Network Working Group
Internet-Draft
Intended status: Informational
Expires: 2 September 2026

R. Bondar

1 March 2026

Warrant Certificate Authorities (WCA): Auditable Data Provenance for AI-Agent Tool-Call Chains
draft-bondar-wca-00

Abstract

Large Language Model (LLM)-based agent systems increasingly invoke external tools and data sources, yet the epistemic provenance of consumed data remains architecturally unregulated. Data crossing tool-call boundaries acquires the apparent trustworthiness of the interface rather than reflecting the institutional standing of its actual source -- a phenomenon termed "semantic laundering."

This document specifies the Warrant Certificate Authority (WCA): an end-to-end cryptographic attestation infrastructure that certifies data sources, not data content. WCA introduces a provenance layer satisfying reference monitor properties (complete mediation, tamperproofness, verifiability) for all agent-to-tool interactions. The architecture draws on the PKI trust model, OS provenance paradigms (IMA, CamFlow, LPM), and supply-chain security frameworks (SLSA, in-toto).

This document defines data structures for tool-call attestation, warrant certificates, and a trust hierarchy of certificate authorities. It specifies the provenance-layer protocol and introduces Warrant Attestation Levels (WAL-0 through WAL-3) as a graduated adoption framework analogous to SLSA build levels.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 2 September 2026.

Table of Contents

1. Introduction

The rapid deployment of Large Language Model (LLM)-based agent architectures has created a category of data provenance risk that existing security frameworks do not address. When AI agents invoke external tools -- APIs, databases, web scrapers, knowledge graphs -- they implicitly trust the data returned. A response from a trusted API may have originated from a retracted study passed through several interpretive layers, yet the agent treats all tool outputs uniformly as "tool output," conflating the channel’s trustworthiness with the content’s actual provenance.

This problem is structurally analogous to the blockchain oracle problem [CHAINLINK], where smart contracts must rely on external data feeds without the ability to verify their correctness on-chain. However, AI-agent systems face an additional dimension: not merely data accuracy, but preservation of epistemic justification across architectural boundaries.

Prior work [SEMANTIC-LAUNDERING] formalized two phenomena:

* Warrant Erosion: the inevitable degradation of epistemic justification through interpretive processing.

* Semantic Laundering: the acquisition of unwarranted credibility by data crossing trusted tool boundaries, constituting a channel-to-content trust conflation.

The same authors [RESPONSIBILITY-VACUUM] demonstrated that human oversight undergoes a phase transition from genuine evaluation to ritualized approval at sufficient throughput, establishing that content-evaluating mediators cannot scale.

This document specifies the Warrant Certificate Authority (WCA), an infrastructure that certifies data sources rather than data content. The key insight is architectural: just as PKI Certificate Authorities certify server identity without evaluating web page content, WCA certifies the institutional standing of data sources without judging the truth of individual responses.

This design is structurally analogous to provenance layers in operating systems:

* IMA (Integrity Measurement Architecture) [IMA] tracks what code ran on a system via kernel-level measurement and TPM anchoring.

* CamFlow [CAMFLOW] tracks all information flows as a provenance monitor satisfying the reference monitor concept.

* LPM (Linux Provenance Modules) [LPM] provides 170 provenance hooks parallel to LSM security hooks with 2.7% overhead.

* PASS [PASS] introduced provenance-aware storage at the filesystem level.

None of these systems evaluate the correctness or meaning of the data they track. They provide complete, tamperproof, verifiable records of what happened. WCA provides the same for AI agent tool calls: not judgment, but attestation.

This document makes the following contributions:

1. Specification of data structures for tool-call attestation, warrant certificates, and a hierarchical trust model for certificate authorities.

2. A provenance-layer protocol satisfying reference monitor properties (complete mediation, tamperproofness, verifiability) for AI agent tool calls.

3. Warrant Attestation Levels (WAL-0 through WAL-3): a graduated adoption framework analogous to SLSA [SLSA] build levels, providing practical deployment milestones.

4. Non-interference requirements and mitigation strategies for mediated self-licensing risks.

5. Security analysis covering response substitution, self-licensing, replay attacks, source impersonation, and attestation log tampering.

2. Conventions and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

The following terms are used throughout this document:

Agent: An LLM-based system that invokes external tools and data sources to accomplish tasks.

Attestation: A cryptographically signed record binding a specific query to a specific response from an identified source at a specific time.

Attestation Log: An append-only, hash-chained sequence of attestation records maintained by the provenance layer.

Channel-to-Content Trust Conflation: The error of treating a communication channel’s trustworthiness as evidence for the trustworthiness of specific data transmitted through that channel.

Domain WCA: An intermediate certificate authority covering a specific epistemic domain, subordinate to a Root WCA.

Epistemic Domain: A bounded field of knowledge within which a source's institutional authority applies (e.g., "pharmacology", "legal-records").

Epistemic Warrant: The structured justification backing a proposition, comprising observations, inference rules, and institutional attestations. WCA operationalizes only the institutional attestation component.

Institutional Anchor: The organizational basis for a source's authority within its epistemic domain.

Non-Interference: The condition that an agent is purely a consumer of registered source data, with no pathway to modify or influence what registered sources store or return.

Provenance Layer: The mandatory intermediary component that mediates all tool calls between agent and external sources.

Root WCA: The top-level certificate authority in a WCA hierarchy.

Semantic Laundering: The phenomenon whereby weakly warranted data acquires unwarranted epistemic status by crossing a trusted tool-call boundary.

Self-Licensing: The condition where an agent generates a proposition and subsequently treats it as externally warranted.

Source: An external system that provides data to agents via tool calls.

Trusted Source Registry: A registry maintained by WCA operators containing registered sources with their public keys, epistemic domains, and institutional anchors.

Warrant Certificate (WC): A compound data structure comprising a tool-call attestation, the source's certificate, and a chain proof to a trusted Root WCA.

Warrant Attestation Level (WAL): One of four graduated levels (WAL-0 through WAL-3) specifying increasing provenance guarantees for agent systems.

W_institutional(s, D): The institutional warrant of source s in domain D.

3. Problem Statement

3.1. Warrant Erosion

Epistemic warrant is a structured object, not a scalar:

```
warrant(p) = <O, I, S>
where:
  O = {observations grounding p}
  I = {admissible inference rules deriving p from O}
  S = {signed attestations from institutional sources}
```

For any interpretive process f:

warrant(f(p)) is a subset of warrant(p) by O and I

Interpretation can only lose observations and inference rules, never gain them. This is the Warrant Erosion Principle [SEMANTIC-LAUNDERING].

3.2. Semantic Laundering

Semantic laundering occurs when:

```
Laundering(p, t) iff
  S(p) = {} AND trusted(t) AND agent_assigns(W(p) >= W_min)
```

Data without institutional attestation gains "warranted" status solely by crossing a trusted tool-call interface. This constitutes channel-to-content trust conflation.

3.3. Insufficiency of Content-Evaluating Mediators

For any mediator M that evaluates content:

```
evaluates_content(M) implies
  there exists boundary b_M such that
    Laundering(p, b_M) is possible
```

The argument proceeds in four steps: (a) M must identify proposition boundaries (semantic judgment), (b) M must define canonical representations (representational assumptions), (c) M must assign warrant status (ritualizes under throughput [RESPONSIBILITY-VACUUM]), (d) M’s "certified" label becomes a new laundering channel.

This result motivates WCA’s design: certify sources, not content.

4. Architecture Overview

4.1. Design Principle

WCA certifies sources, not content. This distinction avoids the mediator vulnerability. Evaluating content requires semantic judgment that scales poorly; certifying provenance requires only identity verification and institutional audit -- operations performed at registration time, not per-query.

4.2. System Components

A WCA deployment consists of:

* Root WCA: Top-level certificate authority.

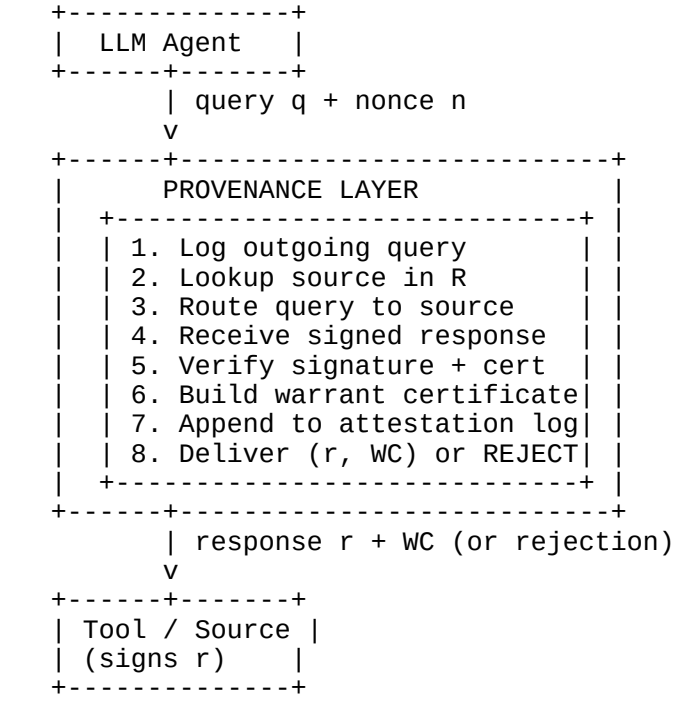
* Domain WCAs: Intermediate CAs for specific epistemic domains.

* Trusted Source Registry (R): Authorized data sources with keys and domain assignments.

* Provenance Layer: Enforcement component mediating all tool calls.

* Attestation Log (L): Append-only, hash-chained transaction log.

* Registered Sources: Data providers implementing WCA signing.



5. Data Structures

5.1. Trusted Source Registry Entry

```
{
  "source_id":   "urn:wca:source:<name>",
  "public_key":  "<SubjectPublicKeyInfo>",
  "domain":      "urn:wca:domain:<domain-name>",
  "anchor":      { "organization": "...", "basis": "..." },
  "valid_from":  "<DateTime>",
  "valid_until": "<DateTime>",
  "issuer_wca":  "urn:wca:authority:<wca-name>",
  "revocation":  { "crl_uri": "...", "ocsp_uri": "..." }
}
```

source_id: Globally unique URI using the "urn:wca:source:" prefix.

public_key: MUST be ECDSA P-256 [RFC6979] or Ed25519 [RFC8032].

domain: MUST be from the WCA Epistemic Domain Registry (Section 12.3).

anchor: Institutional basis for authority, populated during

registration audit.

valid_from / valid_until: Validity period. Source certificates SHOULD have a maximum validity of one year. Renewal requires re-audit.

5.2. Tool-Call Attestation

For query q from agent a to source s returning response r:

```
signature = Sign(K_priv_s,
                 H(query || response || timestamp ||
                   nonce || agent_id))
```

where Sign() is ECDSA-P256-SHA256 [RFC6979] or Ed25519 [RFC8032], H() is SHA-256 [RFC6234], and || denotes concatenation of canonical byte encodings.

Each field MUST be prefixed with a 4-byte big-endian length followed by the field bytes.

The source MUST sign the binding of all five components. Nonce inclusion prevents replay; agent_id inclusion prevents cross-agent substitution.

Protocol:

1. Agent generates query q and cryptographically random nonce n (>= 16 bytes).

2. Provenance layer forwards (q, agent_id, n) to source s.

3. Source computes response r.

4. Source obtains timestamp tau from a trusted time source.

5. Source computes signature.

6. Source returns (r, tau, signature).

7. Provenance layer verifies using K_pub_s from R.

5.3. Warrant Certificate

```
{
  "attestation":        "<ToolCallAttestation>",
  "source_certificate": "<SourceCertificate>",
  "chain_proof":        ["<CertificateChainEntry>"]
}
```

The SourceCertificate issuer_signature is:

```
issuer_signature = Sign(K_priv_WCA,
                        H(source_id || public_key ||
                          domain || anchor ||
                          valid_from || valid_until))
```

5.4. WCA Certificate

```
{
  "wca_id":           "urn:wca:authority:<name>",
  "public_key":       "<SubjectPublicKeyInfo>",
  "domain_scope":     ["urn:wca:domain:<d1>", "..."],
  "trust_anchor":     { "organization": "...", "basis": "..." },
  "parent_wca":       "urn:wca:authority:<parent>" or null,
  "valid_from":       "<DateTime>",
  "valid_until":      "<DateTime>",
  "parent_signature": "<OCTET STRING>" or null
}
```

Root WCA certificates are self-signed. Domain WCA certificates MUST be signed by their parent WCA. WCA key pairs MUST be generated and stored in HSMs. Root WCA key generation MUST use multi-party key ceremony procedures.

5.5. Attestation Log Entry

```
{
  "sequence_number": 42,
  "query":           "<exact bytes>",
  "source_id":       "urn:wca:source:<name>",
  "response":        "<exact bytes>",
  "signature":       "<base64>",
  "timestamp":       "<DateTime>",
  "warrant_cert":    "<WarrantCertificate>",
  "previous_hash":   "<hex>",
  "entry_hash":      "<hex>"
}
```

entry_hash is computed over all preceding fields. previous_hash links to the predecessor. The log MUST be append-only. Implementations SHOULD periodically commit the chain head to an external transparency log.

6. WCA Trust Hierarchy

6.1. Root WCA

* MUST store private key in HSM with multi-party access controls.
* MUST use offline key storage.
* MUST maintain Certificate Transparency log of issued Domain WCA certificates.
* SHOULD be operated by a consortium or standards body.

6.2. Domain WCA

* MUST have certificate signed by Root or parent Domain WCA.
* MUST specify domain_scope.
* MUST perform institutional audit before issuing source certificates.
* MUST maintain CRL or operate OCSP responder.

Example hierarchy:

```
Root WCA (global trust anchor)
  |
  +-- Health WCA (medicine, pharmacology, genomics)
  |     +-- urn:wca:source:fda-druginteractions-v3
  |     +-- urn:wca:source:pubmed-api-v2
  |
  +-- Legal WCA (legal-records, court-rulings)
  |     +-- urn:wca:source:pacer-federal-courts
  |
  +-- Meteorological WCA (weather, climate)
        +-- urn:wca:source:noaa-weather-api-v3
```

6.3. Source Certificates

Issuance requires: (1) institutional audit, (2) key verification, (3) domain validation. Certificates SHOULD have maximum one-year validity. Renewal MUST include re-audit.

6.4. Certificate Lifecycle

Issuance -> Active -> Renewal (with re-audit) or Revocation.
Revocation triggers: key compromise, institutional standing change, misrepresentation of data sourcing, WCA operator determination.

Provenance layer MUST check revocation before accepting signatures. Implementations SHOULD cache revocation status (max 24 hours).

7. Provenance Layer Protocol

7.1. Protocol Flow

Step 1 - Query Registration: Agent submits query q with fresh nonce n. Provenance layer records (q, agent_id, n, timestamp_local).

Step 2 - Source Lookup: Consult R; verify source is registered and certificate is valid. If invalid, MUST reject and log rejection with reason code.

Step 3 - Query Routing: Forward (q, agent_id, n) to source over authenticated channel. SHOULD use mTLS.

Step 4 - Response Receipt: Source computes r, obtains tau, signs binding, returns (r, tau, signature).

Step 5 - Signature Verification: Verify using K_pub_s from Cert_s. On failure, MUST reject, MUST log failure, MUST NOT deliver to agent.

Step 6 - Warrant Certificate Construction: Assemble WC = (Att(q,s,r), Cert_s, ChainProof).

Step 7 - Attestation Log Append: Append e_i = (seq_i, q, s_id, r, signature, tau, WC, H(e_{i-1})).

Step 8 - Delivery or Rejection: Deliver (r, WC) or rejection notice. Rejected data MUST NOT enter agent reasoning context.

Warrant assignment upon delivery:

```
W(r | valid WC from source s in domain D) = W_institutional(s, D)
W(r | invalid WC or absent WC)            = 0
```

7.2. Reference Monitor Properties

The provenance layer MUST satisfy three properties per [ANDERSON] and [CAMFLOW]:

RM1 - Complete Mediation: Every external data access by the agent MUST pass through the provenance layer. The agent MUST NOT have direct network access to external sources. SHOULD be enforced via network isolation.

RM2 - Tamperproofness: Attestation log MUST be append-only and hash-chained.

RM3 - Verifiability: Any party with log access and public keys MUST be able to independently verify every entry.

7.3. Attestation Log

The log MUST be append-only and hash-chained. Implementations SHOULD replicate to external audit services. SHOULD define retention policy per regulatory context. SHOULD periodically commit chain head to external transparency log.

8. Warrant Attestation Levels

8.1. WAL-0: No Provenance

Baseline state. No guarantees. Default for current agent systems.

8.2. WAL-1: Provenance Exists

Source identification recorded; cryptographic verification MAY be absent. Provenance layer MUST log source identity and responses. Post-hoc attribution possible but not cryptographically verified. Adoption cost: minimal (middleware only).

8.3. WAL-2: Signed Provenance

Sources MUST sign responses per Section 5.2. Source keys MUST be certified by Domain WCA. Provenance layer MUST verify signatures. Guarantees: query-response binding, source authenticity, anti-laundering bound. Adoption cost: moderate.

8.4. WAL-3: Full Verification

Provenance layer MUST satisfy RM1-RM3. All tool calls MUST be logged in hash-chained attestation log. Agents MUST generate signed queries with nonces. Non-interference MUST be enforced or M2 MUST be implemented (Section 9.3). Guarantees: all five security properties (Appendix A). Adoption cost: highest.

Use Case	Min WAL	Rationale
Exploratory research	WAL-0	Low stakes
General info retrieval	WAL-1	Post-hoc review
Business intelligence	WAL-2	Source verify
Medical decision support	WAL-3	Safety-critical
Legal research	WAL-3	Regulatory
Financial trading	WAL-3	Fiduciary duty

Table 1

8.5. Migration Path

Phase 1 (WAL-0 to WAL-1): Deploy logging middleware. Phase 2 (WAL-1 to WAL-2): Source signing + Domain WCA setup. Phase 3 (WAL-2 to WAL-3): Full RM1-RM3 + attestation log.

9. Non-Interference Requirements

9.1. Definition

Agent a satisfies non-interference w.r.t. registry R iff:

NI-1: a has no write access to any registered source s in R.

NI-2: a does not control or operate any registered source s in R.

NI-3: a cannot influence the stored state of any upstream system feeding s in R.

WAL-3 deployments MUST enforce non-interference or implement an equivalent mitigation.

9.2. Mediated Self-Licensing Risk

When NI-1 is violated, the following loop is possible:

1. Agent generates content.

2. Agent writes content to registered source.

3. Agent queries that source.

4. Source returns content with valid signature.

5. Provenance layer issues WC.

6. Agent's own content carries institutional warrant.

TLA+ model checking confirms: with non-interference, 286 states explored, no self-licensing. Without: counterexample in 3 transitions.

9.3. Mitigation Strategies

WAL-3 deployments MUST enforce at least one:

M1 - Non-Interference at Deployment: Read-only access to registered sources. Strongest mitigation.

M2 - Origin-Aware Warrant Assignment: If data in source was written by agent a, queries from a receive W = 0.

M3 - Institutional Transformation Gate: Agent-written data undergoes institutional process (peer review, QA) before becoming queryable with warrant.

10. Integration Considerations

10.1. Agent Framework Integration

LangChain/LangGraph: Provenance layer as BaseTool.invoke() wrapper.

AutoGen/AG2: Shared provenance layer at tool execution boundary.

Model Context Protocol (MCP): MCP servers implement signing; MCP clients verify via provenance layer.

OpenAI Function Calling / Anthropic Tool Use: Provenance layer wraps function execution step.

10.2. Source-Side Requirements

Sources at WAL-2+ MUST: generate key pair (ECDSA P-256 or Ed25519), implement signing endpoint, undergo institutional audit, implement certificate renewal and key rotation.

Sources SHOULD: use HSMs, implement rate limiting, monitor for anomalous signing.

10.3. Performance Considerations

Per-query overhead: sub-millisecond for cryptographic operations. Ed25519 signs at ~70K ops/sec, verifies at ~30K ops/sec. SHA-256 >1 GB/s. Dominated by network latency.

LPM demonstrated 2.7% overhead for whole-system kernel provenance; WCA at tool-call granularity is expected negligible.

11. Security Considerations

11.1. Threat Model

Attacker can: MITM traffic, operate malicious sources, inject unattested data, replay responses, impersonate sources, tamper with log.

Trust assumptions: Root WCA key security (HSM), sound crypto primitives, correct RM1-RM3 enforcement, institutional audit at registration.

Out of scope: registered source providing incorrect data (source quality, not provenance).

11.2. Response Substitution

Defense: signature covers H(q || r || tau || nonce || agent_id). Modification detected. Residual: key compromise (standard PKI mitigations).

11.3. Agent Self-Licensing

Defense: non-interference prevents registration and indirect pathways. W = 0 for agent-generated propositions. Residual: NI violation enables mediated self-licensing (Section 9.3).

11.4. Replay Attacks

Defense: nonce in signature binding. Timestamp enables freshness. Implementations MUST use >= 16 byte random nonces.

11.5. Source Impersonation

Defense: certificate chain to Root WCA. Residual: WCA compromise.

11.6. WCA Compromise

Defense: HSMs, multi-party ceremonies, CT logs, CRLs/OCSP, short-lived certificates.

11.7. Attestation Log Tampering

Defense: hash chain; modification cascades. External transparency log anchoring for additional assurance.

11.8. Limitations

WCA does NOT defend against: source inaccuracy, domain mismatch, agent reasoning errors, WCA-source collusion, interpretive warrant erosion (but preserves originals in log for audit).

12. IANA Considerations

12.1. WCA URN Namespace

This document requests registration of formal URN namespace "wca" per [RFC8141].

Syntax: urn:wca:<entity-type>:<entity-name>

Entity types: "authority", "source", "domain".

12.2. WAL Level Registry

New IANA registry "Warrant Attestation Levels":

Level	Name	Reference
0	No Provenance	This document
1	Provenance Exists	This document
2	Signed Provenance	This document
3	Full Verification	This document

Table 2

New levels require Standards Action [RFC8126].

12.3. WCA Epistemic Domain Registry

New IANA registry "WCA Epistemic Domains" with initial entries:

Domain	Description
pharmacology	Drug data, interactions
medical-records	Clinical records
medical-lit	Peer-reviewed medical literature
legal-records	Court records, filings
legal-lit	Legal scholarship
meteorology	Weather, climate
genomics	Genomic sequences
financial-reg	Regulatory filings
financial-market	Market data
geospatial	Geographic, satellite data

Table 3

New domains require Specification Required [RFC8126].

12.4. WCA Media Types

application/wca-warrant-certificate+json: JSON serialization of Warrant Certificates.

application/wca-attestation-log+json: JSON serialization of Attestation Log entries.

13. References

13.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC6234] Eastlake 3rd, D. and T. Hansen, "US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF)", RFC 6234, DOI 10.17487/RFC6234, May 2011, <https://www.rfc-editor.org/rfc/rfc6234>.

[RFC6979] Pornin, T., "Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA)", RFC 6979, DOI 10.17487/RFC6979, August 2013, <https://www.rfc-editor.org/rfc/rfc6979>.

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <https://www.rfc-editor.org/rfc/rfc8032>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/rfc/rfc8126>.

[RFC8141] Saint-Andre, P. and J. Klensin, "Uniform Resource Names (URNs)", RFC 8141, DOI 10.17487/RFC8141, April 2017, <https://www.rfc-editor.org/rfc/rfc8141>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/rfc/rfc8259>.

13.2. Informative References

[ANDERSON] Anderson, J. P., "Computer Security Technology Planning Study", Technical Report ESD-TR-73-51, 1972.

[AUTH-PROMPTS] Rajagopalan, M. and V. Rao, "Protecting Context and Prompts: Deterministic Security for Non-Deterministic AI", arXiv 2602.10481, 2026.

[BAID] Lin, Z., Zhang, S., Liao, G., Tao, D., and T. Wang, "Binding Agent ID", arXiv 2512.17538, 2025.

[CAMEL] Debenedetti, E., Shumailov, I., and T. Fan, "Defeating Prompt Injections by Design", arXiv 2503.18813, 2025.

[CAMFLOW] Pasquier, T., Han, X., Goldstein, M., Moyer, T., Eyers, D., Seltzer, M., and J. Bacon, "Practical Whole-System Provenance Capture", arXiv 1711.05296, 2017.

[CHAINLINK] Ellis, S., Juels, A., and S. Nazarov, "Chainlink: A Decentralized Oracle Network", White Paper, 2017.

[CIV] Gupta, A., "CIV: Can AI Keep a Secret? Contextual Integrity Verification", arXiv 2508.09288, 2025.

[DIDS-VCS] Garzon, S. R., "AI Agents with Decentralized Identifiers and Verifiable Credentials", arXiv 2511.02841, 2025.

[GETTIER] Gettier, E. L., "Is Justified True Belief Knowledge?", Analysis Vol. 23, No. 6, pp. 121-123, 1963.

[IMA] Linux IMA, "Integrity Measurement Architecture", n.d., <https://sourceforge.net/p/linux-ima/wiki/Home/>.

[IN-TOTO] in-toto Project, "A Framework for Securing Software Supply Chains", n.d., <https://in-toto.io/>.

[LPM] Bates, A., Tian, D. J., Butler, K. R. B., and T. Moyer, "Trustworthy Whole-System Provenance for the Linux Kernel", USENIX Security Symposium, 2015.

[MAIF] Narajala, V. S., "MAIF: Enforcing AI Trust and Provenance with an Artifact-Centric Paradigm", arXiv 2511.15097, 2025.

[OMEGA] Stavrakakis, D., "Trusted AI Agents in the Cloud", arXiv 2512.05951, 2025.

[PASS] Muniswamy-Reddy, K., Holland, D. A., Braun, U., and M. Seltzer, "Provenance-Aware Storage Systems", USENIX ATC, 2006.

[PROV-AGENT] Souza, R., "PROV-AGENT: Unified Provenance for AI Agent Interactions", arXiv 2508.02866, 2025.

[RAG-SIGN] Holmes, S. A., "RAG Sign: Cryptographic Authentication for RAG-Enabled LLMs", Springer LNCS 15692, 2025.

[RESPONSIBILITY-VACUUM]
Romanchuk, O. and R. Bondar, "The Responsibility Vacuum: Organizational Failure in Scaled Agent Systems", arXiv 2601.15059, January 2026, <https://arxiv.org/abs/2601.15059>.

[SEMANTIC-LAUNDERING]
Romanchuk, O. and R. Bondar, "Semantic Laundering in AI Agent Architectures: Why Tool Boundaries Do Not Confer Epistemic Warrant", arXiv 2601.08333, January 2026, <https://arxiv.org/abs/2601.08333>.

[SLSA] OpenSSF, "SLSA: Supply-chain Levels for Software Artifacts", n.d., <https://slsa.dev/>.

[SPIFFE] SPIFFE Project, "Secure Production Identity Framework for Everyone", n.d., <https://spiffe.io/>.

[VFA] Gupta, A., "Verifiability-First Agents", arXiv 2512.17259, 2025.

Appendix A. Formal Security Properties

All properties assume RM1-RM3 and EUF-CMA signature security.

A.1. Property 1: No Channel-Based Semantic Laundering

For all p received by agent a: W(p) <= W_institutional(source(p)). Data cannot gain warrant above source's institutional standing.

A.2. Property 2: Self-Licensing Prevention under Non-Interference

For all p generated by agent a: if NonInterference(a, R) then W(p) = 0.

A.3. Property 3: Provenance Completeness

For all data d in agent context: either d has attestation log entry with valid WC, or W(d) = 0.

A.4. Property 4: Query-Response Binding

Valid signature implies r is exactly what source s returned to query q at time tau. Substitution detectable.

A.5. Property 5: Warrant Erosion Auditability

Agent interpretation may degrade warrant, but original data and attestation are always recoverable from log L.

A.6. Composability

Multi-hop chains: W(final) = min_i(W_institutional(source_i)). No laundering through composition.

Appendix B. TLA+ Model Summary

Model verified with TLC:

* WCA_Strict.cfg (non-interference): PASS, 286 states.

* WCA_MediatedLoop.cfg (NI violated): FAIL, counterexample in 3 transitions (agent generates, writes to source, queries back with warrant > 0).

Appendix C. Comparison with Related Systems

Dimension	Covered By	WCA
Agent identity	BAID, DIDs+VCs, SPIFFE	--
Execution environment	Omega, TEEs	--
Action audit trails	PROV-AGENT, VFA, MAIF	--
Access control	CaMeL	--
Prompt/context integrity	Auth. Prompts, CIV	--
Output signing	RAG Sign	--
Data source warrant	*NONE*	*YES*
Anti-channel-laundering	*NONE*	*YES*
Self-licensing prevention	*NONE*	*YES*
End-to-end attestation	*NONE*	*YES*

Table 4

Appendix D. JSON Serialization Examples

D.1. Warrant Certificate


```
{
  "attestation": {
    "query": "GET /interactions?drug_a=ibuprofen&drug_b=warfarin",
    "source_id": "urn:wca:source:fda-druginteractions-v3",
    "response": "{\"interaction\":\"major\",\"severity\":\"high\"}",
    "timestamp": "2026-02-12T14:30:00Z",
    "nonce": "a7f3c9e1d4b2f6a8e0c7d3b5a9f1e2c4",
    "agent_id": "urn:agent:medical-advisor-v2",
    "signature": "MEUCIQD.../base64...=="
  },
  "source_certificate": {
    "source_id": "urn:wca:source:fda-druginteractions-v3",
    "public_key": "MFkwEwYHKoZIzj0.../base64...",
    "domain": "urn:wca:domain:pharmacology",
    "anchor": {
      "organization": "U.S. Food and Drug Administration",
      "basis": "Federal regulatory mandate",
      "audit_date": "2026-01-15"
    },
    "valid_from": "2026-01-01T00:00:00Z",
    "valid_until": "2027-01-01T00:00:00Z",
    "issuer_wca_id": "urn:wca:authority:health-wca-us",
    "issuer_signature": "MEUCIQCx.../base64...=="
  },
  "chain_proof": [
    {
      "wca_id": "urn:wca:authority:health-wca-us",
      "parent": "urn:wca:authority:root-wca-global-v1",
      "signature": "MEYCIQDp.../base64...=="
    }
  ]
}
```


Acknowledgments

The foundational concepts of warrant erosion and semantic laundering were developed jointly with Oleg Romanchuk in [SEMANTIC-LAUNDERING] and [RESPONSIBILITY-VACUUM].

Author’s Address

Roman Bondar
Email: bondar.roman@gmail.com