Zero Trust Secure Agentic AI Architecture (XACML 3.0)

A Full Governance Stack (PDP, PEP, PIP, PAP) based on XACML 3.0 principles, implementable by Open Policy Agent (OPA) or similar YAML-based logic engines, and encapsulating a Deterministic, Out-of-Band Signaling, Agentic Governance Enclave.

Step	Security Principle	Threat Mitigation	Component	Action
1. Validated Intent	Input Sanitization, Fail Safe Defaults (Fail-Closed)	Prompt Injection, Resource Exhaustion, System Failure Exploitation	Agentic AI → PEP	Sends natural language request to PEP after using a small, deterministic model (or regex/guardrail tool like NeMo Guardrails) to ensure the natural language intent matches the requested action before bothering the PDP.
2. Access Request	Mediation	Bypass Attacks (Shadow AI)	PEP → PDP	PEP asks PDP for a verdict.
3. Policy Fetch	Accountability (via Auditable Governance)	Policy Drift, Insider Threat, Unauthorized Changes	PDP ← PAP	PDP uses latest Rego files or YAML config (synced from Git) to evaluate the request.
4. Context Fetch	Least Privilege, Attribute Integrity	Information Disclosure, Stale Data, False Context	PDP ↔ PIP	PDP queries MCP Server (the PIP) for data attributes (e.g., resource.is_sensitive == true or user.clearance_level >= 3).
5. Decision	Policy-as-Code	Logic Flaws, Deterministic Gap	PDP → PEP	PDP issues a "Permit", "Permit with Obligations", "Deny", or "Uncertain" to PEP based on the Git policy and MCP context.
6. Enforcement	Policy-as-Code	Data Exfiltration	PEP	PEP applies any "Permit with Obligations" decisions (e.g., "Mask all PII in the response" or "Log this specific action to the SIEM"), proceeds with any "Permit" decision, denies any "Deny" decision, or PEP pauses the flow for any "Uncertain" decision and requests a manual "Human-in-the-loop" approval before proceeding to Step 7.
7. Token Issuance	Least Privilege	Privilege Escalation, C onfused Deputy, Token Theft, Reply Attacks	PEP ↔ MCP	Upon PEP presenting the PDP verdict, MCP Server issues a "Bound Token" (i.e., a limited token bound not just to the Agent, but to the specific session or task ID to ensure that even if an Agent is compromised, its token cannot be reused for a different "Intent". If using OIDC/OAuth2, these are often referred to as "Demonstrating Proof-of-Possession" or "DPoP" tokens).
8. Handoff	Isolation (or Sandboxing)	Persistence/Backdoors, Resource Exhaustion	PEP → AI Agent	PEP spins up AI Agent (Worker) and hands it the Limited Token.
9. Execution	Least Privilege (Dynamic)	Privilege Escalation, Confused Deputy	AI Agent → MCP	Agent uses the token to perform the task via MCP Server.
10. Audit/Return	Data Loss Prevention, Non-Repudiation	Data Exfiltration, Log Tampering, Deniability	PEP → Log Service PEP → Agentic AI	PEP redacts the output providing egress protection (stopping the data leak), writes the immutable log to an external, write-only logging service (like a secure S3 bucket or Splunk) before the "SIGKILL" in Step 11 occurs to ensure that even if the wipe is catastrophic, the audit trail is preserved, and sends a clean result back to the Agentic AI, closing the loop on the ingress protection (Step 1) that stops an initial attack.
11. Ephemeral Wipe	Forward Secrecy, Immutability	Persistence/Backdoors, Cross-Session Leakage, Resident Malware	PEP → Agent Sandbox	PEP broadcasts a "SIGKILL" to PEP's Agent Sandbox followed by a container deletion, ensuring no residual data exists in memory for the next request.

Policy Enforcement Point (PEP)	Agentic Governance Enclave	An Enclave with No Direct Persistence comprising a Programmable Logic Proxy for intent-validation and an Agent Sandbox Manager for secure, ephemeral execution of Short-lived Scoped Tokens via MCP limited tokens. It physically stops the packet or refuses to spin up the AI Agent if the PDP says Deny.
Policy Decision Point (PDP)	Logic Engine	The logic engine (e.g., Open Policy Agent (OPA) daemon) that sits within the sandbox. It receives the intent from the Agentic AI, fetches attributes from the PIP, and checks them against the Rego files or YAML config provided by the PAP. It issues the final "Permit" or "Deny".
Policy Administration Point (PAP)	Rules Server	It provides the rules (e.g., a Git repo containing Open Policy Agent (OPA) Rego files or a YAML config).
Policy Information Point (PIP)	MCP Server	It provides the attributes or context required for a decision (e.g., resource.is_sensitive == true or user.clearance_level >= 3). It acts as the bridge between the PDP and external data sources via the MCP Server.

NOTE: The PEP, PDP, PAP, and PIP must be logically separated (containers/namespaces), but physical separation (separate hardware or specialized Trusted Execution Environments) is best.

By Terry Raitt, CISM, CISSP
Data 05/02/2026