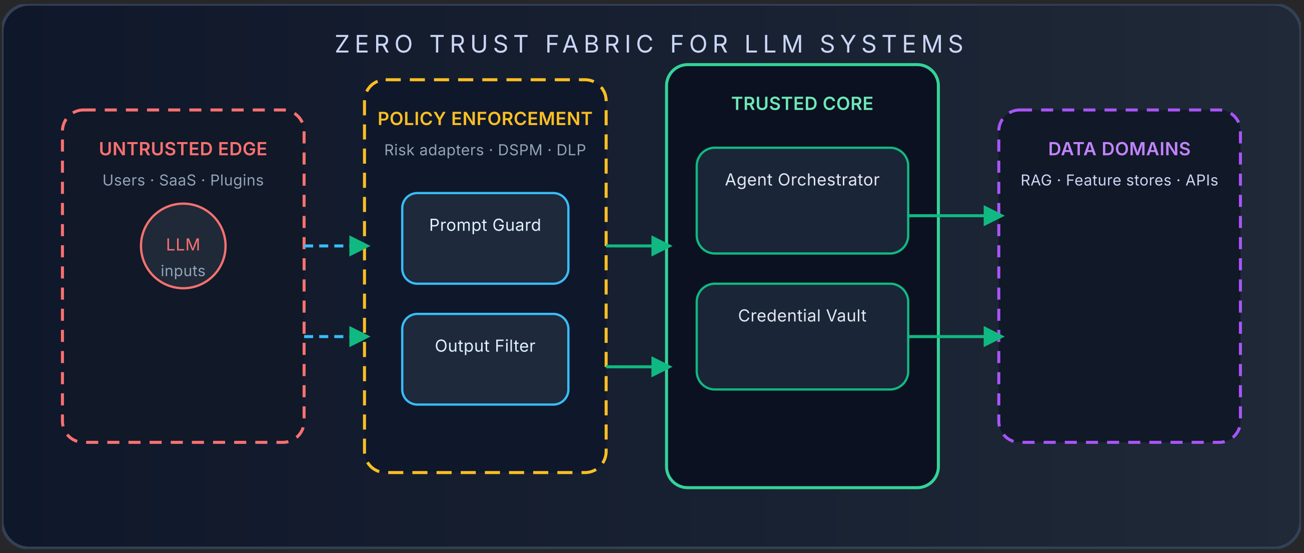
24 FEB 2026 • 19 MIN READ

Zero Trust for AI & LLM Systems: Microsegmentation for Machine Identities

Most Zero Trust programs were scoped for user-to-app traffic. Meanwhile, LLM gateways, agentic orchestrators, and retrieval workers are exchanging secrets through flat service meshes that assume every workload behind the cluster IP is friendly. This guide reframes Zero Trust for AI-first enterprises: segmenting autonomous agents, continuously attesting prompts and outputs, and giving every non-human identity (NHI) the same scrutiny that humans get at the edge.



Zero Trust interlocks policy enforcement, agent orchestration, and data domains. Every hop re-authenticates workload identities and re-scores risk instead of inheriting trust from the VPC.

Start with Control Planes, not Perimeters

Zero Trust guidance anchored in NIST SP 800-207 is explicit: policies ride on identity, device posture, and context instead of VLAN boundaries. AI programs have to add model state and prompt context to that list. The OMB M-22-09 federal Zero Trust strategy calls for enterprise-wide policy enforcement points (PEPs) that evaluate every request. For LLM systems, each prompt assembly, retrieval query, or agent tool call is a new request—even if it comes from serverless code running centimeters away from your gateway. Treat every autonomous hop as untrusted.

That means pulling the policy logic out of application code and into a fabric that can score risk per transaction. Attach inline adapters that enrich each prompt with data classification, user intent, and exposure scoring before it crosses into the LLM. Use outbound filters that evaluate generated text for OWASP LLM Top 10 patterns such as indirect prompt injection or supply chain contamination. When you do this, Zero Trust stops being an enterprise networking program and becomes an AI control plane.

> **Decision checkpoint:** Every LLM call needs three proofs—who is asking, what data the agent wants, and whether the destination policy allows that combination right now. If you cannot answer all three within 200 ms, you still have a perimeter model.

Segment Agent Pipelines like Supply Chains

The CISA Zero Trust Maturity Model 2.0 calls for policy enforcement across five pillars. AI workloads need a sixth: prompt and context integrity. Borrow the BeyondCorp blueprint—Google’s BeyondCorp research shows how to collapse internal networks and make every application a public endpoint protected by continuous verification. Apply that logic to retrieval pipelines by exposing each stage (ingestion, chunking, ranking, synthesis) behind its own service identity, short-lived mTLS certificates, and intent-aware authorization policies.

Microsegmentation for AI is less about IP ranges and more about data contracts. Retrieval workers should only speak to embeddings that correspond to the tenant or project they were launched for. Agent plugins should never inherit RDS credentials just because they live in the same namespace. Build policies that combine subject (agent identity), action (tool capability), resource (dataset or API scope), and environment (risk score, deployment ring). Then log every denied call so your SOC can trend attempts that map to MITRE ATLAS behaviors.

Segmentation Blueprint for Multi-tenant Retrieval

1. **Namespace to tenant mapping:** Use dynamic admission controllers that read tenant metadata (region, data residency, retention tier) and attach it to every retrieval pod. Enforce network policies so pods cannot open sockets to namespaces with different tenant tags.

2. **Context-aware proxies:** Deploy egress proxies that inject tenant IDs and policy tokens into every vector database call. If the downstream index sees a token outside its allowlist, it drops the call before similarity search begins.

3. **Risk-adaptive rate limiting:** Add adaptive throttles that reference policy decisions. If a pod suddenly queries data outside its historical vector neighborhoods, the proxy forces step-up verification, quarantines the embedding set, and pages the on-call AI SRE.

These controls satisfy the CISA data pillar while keeping lateral movement inside AI clusters observable.

Govern Non-Human Identities like Privileged Users

Agents, schedulers, and batch jobs now outnumber humans in most AI stacks. The Microsoft Entra team's workload identity guidance outlines why these NHIs need their own lifecycle—issuance, rotation, revocation, and attestation. Map each NHI to a policy decision point so that prompt routers cannot pull secrets just because the kubelet has access. Borrow the identity pillar from CISA's model and demand phishing-resistant credentials for software too: SPIFFE/SPIRE IDs, hardware-backed attestation (TEE reports or Nitro attestation docs), and just-in-time secrets from your vault.

The NIST AI Risk Management Framework ties identity hygiene to governance functions MAP and MANAGE. Tag every NHI with purpose metadata (feature extraction, inference, evaluation) and feed that into policy decisions. When an evaluation agent suddenly requests write privileges on the feature store, the PDP should deny it because its intended purpose conflicts with the action. This is how Zero Trust prevents "prompt drift" from becoming "credential drift."

Continuous Attestation for Agent Tokens

```
import time
from dataclasses import dataclass

@dataclass
class AgentIdentity:
    subject: str
    workload: str
    risk_score: float
    attested: bool

class TokenAuthenticator:
    def __init__(self, policy_client, attestor, cache):
        self.policy = policy_client
        self.attestor = attestor
        self.cache = cache

    def authenticate(self, token: str, action: str, resource: str) -> bool:
        identity = self.cache.get(token)
        if not identity or identity.expiry < time.time():
            identity = self.attestor.verify(token)
            self.cache.set(token, identity, ttl=identity.ttl)
        if not identity.attested:
            return False
        decision = self.policy.evaluate({
            "subject": identity.subject,
            "workload": identity.workload,
            "risk": identity.risk_score,
            "action": action,
            "resource": resource,
        })
        if decision.deny_reason == "risk_spike":
            self.attestor.quarantine(identity.subject)
        return decision.allowed
```

This pattern keeps latency low by caching attestation results but revalidates when risk scores change. A denied request automatically triggers quarantine, satisfying continuous verification requirements.

Monitor Prompts and Outputs as Policy Objects

Detection teams finally have doctrine for AI abuse because OWASP’s LLM Top 10 codifies failure modes. Feed those patterns into your Zero Trust analytics: flag prompts that contain instruction modifiers (“ignore previous instructions”), output streams that reference deployment secrets, or retrieval calls that pivot tenants. Map each detection to MITRE ATLAS IDs and ship them to the SOC like any other behavioral analytic.

Zero Trust also demands telemetry parity. If users must prove device posture every hour, AI systems must prove prompt posture every request. Instrument your LLM gateway so that every prompt includes provenance (dataset IDs, upstream principal, air-gapped vs connected). When an agent in a semi-trusted zone starts calling tools outside its defined blast radius, the PDP should drop the call and emit an event that your SIEM can correlate with infrastructure logs. Observability is how you keep the “never trust” part honest.

Telemetry Schema for AI Zero Trust

- **prompt_id:** Deterministic hash of the sanitized prompt.

- **subject_chain:** Ordered list of NHIs and humans involved in constructing the request.
- **dataset_scope:** Vector of dataset IDs plus classification tags.
- **policy_snapshot:** UUID reference to the PDP decision so investigators can replay context.
- **atlas_mapping:** Array of MITRE ATLAS technique IDs observed or suspected.

Shipping this schema alongside application logs means your Zero Trust analytics can pivot from user behavior to agent behavior without building a second SIEM.

A 30/60/90 Plan to Operationalize Zero Trust for AI

In the first 30 days, inventory every AI workload and classify NHIs—most organizations underestimate how many GitHub Actions, Airflow DAGs, or Bedrock agents possess permanent tokens. Next 60 days, insert policy enforcement points on both sides of the LLM: pre-context guardrails and post-output sanitizers tied to OWASP signals. Within 90 days, segment your RAG infrastructure by tenant and sensitivity, and route all privileged prompts through attested service identities. Align each milestone to the data, identity, and monitoring pillars from CISA's model so leadership sees continuity with the broader Zero Trust roadmap.

Zero Trust is not another security buzzword for AI—it's the only architecture that assumes every model call could be hostile and then proves otherwise in real time. Treat prompts as requests, treat agents as users, and treat retrieval hops as cross-boundary calls. When you do, AI can ship fast without handing your blast radius to the next jailbreaker.