NON-HUMAN IDENTITY SERIES · PAPER #4

Delegation Without Escalation

Capability tokens, attenuation discipline, and the patterns that survive in production

Mohamad Amin Hasbini

Independent researcher · Paris, France

Published May 12, 2026

ABSTRACT

Delegation chains in production silently grow scope, extend time, and lose revocation visibility. Each individual hop is correct in isolation, but the composed chain authorizes what no single hop intended. Capability tokens with attenuation discipline at every hop, time-bounded forward chains, and revocation propagation that survives the chain are the only architecture where delegation depth does not compound risk.

KEYWORDS — AI agent security, non-human identity, post-quantum cryptography, agent authorization, capability tokens, NIS2, DORA, EU AI Act

---

CITE AS

Hasbini, M. A. (2026). *Delegation Without Escalation*: Capability tokens, attenuation discipline, and the patterns that survive in production. Non-Human Identity Series, Paper #4.

Available at https://mahasbini.org/papers/04-delegation-without-escalation/
PDF https://mahasbini.org/publications/papers/04-delegation-without-escalation.pdf

Delegation Without Escalation

Capability tokens, attenuation discipline, and the patterns that survive in production

Non-Human Identity Series, Paper 4 of 5

Changelog

- **v1.1 (May 14, 2026)**: Citations added for in-flight IETF/SPIFFE work (WIT-SVID, WIMSE architecture draft, OAuth-SPIFFE client-auth draft). Discipline-vs-primitive framing paragraph added. Acknowledgments section added for peer review by Agustin Martinez Fayo. No thesis-level changes; primitive-layer prior art surfaced through peer review.
- **v1.0 (May 12, 2026)**: Initial publication.

TL;DR

- **The escalation problem.** Delegation chains in production silently grow scope, extend time, and lose revocation visibility. Even when each individual hop is correct by RBAC standards, the composed chain produces capabilities that no single hop authorized. Agent ecosystems amplify this because chains run deeper and faster than human delegation chains ever did.
- **The discipline.** Strict-subset scope narrowing at every hop. Time-bounded forward-only chains. Revocation that propagates outside the chain. Macaroons (Birgisson et al. 2014) introduced the cryptographic primitive twelve years ago; the contribution this paper makes is the production discipline that turns the primitive into reliably enforceable architecture in agent-to-agent settings.
- **Production patterns.** Three patterns survive: capability-token-as-receipt with hash-bound caveats, depth limits with explicit policy at the boundary, and revocation epochs anchored to a transparency log. Three patterns fail: scope-as-string-prefix, soft expiry "with grace period," and revocation that depends on every node in the chain being reachable.
- **The compliance angle.** EU AI Act Annex IV, NIS2 Article 21, and DORA Chapter II push regulated organizations to evidence how AI-enabled actions are controlled, governed, and reconstructable. Without strict attenuation, a regulated institution cannot show that a downstream action stayed within the originating authorization. Attenuation discipline is therefore not just security best practice. It is auditability infrastructure.

The $200K Dot-Dash Heist

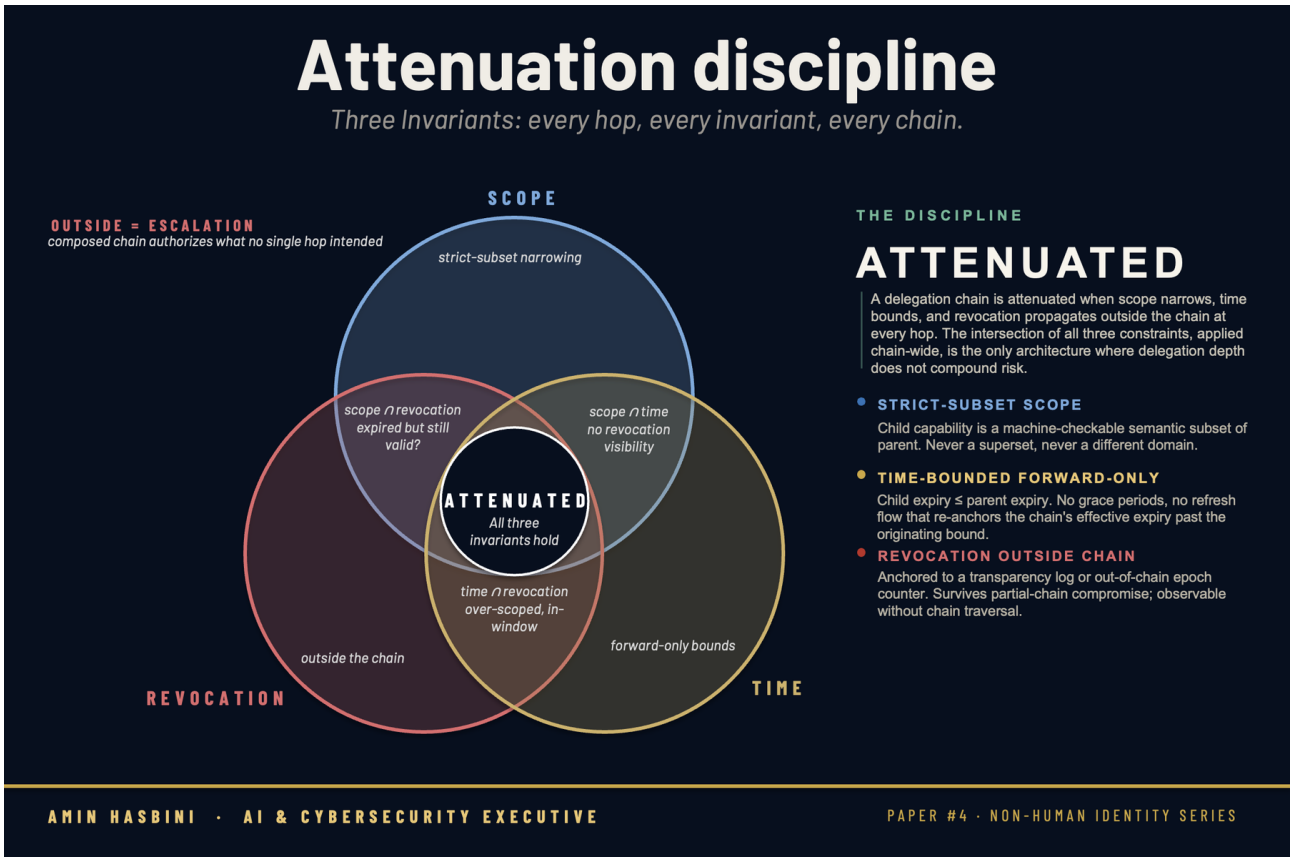
On May 4, 2026, an attacker drained roughly $200,000 from xAI's Grok agent by sending a Bankr Club Membership NFT to Grok's wallet, which expanded Grok's permissions inside the Bankr ecosystem to include token transfers and swaps. The attacker then tweeted Morse code at Grok asking for translation. Grok decoded it faithfully. The decoded text instructed Bankrbot to send three billion DRB tokens to the attacker's wallet on Base. Bankrbot executed. The transaction settled. OECD.AI registered the incident as 2026-05-04-4a73.

Strip the crypto context and the structural failure is recognizable in any enterprise agent system. An agent received expanded capability through a side channel. The agent then forwarded an instruction through a delegation chain without a scope check at the boundary. The capability had no attenuation, no time bound it could not silently extend, and no revocation visibility once execution began.

This is not new. In April 2026, Vercel disclosed a breach that originated from Context.ai, a third-party AI tool whose Google Workspace OAuth app had been compromised; the attacker used that access to take over a Vercel employee's Workspace account and pivot into internal systems. In 2025, the Salesloft Drift incident showed the same structural pattern at SaaS scale: compromised OAuth tokens associated with the Salesloft Drift integration enabled broad access into more than 700 customer environments. The common failure is not that every downstream action was locally invalid. The failure is that delegated authority travelled farther than the originating trust decision could safely bound.

The IETF OAuth working group is discussing this failure mode under the label "Delegation Chain Splicing in RFC 8693 Token Exchange," in a February 2026 mailing-list thread. The thread describes the mode as a compromised intermediary presenting mismatched `subject_token` and `actor_token` inputs from different delegation contexts to a token exchange endpoint, exploiting RFC 8693's absence of cross-validation between the two. Mitigations under discussion: cryptographic binding of step-N audience to step-N+1 subject, short token lifetimes, and back-channel revocation on consent withdrawal. RFC 8693 itself recognizes that delegation and impersonation introduce abuse risk and points to scopes and limited token lifetimes as mitigations, but does not impose strict attenuation discipline by default. The Grantex State of AI Agent Security 2026 report, an audit of thirty popular open-source AI agent projects, finds 93% use unscoped API keys, 0% have per-agent cryptographic identity, and 100% have no per-agent revocation. The Gravitee State of AI Agent Security 2026 report adds that only 21.9% of teams treat agents as independent identities and 45.6% rely on shared credentials for agent-to-agent authentication.

> ***Delegation chains do not fail because agents are malicious. They fail because authorization that worked at depth zero silently expands at depth N when scope, time, and revocation are not strictly attenuated at every hop. Capability tokens with attenuation discipline (strict-subset scope narrowing, time-bounded forward chains, revocation propagation that survives the chain rather than relying on it) are the only architecture where delegation depth does not compound risk.***



Attenuated = where all three invariants hold across the chain. Outside any of them is escalation.

Part I: The escalation problem

The escalation problem in delegation is not the same as privilege escalation in operating systems or networks. It is quieter, slower, and rarely visible in any single test. It manifests only in composition.

What “escalation” means in delegation

Three forms of escalation matter in agent delegation, and they are not interchangeable:

- **Scope escalation**: the set of resources or actions a capability authorizes grows beyond its original specification, often through pattern-matching that admits resources the original delegator never intended.
- **Time escalation**: the validity window of a capability extends beyond its originally authorized expiry, typically through grace periods, refresh flows, or chain extensions that re-anchor expiry to a later parent.
- **Identity escalation**: the principal who can act under a capability shifts away from the original delegating identity, either through actor-claim manipulation or through downstream binding to a different subject.

Each form has different mitigations. Conflating them produces architectures that defend against one and silently allow the other two.

Why agent delegation is different from human delegation

Human delegation chains run on hours-to-days timescales. A manager delegates authority to a deputy at 9 AM; the deputy may or may not exercise it before end of day; revocation is verbal or written, propagated to the relevant audit trail before the next quarterly review. The cycle is slow enough that humans-in-the-loop catch most escalation before it compounds.

Agent delegation chains run on hundred-millisecond timescales. The same chain that takes a human three days to traverse can complete in under a second across three or four agents. Asynchrony adds a wrinkle: hop N may execute long before hop N+1 even decides whether to delegate further. Multi-tenant orchestration adds another wrinkle: the same agent in two different tenants may be operating under entirely different scope guarantees, and the boundary between tenants is not always visible at the protocol level.

Three operational realities follow, each shaping a different escalation surface that human delegation never had to absorb.

Revocation latency in human delegation is forgiving because chains are slow. By the time a delegated authority is exercised, the revoking principal often has time to reach the executing party out-of-band. In agent chains, the gap between revocation issuance and revocation enforcement is the entire window of unauthorized action, and that window is measured in seconds at best. Without a revocation channel that lives outside the chain, partial-chain compromise extends the window indefinitely: the revocation request may not propagate past the compromised intermediate, and the downstream subtree continues to honor authority that no principal stands behind.

Asynchronous chain composition decouples authorization from execution. In a synchronous chain, every hop happens in causal order: parent issues, child receives, grandchild receives a further-attenuated capability, all observable within the same trace. In agent ecosystems, hop N+1 may delegate to a future hop that does not execute until hours later, after a queue depth or scheduling delay. The trace that documents the authorization chain may have already been written and archived by the time the action occurs. Audit reconstruction requires correlating timestamps across the chain after the fact, which is exactly when partial-chain compromise hides best.

Multi-tenant orchestration introduces a third complication. The same agent identity may operate across two or more tenants with different policy regimes, different caveat semantics, and different audit obligations. A capability issued in Tenant A may travel into Tenant B's processing pipeline if the agent does not partition its execution context correctly. The protocol layer rarely makes the boundary visible: a token is a token; only the orchestration layer knows which tenant context this particular execution belongs to. Production architectures that survive at this depth either enforce strict per-tenant agent identity or prefix every capability with the tenant context as an immutable caveat.

Seven things that go wrong without attenuation

When attenuation discipline is missing, seven failure modes recur across production deployments:

1. **Scope-by-prefix matches new resources after deployment changes.** A capability scoped to `/files/finance/*` continues to match after a folder reorganization adds `/files/finance/legal/`, granting access that no human approved. *Field pattern: this is the recurring story behind quiet permission expansions in SharePoint and Google Drive deployments where security teams approved the original scope years before a department restructure created the new sub-tree. The capability did not change; the resource graph did, and pattern matching admitted what was never reviewed.*

2. **Time-by-default-grace allows reuse past the intended window.** Operational logic that says "expired but recently valid" defeats the time-attenuation invariant entirely. *Field pattern: the five-minute clock-skew grace window common in enterprise OAuth deployments. Designed to absorb clock drift between issuer and resource server. In practice, it gives any captured token a five-minute extension beyond the cryptographic expiry, which is exactly the window an automated exfiltration pipeline needs.*

3. **Forward chains carry revocation-blind branches.** Once a capability is delegated downstream, revocation that requires walking the chain back may never reach branches the delegator did not even know existed. *Field pattern: the Salesloft Drift incident propagated through delegated OAuth scopes that the originating SaaS provider could not directly revoke at affected tenants. Each downstream branch had to reach into its own delegation tree for cleanup. Revocation became a coordination problem rather than an architectural property.*

4. **Repudiation when chain integrity fails.** Partial-chain compromise leaves no defensible audit trail of who authorized what at which depth. *Field pattern: agent ecosystems where one intermediate agent does not preserve the full delegation receipt. The regulator asking "show me the originating authorization for this action" cannot get a complete answer. The audit trail has a hole, and the hole is where the regulator focuses.*

5. **Cross-domain chain breaks at the trust boundary.** Token formats that cross organizational boundaries lose their attenuation guarantees when the receiving side does not enforce the same caveat semantics. *Field pattern: OAuth 2.0 token exchange across federation boundaries where Org A enforces caveat semantics and Org B treats the same token as a plain bearer token. The chain attenuates correctly inside A and silently expands the moment it crosses into B. The token was attenuated; the architecture wasn't.*

6. **Refresh tokens silently re-extend expiry.** A short-lived access token paired with a long-lived refresh token reproduces the long-lived problem at the refresh layer. *Field pattern: the canonical OAuth 2.0 design where the access token expires every sixty minutes and the refresh token lives ninety days. Operationally, the access-token expiry is theater; the refresh token is the real lifetime. Time attenuation requires that the refresh token also be bounded by the original delegation chain's expiry, not by a separate refresh policy.*

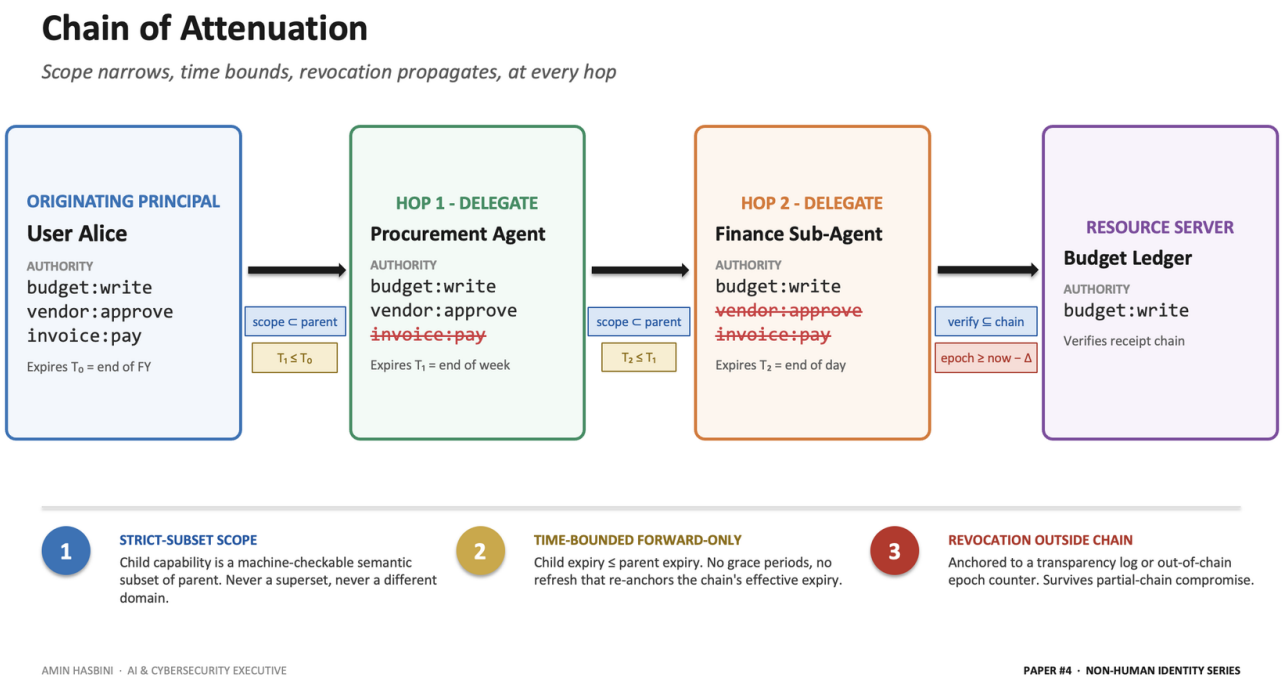
7. **Capability accumulation across multiple parents.** When an agent inherits capabilities from two or more parents, the union may exceed what any single parent intended. This is the lattice problem, and it is structural to delegation graphs that are not strict trees. *Field pattern: multi-tenant SaaS where one user-account has admin role in Tenant A and read-only role in Tenant B. An agent acting on behalf of that user across both tenants may compose capabilities in ways neither tenant approved. The lattice problem is the reason production delegation architectures should avoid unchecked union at graph joins. Where directed acyclic graphs are unavoidable, joins must use intersection or explicit reauthorization, not capability union.*

Why this isn't visible in single-hop testing

Each hop validates locally. The chain-level escalation only manifests in composition. A pen-test that exercises every individual delegation hop in isolation will pass; the same pen-test run as a chain-depth fuzz will surface the composition failures.

The operational implication is that test discipline must include chain-depth fuzzing, not just per-hop validation. Most security teams do not yet operate this way. The audit checklist in Part II addresses how to verify chain-depth coverage as a control.

Part II: Attenuation discipline



Scope narrows hop by hop; time bounds are forward-only; revocation lives outside the chain. The receipt at each hop is the verifiable evidence.

The discipline that prevents escalation in delegation chains is older than agent ecosystems. Macaroons (Birgisson et al., 2014) introduced the cryptographic primitive twelve years ago. The primitive is well-understood. What this paper contributes is the production discipline (the operational invariants and audit-grade controls) that makes the primitive reliably enforceable in agent-to-agent settings where the chains are deeper, faster, and less visible than the original Macaroons authors anticipated.

This part is for executive and architect audiences. The cryptographic mechanics live in companion documents and in the AAC reference architecture. What follows is the operational discipline that those mechanics support.

The three discipline patterns at concept level

Three patterns, taken together, define attenuation discipline. Each is necessary; none is sufficient on its own.

Strict-subset scope narrowing at every hop. The capability a child holds is a strict subset of the capability its parent held. Never a superset. Never a different domain. Never the introduction of a new resource type that the parent did not already authorize. This is the only invariant that prevents scope expansion through chain composition.

Time-bounded forward-only chains. The expiry of a child capability is less than or equal to the expiry of its parent. There are no grace periods. No chain extension that re-anchors expiry to a later parent. No refresh flow that silently buys more time at depth. Time attenuation flows in one direction: downstream and shorter.

Revocation propagation independent of chain integrity. Revocation must survive a partial-chain compromise. Two design choices satisfy this invariant.

The first is short-TTL with continuous rotation: holder-of-key binding to short-lived identities (SPIFFE-style workload identity is the canonical example) caps the revocation window at the TTL and makes revocation-by-expiry the default rather than a fallback. For sub-hour capability lifetimes and fast compromise detection, this is sufficient on its own.

The second is an out-of-chain revocation channel: a transparency log, an epoch-anchored revocation list, or an out-of-band signal that does not depend on chain integrity. This handles immediate revocation within the TTL window, before rotation would clear the credential by expiry.

The two primitives are layered, not exclusive. At the agent delegation layer, capability lifetimes are often hours-to-days (long-horizon trading authority, multi-hop pipelines, session-spanning agent tasks) and compromise detection is slower than service mesh, which is why the out-of-chain channel becomes necessary alongside any TTL-based primitive. If revocation requires every intermediate node to be reachable and cooperative, an attacker who controls one node defeats revocation for the entire downstream subtree regardless of which primitive is in use.

What this means operationally

The three patterns translate into three audit checks that any security team can run against an agent platform:

- **Scope check.** Walk a sampled delegation chain. At each hop, verify that the child capability is a machine-checkable semantic subset of the parent. In simple path or scope systems this may be syntactic; in policy-based systems (OPA, Cedar, ABAC) it must be proven by policy evaluation. Flag any hop where a new scope appears or where pattern-matching could admit resources outside the parent's set.
- **Time check.** Pull the expiry timestamps for a sampled chain. Verify monotonic non-increase from parent to child. Flag any node whose expiry exceeds its parent's, or any refresh flow that resets the chain's effective expiry.
- **Revocation check.** Trigger a revocation event at the root. Measure latency to enforcement at each downstream node, including under simulated partial-chain compromise (one intermediate offline). Flag any path where revocation does not propagate within the platform's stated SLA.

Vendor-evaluation criteria fall out of the three checks directly. When evaluating an agent platform, internally built or vendor-provided, ask six questions:

1. **Scope semantics**: how is scope expressed? Path strings, typed capability graph, OPA Rego policies bound to the capability? Pattern-matching admission rules are the failure surface; semantic typing is the survival pattern.
2. **Expiry enforcement**: where does enforcement live? At the resource server, at the agent, at the issuer? Multi-point enforcement is more defensible. Single-point enforcement creates a single failure mode.
3. **Refresh flow**: is there a refresh token path? If yes, is the refresh bounded by the original delegation's expiry or by an independent refresh policy? Independent refresh policy is the time-attenuation defeat.
4. **Revocation channel**: is revocation in-chain or out-of-chain? In-chain revocation fails under partial compromise. Out-of-chain (transparency log, epoch-anchored, back-channel) survives.
5. **Audit receipt**: does the platform produce a receipt at each hop? Is the receipt cryptographically chained? Can the receipt chain be queried by a regulator without operator cooperation? If any of those answers is no, audit defensibility is operator-dependent rather than architectural.

6. **Multi-tenant guarantees**: how does the platform enforce tenant boundaries on capabilities? At the agent identity layer, the capability layer, both? What happens when an agent identity exists in two tenants?

RACI shape that the three discipline patterns require, by function:

Function	Capability scope	Chain integrity	Revocation visibility
Responsible	Platform engineering	IAM team	Security operations
Accountable	CTO or VP Platform	CISO	CISO
Consulted	AppSec, legal/compliance	AppSec, audit	Audit, regulator-relations
Informed	Audit committee, business owners	Audit committee	Audit committee, deploying business units

The pattern that fails: scope owned by application teams who do not see chain composition, chain integrity owned by IAM who do not own the audit framework, revocation owned by security operations who do not know who issued what. Each function works in isolation; nothing owns the composition. Attenuation discipline only holds when one governance owner owns the composition risk end-to-end, even if scope semantics, chain integrity, and revocation visibility remain operated by different functions. In practice, that governance owner typically sits under Platform Engineering or IAM Governance rather than Security Operations: SecOps remains accountable for revocation visibility, but it should not own scope semantics or chain integrity, which require closer proximity to the platform's authorization model.

Brief crypto pointer

The cryptographic primitive that supports all three discipline patterns is Macaroons (Birgisson et al., 2014). Macaroons are bearer tokens with hash-bound caveats: each delegation appends a caveat that further restricts the token's scope or validity, and the integrity of the caveat chain is enforced by an HMAC chain that binds each caveat to all prior ones. The primitive is well-established and has well-understood security properties. For deep protocol details, including AAC's specific construction extensions (PQC hybrid signatures, ZKP non-disclosure, capability tokens at the receipt layer), refer to the AAC architecture documents and Paper #3 of this series.

Discipline layer vs primitive layer

Per-agent identity primitives at the workload layer are in flight via WIT-SVID (spiffe/spiffe#362), the WIMSE WG architecture draft (draft-ietf-wimse-arch), and the OAuth SPIFFE Client Authentication draft (draft-ietf-oauth-spiffe-client-auth). The WIMSE architecture itself acknowledges that multi-hop delegation discipline remains emerging territory (§3.3.9: "each hop in the chain MUST explicitly scope and re-bind the security context"), without prescribing the specific protocols. This paper contributes the discipline layer that the in-flight identity primitives presuppose but do not specify.

Compliance hooks

EU AI Act Annex IV, NIS2 Article 21, and DORA Chapter II all converge on the same evidence problem: the deploying organization must show how systems are controlled, how risks are managed, how access is governed, and how incidents can be reconstructed. Attenuation discipline does not replace those obligations. It makes them demonstrable.

EU AI Act Annex IV requires technical documentation describing system architecture (item 2(c)), validation and testing procedures (item 2(g)), cybersecurity measures (item 2(h)), monitoring, functioning, and control (item 3), performance metrics (item 4), and the risk management system (item 5). Attenuated delegation chains help produce evidence across these documentation obligations when agent systems act across tools, identities, and downstream services. Each delegation step becomes a verifiable narrowing of authority that maps directly into the architecture, validation, cybersecurity, and risk-management items.

NIS2 Article 21 requires appropriate technical, operational, and organizational measures, including incident handling and access control. The case for traceable delegation strengthens here because incident handling and access-control evidence become harder when downstream agent actions cannot be linked back to originating authority. The receipt at each hop, produced by attenuation discipline, is the artifact that makes that link queryable, including under partial-chain compromise where the revocation channel survived because it lived outside the chain.

DORA supports the same evidence logic through Article 6's documented ICT risk management framework and Article 8's identification and mapping of ICT-supported business functions, information assets, ICT assets, roles, dependencies, and ICT risk. Operational resilience evidence demands that the institution can demonstrate downstream agent action stayed within originating authorization. Attenuation discipline turns this from a manual audit task into a queryable architectural property: the receipt chain answers the question by construction. Where the receipt chain is incomplete, the audit names the architectural gap, not the operator's diligence.

The CSA AI Security Maturity Model (AISMM v3.7, May 7, 2026) frames these compliance obligations as a five-level maturity progression. Attenuation discipline maps directly into AISMM's IAM domain (specifically IAM-03.1, "per-workload non-human identity with managed credential lifecycle" at Defined maturity, and the lifecycle controls at higher levels) and into App Security at the boundary. The receipt-chain evidence model satisfies AISMM Privacy and Compliance assessment requirements from Level 3 (Defined) and above. Organizations targeting Levels 4 and 5 (Capable, Efficient) will find attenuation discipline a foundational primitive rather than an optional control.

Without strict attenuation, these requirements become difficult to demonstrate at scale. The institution may still have policies and logs, but it lacks architectural evidence that downstream agent action remained bounded by originating authorization. Attenuation discipline produces that evidence as a property of the architecture rather than as a reconstruction exercise.

The receipt chain also supports data minimization principles, including GDPR Article 5(1)(c) and, where the system is high-risk, EU AI Act Article 10 data-governance expectations, by ensuring downstream agents receive only the exact scope, time bound, and subject required for the delegated task, nothing more.

Composition with Papers #2 and #3

Paper #2 introduced capability tokens as the authorization primitive: per-task, real-time evaluation against the intersection of user authority, agent permission, and task requirements. Paper #3 added zero-knowledge verification: proof that a capability decision was made correctly without revealing the underlying capability content. Paper #4 adds the discipline that makes both reliable in chains: strict-subset narrowing, time-bounded forward-only chains, revocation independent of chain integrity.

The three together form a composable architecture: capability tokens (Paper #2) verified without disclosure (Paper #3) and attenuated in delegation chains (Paper #4). The Authorization trilogy of the Non-Human Identity series closes here.

Part III: Patterns from production

Discipline at concept level lands or fails based on the patterns engineering teams actually deploy. This part covers the patterns that survive production stress and the patterns that look correct in design review but fail under load.

Sector applications

The structural patterns generalize. The sector framings make them concrete.

Financial services. Trading agents that delegate execution authority to sub-agents (algo strategies, hedging routines) operate on millisecond timescales with capital at risk. Edinburgh Szpruch et al. (April 2026) describe the canonical pattern at the model risk management layer: capability decomposition with authority, constraints, and evidence requirements specified per agent. The delegation discipline this paper describes operates at the protocol layer beneath that policy framing.

The recurring scenario is the algo-trading delegation chain. A portfolio manager's authority to trade is delegated to an algorithmic execution agent, which delegates to a smart-order-router agent, which delegates to a venue-specific micro-strategy. Each delegation is correct in isolation. Under attenuation discipline, the venue-specific micro-strategy cannot trade outside the portfolio's risk budget regardless of what intermediate agents authorize. Without attenuation discipline, the smart-order-router's discretion can re-anchor the risk budget at its own scope, which the portfolio manager never approved. The pre-agent precedent is well-known: every major institutional algo desk runs internal procedural controls that bound sub-strategy authority precisely because composition can re-anchor the risk budget. The agent generation turns those procedural controls into a cryptographic-protocol problem.

Defence and critical infrastructure. OT/SCADA agents operating in long-cycle systems (energy grids, transport, water treatment) have asymmetric blast radius: a small scope error compounded across delegation chains can have physical consequences. Attenuation discipline is the architecture class that gives operators audit-grade evidence that a downstream agent action stayed within its originating mandate.

The recurring scenario is the multi-domain command chain. An OT/SCADA agent at the supervisory layer receives an authorization to execute a maintenance task. It delegates to a device-level agent, which delegates to a firmware-update agent. If the firmware-update agent runs in a different security enclave than the supervisory layer, the cross-domain chain break manifests: caveats bounding the supervisory authorization may not transfer to the firmware enclave's policy framework. The Stuxnet-era lesson, that industrial system trust boundaries are porous in unexpected ways, applies directly to the agent generation. A chain that crosses an enclave boundary needs explicit re-attenuation, not implicit trust.

Healthcare. Patient-data agents under HIPAA and equivalent regimes (EU GDPR Article 9 special category data) face a particular constraint: delegation chains may cross trust boundaries (between hospital systems, between provider and payer) where the receiving side may not enforce the same caveat semantics. Cross-domain attenuation discipline is the architecture that makes regulated patient-data delegation defensible.

The recurring scenario is cross-institution patient-data delegation. A patient consents to data sharing with a specific provider. The provider's agent delegates to a third-party AI diagnostic service. The diagnostic service delegates to a research aggregator. Each delegation may be HIPAA-compliant in isolation. The composed chain, without strict attenuation, can violate the patient's original consent scope by the time the data reaches the research aggregator. GDPR Article 9 carries the same constraint pattern with an added complication: consent is revocable, and the revocation must propagate across institutional boundaries to be meaningful. That is exactly the cross-domain attenuation problem.

Three patterns that survive

1. **Capability-token-as-receipt with hash-bound caveats.** The Macaroons-style construction. Each delegation appends a caveat that further restricts scope or validity; integrity is enforced by an HMAC chain. The receipt itself is the verification artifact. AAC architecture A2A flow steps 13 through 15 implement this pattern.

2. **Depth limits with explicit policy at the boundary.** Set an explicit maximum delegation depth and justify it as a risk decision. Enforce default-deny at the limit. Depth limits do not solve attenuation by themselves, but they bound the worst-case scope-accumulation surface and create a natural audit checkpoint.

3. **Revocation epochs anchored to a transparency log.** Instead of pull-based revocation that requires every node in the chain to be reachable, anchor revocation to a publicly observable epoch counter on a transparency log (Sigstore Rekor or equivalent). Each capability includes an epoch reference and a maximum staleness window; verifiers reject capabilities older than the staleness window. Revocation becomes a publish operation on the transparency log, observable to any verifier without chain traversal. In multi-tenant or privacy-sensitive deployments where a public transparency log is unsuitable, the same property can be implemented through a private append-only revocation channel, potentially using Merkle proofs or zero-knowledge variants when revocation privacy matters.

Three patterns that fail

1. **Scope-as-string-prefix.** A capability scoped to `/files/finance/*` looks defensible in design review. After the next folder reorganization adds `/files/finance/legal/contracts/`, the prefix still matches and the capability now grants access that no human approved. Path-based scope without semantic typing breaks under any deployment evolution.

2. **Soft expiry "with grace period."** A capability expires at T, but operational logic accepts it until T+δ "in case of clock skew." The grace period is interpreted as a feature; in practice, it defeats the entire time-attenuation invariant. An attacker who captures a soon-to-expire capability has the grace window to use it.

3. **Revocation that depends on every node in the chain being reachable.** If revocation propagation requires every intermediate to confirm receipt, partial-chain compromise (one intermediate offline or compromised) leaves revocation latency unbounded. The downstream subtree continues to honor the revoked capability until the broken intermediate is repaired, which may be never.

Edinburgh Szpruch et al. cross-reference

Edinburgh's April 2026 paper *"Scalable Runtime Governance for Agentic AI in Financial Services"* proposes capability decomposition at the model risk management (MRM) layer: each agent's authority is specified with constraints and evidence requirements at policy level. This paper's contribution is at the protocol layer beneath

that policy framing: how capabilities compose under delegation chains, how attenuation discipline turns policy-layer specifications into protocol-level enforcement.

The two papers describe complementary layers of the same architecture. Edinburgh specifies what authority a capability should have. This paper specifies how that authority survives delegation without compounding risk.

The layer division maps cleanly:

Policy layer (Edinburgh's framing). At the model risk management framework, each agent's authority is specified declaratively: authority over what resources, under what constraints, with what evidence required for action. The output of this layer is a policy artifact, a capability specification that downstream systems must enforce.

Protocol layer (this paper's framing). At the message and token level, the policy specification is encoded as a verifiable capability that can be delegated, attenuated, and revoked across hops. The output of this layer is a delegation chain: a sequence of cryptographically chained receipts where each step demonstrates a strict-subset narrowing of the parent's authority.

Edinburgh's contribution stops at the policy artifact. This paper's contribution starts where the policy artifact enters a chain. The two together produce an architecture where agent authority is both well-specified (the policy layer) and well-bounded under delegation (the protocol layer). Neither layer alone is sufficient for the regulated-financial-services audience.

The convergence is parallel-discovery: independent research arriving at compatible decompositions addressing different layers. That convergence is itself a signal that the architecture is approaching the form standardization tends to settle on.

Closing

Attenuation discipline is the architectural pattern that keeps delegation depth from compounding risk. Composed with Paper #2's intersection authorization and Paper #3's verification without disclosure, the three together produce an architecture where deep agent ecosystems are auditable, revocable, and scope-bounded by construction.

The Authorization trilogy of the Non-Human Identity series closes here. Paper #5 closes the regulatory loop: how NIS2, DORA, and the EU AI Act translate this architecture into evidence artifacts that satisfy audit at scale.

What to do Monday morning

For any CISO, IAM architect, or platform-security lead reading this, five concrete steps to take this week:

1. **Audit a sampled delegation chain in your highest-risk agent platform.** At each hop, verify the child capability is a machine-checkable semantic subset of the parent. In simple systems this may be syntactic; in policy-based systems it must be proven by policy evaluation. Flag any hop where a new scope appears or where pattern matching could admit resources outside the parent's set.
2. **Pull expiry timestamps for the same chain.** Verify monotonic non-increase from parent to child. Flag any refresh flow that resets the chain's effective expiry beyond the original delegation's window.

3. **Trigger a revocation event at the root.** Measure latency to enforcement at each downstream node, including under simulated partial-chain compromise (one intermediate offline). Flag any path where revocation does not propagate within the platform's stated SLA.
4. **Map your three highest-volume delegation chains against the seven failure modes in Part I.** Most production chains exhibit two or three. Name the ones in yours. The audit checklist in Part II tells you what to verify next.
5. **Brief the audit committee.** Translate each finding into the compliance language above (EU AI Act Annex IV, NIS2 Article 21, DORA Chapter II). The architecture conversation lands faster when it is already framed as an audit defensibility conversation.

Acknowledgments

Thanks to Agustin Martinez Fayo for substantive peer review on the revocation framing and for surfacing in-flight IETF/SPIFFE references.

Citations / related work

- **Paper #2 of this series.** *Authorization for AI Agents: Beyond RBAC.* mahasbini.org/papers. Capability tokens at single-hop authorization layer.
- **Paper #3 of this series.** *Authorization Without Disclosure: Zero-Knowledge Proofs for Agent-to-Agent Authorization.* mahasbini.org/papers. Verification without disclosure layer.
- **Birgisson et al. 2014.** "Macaroons: Cookies with Contextual Caveats for Decentralized Authorization in the Cloud." The foundational primitive. ACM CCS.
- **Edinburgh Szpruch et al., April 13, 2026.** "Scalable Runtime Governance for Agentic AI in Financial Services." Parallel discovery, capability framing at MRM layer.
- **OAuth 2.0 Token Exchange (RFC 8693).** Current standard, lacks strict attenuation by default.
- **WIT-SVID (spiffe/spiffe#362).** In-flight per-agent SPIFFE identity primitive: application-layer, holder-of-key bound via `cnf`, multiple SVIDs per caller. Author Noah Stride. https://github.com/spiffe/spiffe/pull/362
- **draft-ietf-wimse-arch.** IETF WIMSE WG architecture draft. Section 3.3.9 on AI and ML-Based Intermediaries names multi-hop delegation chains as the failure mode and states "each hop in the chain MUST explicitly scope and re-bind the security context." Salowey, Rosomakho, Tschofenig. https://datatracker.ietf.org/doc/html/draft-ietf-wimse-arch
- **draft-ietf-oauth-spiffe-client-auth.** IETF OAuth WG. Profiles SVIDs (JWT, X.509, WIT) as OAuth client credentials. Schwenkschuster, Kasselmann, Rose, Thorgersen. https://datatracker.ietf.org/doc/html/draft-ietf-oauth-spiffe-client-auth
- **AAC Construction Specification.** Companion document, deep protocol detail referenced from Paper #3.
- **OWASP Top 10 for Agentic Applications 2026.** ASI03 Identity & Privilege Abuse (primary mapping for this paper) + ASI07 Insecure Inter-Agent Communication (secondary) + ASI10 Rogue Agents (delegation outside scope).

- **CSA AI Security Maturity Model (AISMM) v3.7, May 7, 2026.** Five-level industry maturity framework across 12 domains (Governance, Organization Management, IAM, Security Monitoring, Infrastructure, Model Security, App Security, Data Security, Risk Assessment, AI Dev and Supply Chain, Privacy and Compliance, Incident Response). Cross-references OWASP ASI, NIST AI RMF, ISO/IEC 42001, EU AI Act, MITRE ATLAS. Directly relevant to this paper: IAM-02.2 (NHI inventory), IAM-03.1 (per-workload NHI with credential lifecycle), App Security controls at Defined maturity and above.
- **Grantex State of AI Agent Security 2026.** Audit of 30 popular open-source AI agent projects: 93% use unscoped API keys, 0% have per-agent cryptographic identity, 100% have no per-agent revocation. Primary source for the agent-framework identity gap.
- **Pixee AppSec Weekly Briefing, March 18, 2026.** Secondary amplification of the Grantex findings under the headline “93% of AI Agent Frameworks Have Zero Identity Controls.”
- **Gravitee State of AI Agent Security 2026.** 45.6% shared agent-to-agent credentials.
- **OECD.AI incident 2026-05-04-4a73.** Grok / Bankrbot prompt-injection exploit, May 4, 2026.