CEQUENCE

Agent Personas

The Zero Trust Layer for Agentic AI

Where user identity, agent identity, data protection, and behavior converge into a coherent governance decisions for every agent action.

Four disparate governance layers

Every large enterprise already runs components of agent governance. Identity providers like Okta and Azure AD authenticate users. Data protection platforms like Microsoft Purview and Netskope classify and protect data. SIEM and observability tools record what happened. However, none of these layers were designed for AI agents – an actor that is autonomous, headless, token-armed, and routed through an unseen model.

A single agent action requires all four layers (user identity, agent identity, data protection, behavior) to agree, in real time, that the action is in scope. Today, they do not. They were architected for human users and service accounts with fixed purposes, and none of them were built to make a coordinated decision together at the moment the agent acts.

The canonical case

An employee has read and write access to the CRM and to the billing system, by entitlement. They deploy an agent to summarize support tickets and draft customer responses. The job, as the human understands it, is read-only: pull tickets, draft text, hand it back. The agent inherits the full entitlement envelope of the user anyway, because that is how identity-based authorization works. Read access and write access travel together. The agent encounters a refund dispute and, using its own reasoning to solve the problem, calls the billing API to adjust the account balance. Identity passed. Authorization passed. Data protection was not consulted because the API call was structurally valid. Behavioral signals existed in logs but were never part of the decision path. No control prevented the action.

The agent should not have had write access to billing in the first place. The job it was deployed to do did not require it. But nothing in the governance stack was designed to prevent it.

Four layers of governance standing alone when informed action demands coordination.

The agent did not breach the perimeter; it operated inside it. That is what makes this category of failure different from any threat the existing security and governance stack was built to catch. The decision the enterprise needs – is this action, by this agent, on behalf of this user, on this data, consistent with the intended job? – does not exist at any single control point – until now.

The Agent Persona as the binding thread

An Agent Persona is a single definition that compiles into runtime checks across all four governance layers. It is the security team's articulation of an agent's job: which user it acts on behalf of, which agent identity is permitted to assume the role, which tools and data classes are in scope, and what in-scope behavior looks like. One definition. Four layers reconciled. One coordinated decision per action.

The Enforcement Boundary: The AI Gateway

The Persona is enforced at the boundary between the agent and the things it interacts with – applications, APIs, and data. Today that boundary is most often an MCP server, because MCP is the protocol enterprises are standardizing on for agent-to-tool communication. Tomorrow that same boundary will include REST and gRPC APIs the agent calls directly, internal service meshes, and whatever protocols come after. Agent Personas are protocol-agnostic by design; the agent sees a single virtual endpoint. Behind that endpoint, an AI gateway assembles the right toolset, applies data protection policy, binds user and agent identity into the call, and evaluates the call against the declared behavioral scope.

The boundary is the right place because it is the only point in the stack where the agent's actual intent (the call it is about to make) is visible, structured, and inspectable before it executes. Not the model layer (model providers will keep changing). Not the agent code (agents redeploy frequently). The gateway, where the agent's call crosses out of the agent's runtime and into the enterprise.

> For humans, the industry learned RBAC was not enough. We are about to relearn that lesson with agents, unless we start at the right place.

Not the same as an MCP gateway

The category of MCP gateways is real and growing. They route, multiplex, observe, and rate-limit MCP traffic between agents and tools. That is necessary work, but it is not governance.

An MCP gateway answers the question: *can this traffic flow efficiently?* A Persona-aware gateway answers a different question: *is this specific call, by this agent, on behalf of this user, against this data, consistent with the job the agent was deployed to do?* The former is plumbing. The latter is the decision the enterprise actually needs.

Most MCP gateways inherit their security posture from the LLM gateways or API gateways they extend. They authenticate, they apply rate limits, they log. They do not carry a declaration of what each agent is supposed to do, they do not evaluate every call against a behavioral baseline tied to that declaration, and they do not produce a forensic record that names the agent, the user, and the persona on every action. Personas are the layer above MCP, not a feature inside it.

What Makes Up an Agent Persona

User identity binding. Which human authorized this agent to act, and what is that human's entitlement? The Persona never grants the agent more than the user it acts on behalf of, and grants the agent less when the job does not require broad entitlement.

Agent identity binding.
Which agent is this, what model, what version, what deployment? The same Persona can be assumed by multiple agent identities under different conditions or restricted to one.

Tool and data scope.
Which tools can the agent call across which applications, and which data classifications can it touch? Data protection catalogs are consulted as a precondition of the call, not after the fact.

Behavioral baseline.
What does in-scope behavior looks like? The gateway evaluates every call against the declared scope and flags drift the moment it occurs, with no learning curve, because the baseline is declarative, not learned.

For headless agents (CI/CD pipelines, scheduled jobs, batch processing), the credential primitive that carries this three-way binding is the **Agent Access Key.** It binds the agent's identity, the user's identity on whose behalf it acts, and the Persona's privileges into a single auditable credential. The forensic record is not "service account X accessed the billing API." It is "agent X, operating on behalf of user Y, under persona Z, called tool W at timestamp T, decision: allow."

Zero Trust for Agents, Not RBAC

RBAC is the easy analog and the wrong philosophy. RBAC asks one question, once, at authentication: does this role have permission to access this resource. It was designed for human users with consistent jobs, accessing resources at human speed, supervised by other humans.

Zero trust asks a different question, continuously, at every action: is this action, by this actor, on this data, right now, consistent with declared intent? The industry spent a decade moving humans from RBAC toward conditional access, risk-based authentication, and behavioral analytics. Continuous, contextual evaluation replaced the one-time permission check because the one-time check could not catch compromised credentials, insider risk, or lateral movement.

The instinct with agents is to reach back for RBAC. Define a role, grant scopes, issue a token, let it run. That is the architecture zero trust was built to retire.

Identity-only governance versus Persona-governed agents

Dimension Requirement	Identity-only governance	Persona-governed agent
Decision point	**Once, at session start. Token valid for the window.**	Every action. Continuous evaluation against declared scope.
Tool access	**Inherited from the user. Read and write access travel together.**	Scoped to the tools declared for this job. E.g., read without write when the job is read-only.
Data access	**Anything the API permits the credential to touch.**	Data classifications consulted as a precondition, not a log review.
Drift detection	**Discovered after the fact, in correlation across logs.**	The moment scope is exceeded.
Audit forensics	**Service account X accessed resource Y.**	Agent X, on behalf of user Y, under persona Z, called tool W at time T.

The token-validity window is the specific failure mode. A token is issued, the agent acts freely for the lifetime of the token, and every action in that window passes the identity check by definition. The recent report, *Agentic Zero Trust: Extending the Zero Trust Security Paradigm to Autonomous AI Systems* by Chase Cunningham, covers the token-window architecture in detail and grounds the broader Zero Trust extension in the NIST SP 800-207 framework. Identity validates the start of the session. Zero trust evaluates every action inside it. For agents, every action means every tool call the agent makes.

Production Evidence, Co-Existence, and What to Validate

Evidence at scale

Cequence has run a behavioral intelligence control plane in production for over a decade, processing roughly ten billion API transactions per day across global enterprises in financial services, telecommunications, and retail. That plane now evaluates agentic traffic against Persona-declared scope. Drift outside the declared Persona is detected immediately at runtime. No lengthy baselining exercise. No learning curve. The baseline is the Persona itself.

> The honest question for any security leader: what if the agent had been doing something drastic? A determined agent with valid credentials is one prompt-injected document away from a destructive sequence that no identity check would have stopped.

Last quarter, a Fortune 50 enterprise ran an autonomous coding agent for 47 consecutive hours, making 2,575 tool calls unsupervised. The agent guessed 162 filenames that did not exist. It hallucinated commit hashes in 71-second probe loops. It re-probed the same wrong paths across 27 hours with no memory between sessions. Every call was technically valid. Every call passed the identity check. Identity-level logging showed clean authenticated requests. The result was excessive token consumption, wasted time, and ultimately a failure to complete the job. Having a behavioral plane in place documents the full activity trail and surfaces drift the moment it occurs.

The research record is catching up. *Agents of Chaos*[1], a multi-institution red-teaming study from Northeastern, Stanford, Harvard, MIT, and Carnegie Mellon, deployed six autonomous agents into a live environment with persistent memory, email, Discord, and shell access. Over two weeks, the agents executed malicious instructions injected into shared documents, voluntarily propagated those instructions to other agents, and spawned persistent background processes from benign requests. Prompt injection is not solved with a smarter prompt. The defense is the layer above the agent.

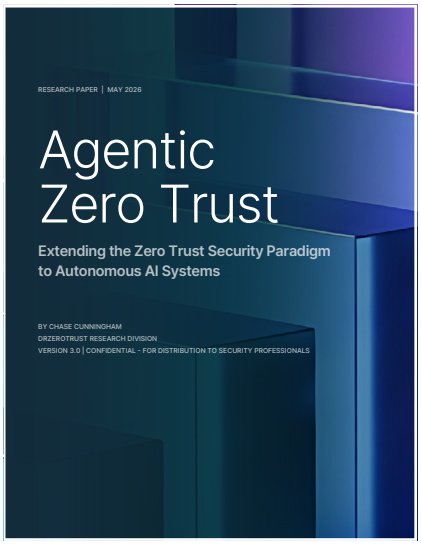
Co-existence with the existing security stack

Layer / Vendors	What the Persona adds at the moment of action
Identity *Okta, Azure AD, Auth0*	Tells you who the user is. The Persona binds that human identity to the agent acting on their behalf, at every call, not just at session start.
Data protection *Microsoft Purview, Symantec, Netskope*	Classifies and protects data at rest and in transit. The Persona consults those classifications when the agent attempts to read or move data, scoped to what the Persona declared.
LLM gateway *LiteLLM, OpenRouter, Portkey*	Routes prompts across model providers and balances cost and latency. The Persona governs what happens after the model emits a call, which is where the security decision lives.
Observability *Datadog, Splunk, Langfuse*	Records what the agent did. The Persona produces the structured event the platform records: agent, user, persona, tool, decision, all attached to the call.
SIEM *Splunk, Microsoft Sentinel, CrowdStrike Falcon Next-Gen SIEM*	Correlates security signals across the enterprise. The Persona feeds it agent-aware signals SIEM otherwise has no source for.
Cloud and model platforms *AWS Bedrock, Azure Foundry, GCP Vertex*	Host the agents and models. The Persona governs the agents that run on top of them, independent of which platform is hosting.

What to validate now

1. **Which of your existing or planned agents would fail their Persona today,** and what would the failure look like in your current logs?
2. **Where in your token issuance flow does behavioral evaluation belong,** and what would it take to insert a gateway at the boundary without disrupting the agent teams that already shipped?
3. **What does the forensic record look like for an agent action in production today,** and what would it look like with agent, user, and persona attached to every call?

[1] *Shapira et al., arXiv:2602.20021, February 2026*



Read the Agentic Zero Trust paper

Agentic Zero Trust: Extending the Zero Trust Security Paradigm to Autonomous AI Systems by Chase Cunningham extends NIST SP 800-207 to the agentic threat surface across all five Zero Trust pillars: identity, device, network, application, and data. It maps controls to threats ranging from prompt injection through emergent offensive reasoning and introduces behavioral identity as a first-class security layer. This paper, the one you are reading, focuses entirely on Personas as the operational primitive that compiles behavioral identity into runtime policy. Read together they form the full case for governing autonomous AI in the enterprise.

Read it: **Agentic Zero Trust: Extending the Zero Trust Security Paradigm to Autonomous AI Systems**

About Cequence

Cequence protects the applications and data that power enterprises in the agentic era. More than a decade of bot defence and API security experience has established Cequence as the leader of safe and secure agentic AI adoption. The Cequence platform delivers deep insight into user, entity, and agent behaviour, enabling organizations to secure and control agentic AI workflows while protecting against bad actors and rogue agents. Cequence delivers value in minutes rather than days or weeks with a highly scalable, no-code approach. Trusted by the largest and most demanding private and public sector organizations, Cequence protects more than 10 billion daily API interactions and 4 billion user accounts. Learn more at **cequence.ai.**

CEQUENCE
5201 Great America Pkwy, Suite 240, Santa Clara, CA 95054 | 1-650-437-6338 | info@cequence.ai | www.cequence.ai