Network Working Group
Internet-Draft
Intended status: Informational
Expires: 21 November 2026

T. Trujillo
Individual Submission

May 2026

Agent-to-Agent Trust, Identity, and Verifiable Provenance
draft-tonyai-a2a-trust-00

Abstract

This document defines a trust model for agent-to-agent (A2A) interactions in multi-agent AI systems. It specifies how agents obtain verifiable identities via CA-signed templates, how spawn chains are cryptographically established and validated, how dynamic policies are governed under a dual-signature model, and how cross-organizational agent interactions are explicitly authorized. The model applies existing PKI primitives (X.509, CRL, CSR) and established identity patterns (OAuth 2.0, On-Behalf-Of) to the problem of agent provenance. This document does not address agent-to-resource access control, human-in-the-loop orchestration, or agent behavior, as those concerns belong to the resource enforcement layer and the orchestration layer respectively.

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 November 2026.

Copyright Notice

Table of Contents

1. Introduction

Multi-agent AI systems introduce identity and authorization gaps that existing standards do not fully address. When Agent A spawns Agent B, and Agent B calls a resource, no current standard defines how the resource verifies that Agent B was legitimately spawned, that its scope has not been escalated, or that its origin template is trusted.

This document proposes a trust model based on:

* Agent templates as first-class, CA-signed identity artifacts.

* Verifiable spawn chains where each agent's provenance is cryptographically traceable to a registered template.

* Two-lane governance separating static identity (cert-based) from dynamic policy (OPA-gated).

* Fail-closed enforcement at every verification step.

The model reuses proven PKI primitives and applies them to a new surface -- agent identity -- rather than inventing new cryptographic mechanisms.

2. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Terminology

Agent: An ephemeral, autonomous process that performs tasks on behalf of a user or another agent.

Agent Template: A CA-signed artifact defining an agent’s identity, allowed scopes, spawn rules, and policy reference.

Template Registry: The authoritative store of approved, signed agent templates for an organization.

Template Registry CA: The Certificate Authority that signs agent templates -- the root of trust for the agent ecosystem.

Spawn: The act of an agent instantiating a child agent from an approved template.

Spawn Chain: The ordered, cryptographically verifiable sequence of agents from the root orchestrator to the current agent.

Policy Authority: The entity responsible for countersigning dynamic policy changes after automated gate validation.

CRL: Certificate Revocation List -- the list of revoked template and agent certificates maintained by the CA.

CSR: Certificate Signing Request -- the template author’s request to the CA for a signed template certificate.

Cross-Org Grant: An explicit, signed authorization allowing agents from one organization to spawn from a template owned by another.

4. Problem Statement

Multi-agent orchestration creates identity and authorization gaps:

```
Agent A --> spawns --> Agent B --> calls --> Resource
```

Figure 1

* Which identity does the Resource see?

* How does authorization propagate across the chain without escalating?

* If Agent B is compromised, can it impersonate Agent A?

* Who owns the audit trail across the full chain?

* How does the Resource prove Agent B was authorized to be spawned?

No current standard (W3C, IETF, or vendor) fully addresses agent provenance in multi-hop chains.

5. Existing Patterns and Gaps

5.1. On-Behalf-Of (OBO)

Microsoft Entra ID OBO provides user-to-service delegation:

```
User --> Agent A (token A) -->
Agent A requests token for Agent B on behalf of user -->
Entra validates the chain -->
Issues scoped delegated token
```

Figure 2

Key insight: a trusted third party validates the delegation chain. Agents do not self-assert trust.

OBO breaks down for agent-to-agent because agents are ephemeral, agent spawning is dynamic, and no registry exists for agent templates.

5.2. OAuth 2.0 Token Exchange (RFC 8693)

[RFC8693] defines scope constraint across delegation hops -- downstream tokens MUST be a subset of upstream grants. This document adopts that principle for agent scope inheritance.

6. Agent Identity

Agent identity MUST be established using X.509 certificate chains [RFC5280]. This reuses the same trust model used by TLS and existing workload identity systems.

6.1. Certificate Signing Request Flow

```
Template Author
    | defines template (scopes, spawn rules, TTL, owner)
    | generates CSR
    v
Template Registry CA
    | validates template conformance
    | signs the template certificate
    v
Signed Agent Template (registered)
    |
    v
Orchestrator spawns agent from signed template
    | agent receives certificate derived from template certificate
    | CA chain proves provenance
    v
Resource validates certificate chain:
    - certificate signed by trusted template?
    - template signed by CA?
    - scope within bounds?
    - ALLOW or DENY
```

Figure 3

6.2. Certificate Chain Structure

```
Template Registry CA (root of trust)
    |
    v
Orchestrator Agent Certificate
    Subject:       orchestrator-v1
    Issuer:        Template Registry CA
    Owner:         owner@example.com
    OrgID:         org-123
    KeyUsage:      spawn, delegate
    AllowedScopes: read:data, write:data
    CanSpawn:      [reader-template-v1], MaxChildren: 5
    PolicyRef:     policy-store/orchestrator-v1/current
    TTL:           1h
         |
         v
    Child Agent Certificate
        Subject:       reader-agent-v1
        Issuer:        orchestrator-v1
        CA Chain:      orchestrator-v1 -> Template Registry CA
        KeyUsage:      read only
        AllowedScopes: read:data (MUST be subset of parent)
        TTL:           15min (SHOULD be shorter than parent)
```

Figure 4

7. Template Structure

7.1. Static Fields

The following fields MUST be present in every agent template certificate and MUST NOT be modified without full re-certification:

Field	Required	Description
Subject	REQUIRED	Unique template identifier
Issuer	REQUIRED	Template Registry CA
Owner	REQUIRED	Verified identity of template owner
OrgID	REQUIRED	CA-validated organization identifier
KeyUsage	REQUIRED	Permitted operations
AllowedScopes	REQUIRED	Maximum scopes this agent may hold
CanSpawn	REQUIRED	Whitelist of permitted child templates
MaxChildren	REQUIRED	Maximum concurrent child agents
ScopeInherit	REQUIRED	Scope inheritance constraint
PolicyRef	REQUIRED	Pointer to dynamic policy store
TTL	REQUIRED	Maximum agent lifetime

Table 1

7.2. Dynamic Policy Bounds

Dynamic policies MUST be bounded by the static template fields. A dynamic policy MUST NOT grant scopes beyond AllowedScopes. A dynamic policy MUST NOT add spawn targets beyond CanSpawn.

8. Spawn Chain Validation

8.1. Two-Check Spawn Rule

Every spawn MUST pass two independent checks. Either check failing MUST result in spawn denial with no fallback.

Check 1 -- Static (CanSpawn in certificate): The requested child template MUST appear in the spawning agent's CanSpawn list.

Check 2 -- Dynamic (Registry live lookup): The requested child template MUST be currently registered, CA-signed, not self-signed, owned by an authorized party, and not present in the current CRL.

Rationale: CanSpawn alone is insufficient because the certificate may be stale and a template may have been revoked since issuance. Registry lookup alone is insufficient because any party could forge a request claiming any template. Both checks MUST pass.

8.2. Spawn Validation Sequence

1. CanSpawn check -- child template in CanSpawn list?
   NO --> DENY, audit log
   YES --> continue

2. Registry check -- registered, CA-signed, not revoked?
   NO --> DENY, audit log
   YES --> continue

3. Scope check -- requested scope subset of parent scopes?
   NO --> DENY, audit log
   YES --> continue

4. MaxChildren check -- current children < MaxChildren?
   NO --> DENY, audit log
   YES --> spawn approved

5. Child certificate issued with:
   - parent identity embedded (delegation chain)
   - scope equal to requested scope (not exceeding parent)
   - spawn timestamp and nonce (replay prevention)
   - audit log entry written

Figure 5

8.3. Scope Constraint

A child agent's AllowedScopes MUST be a strict subset of the parent agent's AllowedScopes. Scope escalation across agent hops is explicitly prohibited.

Example:

```
Parent has:  read:data, write:data
Child gets:  read:data              -- valid
Child gets:  read:data, write:data  -- valid (same as parent)
Child gets:  admin:data             -- MUST be rejected
```

Figure 6

8.4. Audit Requirements

Every spawn event MUST be logged with: spawning agent identity, child template identity, requested scope, granted scope, timestamp, outcome (ALLOWED or DENIED), and reason if denied.

9. Dynamic Policy Governance

9.1. Two-Lane Model

Template certificate (static): WHO the agent is and WHO it can spawn. Changes require full re-certification.

Dynamic policy (fast lane): WHAT the agent can do within the bounds of the template certificate. Changes require dual signature (Section 9.3).

Dynamic policies MUST NOT exceed the bounds defined in the static template certificate.

9.2. Ownership

Template ownership MUST be established at certificate signing time and embedded in the Owner and OrgID fields. Only the verified owner of the organization that signed the template MAY submit policy changes.

9.3. Dual Signature Requirement

Every policy change MUST be signed by two independent parties:

1. Owner -- proves the right to change the policy.

2. Policy Authority -- proves the policy passed automated validation gates.

Neither signature alone is sufficient. Both MUST be present and valid for a policy to be accepted.

```
Owner key  +  Policy Authority key  =  policy accepted
    ^                  ^
who owns it        passed the gates
```

Figure 7

9.4. Policy Change Sequence

1. Identity and ownership verification
   - requester matches Owner in template certificate?
   - requester OrgID matches certificate OrgID?
   NO --> rejected, audit logged

2. Automated gate (OPA):
   - scope within AllowedScopes?
   - spawn targets within CanSpawn?
   - no conflicts with active policies?
   ANY FAIL --> rejected

3. Dual signature applied:
   Owner signs, Policy Authority countersigns

4. Policy stored with dual signature, version, timestamp, and content hash

5. Agent validates at runtime:
   - both signatures valid?
   - version current? (replay prevention)
   - hash matches? (tamper detection)
   - policy within template certificate bounds?
   ANY FAIL --> DENY, audit logged

Figure 8

9.5. Threat Coverage

Scenario	Single Sig Gap	Dual Sig Fix
Rogue Policy Authority	Pushes bad policy	Owner key missing
Rogue owner	Bypasses OPA gates	PA won't countersign
Compromised owner key	Attacker modifies	OPA gates enforced
Compromised Policy	Pushes bad policy	Owner key missing

| Auth | | |
+---+---+---+

Table 2

10. Template Versioning

10.1. Full Re-Verification Required

A new template version MUST undergo full re-verification from scratch. Trust MUST NOT be inherited from a previous version.

Rationale: inheriting trust from v1 would allow a compromised v1 certificate to bootstrap trust for v2.

10.2. Versioning Principle

Everything on the old template chain continues to work unchanged as long as the certificate is valid. New connections and functionality are opt-in -- only available after the new template chain is fully verified and explicitly adopted.

10.3. Non-Inheritance Rules

The following MUST NOT be inherited from a previous version:

* Trust chain
* Cross-organizational grants
* CanSpawn list
* Dynamic policies

10.4. Template Lifecycle

ACTIVE: Template is trusted. New spawns accepted.

DISABLED: No new spawns. Existing agents run to TTL expiry. Reversible.

DELETED: Certificate revoked, CRL updated, registry entry removed. Irreversible. Audit log preserved.

Disable SHOULD precede delete. Implementations SHOULD enforce a mandatory waiting period between DISABLED and DELETED.

10.5. Cross-Organizational Grant Re-Issuance

Cross-organizational grants MUST be explicitly re-issued for each new template version. Grants MUST NOT automatically roll over on version upgrade.

11. Cross-Organizational Agent Interaction

11.1. Explicit Grant Requirement

Cross-organizational agent spawning MUST be explicitly authorized by the resource-owning organization. No implicit trust exists between organizations.

11.2. Grant Structure

A cross-organizational grant MUST contain:

Field	Description
Grantor	Resource-owning organization
Grantee	Requesting organization
Template	Specific template identifier
AllowedScopes	MUST be subset of template scopes
TTL	Grant validity period
MaxSpawns	Maximum concurrent agents permitted
Signature	Dual signature (grantor owner + Policy Authority)

Table 3

11.3. Trust Anchor Options

Federated CA: Both organizations trust a shared root CA.

Explicit CA trust: Org-B explicitly trusts Org-A’s CA.

Third-party CA: Both organizations use a public CA.

11.4. Unilateral Revocation

The granting organization MAY revoke a cross-organizational grant without the cooperation of the receiving organization. Upon revocation, all agents spawned under the affected grant MUST be treated as untrusted on next validation.

11.5. Federated Audit

Each organization MUST maintain its own independent audit trail. Audit records MUST NOT depend on the other organization's systems.

12. Revocation

12.1. Template Revocation

Template revocation MUST be performed by adding the template certificate to the CA's CRL. All agent certificates derived from a revoked template MUST be treated as untrusted on next CRL check.

12.2. Individual Agent Revocation

Individual agents MAY be revoked by explicit certificate revocation or by removal of authorization relationships at the resource layer.

12.3. Automation Requirement

Routine revocation (TTL expiry, task completion) MUST be fully automated. Human involvement SHOULD be reserved for incident response and high-impact decisions.

13. Failure Model

13.1. Fail Closed

Any verification step that cannot be completed MUST result in DENY. This includes:

* CA unreachable
* Registry unreachable
* CRL unreachable
* Certificate expired or revoked
* Scope escalation attempt

* Policy invalid or unsigned

* Dual signature missing or invalid

Implementations MUST NOT provide a degraded mode that allows partial spawning or execution when verification infrastructure is unavailable.

14. Conformance Requirements

14.1. Field Placement

Implementations MUST place every field in the location specified by this document. Variable element placement is NOT permitted.

14.2. CSR Validation

Non-conforming CSRs MUST be rejected by the CA. A non-conforming template MUST NOT be signed.

14.3. Test Vectors

Implementations MUST provide test vectors -- concrete examples of valid and invalid template chains -- for conformance validation.

14.4. Conformance Certification

Implementations claiming conformance with this document MUST pass conformance certification before shipping.

14.5. Reference Implementation

A reference implementation SHOULD be made available including:

* Template Registry CA

* Spawn chain validation

* SDK with certificate chain validation, registry lookup, and CRL check handling

* Template linter for pre-submission CSR validation

15. IANA Considerations

This document has no IANA actions.

16. Security Considerations

16.1. Scope Escalation

Implementations MUST enforce that child agent scopes are a strict subset of parent scopes at every hop. Failure to enforce this allows privilege escalation across the agent chain.

16.2. Replay Attacks

Spawn requests MUST include a timestamp and nonce. Implementations MUST reject spawn requests where the timestamp exceeds a defined freshness window or the nonce has been seen previously.

16.3. Compromised Templates

A compromised template certificate MUST be revoked immediately. A single CRL update invalidates all downstream agent certificates derived from that template.

16.4. Single Point of Compromise

The dual signature requirement (Section 9.3) ensures no single compromised party can push unauthorized policy changes. CA compromise requires full ecosystem re-issuance.

16.5. Cross-Organizational Trust

Cross-organizational grants MUST be explicitly issued. Implicit trust between organizations is prohibited.

16.6. Audit Integrity

Audit logs MUST be tamper-evident. Logs MUST NOT be deletable by agents or orchestrators. Audit log infrastructure SHOULD be outside the control of the agent ecosystem itself.

17. References

17.1. Normative References

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

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC6749] Hardt, D., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <https://www.rfc-editor.org/info/rfc6749>.

[RFC8693] Jones, M., Campbell, A., and C. Mortimore, "OAuth 2.0 Token Exchange", RFC 8693, January 2020, <https://www.rfc-editor.org/info/rfc8693>.

17.2. Informative References

[NIST800-207] Rose, S., Borchert, O., Mitchell, S., and S. Connelly, "Zero Trust Architecture", NIST SP 800-207, August 2020.

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

[OpenFGA] OpenFGA Project, "OpenFGA Specification", <https://openfga.dev>.

[VAULTPKI] HashiCorp, "Vault PKI Secrets Engine", <https://developer.hashicorp.com/vault/docs/secrets/pki>.

Author’s Address

Tony Trujillo (tonyai)
Individual Submission
Email: ajtrujillo68@gmail.com