Appendix: ZLAR Mapping to the NCCoE Concept Paper Questions

Supporting note for public comment on software and AI agent identity and authorization

Executive Thesis

This appendix supports my public comment on the NCCoE concept paper regarding software and AI agent identity and authorization.

My central point is that the key unresolved control problem in agentic systems is **execution-boundary governance**. Existing and emerging standards can help establish agent identity, authentication, and delegated authority. What they do not fully solve is whether a specific proposed agent action should be allowed at the moment of execution, under current context, under bounded delegated authority, with evidence that can later be verified.

In short:

- authenticated is not authorized
- delegated is not governed
- reasoning is not permission

ZLAR is my proposed answer to that gap: an external, deterministic, non-bypassable runtime governance layer for AI agent actions.

Its core design principle is simple:

the gate should have no intelligence

The gate does not reason, does not parse natural language, and does not attempt to decide whether the model "meant well." It evaluates structured action requests against explicit policy and returns allow, deny, or escalate.

This matters because prompt injection, context poisoning, and authority confusion attack the reasoning layer. A deterministic gate does not eliminate those reasoning failures, but it can keep compromise of reasoning from automatically becoming compromise of action.

1. Mapping to NCCoE Question Areas

NCCoE area	What NCCoE is asking	ZLAR response	Current status
Identification	How should agents be identified, and what metadata matters?	ZLAR assumes identity inputs are provided by existing identity and workload identity systems. It is not a replacement identity standard. It consumes identity and delegation context as policy inputs.	Partial / complementary
Authentication	What does strong authentication mean for an agent? How are credentials managed?	ZLAR relies on upstream identity, credential, and attestation systems. Its contribution begins after authentication, at action admissibility.	Out of scope / complementary

Authorization	How should agent actions be allowed or denied in context?	This is ZLAR's primary contribution: per-action policy evaluation at runtime by an external governance layer.	Core contribution
Delegation / human-in-the-loop	How should "on behalf of" authority work? How should human approval be handled?	ZLAR treats delegation as a bounded, enforceable relationship and supports allow / deny / escalate patterns for sensitive actions.	Core contribution, with room to deepen
Auditing / non-repudiation	How can actions be logged in a tamper-evident, verifiable way?	ZLAR generates cryptographically protected evidence: each audit entry carries a FIPS 186-5 EdDSA (Ed25519) signature satisfying SP 800-53 AU-10 (Non-repudiation) and is linked via FIPS 180-4 SHA-256 hash chaining satisfying AU-9(3). Post-quantum hybrid signing (ML-DSA-44 per FIPS 204) is available via configuration.	Strong current contribution
Prompt injection prevention / mitigation	How can agent abuse be prevented or contained?	ZLAR's contribution is containment at the action layer. Even if reasoning is compromised, execution remains bounded by external policy.	Strong architectural contribution

2. Where ZLAR Sits in the Stack

ZLAR should be understood as a **runtime governance layer**, not as a replacement for identity standards.

A simplified stack looks like this:

1. **Identity and workload identity layer** Establishes who the agent is, what software or workload it represents, and what credentials or attestations it carries.
2. **Delegation layer** Establishes on whose behalf the agent may act and under what constraints.
3. **Execution-boundary governance layer (ZLAR)** Evaluates each proposed action at runtime against policy, context, and delegated authority.
4. **Evidence layer** Records what was requested, what policy applied, what decision was made, and what occurred.

The missing layer, in my view, is step 3.

Without it, organizations know who the agent is, but still cannot deterministically control what the agent may actually do at the moment of execution.

3. ZLAR Architecture Summary

ZLAR is built around the claim that the governance boundary must sit **outside the agent process**.

This leads to four architectural properties:

A. External enforcement

The enforcement point should not live inside the model's reasoning layer. The governed system should not be able to rewrite or bypass its own containment.

B. Deterministic decision-making

The gate should operate on structured action requests, not natural-language interpretation.

C. Default-deny posture

If governance fails, the safe outcome is denial, not permissive execution.

D. Evidence generation

Every significant decision should leave behind verifiable evidence suitable for later review.

The design principle can be stated simply:

intelligence may propose; the gate decides admissibility

4. What "The Gate Has No Intelligence" Means

This phrase is central to the architecture.

It means the enforcement layer:

- does not run its own model
- does not interpret prompts
- does not attempt to infer intent from prose
- does not negotiate with the agent
- does not "reason" its way into exceptions

Instead, it consumes structured action requests and explicit policy.

That matters because the gate should not become another persuadable model-layer surface. If the gate itself becomes intelligent, then the same classes of manipulation that affect the agent can begin to affect the governance layer.

The absence of intelligence at the gate is therefore not a limitation. It is the architectural property that preserves determinism.

5. Current Implementation

The current ZLAR reference implementation demonstrates the execution-boundary pattern in practice.

Implemented now

- interception of agent actions at the execution boundary
- signed policy used as the governing source of truth
- a Bash gate for hook-based enforcement
- an MCP gate for client/server tool mediation
- a shared allow / deny / escalate enforcement model
- a hash-chained audit trail with cryptographic integrity (FIPS 186-5 EdDSA per-entry signatures, FIPS 180-4 SHA-256 hash chaining)
- post-quantum hybrid signing (Ed25519 + ML-DSA-44 per FIPS 204) selectable via configuration per IR 8547 transition guidance

Practical meaning

In the current implementation, the agent does not self-govern. Proposed actions are intercepted by an external gate, evaluated against policy, and recorded in a tamper-evident evidence trail.

The governing idea is the same across enforcement surfaces:

the agent does not volunteer to be governed; it is governed by architecture

6. Evidence and Audit Model

A key architectural claim behind ZLAR is that governance should produce evidence, not merely logs.

A governance event should bind together:

- agent identity
- delegating user identity, where applicable
- requested action
- target resource or tool
- governing policy version
- decision result
- timestamp
- audit linkage to prior events

This creates a trail suitable for later review, forensics, and accountability analysis.

The important distinction is this:

- a conventional log says what the system reports happened
- a governance evidence trail is designed to show what happened under which policy constraints

This is especially important for enterprise, public-sector, and regulated environments where organizations need to prove not only that an action occurred, but that it occurred within authorized bounds.

7. Why This Maps to NCCoE's Priorities

I believe ZLAR aligns most strongly with the concept paper's questions in the following areas.

Authorization

ZLAR addresses the question of whether a given action should be allowed right now, under current context, rather than relying only on session establishment or pre-issued scopes.

Delegation

ZLAR supports the idea that "on behalf of" authority should be bounded, reviewable, and technically enforceable rather than merely implied.

Auditing and non-repudiation

ZLAR emphasizes integrity-protected governance evidence rather than ordinary application logging.

Prompt-injection impact reduction

ZLAR does not claim to solve prompt injection at the reasoning layer. Its contribution is containing the consequences at the execution layer.

In this sense, ZLAR complements identity and delegation standards rather than competing with them.

8. Current Limits and Honest Boundaries

For credibility, it is important to state clearly what ZLAR does and does not yet claim.

What ZLAR does claim

- execution-boundary interception
- deterministic external policy enforcement
- non-bypassable governance as an architectural goal
- cryptographically protected audit evidence
- containment of action even when reasoning is imperfect

What ZLAR does not claim

- to replace identity, authentication, or credential standards
- to eliminate prompt injection altogether
- to solve all data lineage and provenance issues by itself
- to represent a finished standards framework across all agent ecosystems

Current development boundary

The reference implementation demonstrates the execution-governance pattern now. Some richer policy semantics and standards alignment work remain future-facing and should be treated as target direction rather than fully realized production scope.

9. Suggested NCCoE Demonstration Scope

If NCCoE moves forward with this project, I would suggest including the following demonstration elements:

5. agent identity and workload identity inputs
6. bounded human-to-agent delegation
7. per-action runtime policy evaluation
8. a non-bypassable enforcement point outside the agent process
9. step-up approval or human escalation for sensitive actions
10. tamper-evident decision records
11. explicit prompt-injection containment testing
12. evidence that an authenticated agent acted within authorized bounds

That combination would move the field from agent authentication alone toward operationally credible agent governance.

10. Closing Observation

The practical question for enterprises is not only whether an agent can authenticate.

The practical question is whether the organization can later prove that the agent acted within authorized limits at the moment of execution.

That is the gap ZLAR is intended to address.