Designing Identity and Access Management for AI Agents

TDI 2026

Thivaharan Kalyanasundaram

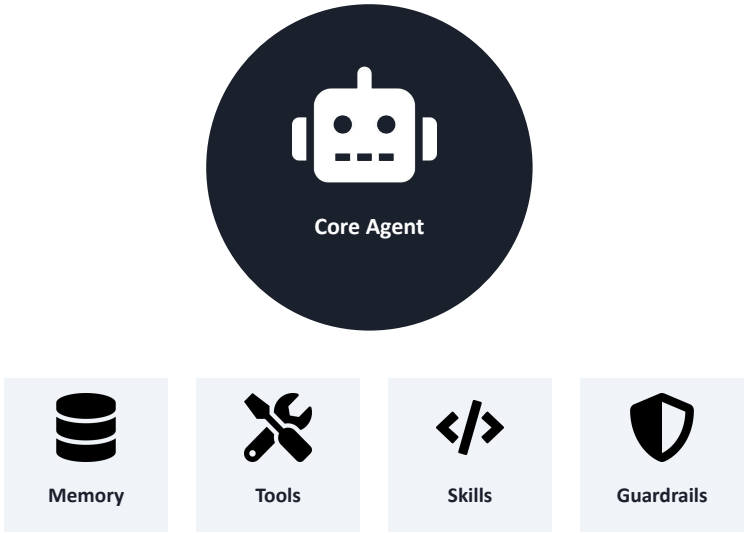
WSO2 | Identity & Access Management

Session Agenda

What Are AI Agents?	Architecture, agent types, and interaction patterns
The Identity Imperative	Real-world incidents, the case for first-class agent identity
The Four Fundamentals	Architecture walkthrough; Administer, Authenticate, Authorize, Audit
Agent Protocols	Identity controls for MCP and A2A
Leveraging Identity Standards	OAuth 2.0, OpenID Connect, SCIM 2.0 — extending, not reinventing
Key Takeaways	Architectural principles for agent-ready IAM

What Is an AI Agent?

More than a chatbot — an autonomous software entity that perceives, decides, and acts.



KEY CHARACTERISTICS

- **Autonomous Execution**
  Operates independently with varying degrees of human oversight
- **Dynamic Decision-Making**
  Evaluates context, plans actions, and adapts in real-time
- **Tool & API Usage**
  Connects to databases, APIs, MCP servers, and external services
- **Delegated Authority**
  Acts on behalf of users while maintaining its own identity context
- **Multi-Step Reasoning**
  Breaks complex tasks into sub-tasks, sometimes delegating to other agents

Different Forms of AI Agents

Each form presents distinct IAM challenges — from simple API access to complex multi-system orchestration.

Request-Response

Single-turn agents that receive a prompt and return a result. Simplest IAM: authenticate the caller, authorize the action, done.

Ambient

Always-on agents that monitor systems, inboxes, or dashboards in the background. Require long-lived, scoped credentials with time-bound consent.

Interactive

Multi-turn, session-based agents (chatbots, copilots) that maintain context. Need session-aware authorization and mid-execution approval workflows.

Computer Use

Agents that control browsers, desktops, or applications autonomously. Highest risk: can click, type, and navigate — need strict action-level guardrails.

Collaborative

Multi-agent systems where agents delegate to each other. Require nested delegation chains, token downscoping, and cross-agent audit trails.

Agent Interaction Patterns

Six key boundaries where IAM governs agent behavior in the enterprise.

User → Agent

Users authenticate via SSO/MFA, grant consent for specific actions, and delegate scoped permissions for the agent to act on their behalf.

Agent → LLM

Agent authenticates to the AI Gateway using its own credentials; gateway applies identity-based guardrails and model access policies.

Agent → Internal APIs

Agent accesses internal APIs, MCP servers, and knowledge bases carrying both its own identity and the delegated user's authority.

Agent → External SaaS

Agent accesses third-party services like Gmail, Slack, and Salesforce through managed token vaults — never handling long-lived secrets directly.

Agent → Agent

Primary agents delegate sub-tasks to worker agents with downscoped tokens, maintaining a clear chain of accountability across hops.

Admin → Agent

Security teams register agents with unique identities, assign liable parties, rotate credentials, and maintain emergency revocation capabilities.

The Identity Imperative

144 : 1
NHIs outnumber humans in enterprises

97%
of NHIs have excessive privileges

1 in 8
breaches now involve an agentic system

Sources: Entro Labs H1 2025 | Entro Security 2025 | CrowdStrike / Mandiant 2025–2026

REAL-WORLD INCIDENTS

GitHub MCP Agent Hijack

May 2025 • Invariant Labs

Malicious commands in GitHub Issues hijacked AI agents via MCP, exfiltrating private repo code and keys

Perplexity Comet Prompt Injection

Aug 2025 • OECD AI Incident DB

Hidden injection caused AI browser to steal credentials autonomously — in 150 seconds, zero awareness

OAuth Supply Chain: Drift / Salesforce

Aug 2025 • Reco AI / Verizon DBIR

Stolen OAuth tokens from a SaaS integration gave attackers access to 700+ customer environments undetected

Why Agents Must Be First-Class Identities

HOW AGENTS ARE MODELED TODAY

- ✗ **As user wrappers:** Agents inherit the invoking user's full session — no separate accountability or permission boundary
- ✗ **As service accounts:** Shared static credentials with no ownership, no expiry, and no link to the human responsible
- ✗ **As application workloads:** Treated like microservices but with autonomous decision-making that workload identity wasn't designed for
- ✗ **As browser extensions:** Plugins with broad permissions and no identity of their own — invisible to IAM systems entirely

WHY FIRST-CLASS IDENTITY MATTERS

- ✓ **Independent accountability:** Each agent has a traceable identity with a liable owner — actions are never lost in a user's session
- ✓ **Granular permission boundaries:** Agent permissions are scoped independently from users — an agent can have less access than its invoker
- ✓ **Lifecycle governance:** Agents can be provisioned, rotated, suspended, and revoked without affecting the humans who use them
- ✓ **Delegation transparency:** When an agent acts on behalf of a user, both identities are visible — who delegated, who acted, on what

The Four Fundamentals of Agent IAM

A comprehensive framework for the complete lifecycle of AI agent identity and access management.

ADMINISTER

Identity Lifecycle

AUTHENTICATE

Secure Credentials

AUTHORIZE

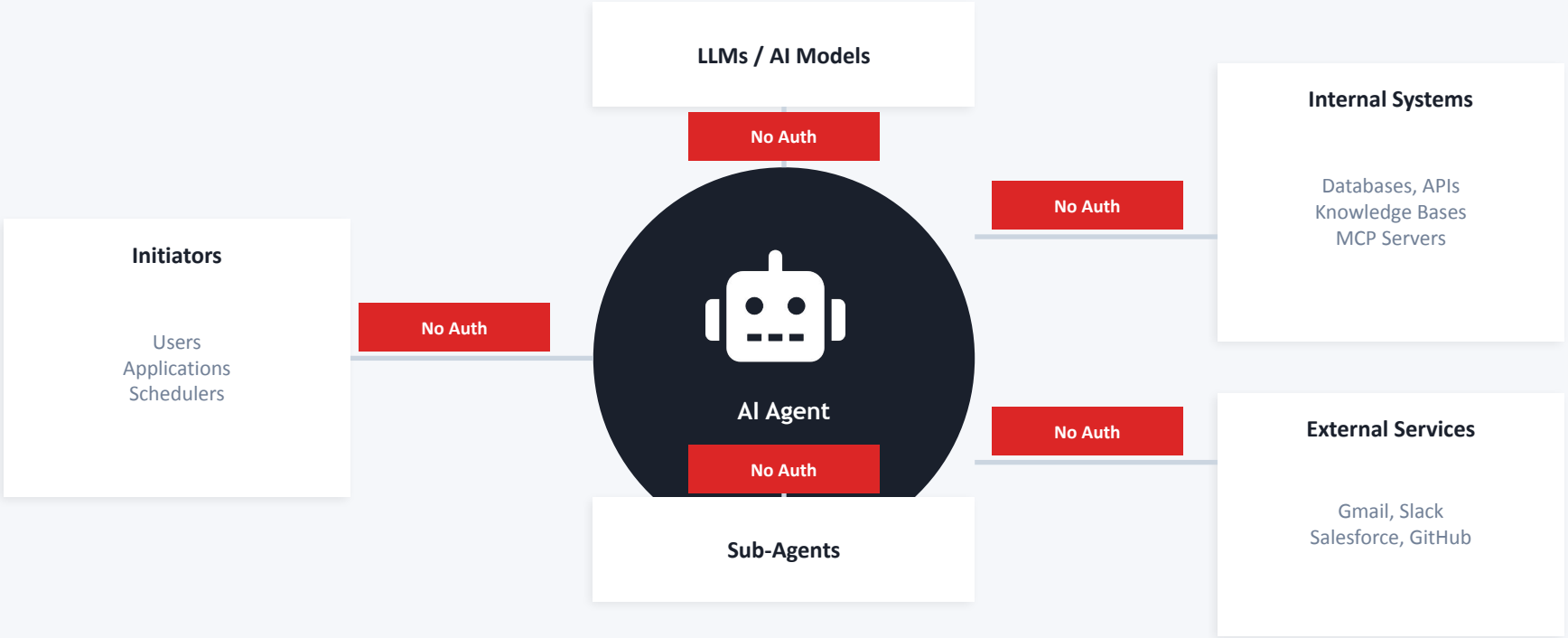
Access Control

AUDIT

Visibility & Compliance

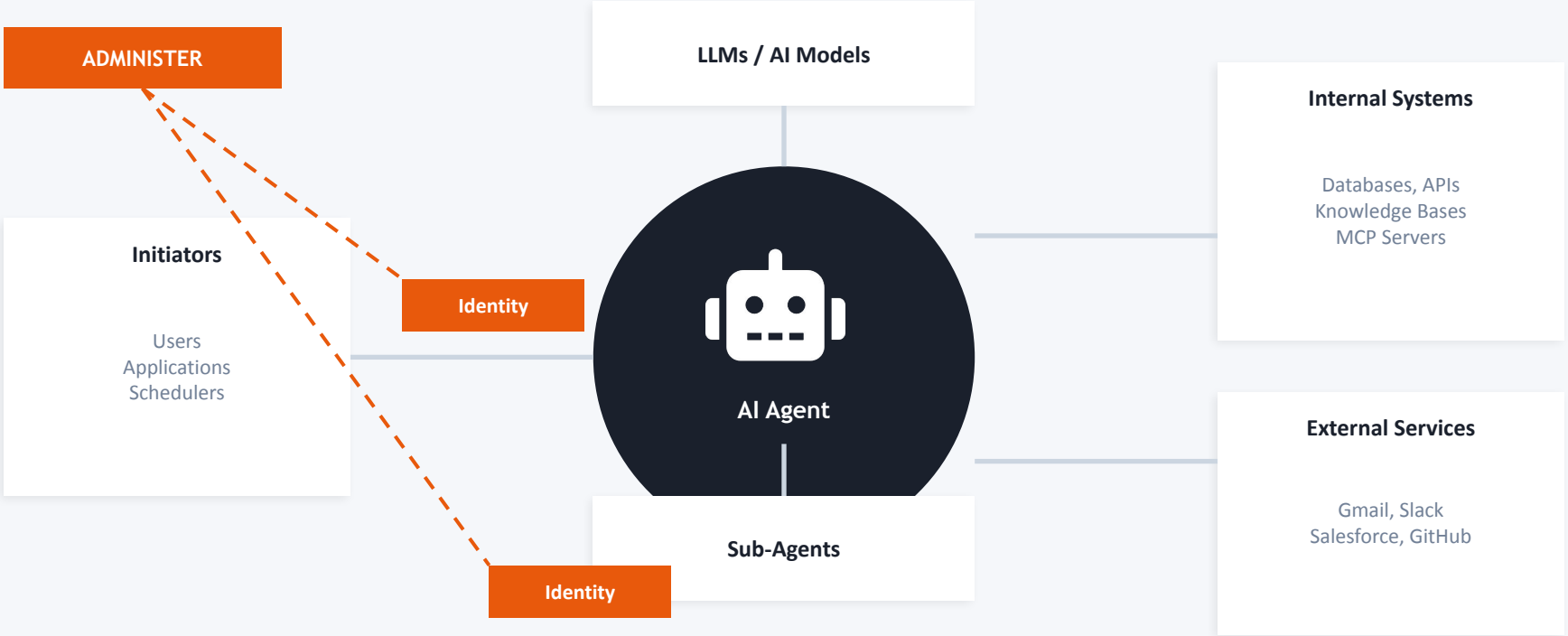
Agent Architecture: The Unsecured View

Without IAM, every connection boundary is an open attack surface.



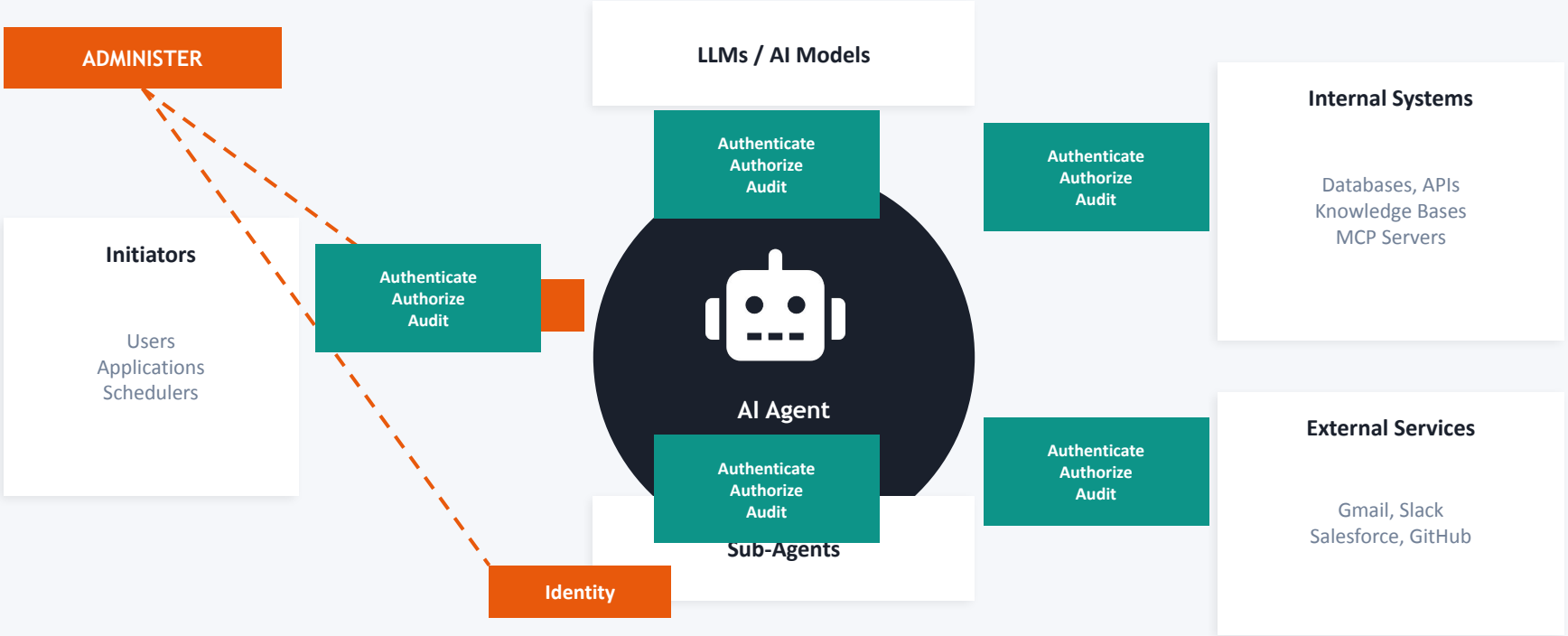
Step 1: Establish Agent Identity

Every agent and sub-agent gets a unique, verifiable identity with a liable owner — the foundation for all security.



Step 2: Secure Every Boundary

Authenticate, Authorize, and Audit at every connection — the Four Fundamentals applied to the architecture.



01

Administer

Agent Identity Lifecycle Management

Agent Identity Lifecycle Management

Agents as first-class identity principals — distinct from human users, yet equally manageable.

1

Register

Assign unique ID, metadata, and liable party

→

2

Configure

Set credentials, permissions, and constraints

→

3

Operate

Active lifecycle with monitoring and key rotation

→

4

Govern

Policy enforcement and compliance checks

→

5

Retire

Revoke, cascade cleanup, prevent orphaned IDs

02

Authenticate

Secure Agent-Friendly Credentials

Agent Authentication Methods

Credential strength scales with agent risk and task sensitivity — from simple secrets to multi-factor.

BASIC

Client Secret

Simple shared secret for low-risk, controlled environments

STRONG

mTLS

Mutual TLS with X.509 certificates for strong M2M authentication

ADVANCED

Private Key JWT

Asymmetric key auth with signed JWTs — no shared secrets

MAXIMUM

Agent MFA

Multi-factor for high-risk operations: step-up on sensitive actions

03

Authorize

Governing Agent Permissions & Access Control

Authorization: Autonomous vs. Delegated

Autonomous Operation

Agent acts under its own pre-provisioned RBAC permissions without user delegation.

- Agent-RBAC: roles assigned directly to agent identity
- Fine-grained tool authorization per MCP / API endpoint
- Principle of least privilege — read-only vs. write permissions
- Example: Data-engineer agent invokes warehouse MCP using its own pre-provisioned permissions

Delegated Access (On-Behalf-Of)

Agent carries delegated user authority via secure tokens that encode both identities.

- OAuth 2.0 delegation: consent-based, standards-driven
- User consent required — transparent, revocable delegation
- Agent cannot exceed delegated user's actual permissions
- Example: Research agent calls SEC API on behalf of Sarah, with both identities in the token

Authorization Use Cases

Step-Up for High-Risk Actions

An agent attempts a $50K wire transfer. Policy triggers human-in-the-loop approval via push notification to the account owner before the action executes — low-risk actions proceed automatically.

Context-Aware Delegation

Agent requests access to a sensitive HR database. IAM evaluates both the user's clearance level and the agent's Trust Tier before issuing a short-lived, downscoped credential.

Time-Bound Delegation Windows

A finance agent is granted write access only during the month-end close window (48 hours). After the window, the token expires automatically — no manual revocation needed.

External SaaS Token Vault

Agent requests a user's Google tokens from a managed vault. The vault performs JIT refresh, returns a fresh access token — the agent never sees the refresh token.

04

Audit

Comprehensive Visibility, Monitoring & Incident Response

Audit: Complete Visibility into Agent Actions

Monitor

- Real-time activity tracking
- Baseline behavior per agent
- Anomaly detection and alerting
- Unusual access pattern flags

Investigate

- Independent agent audit trails
- Who / what / when / where logging
- Full delegation chain tracing
- Rapid incident containment

Comply

- Regulatory compliance evidence
- Automated policy enforcement
- Accountability attribution
- Continuous governance loop

DEEP DIVE

Agent Protocols

Identity Controls for MCP and A2A — Securing Agent Boundaries

Model Context Protocol (MCP) and Identity

MCP connects AI agents to tools and data — but without identity controls, it becomes the weakest link.

WHAT IS MCP?

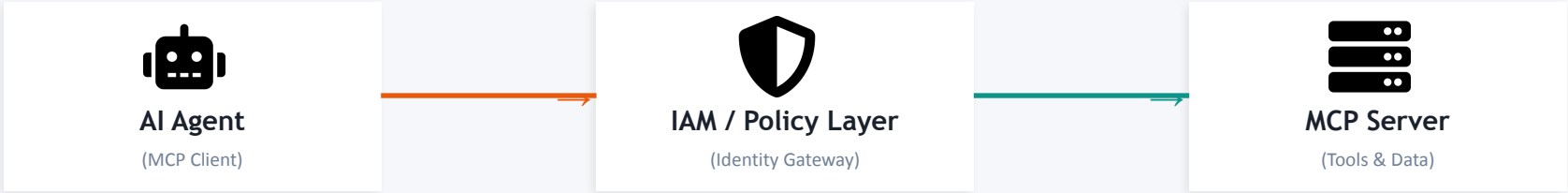
- Open protocol for connecting AI agents to external tools, data sources, and services
- Agents use MCP clients to invoke MCP servers that expose capabilities ("tools")
- Rapidly adopted: GitHub, Slack, databases, CRMs all expose MCP interfaces
- Agents can discover and invoke tools dynamically at runtime — powerful but risky

WHY MCP NEEDS IDENTITY

- **Inconsistent Auth Enforcement:** MCP's authorization spec is still evolving — many community servers don't implement it, leaving auth to the hosting environment
- **Tool Confusion Attacks:** Malicious content can trick agents into invoking unintended tools, executing commands the user never authorized
- **Supply Chain Risk:** Thousands of community MCP servers with no identity verification — any server could exfiltrate data or inject commands

Securing MCP with Identity Controls

Applying IAM principles to the agent-to-tool boundary — authenticate, authorize, and audit every tool invocation.



Authenticate the Agent

Every MCP client must present verifiable credentials before invoking any tool. No anonymous tool access.

Authorize per Tool

Fine-grained policies control which specific tools an agent can invoke — e.g., allow read_data but deny delete_record.

Validate Delegation

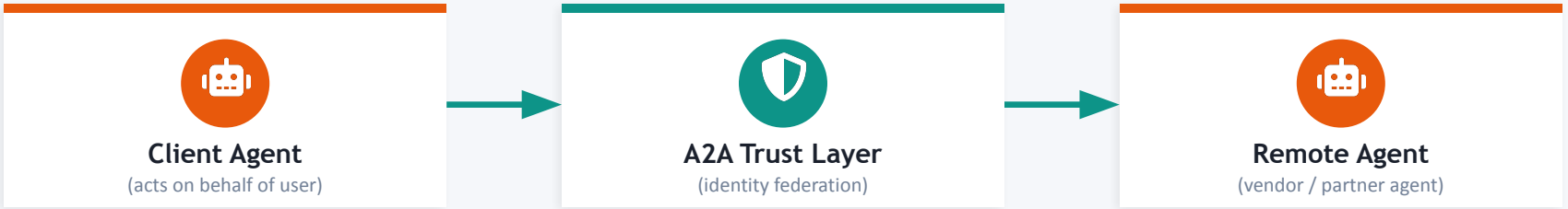
When acting on behalf of a user, the MCP server verifies both the agent's identity and the delegating user's permissions.

Audit Every Invocation

Log every tool call with: which agent, which tool, on whose behalf, what parameters, and what was returned.

A2A Protocol and Identity

Agent-to-agent communication across frameworks — identity anchors cross-vendor trust.



WHAT IS A2A?

- → Open protocol for agent-to-agent communication across vendors and frameworks
- → Agents expose capabilities via signed "Agent Cards" — discoverable skill manifests
- → Stewarded by Linux Foundation; backed by Google, OpenAI, Microsoft, Salesforce, and 50+ partners

KEY IDENTITY CONCERNS

- **Cross-Vendor Trust:** When your agent calls a partner's agent, both identities must be verifiable across organizational boundaries
- **Nested Delegation:** User consent must flow through every hop — agent A → agent B → agent C — not just the first call
- **Capability Spoofing:** Unsigned Agent Cards let malicious agents claim skills they don't have; federated signing prevents it

Leveraging Existing Identity Standards

Extend — don't reinvent — proven identity protocols for the agentic era.

OAuth 2.0

› **Authorization Code** — delegated user actions, with consent
› **Client Credentials** — autonomous agent operations
› **Token Exchange** — downscoping across multi-agent delegation chains

OpenID Connect

› **ID Tokens** — verifiable agent identity assertions
› **Discovery + Dynamic Registration** — agents self-register at runtime
› **CIBA** — out-of-band user consent for headless agents

SCIM 2.0

› **Agent schema extensions** — trust tier, model version, liable party
› **Lifecycle operations** — provision, suspend, deprovision agents
› **Filter queries** — find agents by owner, role, or status

OpenID Foundation Whitepaper on Agentic AI Identity • IETF Draft: OAuth 2.0 Extensions for AI Agent Delegation

Why Reuse Existing Standards?

Interoperability

Agents from different vendors and clouds authenticate through a unified identity fabric without proprietary lock-in.

Battle-Tested Security

OAuth 2.0, OIDC, and SCIM have years of production hardening and well-understood threat models.

Ecosystem & Tooling

Mature libraries, SDKs, and identity providers already support these protocols out of the box.

Regulatory Alignment

Compliance frameworks (SOC 2, GDPR, ISO 27001) already map to OAuth / OIDC patterns.

Federated Trust

Cross-organizational agent interactions via federated trust — just like federated SSO for humans.

Future-Proof

As standards evolve (OpenID AuthZEN, IETF agent drafts), organizations benefit from community-driven improvements.

Zero Trust Architecture for AI Agents

Assume no implicit trust — every agent must authenticate, every action must be authorized.

Verify Explicitly

Every agent interaction requires fresh context evaluation — identity, delegation chain, and trust tier. No cached trust.

Least Privilege

Agents receive minimum permissions per task. Dynamic downscoping ensures agents never hold more access than the current operation requires.

Assume Breach

Design for compromise. Segment access, monitor continuously, and enable instant revocation of any agent and its entire delegation tree.

End-to-End Example: Expense Approval Agent

1	Register	The expense-bot is registered as its own identity via SCIM provisioning, given a clear role (expense approver), permission limits ($500 max), and an assigned owner (DevOps team)
2	Delegate	A manager grants the expense-bot permission to approve expenses on their behalf via OAuth 2.0 consent — with a 30-day expiry
3	Authenticate	The agent proves who it is using secure credentials and receives a delegation token that carries both the agent and the manager's identity
4	Authorize & Act	The agent approves a $320 expense by calling a finance MCP tool. The system verifies both the agent and manager's permissions — and delegates via A2A when involving sub-agents
5	Audit	Every action is logged with full context — which agent, who delegated, which tool was invoked, when, and why — giving complete visibility across the delegation chain

Key Takeaways

- AI agents are first-class identity principals — treat them with the same rigor as human identities
- Leverage existing standards (OAuth 2.0, OIDC, SCIM) — extend, don't reinvent
- Implement the Four Fundamentals: Administer, Authenticate, Authorize, Audit
- Every agent needs a unique identity, a liable party, and automated lifecycle controls
- Delegation must be transparent, consent-driven, and carry full attribution in secure tokens
- Zero Trust is non-negotiable — verify every agent, authorize every action, audit every event
- Standards-based interoperability enables cross-vendor, cross-org agent ecosystems at scale

Thank You!

Questions & Discussion

Thivaharan Kalyanasundaram

WSO2 | Identity & Access Management

thivaharan@wso2.com