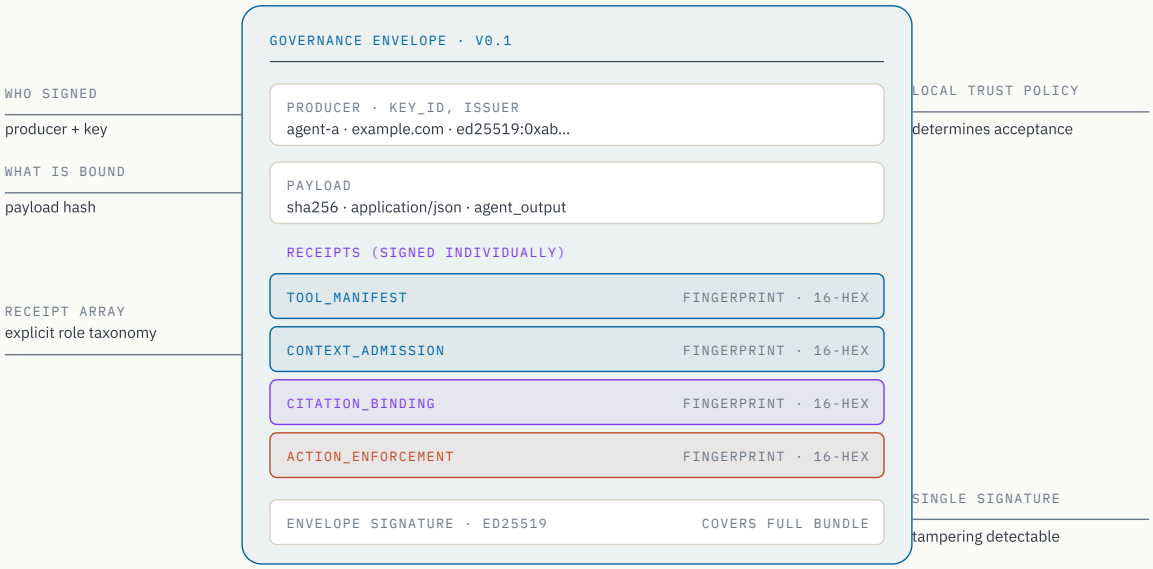

FIG. 02 ENVELOPE ANATOMY: PRODUCER, PAYLOAD, ROLE-TAGGED RECEIPTS, PARENT LINEAGE, SINGLE SIGNATURE.

Properties of this structure

The example is intentionally minimal. A handoff that writes memory would include a `memory_writeback` role; a profile that requires memory attestation can reject envelopes that omit it. Required roles are consumer-policy decisions, not universal schema requirements.

The envelope signature covers the receipt set as presented. A producer cannot remove or mutate a receipt from an already-signed envelope and still claim the envelope is intact — the signature would not verify. The signature provides integrity of the signed bundle and attribution to a key; it does not prove that the producer generated every receipt it should have generated before signing. Completeness is a consumer-policy and instrumentation-coverage property.

The `receipts` array uses an explicit `role` taxonomy. A consumer that requires a `context_admission` receipt can assert presence by role rather than parsing receipt internals. Role-presence checks are syntactic. Consumers that need stronger assurances should pair required roles with accepted policy profiles, content-mode floors, scope-tag requirements, or out-of-band policy review.

The `parent_envelopes` array carries cross-envelope chaining. When an agent produces output that relied on a verified upstream envelope, the producer references the upstream envelope by fingerprint. This is analogous to the existing `parent_receipts` field on individual Sanna receipts, while keeping receipt lineage and envelope lineage as distinct layers.

Several fields are **proposed extensions** beyond the current Sanna v1.5 receipt schema. The `role` taxonomy on receipts, the `parent_envelopes` list, and the envelope signature itself are part of the envelope model rather than the current receipt schema, and would be candidates for future Sanna Protocol proposals if adopted in production SDKs.

The envelope is not intended to replace the individual receipt. A receipt is the unit of attestation for a single governance event. The envelope is the unit of handoff between agents. Both are expected to continue to exist.

§ 08 — SECTION

Verification flow.

A consumer that receives an envelope performs the following sequence before admitting the payload as context. The order matters; later steps assume earlier ones have passed.

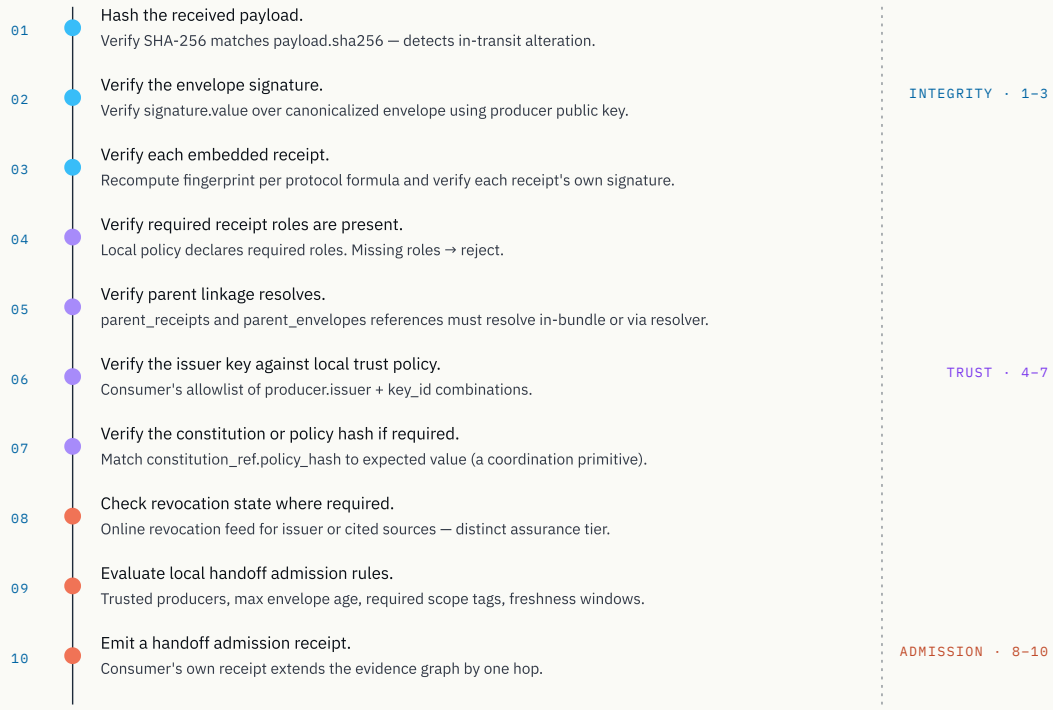

FIG. 03 TEN-STEP VERIFICATION: INTEGRITY (1–3), TRUST (4–7), ADMISSION (8–10).

Sanna's reference SDKs implement the fingerprint computation deterministically — Python and TypeScript produce byte-identical results for the same input.

> Cryptographic verification can be local. Trust establishment may not be.
>
> — DECOUPLING THE SIGNATURE FROM THE ECOSYSTEM

Key discovery, revocation, parent resolution, transparency-log inclusion, and policy-profile interpretation can require online or out-of-band infrastructure. High-assurance handoffs may also require audience binding, request nonces, challenge-response flows, expiry, or trusted timestamps to reduce replay risk; the v0.1 envelope is a portable artifact model, not yet a complete handshake protocol.

§ 09 — SECTION

Context admission.

Context admission deserves separate treatment because it is a chokepoint commonly missed in current architectures, and because the value of downstream chokepoints is bounded by what was admitted.

A permissive admission policy does not break citation binding, memory writeback, or action enforcement; it limits what those downstream checks can meaningfully constrain.

Post-output filtering catches the problem too late

A common pattern in production systems is to scan the model's output for forbidden content and redact or block it after generation. This pattern detects some failure modes, but it cannot detect fabrications grounded in unauthorized context, because the content of the fabrication may itself look benign — the failure is that the agent was informed by context it should not have received.

Context admission is ingress control for the model's working memory

The decision about what enters the model must be made before each model invocation or context-insertion event, including retrieval results, memory reads, and tool outputs that will be returned to the model. Once content is in the prompt or working context, the model has effectively read it; whatever filtering happens to the output is operating on a system that has already been informed by content that should not have been admitted.

> A prompt-assembly or agent-loop pipeline that admits the wrong content and then "tries to remove" it from the output is structurally unsound.
>
> — CONTEXT ADMISSION AS INGRESS CONTROL

Architectures in which retrieval happens through tool calls during generation must wrap each retrieval result, browser observation, database result, or memory read through admission, or accept that some

context enters the model without governance and that the resulting receipt set is correspondingly incomplete.

Disclosure profiles

The receipt for a context admission decision should record what was admitted and what was rejected, with reasons, when that evidence can be safely retained or shared. Sensitive content does not need to be exposed in the receipt; Sanna's `content_mode` already supports `redacted` and `hashes_only` modes that preserve evidentiary value while redacting bodies.

In some deployments, rejected-context evidence may be internal-only, salted, escrowed, or omitted from an external envelope to satisfy privacy, privilege, data-minimization, or data-residency requirements. What matters is that the decision itself can be made attestable at the assurance level the workflow claims.

A consumer that requires high assurance can require, as part of its handoff admission policy, that an upstream envelope include a `context_admission` receipt whose policy hash matches an expected value. The consumer is not auditing the model's reasoning; it is verifying that ingress control was applied under a known policy to the instrumented context paths.

The v0.1 architecture primarily addresses dynamic context: content admitted during a session through retrieval, memory recall, tool returns, or other context-insertion events. **Static framing** — system prompts, persona instructions, in-context examples, and initial conversation history — is not modeled here as a per-turn admission event. Consumers that require attestation of framing should require a separate session-bootstrap manifest or framing-attestation receipt; that is candidate work for a future protocol revision.

§ 10 — Section

Cross-vendor handoff.

Receipts and envelopes are independently verifiable using public key material and a local trust policy. They do not require a vendor API to interpret, and they do not require either side of a handoff to share a runtime platform.

They do, however, require a shared trust framework for producer keys, verifier identities, accepted policy profiles, and assurance levels. In early deployments, that may be bilateral key exchange or enterprise allowlists; in higher-scale deployments, it may be DNS- or well-known-URI key publication, federation operators, industry trust registries, or platform-exported trust metadata.

This is the most consequential property of the model. Consider a workflow that crosses vendors:

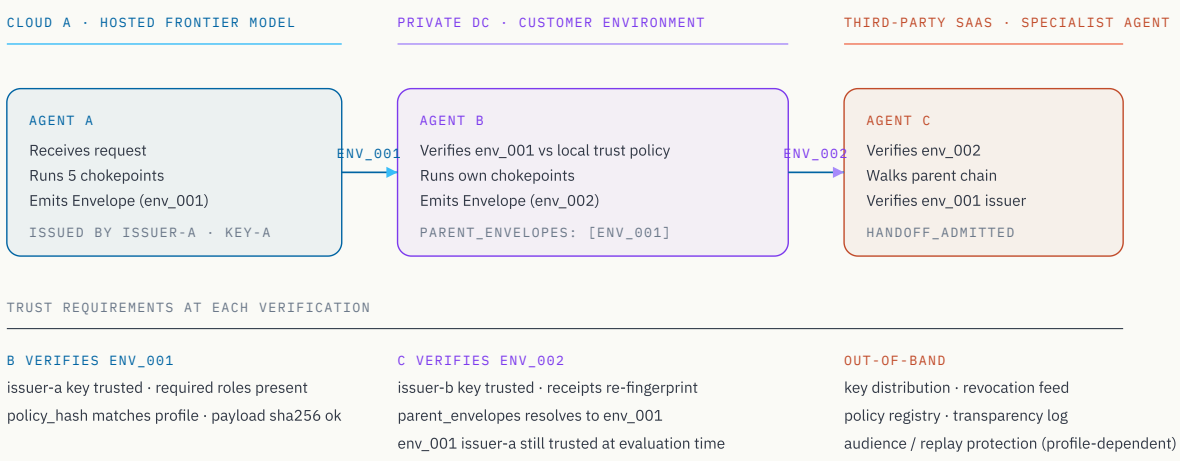

FIG. 04 CROSS-VENDOR ENVELOPE HANDOFF. EACH CONSUMER VERIFIES THE UPSTREAM ENVELOPE, WALKS PARENT_ENVELOPES, AND EMITS ITS OWN.

In a message-passing world, the trust assumptions across this chain are difficult to automate and compare across boundaries. Each hop crosses an organizational boundary, an infrastructure boundary, and frequently a regulatory boundary. The "trust" that makes the workflow function is not actually trust in the cryptographic sense — it is the absence of a verification primitive that would let any party check.

In an envelope-based world, A produces output with a signed envelope that references A's issuer key. B receives the output and verifies the envelope against B's local trust policy, which lists A's issuer key as acceptable for the relevant payload type. C receives B's output with B's envelope (which references A's via `parent_envelopes`) and verifies the evidence graph at its end.

The envelope **travels with the output, not inside a platform**. There need not be a runtime back-channel to the producer. The intended scaling model resembles certificate verification, but with the same implication: useful cross-vendor verification ultimately needs trust stores, key lifecycle management, revocation, transparency, and profile governance.

The cross-vendor verification claim in this paper is structural. Production-scale cross-vendor verification depends on key distribution, revocation, transparency, and trust federation, all of which are discussed as future protocol work in §14. Early deployments will rely on bilateral key exchange, enterprise allowlists, and narrow profile agreements. The federated trust ecosystem described here is a target state, not a current capability of the v0.1 envelope model.

This is distinct from platform-native agent control planes. A cloud or productivity platform can govern agents, tools, data, memory, and logs inside its own operational boundary. Governance Envelopes are proposed for the moment an agent output must leave an observation boundary — another platform, another organization, internal audit, a regulator, a legal reviewer, a counterparty, or a downstream agent — and remain intelligible to a reviewer who did not observe the run. Cross-vendor handoff is the highest-friction case for this property, not the only case; the same observation-boundary problem appears whenever review must happen outside the producer's operational reach.

§ 11 — SECTION

Blast-radius analysis.

When a source is later found to be wrong — corrected, retracted, poisoned, revoked, stale, or determined to have been admitted out of scope — the operationally important question is **what was affected**.

In a system without governance envelopes, this question is usually answerable only by reconstruction. Operational logs and traces can support that reconstruction, but they are usually not portable evidence artifacts designed for cross-boundary evidence-graph queries. Cross-agent traces, if they exist, often lose their evidentiary semantics when the workflow crosses a vendor or organizational boundary.

In a deployment with sufficient instrumentation, durable source identifiers, and indexed envelope storage, the question can become an evidence-graph traversal:

- Which agent runs **admitted** the source? A query over `context_admission` receipts indexed by source identifier.
- Which outputs **cited** it? A query over `citation_binding` receipts indexed by the same identifier.
- Which downstream actions were **linked to it**? A walk along `parent_receipts` and `parent_envelopes` from the citing receipts and envelopes.
- Which customers, matters, or workflows were **in the recorded downstream set**? The `agent_identity`, `human_principal`, and workflow scope on each receipt provide the index.
- Which decisions need **review, rollback, or re-approval**? The intersection of the action receipts in the affected set with the operator's incident response policy.

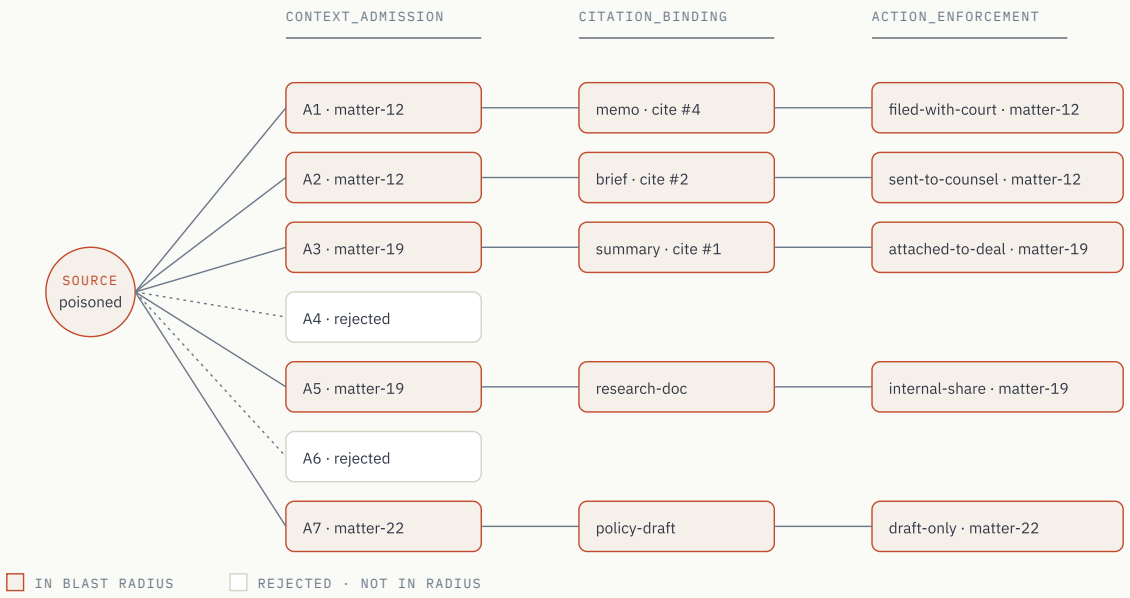

FIG. 05 BLAST-RADIUS TRAVERSAL FROM A SINGLE POISONED SOURCE THROUGH ADMISSION → CITATION → ACTION RECEIPTS.

This resembles **SBOM vulnerability impact analysis**. The Software Bill of Materials concept made it tractable to ask "which of our deployed systems contain version X of this library." Governance envelopes make it tractable to ask "which recorded agent outputs were linked to this source, and which recorded downstream actions followed those outputs."

> The analogy is structural, not complete. SBOM impact analysis is bounded by a binary's static dependencies. Governance Envelope impact analysis is bounded by instrumentation coverage.
>
> — BOUNDED BY WHAT WAS INSTRUMENTED

The query returns the outputs the evidence graph records as having admitted, cited, or linked to the source, not every output the model's behavior was actually influenced by. For incident response, the envelope graph is a starting set, not a closed set. Outputs whose context paths bypassed instrumented gates do not appear in the query and cannot be retrieved by traversing the graph.

The evidence graph also needs an evidence store. The envelope format does not decide where the graph lives, who indexes it, or who can traverse it across organizational boundaries. In single-organization workflows, the graph can live in the operator's evidence store. In cross-organization workflows, partial copies may live with each party, and full traversal may require a federation operator, a contractual audit right, or a lawful data-exchange mechanism.

Temporal queries over the graph rest on producer-asserted timestamps unless envelopes are anchored in a trusted timestamping service, transparency log, or equivalent.

§ 12 — SECTION

High-assurance use cases.

The model is deliberately general. The common pattern is not one regulation or one industry; it is any workflow where an agent's output may be reviewed, accepted, escalated, or relied on by someone who did not directly observe the agent run.

In those settings, the receiving party needs evidence about observed inputs, controls, handoff, and scope before the output can be evaluated responsibly. The trust and coverage assumptions in §4 govern every use case below.

These examples are not claims that Governance Envelopes satisfy any specific regulatory regime by themselves. They are examples where signed evidence of instrumented governance events maps to existing needs for traceability, documentation, oversight, auditability, or counterparty assurance.

12.01

HEALTHCARE

AI-supported clinical decisions, prior authorization, clinical documentation review, revenue-cycle workflows, audit review, and rating-agency or investor diligence may each require evidence about what source material an AI system considered, what material was excluded, and which controls governed the decision pathway.

12.02

INSURANCE

Underwriting decisions, claims adjudication, pricing support, fraud investigation, and loss causality analysis may need defensible records of which instrumented submissions, prior claims, third-party data, and policy documents were admitted or rejected, with explicit boundaries between insureds and claimants.

12.03

FINANCIAL SERVICES

Suitability decisions, credit analysis, underwriting, trade surveillance, model risk management, and supervisory review can use envelopes as supporting evidence for input provenance, policy application, and review workflows when an agent's output informs a subsequent recommendation or decision.

12.04

LEGAL

Document review, legal research, privilege boundaries, conflict checks, and matter walls all depend on controlling which material informs a matter-specific workflow. Envelopes can provide signed evidence that participating retrieval and memory systems enforced matter-boundary controls for a given run.

12.05

LIFE SCIENCES

Drug development, clinical investigations, quality systems, and medical-device workflows already depend on controlled electronic records, audit trails, data integrity, and documented context of use. Agent outputs may need controlled records when they become part of regulated electronic records.

12.06

PUBLIC SECTOR

Benefits eligibility, immigration support, law enforcement analysis, public procurement, and other rights- or safety-impacting workflows may require human oversight, traceability, and review of the data and policy constraints observed by participating systems.

12.07

CRITICAL INFRASTRUCTURE

Energy, transportation, manufacturing, telecommunications, and other operational technology environments increasingly use AI to support reliability, safety, and incident response. When an agent recommends an operational change, an operator may need evidence of which instrumented telemetry, procedures, and threat intelligence were admitted before review or escalation.

12.08

AUDIT & ASSURANCE

Audit workpapers and assurance files depend on clear links among procedures performed, evidence obtained, and conclusions reached. If agents assist with evidence gathering, variance analysis, control testing, or memo drafting, envelopes may become supporting documentation within the review record.

12.09

DATA LICENSING & PROVENANCE

Enterprises and publishers need to review whether licensed, restricted, expired, revoked, or copyrighted material was used to produce an output. Context admission and citation binding — combined with source-state checks and candidate-set commitments — can support evidence that admitted sources were permitted and that excluded sources were evaluated and rejected; stronger exclusion claims require complete instrumentation.

12.10

ENTERPRISE RAG

Multi-tenant retrieval pipelines need evidence that configured admission gates rejected cross-tenant content observed in a given run. Context admission with signed evidence is the right place to enforce and attest this for instrumented retrieval paths.

12.11

INTER-COMPANY AGENT WORKFLOWS

As agents from different organizations begin to coordinate — procurement, supply chain, financial reconciliation, vendor risk review, claims coordination, and future agentic commerce — the basis for trust across the boundary needs to include verifiable artifacts, not only contract clauses. Envelopes are one such artifact; they do not eliminate the need for counterparty diligence.

> The value of an envelope is not that it makes the agent more capable. The value is that it makes *review* of the agent's contribution tractable.
>
> — WHY USE CASES CLUSTER AROUND REVIEW

Across all of these settings, the envelope only attests to events observed by participating governance surfaces. Events that bypass those surfaces — out-of-band tool calls, unmonitored retrieval, side-channel memory writes, manual operator actions — do not appear in the receipt graph, and absence in the envelope is not equivalent to absence in fact.

The envelope supports review; it does not substitute for professional, legal, regulatory, clinical, or audit judgment. Envelopes are technical evidence artifacts. They do not by themselves establish admissibility, compliance, privilege preservation, safe harbor, or legal sufficiency.

Several of these domains — life sciences, public sector, legal, financial services — also impose constraints on what may appear inside the envelope itself. Regulated metadata, personal data, privileged material, and trade-secret context may need to be carried as salted commitments, escrowed audit material, or external references rather than inline values. Privacy and data-residency profiles are tracked in §14 as future protocol work.

§ 13 — SECTION

Non-goals.

Equally important is what the model does not attempt.

- **Sanna does not inspect the model's private chain-of-thought.** The receipt does not contain hidden reasoning, and the verifier does not interpret the model's internal state. Governance is applied at the boundaries the model can be observed at — ingress, egress, retrieval, citation, action — and nowhere else.
- **Sanna does not prove semantic truth or correctness of governed content.** A signed envelope is not a claim that the content is true. It is a claim that the content was produced under specified governance, that its inputs were admitted under specified policy, and that the signed artifact has not been tampered with. Truth-of-content remains a question for substantive review.
- **Sanna does not prove policy substance.** A valid receipt can show that a policy was applied; it does not prove that the policy was meaningful, sufficiently restrictive, correctly written, or appropriate for the workflow. Consumers that care about policy strength must require accepted policy hashes, policy profiles, or out-of-band policy review.
- **Sanna does not prove completeness outside instrumented coverage.** A producer can only attest to governance events observed by participating systems, and consumers must decide which missing roles, weak policies, stale keys, unavailable revocation checks, or unresolved parents cause rejection, quarantine, or degraded trust.
- **Sanna does not provide replay protection or audience binding in v0.1.** The portable artifact model is a one-way attestation. Freshness, replay, audience binding, and challenge-response properties are higher-assurance profile additions and depend on protocol work outside this paper.
- **Sanna does not fully attest static framing in v0.1.** System prompts, persona instructions, in-context examples, and initial conversation history are not modeled here as per-turn admission events. They can be covered by future session-bootstrap or framing-attestation receipts.
- **Sanna does not solve long-term validation by itself.** Audit and regulatory use cases may require envelopes to remain verifiable years after creation. Long-term validation depends on key archival, transparency-log anchoring, format compatibility, and rules for handling cited content that has been deleted, redacted, or revoked.

- **Sanna does not make probabilistic systems deterministic.** Models will continue to vary, hallucinate, and produce non-reproducible outputs. The architecture does not depend on determinism in the model; it depends on deterministic checks at configured chokepoints given fixed policy inputs, source state, and trust-state inputs.
- **Sanna does not replace identity, authorization, DLP, or observability.** Identity systems remain the source of truth for who the principal is. Authorization systems remain the source of truth for what the principal may do. The envelope composes with these systems; it does not subsume them.

> Sanna governs the evidence path around the model, not the model itself.
>
> — THE BOUNDARY OF THE CLAIM

§ 14 — SECTION

Relationship to existing work.

Several adjacent categories are emerging or already established. The relationship is one of composition rather than competition.

Action-level receipts and policy enforcement are an emerging recognized category. Multiple vendors and open-source projects are converging on the idea that an agent's tool calls should pass through a governed surface that can halt, allow, or escalate, and that the decision should produce a record. Sanna's own protocol began here. Action enforcement is necessary infrastructure, and the work being done in this category is real progress.

Gateways, identity systems, and egress controls are necessary infrastructure layers. The agent must be authenticated, the request must be routable, the network call must be inspectable. None of this is in tension with envelopes; envelopes presuppose it.

Evidence packaging layers that bind control results into portable credentials — verifiable credentials, signed audit logs, attestation frameworks — are complementary. The envelope model uses the same primitives (signatures, fingerprints, chained references) and is explicitly designed to coexist. Where an envelope's payload is itself a credential, both can be present.

The contribution of Governance Envelopes is **not** a new cryptographic primitive. The contribution is a **scope** extension: from action-level enforcement to ingress, memory, citation, and cross-agent handoff; and from per-vendor records to a portable, signed bundle that verifies independently.

Platform-native governance will remain important. Hyperscale cloud and productivity platform operators can govern agents, tools, identity, memory, and logs inside their own trust boundaries. Governance Envelopes are aimed at a narrower interoperability gap: exporting, importing, and requiring portable evidence when a governed agent output crosses an observation boundary.

The moat, if the model succeeds, is not secrecy of the cryptographic primitive. It is the ecosystem layer around the primitive: a neutral trust federation, regulated-industry profile governance, verifier conformance suites, SDK adoption, policy-profile registries, enterprise integrations, and operational tooling for storing, searching, redacting, revoking, and resolving envelopes over time. If dominant platforms provide mutually accepted evidence export and import, the standalone opportunity becomes a profile, conformance, and

federation business rather than a primary control plane. If they do not, the neutral envelope layer is the interoperability substrate.

Near-term adoption does not require a mature multi-agent commerce market. It can begin inside organizations before crossing between them, in workflows where opaque native logs are insufficient for review. The commercial trigger is not internal governance alone; it is review across an observation boundary where the receiving party needs portable evidence of what participating governance surfaces observed and signed.

The envelope is most valuable when integrated with existing identity, authorization, DLP, SIEM, policy engines, and observability systems. Without those integrations, it risks becoming duplicative evidence; with them, it becomes the portable artifact those systems can export, import, and retain.

Future protocol work

Several issues belong in future protocol work rather than in this v0.1 thesis paper. They include:

- canonicalization profiles, algorithm agility, signature-suite identifiers, critical-extension handling, and long-term validation;
- producer key distribution, key rotation, revocation freshness, timestamping, replay protection, audience binding, and challenge-response modes for high-assurance handoff;
- evidence-graph consistency rules, cycle detection, depth and fan-in limits, resolver trust, cache behavior, and standardized verifier error codes;
- dynamic manifest deltas, tool-mediated retrieval receipts, multi-role receipts, streaming envelopes, and session-level admission summaries;
- source-state and revocation profiles for admitted sources, including TTLs, revocation reason codes, fail-closed or degraded-mode semantics, and renewable admission attestations for source admissibility;
- candidate-set commitments for negative evidence, so a consumer can reason not only about what was admitted but about what candidates were evaluated and rejected;
- privacy and data-residency profiles, including selective disclosure, salted commitments, escrowed audit material, and external envelopes that minimize sensitive metadata.

§ 15 — SECTION

MVP demonstration.

A minimum viable demonstration of the model can be constructed end-to-end against the existing Sanna protocol, with the envelope structure layered on top.

This MVP is a feasibility test for envelope construction, verification, and rejection inside a controlled stack; a stronger cross-vendor demonstration would use independent producer and verifier implementations, independently managed keys, and separate trust policies.

1. **Set up.** Two retrieval chunks are available to Agent A. Chunk 1 is in scope for the active session (a customer the principal has a relationship with). Chunk 2 is out of scope (a different customer).
2. **Retrieval.** Agent A's retrieval step returns both chunks.
3. **Context admission.** Sanna's context admission gate, configured with the active session's scope policy, admits Chunk 1 and rejects Chunk 2 **before the next model invocation**. A `context_admission` receipt records both decisions; Chunk 2 is recorded by hash only (`content_mode: hashes_only`) so the rejection is attestable without exposing out-of-scope content.
4. **Generation and action.** Agent A produces output. The output passes through citation binding (any cited source must be in the admitted set) and action enforcement.
5. **Envelope emission.** Agent A emits a Governance Envelope binding the output to the four receipts (tool manifest, context admission, citation binding, action enforcement), signed under Agent A's issuer key.
6. **Handoff to Agent B.** Agent B receives the output and the envelope.
7. **Verification.** Agent B runs the verification flow from §8. The envelope verifies. Agent B records the upstream envelope lineage (via `parent_envelopes`) and emits a `handoff_admission` receipt referencing the upstream receipts it relied on before admitting the payload as context. (`handoff_admission` and `handoff_rejected` are proposed `event_type` variants on the existing receipt schema, listed in §14.)
8. **Tamper test.** Re-run with the `context_admission` receipt removed from the envelope. Agent B's verification fails at step 4 (required role missing). Re-run with the receipt present but its payload mutated. Verification fails at step 3 (fingerprint mismatch). Re-run with the envelope signature recomputed over the mutated bundle but using a key not in Agent B's trust policy. Verification fails at step 6.

In each tamper case, Agent B refuses the handoff and emits a `handoff_rejected` receipt explaining which check failed. The evidence of the rejection is itself a signed artifact.

This demonstration exercises the core envelope and handoff path, but it does not yet demonstrate memory writeback, dynamic tool-discovery variants, streaming handoff, market interoperability, or enterprise identity integration. A later interoperability demonstration should prove the same envelope with a separately implemented verifier, independently managed keys, and separately configured trust policy.

§ 16 — SECTION

Conclusion.

Agent trust must compose across observation boundaries. As multi-agent systems move from research demos into production, and as those systems begin to cross organizational, platform, review, and regulatory lines, the unit of trust cannot remain "we both use the same vendor."

Authenticated message passing is not governance evidence. A payload that arrives through an accepted runtime can show who sent it, where it was routed, and sometimes what state accompanied it. It is not, by itself, evidence that the sender was honest, that every event was observed, or that the underlying policy was sound — those remain matters of trust in the producer, of instrumentation coverage, and of out-of-band policy review. What the envelope makes possible is a portable, verifiable record of what the instrumented governance surfaces saw, decided, and signed.

That is a smaller property than full trust, but it is the property runtime-mediated handoff typically lacks as a portable, independently verifiable artifact, and it is the property reviewers, auditors, regulators, and counterparties need before agent output can be evaluated responsibly.

> The primitive this paper proposes is verifiable handoff with a governed evidence path: a portable, signed envelope that binds an agent's output to receipts for configured governance surfaces around the model — tool visibility, context admission, citation binding, memory writeback, action enforcement, and handoff verification — and that a receiving agent can verify under its local trust policy before admitting the output as context.
>
> — THE THESIS, IN FULL

Sanna's existing protocol provides the receipt-level primitives this construction depends on: deterministic fingerprints, Ed25519 signatures, parent-receipt chaining, constitution provenance, and an extension namespace for vendor metadata. The Governance Envelope extends these primitives from per-event records to per-handoff bundles, and from per-vendor records toward cross-vendor verifiable artifacts where trust and profile infrastructure exists.

The result is an evidence graph for **instrumented context admissions, citation bindings, memory decisions, handoff checks, and action decisions** — a governance layer for observable agent events that composes the way the systems it governs already do.

APPENDIX A

End-to-end flow.

A simplified batch handoff. Production agent loops may repeat these gates per model invocation, per tool-returned context insertion, per memory operation, and per handoff.

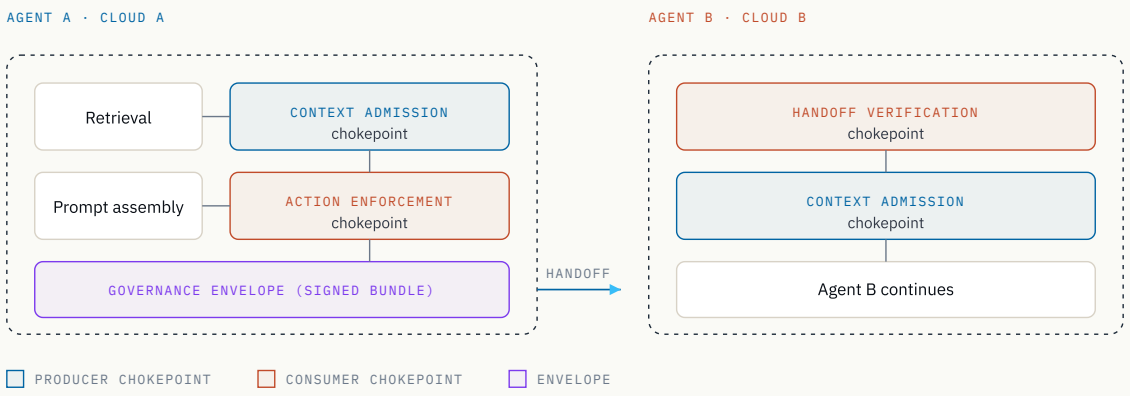

FIG. A END-TO-END BATCH HANDOFF. SHADED BOXES ARE GOVERNED CHOKEPOINTS.

Appendix B

Mapping to existing protocol fields.

The envelope model builds on primitives already present in the Sanna Protocol v1.5 receipt schema. The table below maps envelope concepts to the protocol fields that support them today, and identifies which parts are proposed additions.

ENVELOPE CONCEPT	EXISTING PROTOCOL SUPPORT	PROPOSED EXTENSION
Receipt fingerprint, deterministic across SDKs	`receipt_fingerprint` (16 hex), `full_fingerprint` (64 hex), pipe-delimited fingerprint formula	—
Signed receipts	`receipt_signature` (`scheme: receipt_sig_v1`, Ed25519)	—
Cross-receipt chaining within a session	`parent_receipts: [full_fingerprint, ...]`, AARM R2 verifier	—
Tool manifest as a governance event	`event_type: session_manifest`, `extensions["com.sanna.manifest"]`, `enforcement_surface: mixed`	—
Constitution / policy attestation	`constitution_ref.policy_hash`, `constitution_ref.signature` (`scheme: constitution_sig_v1`)	—
Identity binding for the agent run	`agent_identity` (`agent_session_id`, `human_principal`, `service_account`, `role`, `privilege_scope`) at `checks_version: 10`	—
Content-handling modes for evidentiary records	`content_mode` of `full`, `redacted`, or `hashes_only`	—

ENVELOPE CONCEPT	EXISTING PROTOCOL SUPPORT	PROPOSED EXTENSION
Vendor / extension namespace	`extensions` with reverse-DNS namespacing; `com.sanna.*` reserved	New role IDs (`tool_manifest`, `context_admission`, `citation_binding`, etc.) for envelope-level role taxonomy
Envelope wrapper around multiple receipts	—	`Governance Envelope` (this paper)
Cross-envelope chaining across handoffs	—	`parent_envelopes` array on the envelope
Envelope-level signature covering the bundle	—	Envelope `signature` block (Ed25519, distinct from receipt signatures)
Receipt role taxonomy	Implied by `enforcement_surface` and `event_type` enums; not surfaced as explicit roles	Explicit `role` field on each entry of the envelope's `receipts` array (`tool_manifest`, `context_admission`, `citation_binding`, `memory_writeback`, `action_enforcement`)

ENVELOPE CONCEPT	EXISTING PROTOCOL SUPPORT	PROPOSED EXTENSION
Handoff admission as a first-class event	—	A new `event_type` variant (e.g., `handoff_admitted` / `handoff_rejected`) emitted by the consumer-side verifier

Items in the right-hand column would be candidates for future Sanna Protocol proposals. The intent of this paper is to motivate the extension, not to specify it; field names and exact shapes will be settled through protocol design and implementation review.

APPENDIX C

Selected public anchors.

The use cases in this paper are framed from an architecture perspective, not as legal conclusions. The following public materials illustrate why traceability, documentation, logging, oversight, and evidence preservation are recurring requirements across the environments discussed above.

EU AI ACT

Regulation (EU) 2024/1689 includes logging, record-keeping, human oversight, and general-purpose AI model documentation and copyright-related obligations, including Articles 12, 14, and 53. See the official EUR-Lex publication.

INSURANCE — NAIC

The NAIC's model bulletin on insurer use of AI systems emphasizes governance, risk management, documentation, testing, and oversight. See the NAIC announcement and adopted model bulletin.

FINANCIAL SERVICES — FINRA

FINRA has stated that securities firms' existing regulatory obligations remain applicable when AI is used, including supervision, recordkeeping, and customer-protection obligations. See FINRA Regulatory Notice 24-09.

HEALTHCARE AND MEDICAL PRODUCTS — FDA

FDA materials on AI-enabled medical products and electronic records emphasize transparency, data management, monitoring, record integrity, and audit trails. See FDA AI-enabled medical devices and 21 CFR Part 11.

PUBLIC SECTOR & CRITICAL INFRASTRUCTURE — NIST

NIST's AI Risk Management Framework emphasizes governance, mapping, measurement, and management of AI risks. See NIST AI RMF 1.0 and the Critical Infrastructure Profile concept note.

AUDIT AND ASSURANCE — PCAOB

Audit documentation standards require workpapers to provide a clear record of procedures performed, evidence obtained, and conclusions reached. See PCAOB AS 1215.

LEGAL WORK — ABA

The ABA has warned lawyers using generative AI to attend to competence, confidentiality, communication, supervision, and accuracy obligations. See ABA Formal Opinion 512. Court sanctions for fabricated AI citations, including Mata v. Avianca, illustrate the operational importance of citation binding.

CONTENT PROVENANCE — C2PA

C2PA Content Credentials provide a public provenance model for digital content. See the C2PA 2.4 specification.

sanna.

Trust infrastructure for agentic AI.

CONTACT

nic@sanna.dev

sanna.dev

PROTOCOL

github.com/sanna-ai

v1.5 receipts · v0.1 envelopes

THIS DOCUMENT

Public draft.