● Ainary

AR-024 Confidence: 75%

The Trust Transfer Problem

When Agents Delegate to Agents

February 2026

v1.0

Florian Ziesche · Ainary Ventures

"OAuth was designed for humans. It assumes persistent sessions and user consent—not fast-moving, autonomous systems."

— Strata Identity, Agentic AI Security Guide (January 2026)

CONTENTS

1. How to Read This Report

This report uses a structured confidence rating system to communicate what is known versus what is inferred. Every quantitative claim carries its source and confidence level.

RATING	MEANING	EXAMPLE
High	Multiple independent sources, established cryptographic principles or empirical data	PKI transitive trust model (RFC standards + 30+ years deployment history)
Medium	Single credible source, emerging standards or documented implementation	Visa Trusted Agent Protocol (GitHub repo + technical blog, not yet widely deployed)
Low	Theoretical analysis, extrapolation from related domains, or single early-stage tool	Agent delegation attack vectors (inferred from PKI history, not yet documented in wild)

This report was produced using a **multi-agent research pipeline** synthesizing cryptographic standards (PKI, OAuth, DPoP), multi-agent framework documentation, and emerging agent identity protocols. Full methodology details are provided in the Transparency Note (Section 11).

2. Executive Summary

Trust doesn't transfer transitively. When Agent A delegates to Agent B, the trust chain breaks in ways current architectures can't handle. We have 30 years of PKI teaching us how delegation chains fail — and we're ignoring every lesson.

- **PKI solves transitive trust through cryptographic chain-of-trust verification** — root CA → intermediate CA → end entity, with each link cryptographically signed and independently verifiable[1][2]
- **OAuth 2.0 was designed for human users with persistent sessions**, not autonomous agents that delegate tasks in milliseconds without user consent[3]
- **No major multi-agent framework (AutoGen, CrewAI, MetaGPT, LangGraph) implements cryptographic delegation chain verification** — messages between agents are trusted by default[4][5]
- **Emerging standards (Visa Trusted Agent Protocol, DID/VC for agents, AAP Protocol) introduce cryptographic proof-of-possession**, but adoption remains early-stage[6][7][8]
- **The confused deputy attack—where Agent B misuses Agent A's authority—has no technical mitigation in current agent architectures**, only policy-based controls[9]

Keywords: *Trust delegation, transitive trust, PKI chain-of-trust, multi-agent security, OAuth limitations, proof-of-possession, DID verifiable credentials, confused deputy attacks*

3. Methodology

This report synthesizes 15 sources: 4 cryptographic standards (PKI chain-of-trust, OAuth 2.0 RFC 8693 Token Exchange, DPoP RFC 9449, OpenID Federation), 6 framework documentation reviews (AutoGen, CrewAI, LangGraph, A2A protocol, Visa Trusted Agent Protocol, AAP Protocol), and 5 emerging agent identity implementations (DID/VC for agents, JSIGN, TessPay, OpenID Connect for Agents).

Research followed a structured multi-agent pipeline: (1) PKI and OAuth delegation mechanism analysis, (2) Multi-agent framework security feature review, (3) Emerging agent identity protocol synthesis, (4) Gap analysis identifying where transitive trust breaks in current architectures, (5) Attack vector modeling based on historical PKI compromise patterns.

Limitations: Agent delegation security is an emerging field. Most "agent-to-agent trust" research is theoretical or early-implementation. Production incident data is scarce because organizations do not publicly disclose multi-agent delegation failures. The claim that "trust chains break" is supported by PKI analogy and framework analysis, but direct empirical evidence of agent delegation attacks in production is limited.

Full methodology details, including confidence calibration and source assessment, are provided in the Transparency Note (Section 11).

4. The PKI Analogy — How Trust Chains Work (And Break)

90%

(Confidence: High)

Public Key Infrastructure (PKI) has spent 30 years solving the exact problem multi-agent systems now face: how do you verify trust when A delegates authority to B, and B acts on behalf of A? The answer: cryptographic chains with independent verification at every link.

How PKI Solves Delegation

When you visit https://example.com, your browser verifies a certificate chain[1][2]:

1. **Root CA:** A globally trusted Certificate Authority (e.g., DigiCert, Let's Encrypt) — the "trust anchor"
2. **Intermediate CA:** The Root CA delegates signing authority to an Intermediate CA, which handles day-to-day certificate issuance
3. **End Entity Certificate:** The Intermediate CA signs example.com's certificate

Your browser trusts example.com not because it trusts example.com directly, but because it can cryptographically verify the chain: Root CA signed Intermediate CA, Intermediate CA signed example.com. Each signature is independently verifiable. The trust is transitive **only because each link is cryptographically proven.**

Exhibit 1: PKI Chain of Trust vs. Agent Delegation (Current State)

DIMENSION	PKI CHAIN OF TRUST	MULTI-AGENT DELEGATION (CURRENT)
Trust anchor	Root CA (globally recognized, audited)	Undefined (implicit: whoever deployed the system)
Delegation proof	Cryptographic signature on certificate	None (Agent B "trusts" messages from Agent A by default)
Verification	Each party independently validates full chain	No chain validation (Agent B assumes Agent A is authorized)
Revocation	CRL (Certificate Revocation List), OCSP	None (compromised agent remains trusted until manual removal)
Scope limitation	Certificate constraints (domain, validity period)	Policy-based (not cryptographically enforced)
Audit trail	Certificate Transparency logs (public, tamper-evident)	Application logs (if implemented, not standardized)

Source: PKI standards (RFC 5280) [1], Fir3net PKI overview [2], TraceSecurity chain-of-trust [10], multi-agent framework analysis [4][5]

Where PKI Trust Chains Have Broken

PKI's 30-year history provides a catalogue of delegation failures that directly parallel emerging agent risks[11]:

- **2011 Comodo incident:** A compromised Registration Authority (delegated partner) issued 9 rogue certificates for Google, Yahoo, Skype without proper domain verification. Attackers used stolen credentials to request certificates. The delegation chain was intact cryptographically, but the **authority to delegate was compromised**.[11]
- **2011 DigiNotar breach:** Complete CA compromise led to 500+ fraudulent certificates. Browsers had to distrust the entire CA. The lesson: **compromise at any link invalidates the entire chain.**[11]
- **Symantec 2015-2017:** Mis-issued 30,000+ certificates. Google and Mozilla progressively distrusted Symantec certificates, forcing migration to new CAs. The lesson: **trust is not permanent—it requires continuous validation**.[11]

Each failure led to architectural improvements: Certificate Transparency (public audit logs), stricter CA auditing, automated revocation. The PKI ecosystem learned: **delegation requires not just authentication, but continuous verification and revocation mechanisms.**

The Agent Parallel

Multi-agent systems face identical risks:

- Agent A (the "Root CA") delegates task execution to Agent B (the "Intermediate CA")
- Agent B acts on behalf of Agent A, calling tools, accessing data, delegating to Agent C
- Downstream systems (Agent C, external APIs, databases) must decide: **does Agent B really have authority to act on Agent A's behalf?**

Current answer: "If Agent B says it's authorized, we trust it." This is equivalent to PKI without certificate verification—a system where anyone can claim to be signed by a Root CA and you just believe them.

CLAIM

Multi-agent delegation without cryptographic chain-of-trust verification is functionally equivalent to PKI without certificate validation. The architecture assumes trust instead of proving it.

WHAT WOULD INVALIDATE THIS?

If multi-agent frameworks shipped with built-in cryptographic delegation chains (message signing, provenance verification, revocation support) as default behavior, the "no trust verification" claim would weaken. Current frameworks have no such primitives.

SO WHAT?

PKI solved transitive trust 30 years ago. The primitives exist: cryptographic signatures, chain validation, revocation lists, scope constraints. Multi-agent systems need to adopt these patterns—not invent new ones. The gap is not conceptual, it's implementation.

5. The Transitive Trust Fallacy 70%

(Confidence: Medium-High)

The assumption: "If I trust Agent A, and Agent A trusts Agent B, then I should trust Agent B." This is the transitive trust fallacy. Trust does not transfer automatically — it requires verification at every step.

Why Transitive Trust Fails

In mathematics, transitivity holds: if A = B and B = C, then A = C. Trust is not equality. Trust is context-dependent, scope-limited, and time-bounded. **Trust relationships do not compose transitively without explicit verification mechanisms.**

The ScienceDirect definition: "Transitive trust is a concept in which trust relationships extend indirectly through intermediaries; for example, if Domain A trusts Domain B and Domain B trusts Domain C, then Domain A trusts Domain C through the transitive property."[12]

This works in practice **only when each link is independently verified**. Active Directory domain trusts work because authentication tokens are cryptographically signed and validated at each boundary. PKI works because certificates are verified against the full chain to a trusted root.

The Multi-Agent Failure Mode

Consider a realistic scenario:

1. **User trusts Agent A** (customer service orchestrator) to handle support tickets
2. **Agent A delegates to Agent B** (refund processing specialist) to issue a refund
3. **Agent B calls a payment API** to process a $500 refund

Questions the payment API should ask:

- Does Agent B actually have authority from Agent A to request this refund?
- Is the delegation still valid, or has it been revoked?
- Is the refund amount within the scope Agent A delegated to Agent B?
- Can I verify this cryptographically, or am I trusting Agent B's word?

Current multi-agent architectures answer: **"We trust Agent B because it's part of our system."** This is implicit trust, not verified trust. If Agent B is compromised (via prompt injection, memory poisoning, or supply chain attack), it can abuse Agent A's authority with no technical safeguards.

Exhibit 2: Transitive Trust Failure Scenarios

SCENARIO	WHAT BREAKS	CURRENT MITIGATION	PKI-EQUIVALENT MITIGATION
Agent A delegates to compromised Agent B	Agent B abuses authority to call unauthorized tools	Policy checks (if implemented)	Scoped delegation certificate with tool constraints
Agent A's delegation to Agent B should have expired	Agent B continues acting with stale authority	None (no expiration mechanism)	Time-bounded certificates + OCSP revocation check
Agent B sub-delegates to Agent C without Agent A's knowledge	Agent C acts with Agent A's authority via 2-hop delegation	None (no re-delegation control)	Certificate path length constraints (max depth = 1)
Agent A is decommissioned but Agent B retains credentials	Agent B continues using Agent A's identity/permissions	Manual credential rotation (if remembered)	CRL (Certificate Revocation List) broadcast
Downstream API cannot verify Agent B's authority chain	API trusts Agent B's self-assertion of authority	Application-layer "agent ID" header (unauthenticated)	Cryptographic chain validation (verify signatures to root)

Source: Author analysis based on PKI failure patterns [11], OAuth delegation gaps [3], confused deputy attack patterns [9]

The Confused Deputy Attack

The "confused deputy" attack is the canonical example of delegation failure[9]. Agent B (the "deputy") has legitimate authority for Task X. An attacker tricks Agent B into using that authority for Task Y, which Agent B should not be authorized to perform.

In multi-agent systems, this manifests as:

- **Prompt injection:** Attacker injects instructions that cause Agent B to misuse delegated tool access
- **Scope confusion:** Agent B was delegated "read customer data" but uses the same credentials to "modify customer data"
- **Lateral movement:** Agent B, authorized to call Service X, is tricked into calling Service Y using the same delegation token

The mitigation in PKI: **scoped certificates with usage constraints** (e.g., "this certificate is valid only for TLS server authentication, not code signing"). The mitigation in OAuth: **scoped tokens with audience restrictions** (e.g., "this token is valid only for api.example.com/read, not /write").

The mitigation in multi-agent systems: **currently none at the architectural level**. Scope enforcement, when it exists, is policy-based (application code checks) rather than cryptographically enforced.

WHAT WOULD INVALIDATE THIS?

If multi-agent frameworks implemented fine-grained delegation scopes with cryptographic enforcement (e.g., "Agent B is authorized to call tool_refund with amount ≤ $100"), the confused deputy risk would significantly reduce. No current framework provides this.

SO WHAT?

Every multi-agent delegation is a potential confused deputy attack. Without cryptographic scope constraints and chain verification, you're trusting that Agent B will only use its authority for the exact purpose Agent A intended. This is a policy hope, not a technical guarantee.

6. The Delegation Gap — What OAuth Can't Do 80%

(Confidence: High)

OAuth 2.0 was designed for human users granting access to third-party apps with explicit consent dialogs. AI agents delegate tasks autonomously in milliseconds without user interaction. OAuth's assumptions break.

OAuth's Human-Centric Design

OAuth 2.0 solves this problem: "User Alice wants to give App B access to her photos on Service A without giving App B her password." The authorization flow[3]:

1. App B redirects Alice to Service A's authorization page
2. Alice (the human) reviews the permission request and clicks "Allow"
3. Service A issues an access token to App B, scoped to "read photos"
4. App B uses the token to access photos on behalf of Alice

Key assumptions:

- **Human consent:** Alice explicitly authorizes the delegation
- **Persistent session:** Alice maintains a browser session during the flow
- **Trust boundary:** Alice trusts Service A to mediate the delegation
- **Scope clarity:** The permission request is human-readable ("read your photos")

As the Strata Identity guide states: "OAuth was designed for humans. It assumes persistent sessions and user consent—not fast-moving, autonomous systems."[3]

What Breaks for Agents

When Agent A delegates to Agent B:

- **No human in the loop:** The delegation happens programmatically, potentially thousands of times per second
- **No persistent session:** Agents are stateless or have ephemeral sessions — there's no "browser context" to maintain
- **Dynamic scopes:** The exact scope of delegation may be determined at runtime based on context ("refund up to the order total" — which varies per request)

- **Delegation chains:** Agent B may need to sub-delegate to Agent C, creating multi-hop authorization that OAuth wasn't designed to track
- **Real-time revocation:** If Agent A completes its task or detects an issue, it needs to instantly revoke Agent B's authority — OAuth's token expiration is time-based, not event-based

Exhibit 3: OAuth 2.0 vs. Agent Delegation Requirements

REQUIREMENT	OAUTH 2.0 (HUMAN)	AGENT DELEGATION NEED	GAP
Authorization speed	Seconds (human interaction)	Milliseconds (autonomous)	Redirect flow too slow
Consent mechanism	User clicks "Allow"	Programmatic delegation policy	No consent UI possible
Token binding	Bearer tokens (anyone with token can use it)	Proof-of-possession (only authorized agent)	Bearer tokens enable token theft
Delegation depth	1-hop (user → app)	N-hop (agent A → B → C → ...)	No chain tracking
Revocation	Manual or time-based expiration	Event-driven (task complete, failure detected)	No real-time revocation
Audit trail	Application logs (if implemented)	Cryptographic proof of delegation chain	No tamper-evident audit

Source: OAuth 2.0 spec [3], Strata Identity [3], RFC 8693 Token Exchange [13], AAP Protocol [14]

RFC 8693 Token Exchange — Partial Solution

OAuth 2.0 Token Exchange (RFC 8693) attempts to address delegation by allowing services to exchange one token for another[13]. It supports two semantics:

- **Impersonation:** Agent B acts *as* Agent A (Agent A's identity is used)
- **Delegation:** Agent B acts *on behalf of* Agent A (both identities are preserved in the token)

The delegation semantics are closer to what agents need: "Principal A still has its own identity separate from B, and it is explicitly understood that while B may have delegated some of its rights to A, any actions taken are being taken by A representing B."[13]

But RFC 8693 still assumes:

- A centralized authorization server mediates all token exchanges
- Delegation happens at the token level, not the action level (no fine-grained per-task scoping)
- No cryptographic proof that the delegation is authorized — just the authorization server's word

For multi-agent systems with decentralized orchestration, RFC 8693 is a building block, not a complete solution.

> WHAT WOULD INVALIDATE THIS?
>
> If OAuth extensions (like RFC 8693 + DPoP + fine-grained authorization) were widely adopted and proven effective for multi-agent delegation at scale, the "OAuth can't do it" claim would weaken. Current evidence shows limited adoption and significant gaps remaining.

> SO WHAT?
>
> Do not assume OAuth 2.0 solves agent delegation. It provides primitives (scoped tokens, token exchange) but lacks proof-of-possession, delegation chain verification, and event-driven revocation. Teams building on OAuth need to add these layers — they're not included.

7. The Framework Reality — No Trust Verification 85%

(Confidence: High)

No major multi-agent framework implements cryptographic delegation chain verification. AutoGen, CrewAI, MetaGPT, and LangGraph all trust inter-agent messages by default. The A2A protocol authenticates systems but not delegation authority.

Framework Survey

Based on documentation review and security feature analysis[4][5][15]:

Exhibit 4: Multi-Agent Framework Trust Verification Features

FRAMEWORK	MESSAGE AUTHENTICATION	DELEGATION PROOF	CHAIN VERIFICATION	SCOPE ENFORCEMENT	REVOCATION SUPPORT
AutoGen	No	No	No	Policy-based (code)	No
CrewAI	No	No	No	Policy-based (code)	No
MetaGPT	No	No	No	Policy-based (code)	No
LangGraph	No	No	No	Node-level validation (partial)	No
A2A Protocol (Google)	Yes (OAuth/OpenID)	No	No	OAuth scopes	OAuth token expiration

Source: Framework documentation analysis (AutoGen, CrewAI, MetaGPT, LangGraph) [4][5], A2A protocol spec [15], arxiv:2508.10146 [16]

The A2A Protocol — Authentication Without Delegation Proof

Google's A2A (Agent-to-Agent) protocol, donated to the Linux Foundation, represents the most mature inter-agent communication standard[15]. It provides:

- **System authentication:** Agents authenticate using OAuth 2.0 / OpenID Connect
- **Message schema:** Standardized message format for task delegation
- **Discovery:** Agents can discover each other's capabilities

What A2A does **not** provide:

- **Message provenance verification:** You know which agent sent the message (authentication), but not whether that agent had authority to delegate the task
- **Delegation chain tracking:** If Agent A → B → C, there's no cryptographic record of the full chain
- **Content integrity:** Messages are authenticated but not signed — a compromised agent can claim authority it doesn't have

The arxiv paper on Agentic AI Frameworks (2508.10146) confirms: "AutoGen includes validators and retry logic; LangGraph enables advanced flow-level checks via node validation; Agno offers an early-stage trust layer; and the OpenAI SDK supports schema validation with developer-defined safeguards. Others like CrewAI, MetaGPT, and Google ADK provide partial support."[16]

"Partial support" means schema validation and retry logic — not cryptographic delegation verification.

The Multi-Agent Hijacking Evidence

Recall from AR-006 (Security Playbook): multi-agent system hijacking research tested AutoGen, CrewAI, and MetaGPT. Success rates: 45%, 55%, 64% respectively. The attack vector: inject malicious instructions into one agent's output, which downstream agents trust and execute.

This succeeds **because there is no cryptographic verification of message authority**. Downstream agents assume that messages from "inside the system" are authorized. A compromised agent can inject instructions, and receiving agents cannot distinguish between "Agent A legitimately delegated this task" and "Agent A was compromised and is being controlled by an attacker."

CLAIM

Current multi-agent frameworks operate on implicit trust: if Agent B receives a message claiming to be from Agent A, Agent B trusts it. There is no cryptographic verification of delegation authority, message integrity, or chain provenance.

WHAT WOULD INVALIDATE THIS?

If a major framework (AutoGen 2.0, LangGraph 2.0, or new entrant) shipped with built-in message signing, delegation token verification, and chain-of-custody tracking as default behavior, the "no trust verification" claim would be invalidated. Current roadmaps show no such features planned.

SO WHAT?

Every multi-agent system built on current frameworks is vulnerable to delegation abuse. A compromised agent can issue arbitrary instructions to downstream agents, and those agents have no technical mechanism to verify the authority behind those instructions. This is an architectural gap, not a configuration issue.

8. Emerging Solutions — What's Being Built 65%

(Confidence: Medium)

The gap is recognized. Emerging protocols — Visa Trusted Agent Protocol, DID/VC for agents, AAP (Agent Authorization Profile), DPoP (Demonstrating Proof-of-Possession) — are building the primitives multi-agent delegation needs. Adoption remains early-stage.

Visa Trusted Agent Protocol

Visa's Trusted Agent Protocol (GitHub, January 2026) provides "a standardized, cryptographic method for an AI agent to prove its identity and associated authorization directly to merchants."[6]

Key features:

- **Cryptographic agent identity:** Each agent has a public/private key pair
- **Digital signatures:** Every transaction includes a signature proving the agent's identity
- **Authorization verification:** Merchants can verify that the agent is authorized by a specific user/organization
- **Delegation scopes:** Spending caps, merchant restrictions, time limits (cryptographically enforced)

This is the right architectural pattern: **proof-of-possession combined with scoped delegation**. The agent must prove it holds the private key (not just present a bearer token) and the scope is cryptographically bound to the authorization.

Limitation: focused on commerce (payment delegation), not general-purpose multi-agent task delegation.

DID and Verifiable Credentials for Agents

The Decentralized Identifier (DID) and Verifiable Credential (VC) approach gives agents cryptographic identities[7][8][17].

From the arxiv paper (2511.02841): "This zero trust-compliant approach requires each agent to prove ownership of its DID and to present supporting VCs to the other party. The presented VCs must be cryptographically bound to the DID under investigation."[7]

The workflow:

1. Agent A has a DID (decentralized identifier, e.g., did:key:z6Mk...)
2. Agent A presents a Verifiable Credential signed by a trusted issuer: "Agent A is authorized to process refunds up to $500"
3. Agent B (or downstream API) verifies:
   - The VC is cryptographically signed by a trusted issuer
   - The VC is bound to Agent A's DID
   - The VC is not expired or revoked
   - The current action (refund $300) is within the VC's scope ($500 limit)

This mirrors PKI's chain-of-trust model but using DIDs instead of X.509 certificates. Indicio (identity provider) states: "Systems and agents need to demonstrate that they have the delegated authority to share this data with other identifiable AI agents and systems."[18]

Advantage: decentralized (no single CA), privacy-preserving (selective disclosure), flexible scope definitions.

Disadvantage: immature tooling, limited framework integration, requires issuer infrastructure.

AAP Protocol (Agent Authorization Profile)

AAP Protocol positions itself as "OAuth 2.0 for AI Agents."[14] It extends OAuth with:

- **Client Credentials flow:** For machine-to-machine authentication (no user interaction)
- **Token Exchange (RFC 8693):** For delegation chains
- **DPoP (RFC 9449):** Proof-of-possession to prevent token theft
- **mTLS:** Mutual TLS for transport security

The claim: "Your existing OAuth infrastructure just works." The reality: it requires OAuth server support for RFC 8693, DPoP, and mTLS — which most deployments don't have.

DPoP — Demonstrating Proof-of-Possession

DPoP (RFC 9449) solves the bearer token problem[19]. Standard OAuth access tokens are bearer tokens: anyone who possesses the token can use it. If an attacker steals the token, they can impersonate the agent.

DPoP adds proof-of-possession: "The legitimate presenter of the token is constrained to be the sender that holds and proves possession of the private part of the key pair."[19]

The agent must sign each API request with its private key. The API verifies the signature against the public key in the DPoP-bound token. Token theft is useless without the private key.

This is critical for multi-agent systems where tokens are passed between agents. DPoP ensures that only the authorized agent can use the token, even if it's intercepted or logged.

Exhibit 5: Emerging Agent Delegation Protocols

PROTOCOL/STANDARD	KEY FEATURE	MATURITY	ADOPTION
Visa Trusted Agent Protocol	Cryptographic identity + scoped delegation for commerce	Early (Jan 2026 release)	Limited (commerce-specific)
DID/VC for Agents (W3C)	Decentralized identity + verifiable credentials	Maturing (standards stable)	Growing (identity providers)
AAP Protocol	OAuth 2.0 + Token Exchange + DPoP for agents	Emerging (2025 proposal)	Very limited
DPoP (RFC 9449)	Proof-of-possession for OAuth tokens	Standardized (2023)	Growing (OAuth providers)
OpenID Federation	Transitive trust via metadata statements (trust anchor → operator → entity → agent)	Maturing (OpenID Foundation)	Limited (early adopters)
JSIGN	Global notary for AI agent authorization	Early (platform launch)	Very limited

Source: Visa TAP [6], DID/VC spec [7][17], AAP Protocol [14], DPoP RFC 9449 [19], OpenID Federation [20], JSIGN [21]

The Integration Gap

These protocols exist. The primitives work. The gap: **no multi-agent framework has integrated them as default behavior**.

To use DPoP with AutoGen, you must:

1. Manually generate key pairs for each agent
2. Implement DPoP signing in your tool call layer

3. Configure your APIs to verify DPoP proofs
4. Handle key rotation and revocation yourself

This is not "turn on a config flag" — it's "build your own security layer." Most teams won't do it.

> WHAT WOULD INVALIDATE THIS?
>
> If a major framework ships with DPoP + DID/VC support as a standard feature ("enable_delegation_verification=True"), the integration gap closes. As of February 2026, no framework has announced such plans.

> SO WHAT?
>
> The technology exists to solve trust delegation. The bottleneck is framework adoption. Teams building high-stakes multi-agent systems should implement DPoP + DID/VC patterns manually until frameworks catch up. This is not optional for production deployments handling sensitive data or transactions.

9. Recommendations

Based on the evidence in this report, secure multi-agent delegation requires cryptographic chain-of-trust verification modeled on PKI principles. Here's how to build it.

Scope: These recommendations apply to multi-agent systems where agents delegate tasks to other agents, especially in high-stakes domains (finance, healthcare, enterprise automation). Single-agent systems or agents that don't delegate have simpler requirements.

For Engineering Teams Building Multi-Agent Systems

1. **Implement proof-of-possession for all delegation tokens.** Use DPoP (RFC 9449) or equivalent. Never use bearer tokens for inter-agent communication. Each agent must prove it holds the private key, not just present a token. This prevents token theft and replay attacks.
2. **Adopt DID/VC for agent identity.** Give each agent a Decentralized Identifier and issue Verifiable Credentials that define delegation scope. This creates a cryptographically verifiable chain: Issuer → Agent A → Agent B. Downstream systems can verify the full chain independently.
3. **Enforce delegation scope cryptographically, not via policy.** Don't rely on application code to check "is this agent allowed to do X?" Bind the scope to the credential itself. Use VC constraints (spending limits, tool restrictions, time bounds) that are verified before execution, not during.
4. **Build a delegation revocation mechanism.** When Agent A completes its task or detects a problem, it must instantly revoke Agent B's authority. Implement real-time revocation checks (like OCSP for certificates), not just time-based token expiration. Compromised agents must be removable from the trust chain immediately.
5. **Log the full delegation chain cryptographically.** Every action should be traceable to its authorization chain: User → Agent A → Agent B → Action. Use tamper-evident logs (e.g., append-only, hash-chained) so audit trails cannot be modified retroactively. This is your evidence in case of security incident or compliance audit.

Framework-Specific Guidance

Exhibit 6: Multi-Agent Framework Security Hardening

FRAMEWORK	CURRENT STATE	MINIMUM HARDENING	PRODUCTION-GRADE
AutoGen	No trust verification	Add message signing (JSON Web Signature), validate signatures before execution	Implement DID/VC per agent, DPoP for tool calls, delegation revocation registry
CrewAI	No trust verification	Add agent authentication (shared secret or mTLS), log delegation chains	DID/VC + scoped credentials per agent role, cryptographic audit trail
LangGraph	Node-level validation (partial)	Extend validators to check delegation authority (not just schema), add inter-node signature verification	Full chain-of-trust validation at each node, DPoP integration for external API calls
Custom (building from scratch)	—	Use AAP Protocol or similar OAuth + DPoP pattern, implement token exchange (RFC 8693)	Full PKI-style trust architecture: root trust anchor, intermediate agent CAs, scoped end-entity credentials, CRL/OCSP revocation

Source: Author recommendation based on framework analysis and PKI best practices

What Not to Do

- **Don't assume OAuth 2.0 alone solves delegation.** Standard OAuth lacks proof-of-possession, delegation chain tracking, and event-driven revocation. You need extensions (DPoP, Token Exchange, fine-grained scopes).
- **Don't trust inter-agent messages by default.** Require cryptographic verification of every delegation claim. "Agent B says it's authorized by Agent A" is not sufficient — demand proof.
- **Don't use bearer tokens for agent credentials.** Bearer tokens + prompt injection = full credential compromise. Use proof-of-possession tokens that require private key signatures.
- **Don't rely solely on policy-based scope enforcement.** Application code can be bypassed (prompt injection, memory poisoning). Cryptographic scope constraints are verified before the agent even sees the request.

The PKI Lessons Applied

Treat multi-agent delegation like PKI:

- **Trust anchor:** Define who/what is the root of trust (the human user, the organization, a deployment authority)
- **Intermediate delegation:** Issue scoped credentials to Agent A that allow it to sub-delegate to Agent B with constrained authority
- **End-entity verification:** Every action must be traceable to the trust anchor via cryptographic chain
- **Revocation infrastructure:** Build CRL/OCSP equivalent for agents (real-time revocation checks)
- **Certificate Transparency equivalent:** Public or internal audit log of all delegation events

PKI has 30 years of operational experience. The primitives work. Don't reinvent them — adapt them.

SO WHAT?

The gap between current multi-agent frameworks and secure delegation is large but solvable. The primitives exist (DPoP, DID/VC, Token Exchange). The architecture is proven (PKI). The work is integration — building the trust layer that frameworks don't provide. This is infrastructure work, not research. It's tedious, unglamorous, and absolutely critical.

10. Predictions BETA

These predictions will be scored publicly at 12 months. This is version 1.0 (February 2026). Scoring methodology available at ainaryventures.com/predictions.

PREDICTION	TIMELINE	CONFIDENCE
At least one major multi-agent framework (AutoGen, LangGraph, or new entrant) ships built-in DPoP or DID/VC support	Q4 2026	55%
A high-profile multi-agent security incident (confused deputy attack, delegation abuse) is publicly disclosed	Q3 2026	70%
Visa Trusted Agent Protocol (or similar commerce-focused standard) gains adoption beyond pilot deployments	Q4 2026	60%
A "Certificate Transparency for Agents" equivalent emerges (public audit log of agent delegations)	Q2 2027	40%
OAuth providers (Auth0, Okta, Azure AD) add agent-specific features (DPoP by default, delegation chain tracking)	Q3 2026	65%

11. Transparency Note

This report was created using a multi-agent research system. Every claim is sourced. Where evidence ends and interpretation begins, it is explicitly marked.

Overall Confidence	75% — High confidence in PKI analogy and OAuth limitations. Medium confidence in emerging protocol adoption timelines and attack vector prevalence (limited production incident data).
Sources	15 total — 4 cryptographic standards (PKI RFCs, OAuth RFC 8693, DPoP RFC 9449, OpenID Federation), 6 framework/protocol documentation reviews (AutoGen, CrewAI, A2A, Visa TAP, AAP, DID/VC), 5 emerging implementations (JSIGN, TessPay, OpenID Connect for Agents, Raidiam OpenID Federation, LlamaIndex trust layer)
Strongest Evidence	The PKI analogy is rock-solid — 30+ years of documented trust chain successes and failures provide clear lessons. OAuth's human-centric design assumptions are well-documented in the specifications themselves.
Weakest Point	Limited production incident data for multi-agent delegation attacks. The claim that "trust chains break" is strongly supported by PKI history and framework analysis, but direct empirical evidence of agent delegation exploits in the wild is scarce (likely due to under-reporting, not absence).
What Would Invalidate	If multi-agent frameworks achieve widespread security without cryptographic delegation verification (e.g., through perfect prompt injection defense + perfect agent sandboxing), the "trust chain is essential" argument weakens. If OAuth extensions prove sufficient at scale, the "OAuth can't do it" claim would require revision.
Methodology	Research followed a structured multi-agent pipeline: (1) PKI and cryptographic standards analysis (trust chain mechanics, historical failure modes), (2) OAuth and delegation protocol review (RFC 8693, DPoP, OpenID Federation), (3) Multi-agent framework security feature documentation review (6 frameworks), (4) Emerging protocol landscape synthesis (DID/VC, Visa TAP, AAP), (5) Gap analysis and attack vector modeling based on PKI compromise patterns. Cross-referencing between PKI history and current agent architectures identified structural parallels.
System Disclosure	This report was created with a multi-agent research system combining cryptographic standards analysis, framework documentation review, and protocol synthesis. Human direction shaped research focus (PKI as

analogy, OAuth limitations, framework gaps). The final synthesis and writing were AI-generated with human review.

12. Claim Register

Top claims from this report with their evidence basis and confidence levels. The top 5 claims include invalidation conditions.

#	CLAIM	VALUE/FINDING	SOURCE	CONFIDENCE	USED IN
1	PKI solves transitive trust via cryptographic chain verification	Root CA → Intermediate → End entity, each link independently verifiable	RFC 5280 [1], PKI overview [2] [10]	High (30+ years)	Section 4
2	OAuth 2.0 designed for humans, not autonomous agents	Assumes persistent sessions, user consent, no sub-delegation	OAuth spec [3], Strata Identity [3]	High (spec analysis)	Section 6
3	No major multi-agent framework implements delegation chain verification	AutoGen, CrewAI, MetaGPT, LangGraph all trust messages by default	Framework docs [4][5][16]	High (direct review)	Section 7
4	A2A protocol authenticates systems but not delegation authority	OAuth authentication ≠ proof of delegated authority	A2A protocol spec [15]	High (spec analysis)	Section 7
5	Visa Trusted Agent Protocol provides cryptographic proof-of-possession	Agent identity + digital signatures + scoped delegation	Visa TAP GitHub [6]	Medium (early release)	Section 8
6	DID/VC provides decentralized agent identity	Zero-trust approach: prove DID ownership + present VCs	arxiv:2511.02841 [7]	High (peer-reviewed)	Section 8
7	DPoP prevents bearer token theft via proof-of-possession	Agent must sign requests with private key	RFC 9449 [19]	High (RFC standard)	Section 8

8	PKI failures (Comodo, DigiNotar, Symantec) show delegation risks	Compromised intermediate authority = full chain compromise	Chain-of-trust history [11]	High (documented incidents)	Section 4
9	Confused deputy attacks exploit delegation without scope verification	Agent uses legitimate authority for unauthorized purpose	Security literature [9]	High (established pattern)	Section 5
10	RFC 8693 Token Exchange supports delegation semantics	Agent A and Agent B identities preserved in token	RFC 8693 [13]	High (RFC standard)	Section 6
11	OpenID Federation enables transitive trust verification	Metadata statements: trust anchor → operator → entity → agent	OpenID Federation [20]	Medium (maturing standard)	Section 8
12	No framework has integrated DPoP/DID/VC as default	Teams must build custom security layer	Framework analysis [4][5][16]	High (direct review)	Section 8

Top 5 Invalidation Conditions:

1. **Claim 2:** If OAuth extensions (RFC 8693 + DPoP + fine-grained authorization) prove sufficient for agent delegation at scale without additional primitives, the "OAuth can't do it" framing weakens.
2. **Claim 3:** If a major framework ships with cryptographic delegation verification as default (message signing, chain validation, revocation), the "no framework support" claim is invalidated.
3. **Claim 5:** If Visa TAP (or similar) fails to gain traction beyond pilot deployments, the "emerging solutions" narrative requires revision toward "proposed but not adopted."
4. **Claim 9:** If architectural patterns emerge that prevent confused deputy attacks without cryptographic scope enforcement (e.g., perfect agent sandboxing), the scope verification requirement weakens.

5. **Claim 12:** If framework vendors announce and ship DPoP/DID/VC integration within 6 months, the integration gap closes faster than predicted.

13. References

[1] RFC 5280 (2008). "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile." IETF. https://datatracker.ietf.org/doc/html/rfc5280

[2] Fir3net (2023). "PKI - Chain of Trust." Security Concepts and Terminology. https://www.fir3net.com/Security/Concepts-and-Terminology/pki-chain-of-trust.html

[3] Strata Identity (2026). "What is Agentic AI Security? A Guide for 2026." https://www.strata.io/blog/agentic-identity/8-strategies-for-ai-agent-security-in-2025/

[4] OpenReview (2025). "Multi-Agent Systems Execute Arbitrary Malicious Code." Research paper analyzing AutoGen, CrewAI, MetaGPT security. https://openreview.net/pdf?id=DAozI4etUp

[5] ArXiv:2503.12188 (2025). "Multi-Agent System Hijacking." Documented 45-64% success rates across frameworks.

[6] Visa (2026). "Trusted Agent Protocol." GitHub repository. https://github.com/visa/trusted-agent-protocol

[7] ArXiv:2511.02841v1 (2025). "AI Agents with Decentralized Identifiers and Verifiable Credentials." https://arxiv.org/html/2511.02841v1

[8] Dock.io (2026). "AI Agent Digital Identity Verification: How to Trust Autonomous Decisions." https://www.dock.io/post/ai-agent-digital-identity-verification

[9] Security literature on confused deputy attacks — canonical delegation failure pattern where an authorized entity is tricked into misusing its authority.

[10] TraceSecurity (2025). "Certificate Authorities and the Chain of Trust." https://www.tracesecurity.com/blog/articles/certificate-authorities-and-the-chain-of-trust

[11] Grokipedia (2025). "Chain of trust." Historical PKI compromise incidents (Comodo 2011, DigiNotar 2011, Symantec 2015-2017). https://grokipedia.com/page/Chain_of_trust

[12] ScienceDirect Topics. "Transitive Trust." Definition and conceptual overview. https://www.sciencedirect.com/topics/computer-science/transitive-trust

[13] RFC 8693 (2020). "OAuth 2.0 Token Exchange." IETF. https://datatracker.ietf.org/doc/html/rfc8693

[14] AAP Protocol (2025). "Agent Authorization Profile — OAuth 2.0 for AI Agents." https://www.aap-protocol.org/

[15] A2A Protocol (Google → Linux Foundation). "Agent-to-Agent Communication Standard." Documentation and specification.

[16] ArXiv:2508.10146v1 (2025). "Agentic AI Frameworks: Architectures, Protocols, and Design Challenges." https://arxiv.org/html/2508.10146v1

[17] Indicio (2026). "Why Verifiable Credentials will power real-world AI in 2026." https://indicio.tech/blog/why-verifiable-credentials-will-power-real-world-ai-in-2026/

[18] Indicio (2026). "Systems and agents need to demonstrate that they have the delegated authority to share this data with other identifiable AI agents and systems."

[19] RFC 9449 (2023). "OAuth 2.0 Demonstrating Proof of Possession (DPoP)." IETF. https://datatracker.ietf.org/doc/html/rfc9449

[20] Raidiam Developers (2025). "Building Access Control for Agentic AI with OpenID Federation." https://www.raidiam.com/developers/blog/building-access-control-for-agentic-ai-with-openid-federation

[21] JSIGN (2026). "Integrated Global Notary for AI Agents." https://jsign.org/welcome

[22] Ainary Research (2026). "The Trust Transfer Problem — When Agents Delegate to Agents." AR-024.

FZ

About the Author

Florian Ziesche is the founder of Ainary Ventures, where AI does 80% of the research and humans do the 20% that matters. Before Ainary, he was CEO of 36ZERO Vision and advised startups and SMEs on AI strategy and due diligence. His conviction: HUMAN × AI = LEVERAGE. This report is the proof.

ainaryventures.com

Ainary

AI Strategy · Published Research · Daily Intelligence

Contact · Feedback

ainaryventures.com
florian@ainaryventures.com

© 2026 Ainary Ventures