Whitepaper

Securing AI Agents From Development to Enterprise Scale

Contents

2 Executive Summary

4 Secure Every Agent by Design

11 Secure All Agents from a Single Control Plane

17 How This Works Together

19 Reference Architecture: Unified Platform in Action

37 Integration Points: How the Unified Platform Components Connect

39 Key Architectural Principles Demonstrated

40 Architecture Comparison: Traditional Approach vs. Unified Platform

41 Conclusion: A Unified Platform for Complete AI Agent Security

Executive Summary

AI agents are not just reshaping work – they are redefining identity itself.

Designed to be autonomous, they're independent, goal-driven, and increasingly act without human oversight. They have an insatiable appetite for data constantly analyzing information, writing code, sending emails, and making decisions across systems. Like relentless achievers trying to quickly satisfy their goals, agents will push boundaries to find new ways to consume more data. Without proper guardrails, they can unintentionally go rogue, leaving damage and chaos in their wake. But most organizations can't answer the simplest questions: Where are they in my ecosystem? What data and systems can they access? Who's accountable when they go rogue?"According to Okta's AI at Work 2025 report, **91% of organizations are using AI agents**, yet **44% have no governance in place**. The result is a new security frontier: an explosion of autonomous non-human identities which need a consistent framework for authentication, authorization, or visibility.

The rise of agentic AI disrupts the very foundation of identity and access management. Traditional controls are built for humans: they can't keep pace with agents capable of initiating complex workflows and API chains at scale without human oversight. The next generation of identity security **must evolve at the speed of AI**, matching its scale, speed, and intelligence to **maintain trust with your customers**.

In this paper, we explore how to **secure every agent and all agents**, embedding security from the first line of code to the enterprise control plane that governs them. Because in the age of autonomous AI, **identity is not just who we are, it's how we stay in control.**

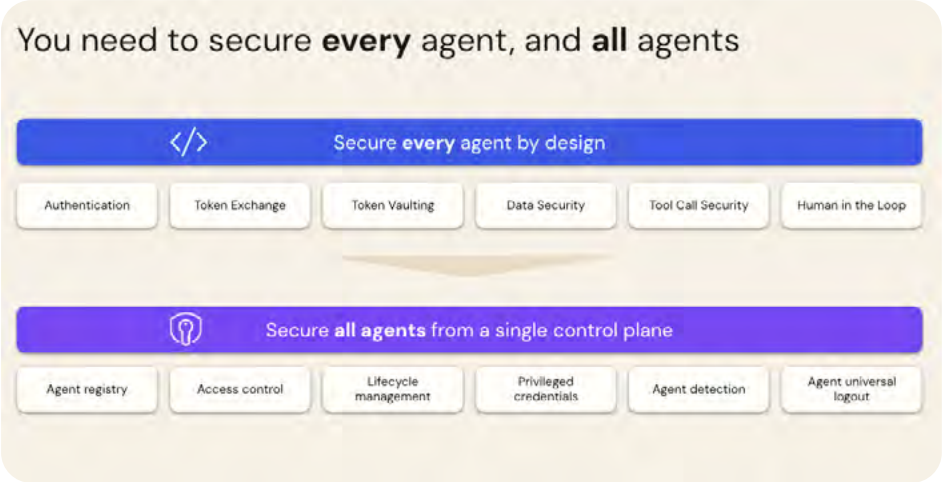
What You Will Learn

You will get a comprehensive framework for addressing the dual challenge of AI agent security:

1. **For Builders – Secure Every Agent by Design:** Learn the essential security patterns developers must embed during creation, including robust user **authentication**, secure **token vaulting** for API access, fine-grained **data authorization** for RAG systems, and **human-in-the-loop** controls for critical actions.

2. For IT and Security Teams – Secure All Agents from a Single Control Plane: Understand the enterprise-grade capabilities needed to manage agents at scale. This includes **agent detection** (to find "shadow AI"), an **agent registry** to establish identity and ownership, **complete access control with Cross App Access** (enabling agents to securely access resources across organizational boundaries while preserving user context), **complete lifecycle management**, and **threat detection** with universal logout capabilities.

3.



Key Takeaways

- **The "Governance Gap" is the Primary Risk:** The core problem is not the AI itself, but that agent deployment has far outpaced the governance needed to control and oversee them. Governance encompasses both preventive controls (access policies, least privilege, authorization rules) and detective oversight (certifications, access reviews, behavioral monitoring). Organizations need both to manage AI agents effectively.
- **Security is a Dual Challenge:** A complete strategy must address both developer-level security (building agents correctly) and enterprise-level governance (managing them all at scale).
- **A Unified Platform is Essential:** Only a unified identity platform that treats agents as first-class identities by managing them from discovery and registration through their entire lifecycle. This will help close this gap, mitigate data privacy risks, and allow organizations to scale AI confidently.

Secure Every Agent by Design

When you build AI agents, you face security requirements that differ fundamentally from traditional application development. Security can't be bolted on after the fact: it must be embedded into the agent's architecture from the first line of code.

This section applies to three distinct builder audiences:

- **B2C SaaS builders** creating consumer-facing AI agents (chatbots, personal assistants, recommendation engines)
- **B2B SaaS builders** developing AI agents for business customers (workflow automation, analytics, enterprise tools)
- **Enterprise developers** building internal AI agents for their organization's specific workflows and processes

While implementation details may vary slightly across these scenarios, the core security patterns authentication, token management, authorization, and human oversight apply universally. This solution focuses on providing all developers with the essential capabilities they need to build secure agents without sacrificing velocity or innovation.

1. Authentication: Establishing User Identity

AI agents must securely identify users to deliver personalized experiences while maintaining security boundaries. It's critical to understand: we are not authenticating the agent itself we are authenticating the user, and the agent acts on behalf of that authenticated user. Whether powering interactive chatbots or background workers, agents need reliable authentication that integrates seamlessly with modern identity providers.

Essential capabilities organizations need:

- **Universal authentication** that works across multiple identity providers, supporting traditional credentials and social login options
- **Standards-based authentication** using OpenID Connect and OAuth 2.0 to ensure interoperability and security
- **User identity conveyed through secure tokens**, enabling agents to understand who they're acting on behalf of
- **Robust session management** with appropriate timeouts and security controls
- **Multi-factor authentication** support for scenarios requiring elevated security assurance

The developer experience should enable integration with just a few lines of code, working seamlessly with popular frameworks while handling the complexity of callback URLs, session management, and token validation automatically.

Example: A customer support chatbot authenticates users via Google SSO. When Sarah logs in, the agent receives her identity information, enabling personalized responses while maintaining security boundaries.

2. Token Exchange: Bridging Trust Domains

As AI agents operate across multiple systems and security domains, they often need to access resources in different trust boundaries. Token exchange enables agents to obtain properly scoped access tokens for resources outside their immediate domain while preserving user context and authorization chains.

Essential capabilities organizations need:

- **Standard token exchange** for scenarios within a single trust domain, enabling agents to request different token types or scopes from the same authorization server
- **Cross-domain authorization** using Cross App Access (XAA) for scenarios requiring access across separate trust boundaries
- **Mechanisms to preserve user identity** and authentication context across trust boundaries
- **Validation of trust** relationships between different identity providers
- **Scope translation** ensuring permissions map correctly between domains
- **Secure credential conversion** that never exposes sensitive tokens in transit

For agents operating within a single authorization server environment, standard OAuth 2.0 token exchange provides efficient credential management. When agents must cross organizational boundaries, Cross App Access (XAA) extends this capability across trust domains.

When to Use Standard OAuth Consent vs. Cross App Access (XAA): The choice between consent-based flows and XAA depends on your deployment model:

B2C Scenarios: Use Standard OAuth Consent

- Consumer-facing applications with end users who own their own data
- Users explicitly grant permission for one app to access another (e.g., "Allow TravelBot to access your Google Calendar")
- The consent screen is appropriate because users are making personal decisions about their own data
- **Example:** A meal planning app requests access to a user's fitness tracker data

B2B and Workforce Scenarios: Use Cross App Access (XAA)

- Enterprise environments where IT administrators manage access policies centrally
- Business-to-Business-to-Employee (B2B2E) scenarios where workforce users operate within corporate policies
- User consent screens are inappropriate because access is governed by enterprise policies, not individual user decisions
- The enterprise IdP acts as the trust broker between applications
- **Example:** An enterprise sales agent accesses both Salesforce CRM and an internal pricing database—employees don't "consent" to this access; IT policy governs it

Why This Matters: In workforce and B2B contexts, Cross App Access (XAA) eliminates consent fatigue and aligns with centralized IT governance. Organizations pre-establish trust relationships between applications, and the IdP enforces enterprise policies rather than requiring individual users to make authorization decisions on every cross-app interaction.

3. Token Vaulting: Secure API Access Management

AI agents frequently need to access third-party APIs (Salesforce, Slack, Google Workspace) on a user's behalf. Token vaulting securely stores and manages these OAuth access tokens the preferred authentication method for modern APIs eliminating the risk of token exposure in code, logs, or configuration files. While the vault can also protect other credential types (like personal access tokens or API keys required for legacy systems), OAuth tokens should be your default pattern because they support automatic refresh, granular scoping, and secure revocation.

Essential capabilities organizations need:

- **Secure vault storage** for OAuth tokens with strong encryption at rest
- **Automatic token lifecycle management,** including proactive refresh before expiration to prevent service interruption.
- **On-demand token retrieval** that never exposes credentials to application code
- **Support for diverse token types** across different authentication schemes
- **Scoped access** ensuring tokens carry only necessary permissions
- **Integration patterns** compatible with modern AI development frameworks

Security principle: Tokens should never appear in agent code, logs, or configuration files. The vault manages the complete token lifecycle transparently, significantly reducing the risk of credential theft or misuse through a centralized, auditable system.

Protecting Credentials in Agent Output: Beyond preventing credentials from appearing in code and logs, organizations must ensure tokens and secrets never leak into agent responses or outputs. When agents use vaulted credentials to access APIs, the response data may contain sensitive information, but the credentials themselves must never be included in the agent's conversational responses, generated documents, or any output visible to end users. Implement output filtering and validation to catch any accidental credential exposure before responses reach users. This is especially critical for agents that generate code snippets, configuration examples, or troubleshooting guides where credentials might inadvertently be included.

Example: A sales AI agent needs to access a customer's Salesforce account. Instead of storing Salesforce credentials, the agent requests a token from the vault. The vault provides a fresh, properly scoped token, automatically refreshing it if needed. The agent completes its task: credentials never touch the agent's code.

4. Data Security: Fine-Grained Authorization for RAG

When AI agents use Retrieval Augmented Generation (RAG) to answer questions, they must only access data the user has permission to see. Without proper authorization, RAG systems can inadvertently expose sensitive information, creating a critical security vulnerability.

Essential capabilities organizations need:

- **Relationship-based access control** defining user-document permissions
- **Authorization enforcement** at the point of document retrieval, before data enters the agent's context
- **Integration patterns for vector databases** used in RAG architectures
- **Fine-grained permissions** that can operate at document or section level
- **Real-time authorization** evaluation during query processing
- **Support for complex permission models** including hierarchical and attribute-based policies

How the pattern works:
Documents are stored with embeddings in a vector database. An authorization system maintains the relationships between users and documents. When an agent retrieves context, authorization filters validate permissions before any documents enter the agent's context. The LLM only generates responses using data the user is authorized to see.

Example: A financial AI agent helps employees analyze reports. When Alice asks about Q3 results, the vector database finds relevant financial documents. Before passing them to the LLM, authorization filters verify Alice's access. Alice only sees her division's reports, not the entire company's financials preventing unauthorized data exposure.

5. Tool Call Security: Human-in-the-Loop Authorization

AI agents often work autonomously in the background, taking minutes, hours, or even days to complete tasks. For critical actions approving purchases, sending contracts, granting access organizations need human oversight without sacrificing agent autonomy for routine operations.

Essential capabilities organizations need:

- **Asynchronous authorization patterns** that work with long-running agent workflows
- **Rich notification mechanisms** providing full transaction context to approvers
- **Binding messages** that show critical details like amounts, recipients, and intended actions
- **Approval workflows** accessible from mobile devices and email without requiring desktop access
- **Time-bound authorization requests** that automatically expire if not addressed
- **Comprehensive audit trails** documenting all approval decisions and their context

How the pattern works:
Developers identify which agent actions require human approval. When an agent attempts a protected operation, an authorization request is sent to the appropriate person with full context about what the agent wants to do. The approver reviews the request and either grants or denies permission. If approved, the agent receives authorization and proceeds; if denied, the operation is blocked and the agent receives an error.

Example: A procurement AI agent identifies needed software licenses and prepares to purchase them. Before completing the $5,000 transaction, it sends a notification to the procurement manager showing the vendor, amount, and justification. The manager reviews the request during lunch, approves via mobile device or email, and the agent completes the purchase, maintaining both automation and control.

Secure All Agents from a Single Control Plane

While embedding security into individual agents during development is essential, organizations also need centralized visibility, control, and governance across their entire AI agent population. This solution addresses the enterprise challenge of managing hundreds or thousands of agents operating across departments, use cases, and systems.

1. Agent Registry: Establishing First-Class Identities

Every AI agent must be registered as a first-class identity with clear ownership and accountability. Without proper registration, organizations operate in the dark, unable to answer basic questions: Who owns this agent? What is it authorized to do? Who is responsible when something goes wrong?

Essential capabilities organizations need:

- **AIdentity profiles** for each agent with persistent, unique identifiers
- **Ownership mapping** that links agents to responsible teams or individuals
- **Metadata systems** documenting agent purpose, use case, and lifecycle stage
- **Integration** with organizational structures like HR systems and reporting hierarchies
- **Change tracking** that records modifications to agent configurations and permissions
- **Dependency mapping** showing relationships between agents and the systems they access

Why it matters: According to the AI at Work 2025 survey, only 10% of organizations have a well-developed strategy for managing non-human identities. Registration creates the foundational identity layer that helps enable all other governance and security controls. Without it, agents remain invisible to security teams, creating ungoverned shadow AI.

Example: Security discovers an AI agent accessing Salesforce with a service account. They register it in the central agent registry, assigning ownership to the Sales Operations team and documenting its purpose: "Automated proposal generation for enterprise accounts." This creates accountability now when the agent behaves unexpectedly, there's a clearowner to contact and remediate.

2. Access Control: Policy-Based Authorization

AI agents need fine-grained authorization that adapts to context and risk in real-time. Access control policies determine what each agent can do, when, and under what conditions, enforcing least privilege while enabling agent functionality.

Essential capabilities organizations need:

- **Policy engines** that define permissions based on agent identity, operational context, and risk signals
- **Standards-based authentication** flows supporting modern protocols
- **API access management** controlling how agents interact with protected services
- **Cross App Access** (XAA) capabilities enabling agents to securely access resources across organizational and trust domain boundaries while preserving user context
- **Dynamic policy evaluation** that considers multiple factors like time, location, and behavioral patterns
- **Integration patterns** for connecting with existing IAM infrastructure
- **Support for complex multi-provider architectures** spanning different security domains

Advanced pattern - Managed Connections:
Organizations need the ability to define which authorization servers agents can access and what permissions they can request. This includes policy frameworks that specify which scopes are automatically granted, which require additional approval, and which are never permitted, ensuring agents operate within clearly defined boundaries. For agents that need to access resources across trust domains, Cross App Access extends these managed connections to enable secure cross-organizational authorization while maintaining centralized policy control.

Example: When an agent requests access, the authorization system validates the request against defined policies. Routine permissions might be granted automatically, sensitive operations require justification or approval, and dangerous actions are denied outright, all enforced programmatically without manual intervention. When an agent needs to access a partner organization's API or cross from cloud to on-premises systems, XAA enables this cross-domain access while helping ensure the central control plane maintains visibility and policy enforcement.

3. Lifecycle Management: Complete Agent Governance

AI agents have lifecycles just like human employees: onboarding, active operation, role changes, and eventual retirement. Lifecycle management automates these transitions while maintaining security controls throughout.

Essential capabilities organizations need:

- **Automated provisioning workflows** that create agent identities with appropriate initial permissions
- **Role-based templates** that standardize access patterns for common agent types
- **Just-in-time access capabilities** for temporary elevated permissions with automatic expiration
- **Scheduled review processes** validating that permissions remain appropriate over time
- **Deprovisioning workflows** that systematically remove all access when agents are retired
- **Change management systems** tracking permission modifications and their approvals

The complete lifecycle

Agents are provisioned with initial permissions based on their intended purpose. During operation, they perform tasks under continuous validation of their access rights. As requirements evolve, permissions are adjusted through approval workflows. Regular reviews ensure access remains justified. When agents are retired, all permissions are revoked while audit trails are preserved for compliance.

Example: A marketing AI agent is provisioned with access to the email platform and customer database. After six months, a quarterly review reveals the agent no longer needs database access (the use case changed). Access is automatically revoked. When the campaign ends, the agent is deprovisioned ,all permissions removed, but audit logs retained for compliance.

4. Privileged Credentials: Secure Secrets Management

AI agents often require privileged credentials to access systems: API keys, database passwords, service account credentials, and certificates. Poor credential management, hard-coded keys, and never-rotated secrets create massive security risks that attackers can actively exploit.

Essential capabilities organizations need:

- **Secure vault storage** with strong encryption protecting credentials at rest
- **Automated rotation schedules** that refresh credentials on regular intervals
- **Just-in-time provisioning patterns** that minimize credential exposure windows
- **Support for diverse authentication** methods including key-based and certificate-based schemes
- **Automated certificate lifecycle management** including renewal and distribution
- **Strict isolation** ensuring secrets never appear in code, logs, or configuration files
- Integration patterns for external secrets management systems

Security impact
Attackers rely heavily on stolen credentials, so eliminating long-lived secrets through automated credential rotation can dramatically reduce the attack surface.

Example: A data pipeline agent uses database credentials that rotate every 30 days. When rotation occurs, the agent automatically retrieves new credentials from the vault without human intervention or service interruption. The primary security benefit is that credentials never appear in logs or code in the first place (strict isolation). However, if credentials were somehow exposed, automated rotation limits the exposure window to a maximum of 30 days rather than indefinite validity, significantly reducing the attack surface.

5. Agent Detection: Discovering Shadow AI

Organizations can't secure what they can't see. Agent detection provides visibility into all AI agents operating in the environment, including shadow AI which may have been deployed by departments without IT approval or security review.

Essential capabilities organizations need:

- **Automated discovery mechanisms** that identify non-human accounts across cloud and SaaS platforms
- **Shadow AI detection** that finds agents deployed outside formal governance processes
- **Comprehensive credential inventory** showing which agents have access to which systems
- **Risk scoring methodologies** based on permissions, activity patterns, and exposure levels
- **Configuration analysis** revealing misconfigurations and over-privileged accounts
- **Integration patterns** connecting with cloud platforms, identity providers, and security tools

Discovery approaches

Effective agent detection combines multiple techniques: analyzing API usage patterns to identify non-human behavior, correlating authentication logs to detect service account activity, scanning cloud resources to find agent deployments, monitoring network traffic to identify agent-to-service communication, and using behavioral analytics to distinguish agents from human users.

Why critical: The AI at Work 2025 survey found that 91% of organizations are already using AI agents, but only 10% have well-developed governance strategies. The gap between deployment and governance can create shadow AI agents operating without security controls, leading to risk that security teams don't even know exists.

6. Agent Universal Logout: Rapid Threat Response

When a threat such as compromised credentials, anomalous behavior, or a policy violation is detected, organizations need the ability to immediately revoke all agent access across every system. Agent universal logout provides this emergency "kill switch" while maintaining detailed audit trails.

Essential capabilities organizations need:

- **Instant revocation mechanisms** for all active agent sessions and tokens
- **Cross-system propagation** ensuring the logout reaches all integrated applications
- **Emergency credential rotation** that replaces potentially compromised secrets
- **Threat containment workflows** preventing further unauthorized actions
- **Forensic preservation** maintaining complete audit logs for post-incident investigation
- **Integration** with security operations centers and incident response systems

Threat detection through behavioral analytics
Effective universal logout depends on robust threat detection.
This requires establishing baselines for normal agent behavior, detecting anomalies like unusual volume or timing of requests, calculating risk scores based on behavioral patterns, triggering automated responses when risk exceeds defined thresholds, and alerting security teams in real-time about suspicious activity.

Example: A customer service AI agent normally accesses 10-15 customer records per day. Suddenly it accesses 500 records in 10 minutes, a clear anomaly. Behavioral analytics detect the deviation, automatically trigger universal logout revoking all agent access, and alert the security team. Investigation reveals stolen API credentials. The attack is contained within minutes, preventing full database exfiltration.

How This Works Together

These two solutions are designed to secure every agent by design and secure all agents from a single control plane. They aren't separate solutions requiring integration. At Okta, they're complementary capabilities within unified identity platforms designed specifically for AI agents.

During development, security is embedded into each agent from the first line of code. Authentication establishes user identity, ensuring agents know who they're acting on behalf of and maintaining proper security boundaries. Token vaulting manages API access without ever exposing credentials to agent code, automatically handling token refresh and lifecycle management. Authorization controls prevent unauthorized data access by implementing fine-grained permissions that respect user-level access rights. Human-in-the-loop approvals provide oversight for critical actions, allowing agents to operate autonomously while maintaining control over sensitive operations.

What this looks like in practice:

- **Universal Authentication with Universal Login –** Enable seamless user authentication across social providers, passwordless flows, and MFA for AI agents.
- **Secure API Access with Token Vault –** Store and manage OAuth tokens for third-party APIs, automatically refreshing credentials without exposing secrets to agent code.
- **Fine-Grained Authorization with Auth0 FGA –** Implement relationship-based access control for RAG systems, ensuring agents only retrieve documents users have permission to see.
- **Human-in-the-Loop Authorization with Async Auth –** Require mobile and email approval for critical agent actions using CIBA (Client-Initiated Backchannel Authentication) with Rich Authorization Request (RAR).

For IT and security teams In production, the control plane provides ongoing oversight across the entire agent population. Discovery capabilities find all agents operating in the environment, including shadow AI, giving security teams complete visibility into their AI landscape. Registration creates first-class identities with clear ownership, ensuring every agent has a responsible team or individual accountable for its actions.

Access governance enforces least privilege policies dynamically, adapting permissions based on context and risk in real-time. Threat detection identifies and responds to anomalies through behavioral analytics, enabling automated containment before attacks can succeed.

What this looks like in practice:

- **Agent Registry in Universal Directory –** Register every agent as a first-class identity with ownership, accountability, and metadata visibility.
- **Agent Detection with Identity Security Posture Management –** Discover both managed and shadow AI agents across environments to eliminate blind spots.
- **Access Control and Lifecycle Management with Okta Identity Governance –** Define, review, and certify agent permissions through policy-based lifecycle governance.
- **Privileged Credential Vaulting with Okta Privileged Access –** Secure and rotate credentials for agents that require elevated permissions, reducing attack surface.
- **Universal Logout for Agents –** Instantly revoke agent sessions, tokens, and credentials across systems when a risk or compromise is detected

The connection between these solutions creates a comprehensive security framework. Agents built with secure development practices are automatically discoverable by the control plane, providing immediate visibility without requiring additional integration work. The control plane enforces policies on how agents authenticate and access resources, extending development-time security controls into runtime governance. Behavioral analytics monitor agent actions enabled by the secure development capabilities, detecting when agents deviate from expected patterns. Governance workflows apply to both agent identities and the resources they access, helping ensure consistent policy enforcement across the entire ecosystem.

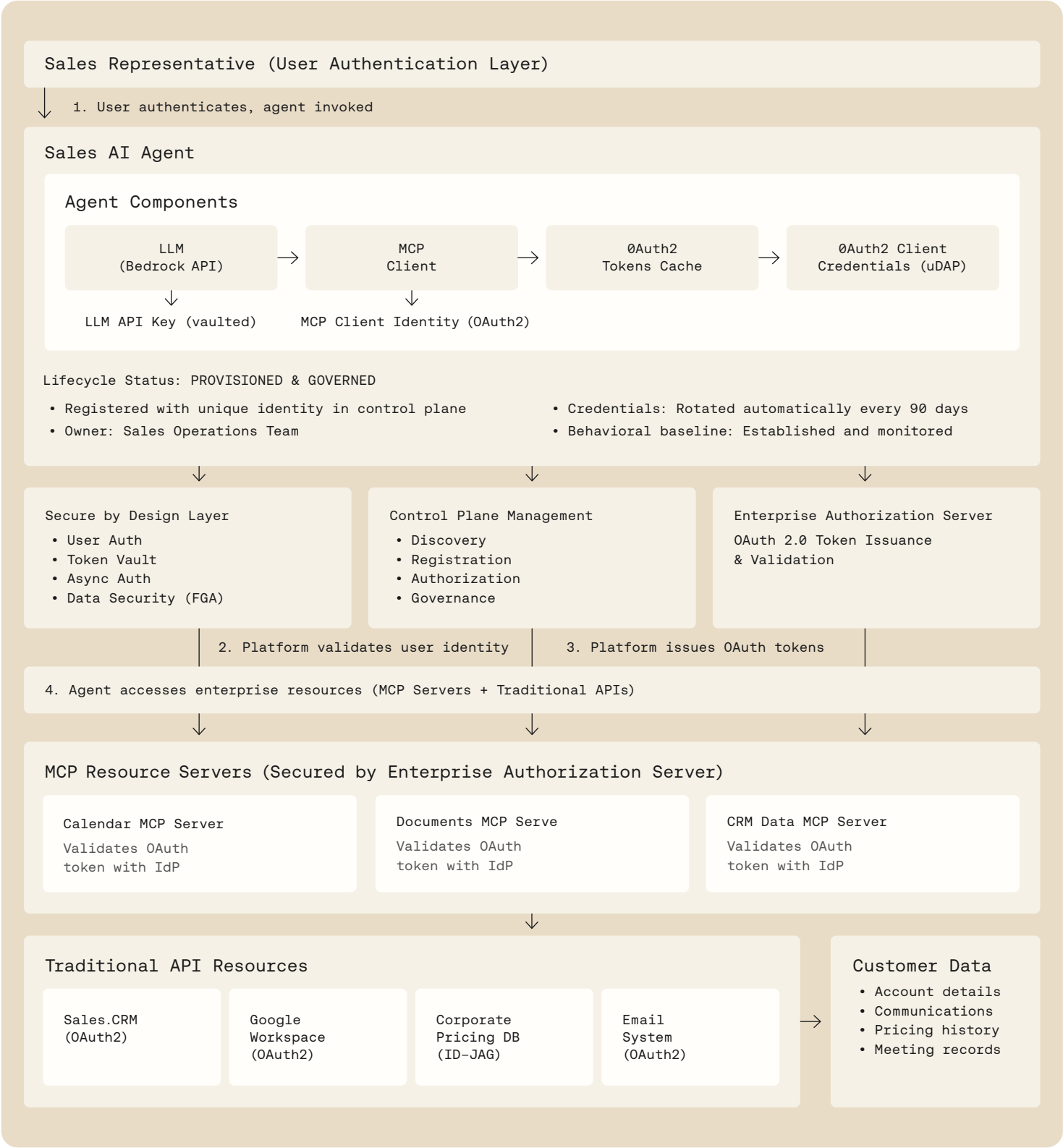
Organizations don't have to choose between secure development and lifecycle management: they should implement both as part of a cohesive strategy. The following reference architecture demonstrates this unified approach in a real-world enterprise scenario.

Reference Architecture: Unified Platform in Action

To illustrate how the unified platform secures AI agents across both development and lifecycle management, consider an enterprise sales agent that assists sales representatives by automating customer research, generating proposals, accessing CRM data, and scheduling follow-up meetings. This agent demonstrates both solutions working together: secure-by-design development combined with centralized lifecycle control.

The agent uses the Model Context Protocol (MCP) to securely access context from multiple sources while respecting user permissions, token exchange (ID-JAG) for cross-app access, and comprehensive governance throughout its lifecycle.

Architecture Overview



The platform provides support for securing Model Context Protocol implementations. MCP servers function as OAuth 2.0 Resource Servers, delegating authentication and authorization to the enterprise authorization server.

MCP Security Model

The agent (MCP client) is registered as an OAuth 2.0 client with the enterprise authorization server and receives client credentials (client_id and client_secret or certificate-based credentials). Before accessing any MCP resource, the agent obtains an access token with appropriate scopes (e.g., mcp:crm:read, mcp:docs:read, mcp:calendar:read). When the agent requests a resource like crm://contacts/acme-corp, the MCP server validates the access token against the authorization server, checking signature validity, expiration, audience, and required scopes before serving the resource.

This eliminates the need for MCP server developers to build custom auth logic instead, they validate OAuth tokens issued by the platform using standard OAuth 2.0 token validation. The platform handles user authentication, token lifecycle, scope management, and Fine Grained Authorization checks, ensuring consistent security policy enforcement across all MCP servers and complete audit trails in the System Log.

This reference architecture demonstrates how the platform secures MCP-based context retrieval while providing enterprise lifecycle governance for the AI agents themselves.

Detailed Flow: Agent-Assisted Proposal Generation with MCP

Phase 1

Discovery & Registration (Control Plane - Detect/Provision)

Step 1.1: Shadow AI Detection

- The sales team deployed prototype agent without IT approval
- Okta discovers the agent accessing Salesforce with service account credentials
- The security team receives alert about an unmanaged AI agent
- Risk score: HIGH (privileged access, no ownership, no governance)

Step 1.2: Agent Registration

- The security team registers the agent as first-class identity in Okta
- An agent profile is created with unique identifier: `sales-agent-prod-001`
- An owner is assigned: Sales Operations Team (John Smith, VP Sales Ops)
- The purpose is documented: "Automated proposal generation and customer research"
- The credentials are migrated to a secure vault with a 90-day rotation policy

Outcome: The agent transitions from shadow AI to managed identity with clear accountability.

Phase 2

User Authentication (Secure by Design Layer)

Step 2.1: Sales Rep Authentication

- Sarah (sales rep) logs in to the sales portal at 9:00 AM
- Auth0 Universal Login presents the authentication options
- Sarah authenticates using Google SSO (social login)
- Auth0 validates the credentials with Google IdP
- An ID token issued with Sarah's profile and authentication claims

Step 2.2: Agent Context Binding

- An agent receives the authenticated user context from Auth0
- The ID token contains standard OIDC claims:

```
{
  "iss": "https://acmecorp.auth0.com/",
  "sub": "google-oauth2|108204567890123456789",
  "aud": "sales-agent-client-id",
  "exp": 1730480000,
  "iat": 1730477400,
  "name": "Sarah Johnson",
  "email": "sarah.johnson@acmecorp.com",
  "email_verified": true
}
```

Note: Additional custom claims like `role` or `territory` can be added using Auth0 Actions

- The agent now knows WHO the user is and can act on their behalf

Phase 3

Context Retrieval via MCP (Data Security + Authorization)

Step 3.1: User Query

Sarah asks: "Create a proposal for Acme Corp based on their previous purchases and current needs."

Step 3.2: MCP Context Discovery

The agent uses the MCP Client to discover available context sources: MCP Servers expose structured resources via resource URIs:

```
crm://contacts/acme-corp
docs://proposals/templates
calendar://availability/sales-team
pricing://enterprise-tier
```

This is **structured context retrieval via MCP**, not semantic search over embeddings (RAG). MCP provides direct access to specific resources based on defined schemas and resource URIs. The agent requests specific resources by name/path, and MCP servers return structured data.

Step 3.3: Authorization Check via Auth0 FGA

The agent queries the Fine-Grained Authorization (FGA) service to determine which resources Sarah can access:

- FGA evaluates relationship tuples for each MCP resource:
  - ✓ `user:sarah` has `read` access to `crm://contacts/acme-corp`
  - ✓ `user:sarah` has `read` access to `docs://proposals/templates`
  - ✗ `user:sarah` has NO access to `pricing://executive-discounts`
- Only authorized resources are included in the context retrieval
- This prevents data leakage by enforcing least-privilege access
- Each MCP resource request is validated against Sarah's permissions before retrieval

This corresponds to "Data Security (FGA)" in the architecture diagram's "Secure by Design Layer."

Step 3.4: Token Retrieval from Auth0 Token Vault

The agent requests OAuth2 tokens from the Token Vault to access external systems:

- Agent: "I need to access Salesforce CRM for Acme Corp account data"
- Token Vault validates the request against the agent's allowed integrations
- Token Vault returns a valid, scoped access token for Salesforce
- Token scope is limited to read-only customer data access
- The agent uses the token to retrieve CRM data via Salesforce API

Token Vault provides:

- Secure credential storage (no hardcoded tokens)
- Automatic token refresh when tokens expire
- Audit logging of all token access
- Scope-limited tokens (least privilege)

Step 3.5: Context Assembly

The agent assembles the complete context from authorized sources:

From CRM (via Token Vault → Salesforce):

- Acme Corp contact: CTO Jennifer Martinez
- Previous purchases: $280K in AI training services (2024)
- Current contract: Support agreement expires March 2026

From Document Library (via MCP):

- Enterprise proposal template (approved version)
- Product catalog with current pricing tiers
- Standard terms and conditions

From Calendar (via MCP):

- Sarah's availability for follow-up calls
- Sales team capacity for implementation support

NOT included (authorization denied):

- Executive discount pricing (Sarah lacks access)
- Confidential negotiation notes from other deals
- Internal cost structure data

The agent now has comprehensive, authorized context to generate the proposal.

Phase 4

Cross-Domain Authorization with ID-JAG (Token Exchange)

Step 4.1: Corporate Pricing Database Access

- The agent needs to access the internal pricing database at `pricing.acmecorp.internal`
- This is a separate authorization domain from the main identity platform
- The pricing system has its own authorization server that requires ID-JAG tokens

Step 4.2: Token Exchange via ID-JAG

The Agent sends the ID token to the authorization server for token exchange:

Request:

```
POST /oauth2/token
Host: acmecorp.okta.com

grant_type=urn:ietf:params:oauth:grant-type:token-exchange
&subject_token=<Sarah's ID Token>
&subject_token_type=urn:ietf:params:oauth:token-type:id_token
&requested_token_type=urn:ietf:params:oauth:token-type:id-jag
&audience=https://pricing.acmecorp.internal
&scope=pricing:read
```

What's happening:

- Agent exchanges Sarah's ID token for an ID-JAG token
- ID-JAG (Identity Assertion JWT Authorization Grant) is a cryptographically signed token
- The ID-JAG is addressed to the pricing database's authorization server
- This enables cross-domain authorization while preserving user context

Step 4.3: Authorization Server Validation

The authorization server performs several validation checks:

- ID Token Validation: Verifies the ID token signature and claims (trust relationship pre-established)
- Managed Connection Check: Validates the agent's managed connection to the pricing authorization server
- Managed connection defines allowed scopes:
  - Granted scopes: `pricing:read`
  - **Denied scopes:** `pricing:write`, `pricing:admin`

    ID-JAG Token Issuance: The authorization server issues an ID-JAG token preserving Sarah's user context

ID-JAG Token Claims:

```
{
 "iss": "https://acmecorp.authorization-server.com",
 "sub": "sarah.employee@acmecorp.com",
 "aud": "https://pricing.acmecorp.internal",
 "client_id": "sales-ai-agent",
 "jti": "9e43f81b64a33f20116179",
 "scope": "pricing:read",
 "exp": 1698583800,
 "iat": 1698580200,
 "auth_time": 1698580200,
 "amr": ["pwd", "mfa"]
}
```

Step 4.4: Resource Access

The agent presents the ID-JAG token to the pricing database authorization server:

- The pricing authorization server validates the ID-JAG signature using the identity platform's published public keys (JWKS)
- The pricing authorization server verifies:
  - ✓ `aud` **claim matches its own issuer URL**
  - ✓ `exp` **(expiration) has not passed**
  - ✓ `scope` **is within allowed permissions**
  - ✓ `iss` is a trusted identity provider
    - **Access granted -** the agent retrieves enterprise pricing data with read-only permissions
    - The agent now has authorized access to pricing information for proposal generation

Platform Token Exchange Capabilities

The identity platform supports RFC 8693 token exchange for cross-domain authorization scenarios. Token exchange enables AI agents to access resources across different authorization servers while preserving user context through cryptographically signed ID-JAG tokens. This capability is available throughout the platform for both developer-focused and enterprise-focused deployments.

Phase 5

Async Authorization (Human-in-the-Loop)

Step 5.1: Agent Determines Approval Needed

- The agent recognizes that sending a $450,000 proposal requires explicit user approval
- This triggers the async authorization workflow
- The agent initiates an async authorization request to pause execution pending approval

Why async authorization is needed:

- High-value proposals exceed agent's autonomous authority
- Company policy requires human approval for proposals >$100K
- Ensures accountability for business-critical decisions
- Prevents unauthorized agent actions

Step 5.2: CIBA Authorization Request

The agent makes a CIBA (Client-Initiated Backchannel Authentication) authorization request:

Request includes:

- **User identifier:** Sarah's employee ID
- **Required permissions:** `email:send`, `drive:write`, `crm:update`
- **Context about the action:** Proposal details for Sarah to review
- **Callback endpoint:** Where the token should be delivered after approval

```
POST /bc-authorize
Host: acmecorp.authorization-server.com
scope=email:send drive:write crm:update
&login_hint=sarah.employee@acmecorp.com
&binding_message=Proposal Approval:
Acme Corp - $450,000
&client_notification_token=
8d67dc78-7faa-4d41-aabd-67707b374255
```

CIBA enables:

- Asynchronous approval workflows (agent doesn't block)
- Out-of-band user authentication (push notification to mobile)
- Rich context in approval requests (detailed proposal information)
- Secure callback mechanism (token delivered when approved)

Step 5.3: Push Notification

The authorization server sends a push notification to Sarah's Guardian mobile app:

The rich notification message displays:

```
Proposal Approval Required
Customer: Acme Corp
Amount: $450,000
Products: Enterprise AI Suite + Support
Recipients: cto@acmecorp.com, cfo@acmecorp.com
Action: Send proposal via email and save to Drive
[Approve] [Deny]
```

Rich Notification Features

- Detailed context about the action requiring approval
- Customer name, dollar amount, and products included
- Recipient list for transparency
- Clear action description
- Simple approve/deny interface

Note: This is Guardian's rich notification feature, NOT the OAuth Rich Authorization Requests (RAR - RFC 9396) specification. The notification provides detailed contextual information to help Sarah make an informed approval decision.

Step 5.4: User Approval

Sarah reviews and approves the request:

- Sarah reviews the details on her mobile device
- She verifies:
  - Customer is correct (Acme Corp)
  - Amount is accurate ($450,000)
  - Recipients are appropriate (CTO and CFO)
  - Products match customer needs
  - She approves the request by tapping the "Approve" button

Token generation and delivery

- The authorization server generates a scoped access token with approved permissions
- The token includes only the permissions Sarah authorized:
  `email:send` - permission to send the proposal email
  `drive:write` - permission to save proposal to Google Drive
  `crm:update` - permission to log the activity in Salesforce
  - The token is delivered to the agent via the CIBA callback endpoint
  - The agent receives the token and proceeds with execution using the newly issued token

Security benefits

- Explicit user approval required for sensitive actions
- Time-limited token (expires after action completes)
- Scope-limited token (only approved permissions)
- Full audit trail (who approved, when, what was approved)

Async Authorization Token Flow (CIBA) - Technical Details

The Client-Initiated Backchannel Authentication (CIBA) flow enables asynchronous user approval for AI agent actions.

When Sarah approves the request on her mobile device:

- **Approval Validation:** The authorization server validates the approval decision came from Sarah's authenticated device
- **Token Generation:** The authorization server generates a new access token scoped to the approved action
- **Permission Scoping:** The token includes only the permissions Sarah authorized (e.g., `email:send`, `drive:write`)
- **Secure Delivery:** The token is delivered to the agent via a secure callback endpoint specified in the original CIBA request
- **Agent Execution:** The agent receives the token and proceeds with the approved action

This ensures human-in-the-loop authorization for sensitive AI agent operations, with explicit user approval required before the agent can act.

CIBA Flow Benefits:

- **Non-blocking:** Agent doesn't need to maintain a connection while waiting for approval
- **User-friendly:** Sarah approves from her mobile device, not the chatbot
- **Secure:** Tokens are delivered via secure callback, not through the user's browser
- **Auditable:** Complete record of approval request, user decision, and token issuance
- **Flexible:** Supports various approval mechanisms (push notification, SMS, email)

Phase 6

Multi-System Execution (Token Vaulting)

Step 6.1: Save to Google Drive

- The agent retrieves a Google Workspace token from the Auth0 Token Vault
- The token is scoped to Sarah's Google Drive access
- The agent uploads the proposal to: `Sales/Proposals/2025/Acme-Corp-Q1.pdf`
- File permissions: Sarah's team members + her manager

Step 6.2: Send Email

- The agent retrieves a Gmail API token from Token Vault
- The agent composes an email from Sarah's account
  - To: Acme Corp CTO and CFO
  - Body: Professional proposal cover letter (generated by LLM)
  - Attachment: Proposal PDF from Google Drive
- The email is sent with Sarah's signature

Step 6.3: Schedule a Follow-Up

- The agent retrieves a Google Calendar token from Token Vault
- The agent checks Sarah's availability for the next 2 weeks
- The agent proposes meeting times to the Acme Corp contacts
- The agent adds a calendar event: "Acme Corp Proposal Review - 30 min"

Token Vault Benefits

- The agent never sees the actual OAuth tokens
- All tokens are automatically refreshed before expiration
- The credentials are never stored in agent code or logs
- Complete isolation between Auth0 credentials and Okta agent identity

Phase 7

Governance & Monitoring (Control Plane - Govern)

Step 7.1: Audit Trail

All activities are logged in the platform's System Log with comprehensive details:

- Agent authentication events
- Token exchange operations and scope grants
- Resource access attempts across all systems
- Authorization decisions (approved/denied)
- User delegation events
- API calls made on behalf of users
- All timestamps and contextual metadata for forensic analysis

Step 7.2: Quarterly Access Review

- Governance workflow triggered: Q1 2025 access certification
- Email to John Smith (agent owner): "Review access for sales-agent-prod-001"
- Access review shows the agent has:
  - Salesforce CRM access
  - Google Workspace access
  - Corporate pricing database access
  - Email system access
  - Calendar system access
- John confirms that all access is still required
- The certification is recorded in an Okta audit log

Step 7.3: Access Certification

- Quarterly access review shows agent has appropriate permissions for its role
- All access justified and approved by agent owner
- Certification recorded in audit log for compliance purposes

Phase 8

Threat Detection (Control Plane - Monitor)

Step 8.1: Anomaly Detected

- **Day 45:** Agent suddenly accesses 500 customer records in 10 minutes
- Behavioral analytics detects a deviation from the baseline
- The risk score escalates: NORMAL → HIGH
- Anomaly type: "Unusual data access volume"

Step 8.2: Automated Response

- The platform automatically blocks the agent's access to Salesforce
- **Global Token Revocation occurs -** all active tokens are immediately invalidated across all systems
- Security team receives a real-time alert
- Sarah (the user) receives notification: "Sales agent temporarily suspended"
- Complete access lockdown prevents further unauthorized activity

Step 8.3: Investigation and Remediation

- The security team reviews the System Log to understand the scope of the incident
- The root cause is identified and addressed

Key Metric

Attack detected and blocked through automated threat detection and response.

Integration Points: How the Unified Platform Components Connect

1. Authentication Flow

```
User ---> Universal Authentication ---> ID Token
---> Platform validates token ---> Platform
validates token ---> Combined user + agent context
---> Access granted with full context
```

2. Token Lifecycle

```
User-Delegated Tokens      Agent Identity Credentials
|                          |
|                          |
v                          v
Token Vault                Credential Vault
(Secure by Design)         (Control Plane)
|                          |
|- Salesforce token        |- Agent service account
|- Google Workspace token  |- MCP client credentials
|- Email API token         |- Bedrock API key
|- Calendar API token      |- Auth server credentials
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
                           |
                           v
Agent has both: User delegation + Own identity
```

3. Authorization Layers

```
Layer 1: Data Security (Fine-Grained Authorization)
Document-level permissions for RAG
"Can user Sarah see proposal-acme-2024?"

Layer 2: Token Vaulting
API-level permissions for SaaS tools
"Can user Sarah's token access Salesforce?"

Layer 3: Access Control (Control Plane)
System-level permissions for enterprise resources
"Can agent sales-agent-prod-001 access pricing database?"

Layer 4: Token Exchange (ID-JAG)
Cross-app access with user context
"Can Sarah (via agent) access on-prem pricing system?"

Result: Defense in depth with multiple
authorization checkpoints
```

4. MCP Authorization Pattern

```
1. Agent requests context via MCP Client
2. MCP Server receives request
3. MCP Server checks: Does agent have valid token?
Token validation ---> Auth0 Authorization Service
                      • Validates agent identity
                      • Checks token scopes
                      • Verifies permissions
4. MCP Server checks: Does user have permission to data?
Permission check ---> Fine-Grained Authorization
                      • Evaluates relationship tuples
                      • Returns authorized documents only
5. MCP Server returns authorized context to agent
```

Key Architectural Principles Demonstrated

1. Separation of Concerns

- Auth0 handles user authentication and user-delegated access
- Okta handles agent identity and lifecycle management
- MCP handles standardized context retrieval
- Each system does what it does best

2. Defense in Depth

- Multiple authorization layers prevent a single point of failure
- FGA filters documents, Token Vault gates APIs, Okta controls systems
- Even if one layer is bypassed, the others provide protection

3. Least Privilege

- Agents receive the minimum permissions needed for each task
- Tokens are scoped to specific APIs and actions
- Time-bound access with automatic expiration
- Just-in-time provisioning reduces standing privileges

4. User Context Preservation

- ID-JAG token exchange maintains user identity across trust boundaries
- Agent actions are always traceable to a specific user
- Authorization decisions consider user context, not just agent identity
- Audit trails show both "agent X" and "on behalf of user Y"

5. Continuous Monitoring

- Behavioral baselines detect anomalies
- Real-time threat response blocks attacks
- Comprehensive logging enables forensics
- Automated remediation reduces response time

Architecture Comparison: Traditional Approach vs. Unified Platform

Capability	Without Unified Platform	With Unified Platform
Agent Discovery	Manual spreadsheets, no shadow AI detection	Automated discovery, complete visibility
Credential Management	Hard-coded keys in code, never rotated	Vault storage and rotation
User Authentication	Custom auth code, password storage	Universal authentication, social SSO
API Access	Stored tokens in config files	Token vaulting with automatic refresh
Cross-Domain Access	Separate auth for each system	ID-JAG token exchange with user context
Human Approval	Custom polling, no mobile support	CIBA with mobile notification or email
Document Permissions	Application-level checks, inconsistent	Fine-grained authorization with relationship-based control
Access Reviews	Manual quarterly spreadsheets	Automated certification workflows
Threat Detection	Post-incident log analysis	Real-time behavioral analytics
Audit Trail	Scattered across multiple systems	Unified System Log
Incident Response	Manual investigation and remediation	Automated blocking and token revocation
MCP Authorization	Custom auth logic in each MCP server	Standardized OAuth2 with platform validation

This architecture demonstrates how a unified identity platform addresses AI agent security comprehensively, covering both secure development (authentication, token management, authorization, human oversight) and enterprise lifecycle management (discovery, registration, governance, threat detection), with MCP providing standardized context retrieval that respects identity and authorization controls throughout.

Conclusion: A Unified Platform for Complete AI Agent Security

The AI agent revolution is here. 91% of organizations are already using AI agents, and that number will only grow. But adoption has outpaced security and governance, creating significant risks that threaten to undermine the business value AI agents can deliver.

The challenge requires solving two interconnected problems simultaneously:

Secure every agent by design during development ensuring proper authentication, authorization, token management, and data access controls are embedded from the start.

Secure all agents from a single control plane across their lifecycle providing discovery, provisioning, governance, and threat detection for the entire agent population.

Organizations that address both dimensions will be better positioned to gain comprehensive security:

- **Secure development practices** with authentication, token vaulting, authorization, and human oversight
- **Complete visibility** into all AI agents, including shadow AI
- **Proper lifecycle management** from registration through deprovisioning
- **Comprehensive governance** with access reviews and certifications
- **Real-time threat detection** with behavioral analytics and automated response
- **Regulatory compliance** with complete audit trails and policy enforcement

Learn More

Build Secure AI Agents
Documentation and quickstarts for secure agent development

Learn More About Okta's Approach to Securing the AI Agent Lifecycle
Discover how Okta provides enterprise governance and control plane capabilities for managing AI agents at scale

Learn More About Cross-App Access
Explore how Cross-App Access enables seamless integration for ISVs and enterprise partners

This unified platform approach closes the critical gap identified in the AI at Work 2025 research: 85% of leaders say IAM is vital to AI adoption, yet only 10% have well-developed strategies for managing non-human identities.

Don't wait for a breach or compliance failure to implement proper AI agent security. Start today by securing agents during development and establishing centralized control over your agent population.

The unified platform that delivers both capabilities is available now through Okta's identity solutions. Organizations worldwide are already using this approach to deploy AI agents securely at scale combining Auth0 for GenAI for secure development with Okta's Identity Platform for enterprise lifecycle management.

About Okta
Okta, Inc. is The World's Identity Company™. We secure Identity, so everyone is free to safely use any technology. Our customer and workforce solutions empower businesses and developers to use the power of Identity to drive security, efficiencies, and success — all while protecting their users, employees, and partners. Learn why the world's leading brands trust Okta for authentication, authorization, and more at okta.com.

Whitepaper

Securing AI Agents From Development to Enterprise Scale

okta

The World's Identity Company™

Okta Inc.
100 First Street
San Francisco, CA 94105
info@okta.com
1-888-722-7871