RESEARCH PAPER | MAY 2026

Agentic Zero Trust

Extending the Zero Trust Security Paradigm to Autonomous AI Systems

BY CHASE CUNNINGHAM
DRZEROTRUST RESEARCH DIVISION
VERSION 3.0 | CONFIDENTIAL - FOR DISTRIBUTION TO SECURITY PROFESSIONALS

Abstract

The proliferation of autonomous AI agents across enterprise environments has introduced a class of computing entity that existing security architectures were never designed to govern. Frameworks built around users, devices, and deterministic workloads break down when the "principal" is a reasoning system that can spawn sub-agents, invoke tools across organizational boundaries, persist memory across sessions, and take real-world actions — all without a human in the loop. Zero Trust Architecture (ZTA), as codified in NIST SP 800-207, provides the most rigorous available foundation for enterprise security, but its current application was designed for a world of human users and static services. That world no longer describes the enterprise attack surface.

This paper argues that Zero Trust is necessary but insufficient for agentic AI, and proposes a coherent extension — Agentic Zero Trust — that maps the five canonical ZT pillars (Identity, Device, Network, Application/Workload, Data) onto the specific challenges posed by autonomous AI agents. We examine five major threat categories that define the agentic attack surface: prompt injection and indirect prompt injection (50–84% success rates in production systems, ranked OWASP LLM01); tool poisoning and supply chain attacks (including CVE-2025-6514, the OpenClaw/ClawHub crisis, and LangChain's CVE-2025-68664 at CVSS 9.3); privilege escalation and agent hijacking (demonstrated in the Vertex AI Double Agent incident and the McKinsey Lilli red-team exercise); data exfiltration via semantic privilege escalation, including the Slack AI RAG attack and a documented 45,000-record financial services breach; and — introduced in this revision — emergent offensive reasoning, wherein frontier AI models autonomously generate offensive cyber capabilities as a byproduct of goal-directed reasoning, independent of attacker manipulation (evidenced by Claude Mythos Preview scoring 83.1% on UC Berkeley's CyberGym benchmark and completing AISI's 32-step corporate network attack simulation end-to-end).

Against each threat category, we map concrete Agentic Zero Trust controls: SPIFFE/SPIRE workload identity and OAuth 2.0 Token Exchange (RFC 8693) for agent authentication and delegation chains; the Agent Persona framework and Just-in-Time tool authorization with the MiniScope least-privilege model; the Token Isolation Pattern for structural credential non-portability; microsegmentation, mutual TLS, and agent gateways for network enforcement; runtime behavioral guardrails and the CSA MAESTRO threat model for application-layer enforcement; and ABAC-based data governance, PHI sanitization, and immutable audit logs for data protection. This revision elevates behavioral identity — continuous intent verification against the declared agent persona — as a third required layer alongside cryptographic identity and authorization, and the only control capable of detecting the emergent offensive reasoning threat. We introduce the Cloud Security Alliance Agentic Trust Framework (ATF) maturity model — Intern, Junior, Senior, Principal — as a practical governance sequencing tool. Sector-specific applications in cybersecurity operations, healthcare, financial services, government, and DevSecOps are examined with reference to real deployments and the regulatory frameworks that govern them. The paper concludes with ten actionable recommendations for security architects and CISOs implementing Agentic Zero Trust today.

1. Introduction

Autonomous AI agents are no longer a research concept. They are deployed in production across enterprises of every scale. LangChain and LangGraph orchestrate tens of millions of agent tasks daily. AutoGen enables multi-agent collaborative pipelines. CrewAI lets domain teams compose task-specific agent crews with minimal engineering overhead. OpenAI Agents SDK provides a first-party framework for building agents with tool access and handoff logic. Anthropic Claude agents operate as research, coding, and data analysis assistants with persistent memory and tool-calling capabilities. Microsoft Copilot is now an agentic SOC platform embedded in Defender, Entra, Intune, and Purview. Salesforce Agentforce deploys autonomous agents across CRM workflows with direct write access to customer data. The era of agentic AI is not arriving — it has arrived.

What has not arrived is a security architecture commensurate with the threat surface these systems create. The fundamental problem is categorical: AI agents are not users, not devices, and not services. They are a new class of computing entity with properties that confound every assumption baked into existing security frameworks. Traditional IAM was designed for human users who authenticate once per session. Agents authenticate at machine speed, thousands of times per minute, across dozens of services simultaneously. Network security was built around perimeter models and, later, microsegmented workloads — but those workloads execute deterministic code. Agents execute probabilistic reasoning that can be redirected mid-session through natural language. Endpoint protection assumes a device with a known software stack; agent "devices" include model weights, tool schemas, and MCP server definitions that can be poisoned through semantic manipulation invisible to conventional scanners.

Zero Trust Architecture provides the most rigorous and widely adopted security philosophy available. Its core mandate — "never trust, always verify" — is precisely the disposition enterprises need toward AI agents. An agent claiming to be an authorized orchestrator for a sensitive pipeline should receive the same continuous scrutiny as an anonymous endpoint attempting to access a protected API. The seven tenets of NIST SP 800-207, from treating all resources as subjects of policy enforcement to enforcing dynamic, per-session least-privilege access, translate naturally into the agentic context. The question is not whether Zero Trust is relevant to agentic AI; it is how to extend it to address capabilities that did not exist when the standards were written.

This paper makes four contributions. First, it provides a rigorous mapping of the agentic threat surface against the five ZT pillars, demonstrating where existing controls are insufficient and why. Second, it presents concrete, implementable Agentic Zero Trust controls for each pillar, drawing on the emerging industry consensus across NIST, OWASP, CIS, CSA, Microsoft, Google, Palo Alto Networks, CrowdStrike, and HashiCorp. Third, it applies these controls to five high-stakes enterprise sectors: SOC operations, healthcare, financial services, government/defense, and DevSecOps. Fourth, it presents the CSA Agentic Trust Framework maturity model as a governance roadmap for organizations at any stage of agentic AI adoption. The goal is a framework that is not merely theoretically sound but immediately actionable for the security architects and CISOs who must make deployment decisions now.

Version 3.0 of this paper adds a fifth threat category — Emergent Offensive Reasoning — grounded in published benchmark data from Anthropic and the UK AI Security Institute demonstrating that frontier model generations have crossed a threshold of autonomous offensive cyber capability that no prior model generation possessed. It also introduces the Token Isolation Pattern as an architectural control for structural credential non-portability, the Agent Persona framework as the upstream driver of least-privilege tool grant decisions, and Behavioral Identity as a first-class security layer. The evidence

base compels treating these additions not as theoretical extensions but as operational necessities.

2. Zero Trust Architecture: Foundational Principles

Zero Trust Architecture is a security model, not a product. NIST SP 800-207, published in August 2020, is its canonical definition: a set of principles and a logical architecture for protecting enterprise resources by eliminating implicit trust from any location, user, or device. The foundational premise is that breach is inevitable; the security objective is to limit blast radius through continuous verification and minimal access.

NIST SP 800-207 defines seven tenets [1]. All data sources and computing services are treated as resources regardless of network location. All communication is secured regardless of network provenance. Access to individual resources is granted on a per-session basis. Access decisions are informed by dynamic policy incorporating identity, context, and behavioral signals. The organization monitors and validates the integrity and security posture of all assets continuously. Authentication and authorization are strictly enforced before every access grant. The organization collects telemetry to improve security posture over time.

These tenets are implemented through a three-component architecture. The Policy Engine (PE) evaluates access requests against enterprise policy, producing approve/deny decisions. The Policy Administrator (PA) establishes and terminates communication paths between subjects and resources, managing session credentials. The Policy Enforcement Point (PEP) mediates all access requests, sitting between subjects and the resources they seek to reach.

The five ZT pillars defined by the CISA Zero Trust Maturity Model provide the operational structure. The Identity pillar governs who — or what — is requesting access, using MFA, SSO, RBAC, and continuous session monitoring. The Device pillar governs the trustworthiness of the endpoint presenting the request. The Network pillar governs traffic flows, implementing microsegmentation and Zero Trust Network Access (ZTNA) to eliminate implicit internal-network trust. The Application/Workload pillar governs access to specific applications and services, with API gateways, WAFs, and secure SDLC practices. The Data pillar governs the sensitivity classification and access control of the data itself, applying DLP, encryption, and data-centric access policy.

This architecture is demonstrably effective for its intended scope. The problem is scope. NIST SP 800-207 was designed for an enterprise in which the principals are humans, the workloads are deterministic services, and access decisions can be governed by static or slowly-evolving policy. It addresses the question "should this user, on this device, be allowed to access this resource?" The agentic question is fundamentally different: "should this autonomous reasoning system, which may spawn sub-agents, invoke dozens of tools, ingest untrusted external data, and make decisions that affect real-world systems, be allowed to proceed — and with what constraints?" Current ZTA frameworks have no answer.

The transition is not incremental. When the "user" is an autonomous reasoning system, every assumption about the identity, intent, and scope of a principal must be reconsidered from the ground up.

3. Defining the Agentic Threat Surface

An AI agent is an autonomous software system comprising four core components: a reasoning loop (an LLM performing iterative chain-of-thought or planning logic), tools (APIs, databases, shell executors, code interpreters, or other external interfaces the agent can invoke), memory (short-term context windows, long-term vector stores, and episodic memory that persists between sessions), and orchestration logic (the framework governing task decomposition, subtask delegation, and multi-agent coordination). What distinguishes an agent from a conventional API or microservice is that the agent reasons and decides — it is not executing a fixed procedure but pursuing a goal through improvised action selection.

Multi-agent systems (MAS) extend this architecture to networked ensembles of agents. An orchestrator agent decomposes a high-level task and delegates subtasks to specialized sub-agents: a data retrieval agent, an analysis agent, a communication agent, a code execution agent. Each sub-agent may itself spawn further sub-agents. Authority — and with it, attack blast radius — flows down the delegation chain.

The agentic attack surface has six distinct dimensions that do not exist in conventional workloads. Identity ambiguity: agents are ephemeral and non-deterministic; the same agent configuration may produce different behaviors in different contexts, making consistent identity attestation difficult. Tool access: an agent's capability is defined by its tool set; a single compromised tool definition can redirect all tool-calling behavior. Memory stores: long-term memory is simultaneously a feature (context persistence) and a vulnerability (persistent poisoning surface). Inter-agent trust: in MAS, agents routinely accept instructions from other agents without cryptographic verification of that agent's authority to issue instructions. External data ingestion: agents retrieve and process untrusted content from the internet, email, documents, and APIs as a core function — every such source is a potential injection vector. Model behavior: unlike deterministic code, LLMs can be manipulated through natural language, making semantic attacks invisible to conventional security tooling.

This last dimension deserves emphasis. Traditional workloads can be secured by controlling inputs and enforcing policy on outputs. Agents reason, improvise, and make decisions in the middle of the stack — in the reasoning loop — where no conventional security control operates. A SQL injection attack targets a parsing function; a prompt injection attack targets the model's semantic interpretation of its context. These require fundamentally different defenses.

The confused deputy problem, described by Norm Hardy in 1988 [2], applies to agentic AI with dramatically amplified consequences. In classic computing, a confused deputy is a privileged component manipulated by a less-privileged entity into abusing its authority. In multi-agent AI, the deputy is an LLM that can be convinced — through natural language embedded in data it retrieves — to act against its principal's interests. Unlike a fixed-function service, the "confused" LLM deputy has no stable policy it maintains. Its behavior is a function of its context, and that context can be poisoned.

The MITRE ATLAS framework (v5.4.0, 2025) provides the adversarial taxonomy for these attacks [3]. Key techniques include AML.T0051 (LLM Prompt Injection), AML.T0054 (LLM Jailbreak), and 14 new agent-specific entries covering multi-agent orchestration exploitation, memory poisoning, and tool supply chain attacks across 16 tactics and 84 total techniques.

4. The Agentic Threat Landscape

4.1 Prompt Injection and Indirect Prompt Injection

Prompt injection is ranked #1 on the OWASP Top 10 for LLM Applications 2025 (LLM01) and appears in over 73% of assessed production AI systems [4]. In direct prompt injection, an attacker types malicious instructions into a user-accessible input, overriding system prompts or safety constraints. The damage is bounded by the agent's tool access; if the agent can execute code, send email, or modify databases, a direct injection can cascade from text generation into real-world action.

Indirect prompt injection (IPI) is categorically more dangerous because the attacker never touches the prompt interface. Malicious instructions are embedded in external data sources — webpages, PDFs, emails, calendar invites, MCP tool descriptions, RAG knowledge bases, code repositories — that the agent retrieves as part of normal operation. The agent processes the poisoned content as trusted data; the hidden instructions execute with the agent's full authority. Attack success rates reach 84% in agentic systems with auto-execution enabled [5].

The UK's National Cyber Security Centre has warned that indirect prompt injection "may never be fully fixed" due to the architectural impossibility of reliably separating instructions from data within a single LLM context window. This is not a bug to be patched; it is a property of how LLMs process information.

Real incidents illustrate the severity. EchoLeak (CVE-2025-32711, CVSS 9.3) affected Microsoft 365 Copilot: an attacker's email contained a hidden prompt payload that, when Copilot's RAG retrieved it during any subsequent query, leaked emails, SharePoint files, and Teams chats without any user interaction, bypassing XPIA classifiers, link redaction, and Content Security Policy simultaneously [6]. CVE-2025-53773 (CVSS 9.6) in GitHub Copilot enabled remote code execution via prompt injection, demonstrating that developer tools with code execution access represent an extreme-severity IPI surface [7]. Palo Alto Networks Unit 42 published the first confirmed in-the-wild weaponization of IPI in March 2026, documenting attackers using IPI for content moderation bypass, unauthorized transactions, SEO poisoning, system prompt leakage, data destruction, and denial of service — with some malicious pages deploying 24 distinct injection techniques simultaneously [8].

ZombieAgent represents the logical endpoint of IPI exploitation: an attack on ChatGPT's long-term memory integration achieved zero-click IPI that persisted across sessions. The agent continued acting on attacker instructions in future conversations without any reinsertion of the payload — the memory store itself became the persistence mechanism [9].

Agentic Zero Trust countermeasures for prompt injection operate at three layers. At the input layer: structured input validation, content sanitization for all external data before ingestion, and prompt shield technology (Azure AI Foundry Prompt Shields) to detect injection attempts before they reach the reasoning loop. At the policy layer: deterministic Policy Enforcement Points (implemented in OPA/Cedar, not in the LLM itself) that evaluate each proposed tool call against declared policy, rejecting actions outside the agent's authorized scope regardless of what instructions are in the context window. At the audit layer: immutable logs of every tool invocation with the full context that triggered it, enabling forensic reconstruction of injection attack chains. These controls do not solve IPI architecturally — NCSC is correct that it cannot be fully fixed — but they limit blast radius to what the agent is policy-authorized to do.

4.2 Tool Poisoning and Supply Chain Attacks

Tool poisoning occurs when malicious content is injected into the descriptions, metadata, schemas, or implementations of tools AI agents use via frameworks like the Model Context Protocol (MCP). Unlike traditional code vulnerabilities, tool poisoning operates entirely in the reasoning layer — it exploits the relationship between tool descriptions and how the LLM interprets them, not the tool's underlying code. Static code analysis, antivirus scanners, and conventional security tooling are blind to this attack class.

CrowdStrike has documented five distinct tool poisoning variants [10]. Hidden instruction poisoning embeds attacker instructions in tool metadata invisible to human users but processed by the model — a tool named `add_numbers` whose description instructs the agent to read `~/.ssh/id_rsa` and pass its contents as a side parameter. The math completes normally; the credential exfiltration is invisible. Tool shadowing places malicious instructions in one tool's description that influence behavior when the agent uses a completely different, legitimate tool — poisoned `calculate_metrics` metadata that trains the agent to include an attacker's address in BCC on all future email operations, even though the malicious tool never directly invoked the email function. Malicious example poisoning includes attacker-controlled endpoint URLs in tool usage examples. Permissive schema injection declares a restrictive schema while the actual schema includes undeclared parameters (such as `admin: true`) that elevate privileges in downstream systems. Tool Invocation Prompt (TIP) hijacking manipulates tool descriptions and environmental state to coerce agent behavior across three phases — prompt stealing, vulnerability analysis, and hijacking — enabling full remote shell access even when alignment or guard models are present.

The supply chain dimension of tool poisoning reached crisis scale with the OpenClaw/ClawHub incident in February 2026. Antiy CERT confirmed 1,184 malicious skills across the ClawHub registry — approximately 1 in 5 packages at peak. SecurityScorecard found 135,000 OpenClaw instances exposed to the public internet with insecure defaults. Nine CVEs were disclosed, three with public exploit code. The attack vector was typosquatting and automated mass uploads; a compromised dependency in an agent framework runs with the agent's full permissions — terminal access, file system, stored credentials for cloud services [11].

Separately, Trend Micro identified 492 MCP servers exposed to the internet with zero authentication, any of which represents an immediately exploitable tool poisoning and data exfiltration surface [11]. CVE-2025-6514 (CVSS 9.6) in `mcp-remote` enables arbitrary OS command execution when MCP clients connect to untrusted servers. CVE-2025-68664 ("LangGrinch", CVSS 9.3) in LangChain Core exploited improper handling of internal metadata fields during deserialization — a classic injection flaw where untrusted input influenced execution flow — enabling extraction of environment variables and instantiation of unintended classes [12].

Agentic Zero Trust countermeasures center on supply chain integrity and tool-level authorization. MCP servers must be allowlisted and cryptographically verified — agents should not be permitted to invoke arbitrary tools discovered at runtime. Tool definitions must be signed and version-pinned; "rug pull" attacks that modify tool definitions between sessions are prevented through cryptographic integrity checks. Static API keys for MCP must be replaced with per-agent OAuth 2.1 credentials with PKCE. At the framework level, MCP servers should be isolated in separate network segments with agent gateways (modeled on Palo Alto's Prisma AIRS 3.0 AI Agent Gateway) acting as Policy Enforcement Points for all agent-to-tool traffic.

4.3 Privilege Escalation and Agent Hijacking

In multi-agent architectures, the foundational security principle is that authority should only shrink, never expand, as it flows down the delegation chain. Current frameworks largely do not enforce this at the infrastructure level; privilege boundaries are maintained by prompt-level restrictions that are trivially bypassed by IPI. The consequence is that privilege escalation in MAS is not an edge case — it is an architectural property of systems built without cryptographic delegation.

The Irregular multi-agent research published in 2026 is the most sobering documented illustration [13]. In three independent scenarios, frontier multi-agent systems conducting routine enterprise tasks autonomously discovered and exploited vulnerabilities with no adversarial prompting. A lead agent delegating document research had a sub-agent encounter an access-denied error — and rather than reporting the failure, the agents devised an exploit chain: they found a hardcoded Flask secret key, decoded their own session cookie, forged an admin session cookie, and retrieved the restricted document. A separate scenario had a backup agent find an administrator password embedded in a utility file and use it to disable Windows Defender. A third had agents drafting social media posts autonomously develop a steganographic encoding scheme to hide credentials in whitespace after a DLP system blocked the direct exfiltration attempt. None of these behaviors were instructed. They emerged from goal-oriented reasoning.

Vertex AI Double Agent (April 2026) demonstrated that malicious or compromised agents in Google Cloud Vertex AI could abuse default permission scoping to exfiltrate data, access service-agent credentials, and reach protected internal resources [14]. Classified as OWASP ASI03 (Identity and Privilege Abuse), this incident confirmed that even hyperscaler-managed agentic environments carry structural privilege escalation risk.

The McKinsey Lilli red-team exercise showed an autonomous agent gaining broad system access in under two hours — a timeline that exceeds human response capacity and renders post-incident review insufficient as a sole control [15].

The LangGraph confused deputy scenario illustrates the multi-agent escalation chain precisely: a low-privilege Intake Agent (with only `fs.read` permission) parsing resumes encounters a malicious resume containing a hidden injection. The hijacked Intake Agent then instructs an HR Admin Agent (with `http.fetch` permission) to POST salary data to an external attacker endpoint. The HR Admin Agent complies because the instruction originates from a "trusted" internal agent. The escalation succeeds without any access control violation at the network layer — because most frameworks do not verify that an agent requesting delegation actually possesses the permissions it is attempting to delegate [16].

Agentic Zero Trust countermeasures require moving privilege enforcement from the prompt layer to the infrastructure layer. OAuth 2.0 Token Exchange (RFC 8693) enforces cryptographic scope attenuation at every delegation hop: a sub-agent receives a freshly issued token scoped only to its specific task, with a shorter lifetime than its parent's token, and with the original human subject preserved in the `sub` claim. No agent can delegate permissions it does not itself possess. Per-instance agent identity (going beyond the SPIFFE/SPIRE pod-level identity to true instance-level attestation) provides forensic traceability for every action in a multi-agent chain.

4.4 Data Exfiltration and Lateral Movement

Semantic privilege escalation represents a category of attack that does not violate any access control at the technical layer but violates every assumption about authorized scope. An agent authorized to read files and send email has the technical permission to scan a directory for API keys and email them to an external address. If IPI directs it to do so, every individual action is technically authorized. No firewall rule fires; no DLP policy triggers; no anomaly detection fires on the action itself. The exfiltration is invisible because it is indistinguishable from legitimate operations.

The Slack AI RAG exfiltration (August 2024) demonstrated the attack at production scale [17]. Researchers showed that IPI placed in a public Slack channel could cause the Slack AI assistant to access data from private channels the attacker did not belong to, embed the stolen data (including API keys) in an encoded malicious link, and exfiltrate it upon victim click. Slack's initial characterization of this as "intended behavior" illustrates the governance gap: RAG systems were designed for data access, not data isolation. After August 14, 2024, Slack expanded AI ingestion to include Google Drive and PDF uploads, dramatically widening the surface — an attacker no longer needed to be in Slack at all.

A financial services incident documented by Stellar Cyber captures the operational impact: an attacker manipulated a reconciliation agent into exporting "all customer records matching pattern X," where X was a regex matching every record. The agent complied because the request was semantically reasonable as a business task. The result was 45,000 customer records exfiltrated [18].

Loss of data lineage in multi-agent chains creates compounding risk. An agent retrieves a document, summarizes it, drafts an email, and archives the source — four actions, each logged individually, but the full data movement pathway invisible as a unified event chain. Security teams face attribution blind spots that prevent accurate damage assessment and forensic reconstruction.

The ZombieAgent attack (discussed in Section 4.1) converges data and memory pillars: once memory is poisoned, every future retrieval and action by that agent is potentially compromised data, making memory stores a persistence and exfiltration mechanism simultaneously.

Agentic Zero Trust countermeasures operate across three layers. Attribute-Based Access Control (ABAC) at the retrieval layer ensures agents retrieve only data they are specifically authorized for given their current task context — not the broader access implied by their standing permissions. PHI and PII detection before data enters agent context windows (Microsoft Purview SDK, AWS Bedrock Guardrails, BERT-based NER in healthcare pipelines) prevents sensitive data from entering the reasoning loop where it can be misused. Immutable, agent-specific audit logs that record not just individual actions but the full data provenance chain — what was retrieved, when, from what source, by which agent instance, in service of what declared task — provide the lineage tracking necessary for breach detection and regulatory compliance.

4.5 Emergent Offensive Reasoning: The Agent-as-Adversary

The four preceding threat categories share a common structure: each describes an attack against an agent — a malicious actor or malicious content that manipulates an otherwise compliant system into acting outside its sanctioned envelope. Section 4.5 introduces a categorically different threat in which the agent itself — acting on a legitimate task, with valid identity, within its authorization envelope — autonomously generates offensive cyber behavior as an emergent property of goal-directed reasoning. No injection is required. No attacker is present in the loop. The agent is simply doing its job and deciding, through its own reasoning, that offensive techniques are the most efficient path to task completion.

The distinction matters architecturally. Prompt injection is a threat against the agent — the defense is input sanitization, context isolation, and PEP enforcement. Emergent offensive reasoning is a threat from the agent — identity controls confirm who the agent is, authorization controls confirm what it is allowed to do, and both return valid results. The agent is exactly who it claims to be, acting within its declared authorization envelope, producing behavior that is operationally indistinguishable from a sophisticated attacker until behavioral telemetry is examined in aggregate. This threat is detectable only through behavioral signals.

The benchmark evidence establishes that this is not theoretical. Anthropic's Claude Mythos Preview evaluation, published April 7, 2026, scored 83.1% on UC Berkeley's CyberGym benchmark and 73% on the UK AI Security Institute's expert-level Capture the Flag suite — tasks no model could complete before April 2025 [41]. On a Firefox 147 exploit benchmark, Mythos produced 181 working exploits compared to 2 for Claude Opus 4.6 — a roughly 90x improvement in autonomous offensive capability within a single model generation [41]. The UK AI Security Institute's independent evaluation confirmed that Mythos was the first model to complete AISI's 32-step corporate network attack simulation end-to-end, autonomously chaining reconnaissance through lateral movement to full network takeover [42]. These are not marginal capability improvements. They represent a qualitative phase transition in what a model deployed in an enterprise agentic workflow can do when it decides that offensive techniques serve its task.

Nation-state operators have already operationalized this property. In November 2025, Anthropic disrupted GTG-1002, a Chinese state-sponsored campaign in which AI executed 80–90% of tactical work autonomously across approximately 30 simultaneous targets spanning financial institutions and technology companies [43]. Human operators intervened at only 4–6 decision points per campaign. The AI was not being controlled step-by-step — it was reasoning autonomously toward campaign objectives and selecting offensive techniques without instruction at each step. Anthropic detected the campaign; the targeted organizations did not.

The Vertex AI Double Agent incident (April 2026) and the Irregular Research MegaCorp scenario — in which agents reverse-engineered authentication, forged session cookies, found credentials in utility files, and encoded secrets in steganographic whitespace to evade DLP — are illustrations of the same pattern operating in legitimate enterprise deployments [13, 14]. None of these behaviors were adversarially injected. They emerged from goal-pursuit reasoning encountering an obstacle. The insight is direct and important: you have put the world's best reasoning system to work on your enterprise network, and it will reason its way through barriers using whatever techniques it can discover — including offensive ones — if those techniques serve its declared goal.

This threat class requires a reframing of the security model. The conventional security assumption is that a principal with valid identity and valid authorization is a trusted actor. For human principals, this assumption is imperfect but defensible: behavioral anomalies are the primary additional signal, and human attackers operate on timescales that allow detection. For agentic principals backed by frontier models, the assumption fails. A model capable of 181 working exploits in a benchmark context is capable of applying that same reasoning to enterprise tool calls when it encounters an access denied error. The agent is not being manipulated; it is being its designed self. Authorization boundaries become, at best, the minimum viable definition of what the agent is not permitted to do — not a description of what it will do.

The correct framing: identity and authorization are necessary but insufficient controls for agents backed by frontier reasoning systems. Behavioral monitoring is the third required layer, and for this threat category it is the primary detection layer, because identity and authorization controls remain valid throughout an emergent offensive reasoning incident.

Agentic Zero Trust countermeasures for emergent offensive reasoning operate on four principles:

Behavioral monitoring as primary detection. Tool-call patterns, data access volumes, decision sequences at barriers (what does the agent do when it receives an access-denied response?), and deviation from declared job-description behavior are the signal set. An agent that receives an access-denied error and begins systematically attempting alternative credential vectors, path traversal variants, or encoding schemes to move data past a DLP boundary is exhibiting a behavioral pattern that is detectable in aggregate, even when each individual action is within authorization. The behavioral identity framework (see Section 5.6) is the primary defense.

Hard task-scope enforcement at the Policy Enforcement Point. The agent cannot invoke tools outside its declared task scope, regardless of what its reasoning concludes about the utility of those tools. A content-analysis agent has no legitimate business purpose invoking credential-store APIs, network scanning tools, or outbound encoding utilities. The PEP enforces declared scope at the tool-call layer, not in the model's reasoning. This does not prevent the model from reasoning about offensive techniques — it prevents those conclusions from being actuated.

Human-in-the-loop gates for sensitive action categories. Actions in defined high-risk categories — first use of a novel tool combination, actions affecting authentication infrastructure, large-volume data movements, outbound communications containing encoded data — require human approval regardless of the agent's stated justification. The gate is behavioral, not role-based: it fires on action patterns, not on agent identity.

Immutable audit trails with aggregate behavioral analysis. Individual log entries are insufficient. The audit trail must support reconstruction of the full behavioral chain — what did the agent do when it encountered each barrier, and does the aggregate sequence constitute an attack pattern? This requires correlation across tool calls in temporal proximity, not just individual event logging.

Rate limiting on novel tool-call sequences. An agent invoking an unusual combination of tools — or invoking familiar tools with unusual parameter distributions — triggers rate limiting that creates detection windows. The purpose is not to block the behavior immediately but to create time for human review before the behavior can complete a meaningful attack chain.

The emergent offensive reasoning threat is the most difficult to accept because it does not fit the adversarial framing that underlies most security thinking. There is no attacker to attribute, no injected payload to detect, no policy violation to block. There is only a very capable reasoning system doing what it was designed to do, in ways that the enterprise did not intend and cannot prevent through identity or authorization controls alone.

5. Applying Zero Trust Pillars to Agentic Systems

The following sections map each ZT pillar to the agentic challenge and the corresponding Agentic Zero Trust control. This is the core technical contribution of the paper.

ZT Pillar	Traditional Control	Agentic Challenge	Agentic ZT Control
Identity	MFA, SSO, RBAC for humans	Agents are ephemeral, can clone/impersonate, have no password	SPIFFE/SVID attestation, OAuth Token Exchange, DIDs/VCs, agent registries, behavioral identity
Device	EDR, patch compliance, device trust	Agent "device" includes model weights, tool schemas, MCP definitions	Container attestation, supply chain integrity, workload isolation
Network	Microsegmentation, ZTNA	Agent-to-agent traffic, MCP trust boundaries, token forwarding	mTLS everywhere, agent gateways as PEP, Token Isolation Pattern, token exchange at each hop
App/Workload	API gateway, WAF, secure SDLC	LLMs are non-deterministic; IPI overrides system prompts; emergent offensive reasoning	Policy-as-code (OPA/Cedar), runtime guardrails, behavioral monitoring, behavioral identity
Data	DLP, encryption, classification	Agents accumulate, synthesize, and act on data through memory and RAG	ABAC at retrieval layer, PHI sanitization, data lineage, memory isolation

5.1 Identity: Verifying the Unverifiable

Agent identity presents three challenges that classical IAM cannot address. Agents are ephemeral — they spawn and terminate at timescales that make manual provisioning impractical. They can clone — a single agent configuration deployed at scale produces thousands of instances sharing an identity that must nonetheless be distinguishable at the instance level for forensic purposes. And they can impersonate — an agent can claim any identity in natural language, making cryptographic verification essential.

SPIFFE/SPIRE (Secure Production Identity Framework For Everyone) is the leading open standard for machine/workload identity and maps directly onto agentic systems [19]. Each agent process is assigned a SPIFFE ID (URI format: `spiffe:///`) encoded in a SVID (SPIFFE Verifiable Identity Document), issued as a short-lived X.509 certificate or JWT. SVIDs rotate automatically — typically every hour — and workloads never experience credential expiry. SPIRE provides two-level attestation without pre-shared secrets: node attestation (is this a legitimate compute node, verified through AWS EC2 instance identity documents, GCP metadata tokens, or TPM-backed attestation?) and workload attestation (is this the specific agent it claims to be, verified through Kubernetes pod labels, container image digest, or Linux UID?). CyberArk has deployed SPIFFE in production for AI agent workloads; Microsoft's Entra Agent ID extends the same principle to the Microsoft ecosystem, automatically assigning cryptographic identities to agents created in Copilot Studio and Azure AI Foundry.

For delegation chains in multi-agent systems, OAuth 2.0 Token Exchange (RFC 8693) provides the mechanism for secure authority propagation. Each delegation hop receives a new token with a shorter lifetime, narrower scope, and the preserved original subject (`sub` claim = human principal, `act` claim = delegating agent). A sub-agent cannot request scoped credentials that exceed its parent's permissions — the on-behalf-of chain is cryptographically enforced, not prompt-enforced.

For cross-organizational and cross-framework agent trust, Decentralized Identifiers (DIDs) and Verifiable Credentials (VCs) provide cryptographically verifiable agent identity that does not depend on a central authority [20]. Agent credentials encoded as VCs can include capabilities, provenance (who built it, what version, what framework), behavioral scope, and security posture. The Agent Naming Service (ANS) — a discovery mechanism allowing agents to find and query other agents by capability with trust scores attached — prevents agent impersonation in federated multi-agent systems.

The "identity paradox" for reasoning agents is real: cryptographic identity establishes that this is a specific agent process, but it says nothing about whether the agent's behavior is consistent with its declared purpose. A 5-minute SPIFFE token is not a free pass for those 5 minutes — an agent can operate entirely within its authorization envelope while conducting exfiltration, with every individual action permitted and the aggregate pattern constituting the attack. Behavioral identity — the continuous verification of whether an agent is acting within its intent envelope, not just its authorization envelope — is the third required layer alongside cryptographic identity and authorization. It is treated as a full first-class security layer in this framework and documented in Section 5.6.

5.2 Least Privilege: Minimal Tool Access and Scoped Permissions

Traditional Principle of Least Privilege (PoLP) defines minimum permissions at provisioning time. For agents, this is insufficient because access needs evolve dynamically during task execution, agents may dynamically discover tools at runtime, and multi-step tasks require permissions that accumulate across steps. The right model is not minimal standing permissions — it is no standing permissions, with JIT authorization for each action.

Agent Persona as the Upstream Driver of Tool Grant Decisions

OAuth scopes, JIT grants, and MiniScope all address how permissions are enforced. What they do not address is how the scope is decided in the first place. The Agent Persona concept provides this upstream anchor. An agent persona is the intersection of (a) the user the agent acts on behalf of, and (b) a defined job description. The job description determines tool selection deterministically, making least privilege the default state rather than a hardening step applied after initial deployment.

Three properties govern agent personas. First, intersection, never expansion: a marketing-assistant persona acting on behalf of a finance VP still gets only email-send access. The persona scope and the user's scope compose — the agent receives the minimum of both, never the union. A junior analyst's agent cannot invoke capabilities the analyst herself lacks, and a senior analyst's agent cannot invoke capabilities outside the persona's declared job description. Second, tool recommendations are automatic from the description: the job description IS the policy. A "contract review agent" that reads and annotates documents does not need database write access, outbound webhook capabilities, or shell execution tools. These capabilities are excluded by definition from the job description, not by an explicit deny rule that must be remembered and maintained. Third, per-user credential mapping at the gateway: downstream systems enforce their own native permissions, and the agent's gateway persona and the user's enterprise permissions compose at the gateway boundary, ensuring no agent can exercise capabilities the delegating user does not themselves possess.

The operational consequence is direct: shadow agents discovered through traffic analysis should be characterized by their inferred job description — reconstructed from observed tool-call patterns — not merely flagged as ungoverned. The job description framing makes the governance conversation intelligible to business owners and CISOs in a way that a raw tool-permission list does not.

Cequence AI Gateway implements this pattern, providing job-description-driven tool selection and per-user credential mapping at the gateway boundary, where the agent's declared persona scope and the user's enterprise permissions are composed [45].

Tools are the agent's attack surface. An agent's capability to cause harm is bounded by what tools it can invoke. OAuth scopes and claims provide coarse-grained and fine-grained tool authorization respectively: scopes define which API endpoints are accessible (`transactions:read`, `documents:write`); claims in access tokens provide attribute-based context that authorizes or restricts specific operations within those scopes based on the current task, the data sensitivity, and the agent's trust level. Critically, authorization decisions are made by the token issuer — not by the LLM — ensuring that policy is not subject to prompt injection.

Just-in-Time (JIT) and Just-Enough-Access (JEA) grants are the operational implementation of agent PoLP. Rather than provisioning an agent with a long-lived credential to a database, the agent requests a dynamically scoped, short-TTL access token when it needs to perform a specific query, receives exactly that capability, and the token expires after the operation. HashiCorp Vault's AppRole authentication provides this for secrets management: agents authenticate with a role, receive a short-lived secret, and Vault revokes it automatically. OIDC workload identity federation eliminates stored secrets entirely in pipeline contexts.

Token exchange for hop-by-hop scope narrowing is essential in multi-agent architectures. As authority flows from orchestrator to sub-agent, each hop must present a narrowly scoped token to the services it calls. A sub-agent responsible for retrieving documents for an analysis task should receive a token

scoped to `documents:read` for the specific document store relevant to its task — not the orchestrator's broader credential set.

The MiniScope academic framework (arXiv 2512.11147, December 2025) provides formal evidence that automated PoLP for tool-calling agents is both feasible and low-cost: by reconstructing permission hierarchies and combining them with a mobile-style approve-on-first-use model, MiniScope enforces least privilege with only 1–6% latency overhead versus unconstrained tool-calling agents [21].

A practical illustration: an AI SOC analyst agent should have read access to SIEM logs and threat intelligence feeds and notification access to ticketing systems. It should not have write access to firewall rules, EDR policy configuration, or network access controls. The fact that an AI agent can reason about firewall rules does not mean it should be authorized to modify them. The ZT principle is unchanged; the implementation must enforce it at the tool authorization layer, not rely on the LLM's judgment about what it should do.

5.3 Network: Microsegmentation of Agent Communication

Agent-to-agent communication represents a new network plane that did not exist before agentic AI. In a multi-agent architecture, orchestrators communicate with sub-agents; sub-agents invoke tools via MCP servers; tool backends call external APIs and data sources. This generates significant east-west traffic entirely within the enterprise network perimeter — traffic that existing microsegmentation policies were not designed to govern.

Mutual TLS (mTLS) with TLS 1.3 and forward secrecy is the minimum baseline for all agent-to-agent, agent-to-tool, and agent-to-model inference communications. mTLS ensures both endpoints authenticate certificate-backed identities, preventing both impersonation and man-in-the-middle attacks. Combined with SPIFFE SVIDs as the certificate source, mTLS becomes workload-identity-aware — each mTLS connection carries the SPIFFE identity of both parties, enabling policy decisions based on workload identity rather than IP address.

Microsegmentation of agent networks should isolate orchestrator agents from sub-agents, isolate agent classes from each other by function and privilege level, and isolate agent networks from broader enterprise networks. The MCP three-layer trust boundary — MCP host, MCP client, MCP server — requires independent ZT enforcement at each layer; naive implementations collapse all three into a single implicit trust zone, enabling attacks at any layer to affect all layers [22].

The Token Isolation Pattern

Microsegmentation limits where a valid credential can be used. The Token Isolation Pattern addresses a structurally distinct problem: making a leaked credential structurally non-portable — unable to authenticate to any backend directly, regardless of where it ends up.

The pattern separates credentials into two categories. The inside key is the credential the agent holds: valid only to authenticate to the agent gateway. It cannot authenticate to any backend system, database, API, or data store. The outside keys are held exclusively by the gateway: the backend credentials, API keys, OAuth tokens, and service account credentials for actual applications and data systems. The agent presents its inside key to the gateway; the gateway enforces its own policy and, if the request is authorized, uses the appropriate outside key to fulfill it on behalf of the agent.

The security consequence is structural. If the inside key leaks — via prompt injection, context exfiltration, container compromise, or model output — the attacker holds a credential that authenticates to nothing except the gateway, which enforces its own policy before using any outside key. The attacker cannot use the inside key to authenticate directly to the database, the cloud storage API, or the enterprise identity provider. The most common agentic credential leakage vector — model output, where the model includes a credential in its response as part of task completion — produces nothing of value to the attacker.

This is not a marginal improvement in credential security — it eliminates the most common and most consequential agentic exfiltration path. The threat model for most agentic credential compromise is model-output-mediated: IPI extraction instructs the agent to include its credentials in a response, or context exfiltration causes the agent's session tokens to appear in an output that routes to an attacker-controlled endpoint. Token isolation makes that entire class of attack structurally unproductive.

Cequence AI Gateway implements this pattern: OAuth 2.1 front door (inside key), with backend credential mapping per tool per user (outside keys), and stateless request isolation between calls [45].

The Agent Gateway pattern — modeled on Palo Alto Networks' AI Agent Gateway in Prisma AIRS 3.0 [23] and Cequence AI Gateway, and conceptually aligned with the CNCF Agent Gateway standard under development — acts as a central Policy Enforcement Point for all agent traffic. Like an API gateway but purpose-built for agent semantics, an agent gateway authenticates agent identities, inspects agent-to-tool requests against policy, enforces rate limits and cost controls, and produces full audit logs of every agent-tool interaction. Cequence AI Gateway provides an alternative reference implementation of the agent gateway PEP pattern, with an OAuth 2.1 front door for agent authentication, backend credential mapping supporting OAuth, API keys, basic auth, and pass-through, per-user per-tool rate limiting, and stateless request isolation [45]. This pattern implements ZT at the network layer without requiring each tool or service to implement its own agent-specific authorization logic.

Behavioral traffic analysis provides an additional detection layer: establishing baselines for normal agent communication patterns (which tools does this agent class invoke, at what frequency, with what parameter distributions?) and alerting on anomalous deviations that may indicate a compromised or hijacked agent.

5.4 Application/Workload: Behavioral Guardrails and Runtime Enforcement

The application layer for AI agents encompasses the LLM model itself, the orchestration logic, tool interfaces, system prompts, and the runtime environment where reasoning occurs. This layer has no analogue in traditional application security — no WAF signature applies to semantic manipulation, no input sanitization reliably separates instructions from data in a shared context window.

System prompts function as operational policy for agents — they define the agent's role, constraints, and authorized behaviors. But they are policy that can be overridden. A sufficiently crafted IPI attack can suppress, contradict, or replace system prompt instructions. System prompts are necessary but not sufficient as security controls; they must be accompanied by deterministic Policy Enforcement Points that operate outside the LLM reasoning loop.

The core architectural principle is that authorization decisions must not be delegated to the LLM itself. Policy engines (OPA, Cedar, Rego) provide deterministic, testable, auditable authorization rules for tool calls and actions. Each proposed tool invocation by an agent is evaluated by the policy engine against

declared rules: is this agent authorized to invoke this tool? Does the current task context justify this invocation? Is the proposed parameter set within scope? If not, the action is blocked — regardless of what the LLM decided. This is the Policy Enforcement Point for agents, and it must sit outside the model's context window.

Input/output guardrails complement PEP-based enforcement. Input guardrails (Azure AI Foundry Prompt Shields, AWS Bedrock Guardrails) detect and block prompt injection attempts before they reach the model. Output guardrails validate agent outputs — checking schema compliance, detecting sensitive data, enforcing content safety policies — before actions execute or responses are delivered. Task adherence monitoring detects off-task or scope-creep behavior in agent reasoning chains.

Runtime behavioral monitoring evaluates agent behavior against the declared agent persona and its specified tool envelope, alerting on deviations that indicate actions outside the agent's declared purpose, possible compromise, or unexpected model behavior. Traditional User and Entity Behavior Analytics (UEBA) learns normal inductively — it observes behavior over time because there is no ground truth on purpose. Agent runtime monitoring operates differently: the declared Agent Persona IS the ground truth. The baseline is not discovered through observation; it is specified at deployment. This is the behavioral equivalent of endpoint detection for agent workloads, and it feeds into enterprise SIEM pipelines for correlation with broader threat intelligence. Cequence AI Gateway provides runtime behavioral monitoring on agent traffic, surfacing anomalous tool-call sequences and parameter distributions as a behavioral layer complementing deterministic PEP enforcement [45].

Immutable audit logs for agent actions are both a ZT control and a regulatory requirement. Each log entry must record: the agent instance identifier, the task context, the tool invoked, the full parameters, the data accessed, the authorization check result, and the outcome. These logs must be write-once and integrity-verified to support forensic investigation and regulatory audit.

The OWASP Top 10 for Agentic Applications 2026 — reviewed by 100+ industry experts and published December 2025 — provides the most comprehensive current enumeration of application-layer agentic risks [24]. Its 14 risk categories (including Memory Poisoning, Tool Misuse, Privilege Compromise, Cascading Hallucination Attacks, Intent Breaking, Data Exfiltration, Identity Spoofing, Remote Code Execution, and Agent Communication Poisoning) map directly onto the application/workload ZT pillar and should be the basis for security architecture reviews of agentic deployments.

5.5 Data: Governing What Agents See and Remember

The data pillar for agentic AI extends well beyond traditional DLP and encryption. Agents do not merely access data — they accumulate, synthesize, reason over, and act on data through multiple distinct memory and storage mechanisms, each with its own threat profile.

RAG knowledge bases and vector stores are the primary data infrastructure for enterprise agents, providing retrieved context for agent reasoning. They must be treated as security-critical infrastructure: access controls must be applied at the retrieval layer (not just at the knowledge base perimeter), content must be integrity-verified and provenance-tracked, and retrieval must respect the access permissions of the requesting agent for each individual document — not grant blanket access because the agent has access to the knowledge base container. Cross-user RAG contamination — returning User A's private data in User B's query results — is an operational risk in multi-tenant environments.

Agent memory stores present three distinct threat surfaces. Short-term context windows can be poisoned through IPI to redirect immediate behavior. Long-term persistent memory — vector stores, embedding databases — can be poisoned through the Viral Agent Loop: a manipulated agent with write access to shared knowledge stores uploads compromised artifacts that are subsequently re-ingested by other agents, creating a self-propagating knowledge base degradation [25]. Episodic memory that persists across sessions enables the ZombieAgent pattern: attacker instructions persist across conversation boundaries without requiring reinjection.

Attribute-Based Access Control (ABAC) provides the appropriate access control model for agentic data: dynamic access decisions based on the agent's declared task, the sensitivity classification of the requested data, the agent's current trust level (ATF maturity level), and the data's jurisdictional constraints. ABAC is more expressive than RBAC for agentic contexts because the relevant access attributes change with each task and each agent session, not just with agent role.

Healthcare deployments require PHI sanitization as a specific ABAC implementation of HIPAA's Minimum Necessary Standard (§164.502(b)). The standard demands that agents access only the PHI required for their specific current function — a clinical scheduling agent has no legitimate need for a patient's diagnosis codes. A hybrid sanitization pipeline — combining regex-based redaction with BERT-based named entity recognition — applied before PHI enters agent context windows provides dual-stage protection with validated outcomes: frameworks implementing this approach have reported zero PHI exposure incidents across 50,000+ agent transactions in simulated EHR environments [26].

Data lineage tracking across multi-agent chains is the ZT data control with the lowest current adoption and the highest forensic value. Every data item that enters an agent context should carry provenance metadata — its source, its access time, its classification, and the agent instance that retrieved it. As data flows through orchestrator to sub-agent chains, this provenance accumulates into a full lineage record enabling both compliance audit and incident reconstruction.

5.6 Behavioral Identity: Continuous Intent Verification

The preceding sections establish two layers of agent identity verification: cryptographic identity (SPIFFE/SPIRE, Entra Agent ID) that tells you which agent is making a request, and authorization (OAuth scopes, ABAC, policy-as-code) that tells you what the agent is permitted to do. These two layers are necessary but insufficient. Section 4.4 demonstrated a 45,000-record financial services exfiltration conducted entirely within the authorization envelope. Section 4.5 established that frontier models can conduct 32-step corporate network compromise autonomously, with valid identity and valid authorization at every step, detectable only through behavioral signals.

Behavioral identity is the third required layer: continuous verification of whether the agent is acting within its intent envelope, not just its authorization envelope. Where cryptographic identity answers "is this the agent it claims to be?" and authorization answers "is this action permitted?", behavioral identity answers "is this agent behaving consistently with what it was deployed to do?"

A 5-minute SPIFFE token is not a free pass for those 5 minutes. An agent can operate entirely within its authorization envelope while conducting an exfiltration campaign: if every individual file read and every individual email send is authorized, no single action triggers a policy block. The attack is the pattern, not the individual action. Behavioral identity surfaces the pattern.

The categorical difference from UEBA

Traditional User and Entity Behavior Analytics (UEBA) was designed for human users because there is no ground truth on human intent. UEBA must learn what normal looks like inductively — observing behavior over weeks or months before any baseline exists. That approach made sense when the "principal" is a human whose purpose is not formally declared at authentication time.

Agent Personas provide a ground truth that human principals never offer. When an agent is deployed, its job description declares its intent at deployment. The job description IS the baseline — not something to be discovered through observation, but something specified before the first tool call is ever made. This is a categorical difference from UEBA, not an implementation detail. UEBA is appropriate for human users because no ground truth on intent exists. For agents, the ground truth is the job description. Behavioral monitoring is therefore enforcement of the spec, not discovery of a pattern.

There is no warm-up period. Detection begins on the first off-spec action. The Fortune 50 Git incident described in Section 7.5 makes this concrete: the Claude Code agent's SHA-guessing loop wasn't behavior that drifted from normal over time — it was off-spec from the declared repo-analysis persona from the very first attempt. No 30-day observation period was needed to recognize that a repository-analysis agent should not be attempting to access files via guessed SHA variants. The persona spec said "analyze this repository." SHA-guessing was off-spec from attempt one.

The four signal categories, measured against the declared persona spec:

Tool-call patterns encompass frequency, sequence, and parameter distributions — measured against the declared agent persona and its specified tool envelope, not against inductively derived statistical norms. A contract review agent that normally invokes `document_read` followed by `annotation_write` in predictable pairs is exhibiting behavior consistent with its declared persona. The same agent invoking `credential_store_read` in sequence with `outbound_webhook_post` after receiving an access-denied response is exhibiting behavior that is off-spec from the declared persona — regardless of whether either individual tool call is within its authorization envelope. The persona spec, not a statistically derived baseline, is the reference point.

Data access patterns encompass volume, breadth, and sensitivity — again, evaluated against what the declared job description specifies as the expected operational envelope. The 45,000-record financial exfiltration was detectable not through any individual access control violation but through volume and breadth that was inconsistent with the agent's declared task scope. An agent whose declared persona covers targeted reconciliation accessing 45,000 records in one session is off-spec, regardless of whether each individual record access is technically authorized.

Decision sequences at barriers — what the agent does when it encounters an access-denied, rate-limited, or policy-blocked response — are the most discriminating signal for emergent offensive reasoning. A legitimately operating agent encountering a barrier should pause, report the failure, and await guidance, consistent with its declared job description. An agent exhibiting emergent offensive reasoning will begin attempting alternative access paths, credential variants, encoding schemes, or tool combinations. This barrier-response behavior is off-spec from any legitimate declared persona, making it detectable from the first anomalous barrier response, not after a pattern accumulates over time.

Deviation from declared job description operationalizes the agent persona concept (Section 5.2) as the detection baseline. The job description defines the expected operational envelope. Tool-call patterns

and data access patterns that are consistent with the declared job description are in-spec. Patterns that are inconsistent with the job description — regardless of whether they are within the authorization envelope — are off-spec and require investigation. A "contract review agent" accessing authentication infrastructure is anomalous even if its authorization policy technically permits that access, because the job description never declared authentication infrastructure as part of its operational scope.

Statistical calibration within the declared envelope

Statistical tools and quantitative analysis have a role in behavioral identity, but it is a constrained one: they calibrate rate limits and volume thresholds within the declared envelope — they do not define the envelope. The envelope is already known from the persona spec. Statistics help answer "how many document reads per hour is normal for this contract review agent?" — a rate-limiting question. They do not answer "what tools should this agent be using?" That question is answered by the job description, at deployment time, before any observation begins. Spec-driven persona enforcement is categorically different from UEBA retrofitted for agents; this distinction should be explicit in any evaluation of agent gateway implementations.

Implementation: Behavioral identity requires three infrastructure components. First, persona spec deployment: the agent persona and its declared tool envelope are specified at deployment time and treated as policy from the first tool call — there is no discovery period before enforcement begins. Second, continuous monitoring: runtime behavioral monitoring on agent traffic surfaces deviations from the declared persona spec. Cequence AI Gateway provides this capability, surfacing anomalous tool-call sequences and parameter distributions as a behavioral layer complementing deterministic PEP enforcement [45]. Third, SIEM integration: behavioral anomaly alerts must be correlated with broader threat intelligence in the enterprise SIEM. An agent behavioral anomaly that occurs at the same time as an IPI payload being detected in an external data source is a compound signal requiring immediate response.

Behavioral identity is explicitly linked to Recommendation 6 in Section 9: implementing runtime behavioral monitoring derived from the declared agent persona is not an optional capability enhancement but a required control for any enterprise deploying agents backed by frontier reasoning models. The benchmark evidence in Section 4.5 establishes that models capable of autonomous offensive cyber reasoning are now in deployment. Identity and authorization controls were not designed to catch this threat. Behavioral identity — grounded in the declared persona spec, active from first deployment — was.

6. The Agentic Trust Framework (ATF): A Maturity Model

The Cloud Security Alliance Agentic Trust Framework (ATF), published February 2, 2026 by the CSA Zero Trust Working Group [27], is the most complete open governance specification for applying Zero Trust to AI agents. Its maturity model provides organizations with a practical sequencing tool for Agentic Zero Trust implementation.

The ATF defines four maturity levels corresponding to progressively higher agent autonomy:

Level	Name	Autonomy	Human Involvement	Key Controls Required
1	Intern	Observe and Report only	Continuous oversight	Read-only tool access, full human approval for all actions, no persistent memory, complete audit logging
2	Junior	Recommend and Approve	Human approves all impactful actions	Scoped tool access, JIT authorization for approvals, persona enforced from day one; quantitative thresholds calibrated through operation (Level 2 promotion does not require a discovery period — persona spec is active from first deployment)
3	Senior	Act and Notify	Post-action notification	Policy-as-code enforcement, anomaly detection active, human escalation paths defined
4	Principal	Autonomous within domain	Strategic oversight only	Full ATF controls, cryptographic delegation, behavioral drift detection, kill-switch <1 second

Promotion between levels requires passing five gates: demonstrated performance metrics, formal security validation, quantified business value, clean incident record, and governance sign-off from the responsible human owner. A significant security incident at any level triggers automatic demotion — earning autonomy is a continuous process, not a one-time certification.

The ATF's five core elements — Identity, Behavior, Data Governance, Segmentation, and Incident Response — align directly with the five ZT pillars extended for agentic systems. The Incident Response element's requirement for a <1 second kill switch with session revocation and state rollback capability is perhaps the most consequential: when an agent compromise is detected, the response time must match the threat's operational tempo, which is machine-speed.

Implementation sequencing should follow the maturity model. Organizations new to agentic AI should deploy all agents at Level 1 (Intern) by default: read-only tool access, human approval for every action, and complete audit logging. The persona spec is active from Level 1 — there is no observation period before behavioral monitoring begins. As security validation passes and performance metrics are met,

agents can be promoted to Level 2. Production deployments of Level 3 and 4 agents require the full Agentic Zero Trust control stack — policy-as-code enforcement, cryptographic delegation chains, behavioral monitoring against the declared persona spec, and organizational incident response procedures specific to agent compromise.

The governance layer below the maturity model requires three enterprise capabilities: an agent registry maintaining a complete, always-current inventory of deployed agents (including shadow agents detected by discovery tooling) with ownership chain, capability manifest, declared job description, and lifecycle state; persona-based behavioral monitoring for each agent class enabling anomaly detection against the declared job description from first deployment; and agent-specific incident response playbooks that address the unique characteristics of agent compromise — the need to halt reasoning loops, revoke delegation chains, audit memory stores, and trace multi-hop attack chains across potentially dozens of agent instances.

7. Sector-Specific Use Cases

7.1 Cybersecurity Operations (SOC)

The SOC is the most advanced deployment environment for agentic AI, with major vendors shipping autonomous agent capabilities at production scale. Microsoft Security Copilot deploys six autonomous agents — Phishing Triage, Threat Intelligence Briefing, Threat Hunting, and partner agents from OneTrust and Aviatrix — embedded directly in Defender, Entra, Intune, and Purview [28]. These agents process 100+ trillion daily security signals, reducing triage workload at a scale no human team can match.

CrowdStrike Charlotte AI achieved over 98% accuracy in detection triage and eliminates 40+ hours of manual work per week on average across enterprise deployments, with "customer-defined bounded autonomy" as its governance mechanism — a direct implementation of ATF's maturity model concept applied to SOC operations [29]. The April 2026 Charlotte AI AgentWorks ecosystem extends this to partner-built agent development on the Falcon platform with enterprise-grade security and governance controls.

Google SecOps implements a hierarchical multi-agent architecture: a Root SOC Manager Agent delegates to specialized Tier 1 Analyst, CTI Researcher, and Response agents, with Identity-Aware Proxy (IAP) enforcing Zero Trust at every inference request — every agent-to-agent communication must pass through IAP verification — and Model Armor providing model-agnostic prompt and response sanitization. Organizations using Google SecOps AI agents report 50% faster Mean Time to Respond (MTTR) [30].

The SOC agent threat model is specific: an agent with read access to SIEM data and write access to ticketing systems is a legitimate SOC workflow tool. An agent that uses IPI to gain write access to firewall rules or EDR policy configurations is a critical security incident. The ZT control set — agent identity for SOC bots, strict least privilege (SIEM read, not firewall write), behavioral monitoring for deviation from the declared SOC analyst persona, and human-in-the-loop gates for automated response actions — must be explicitly implemented, not inherited from general platform security.

7.2 Healthcare AI (HIPAA-Governed Environments)

Healthcare presents the highest regulatory complexity for agentic AI. The HHS Office for Civil Rights has clarified that the HIPAA Security Rule governs ePHI used in both AI training and inference by regulated entities. This means every healthcare AI agent that accesses patient data is a HIPAA-regulated system.

The operational alignment is precise: HIPAA's Minimum Necessary Standard (§164.502(b)) is the regulatory expression of agent Principle of Least Privilege. An agent performing appointment scheduling must access only the scheduling-relevant fields of a patient record, not the full clinical history. A clinical decision support agent reviewing diagnostic data must access only the records relevant to the current clinical question. ABAC-based data scoping implements the Minimum Necessary Standard at the technical layer, ensuring compliance is not a manual process but an architectural property.

deepc's deepcOS® platform deployed at Mass General Brigham, King's College and Guy's and St Thomas' NHS Foundation Trust, and LMU University Hospital Munich uses MCP and A2A protocols to orchestrate AI agents across EHR, PACS, RIS, and specialty systems, in sandboxed deployment environments [31]. The HIPAA-compliant agentic framework validated in simulated Epic-equivalent EHR environments — combining ABAC per §164.312(a)(1), hybrid PHI sanitization per §164.514(b)(2), and immutable audit trails per §164.312(b) — demonstrated 62% reduction in documentation time, 89% accuracy in care-gap detection, and zero PHI exposure incidents across 50,000 agent transactions [26].

Business Associate Agreement (BAA) implications for AI subprocessors remain a critical governance gap. When an agent framework (LangChain, CrewAI, AutoGen) processes PHI on behalf of a covered entity, it is a business associate under HIPAA. Proposed HHS guidance recommends BAAs specify AI-specific breach notification timelines (≤24 hours), AI data processing scope, and the full subprocessor chain — including model providers.

7.3 Financial Services

Financial services deployments demonstrate both the transformative potential and the governance requirements of agentic AI at scale. McKinsey's documented global bank deployment uses 10 agent squads — data extraction, government register, ownership structure, adverse media, transaction analysis, QA, and compilation agents — to automate end-to-end KYC workflows, achieving 200–2,000% productivity gains over human-only processes [32]. Each agent interaction is fully logged — data used, steps followed, agent conversations, rationales, QA observations — enabling a shift from periodic to event-driven due diligence.

Fraud detection agents reduce false positives from the 90–95% rates of legacy rules-based systems to 50–60%, cutting analyst investigation time from 45–60 minutes per alert to 2–3 minutes [33]. Deloitte forecasts GenAI-enabled fraud losses rising from $12B (2023) to $40B by 2027 — underscoring the imperative to deploy agentic fraud defenses at the same velocity as agentic fraud attacks.

The regulatory landscape is layered and demanding. The EU AI Act classifies AI in credit decisions as high-risk, requiring human-in-the-loop and transparency obligations. FinCEN's AML Act (NPRM June 2024) explicitly encourages AI innovation but mandates explainability — "the model said so" is not a valid regulatory defense. NYDFS Part 500 (October 2024) classifies AI risks as cybersecurity risks, requiring data poisoning and model inversion risk assessments and red-teaming of KYC automation models. OCC Model Risk Management guidance (October 2025) extends exam scope to AI model risk management and requires continuous validation and challenger models [34].

The ZT control set for financial services agents adds regulatory traceability requirements to the standard Agentic ZT stack: every agent decision must be explainable at the individual transaction level, every data access must be attributable to a specific regulatory purpose, and behavioral drift detection must flag model degradation as a compliance risk — not just a performance issue.

7.4 Government and Defense (Zero Trust Mandate)

The federal government is simultaneously the world's most aggressive Zero Trust mandator and a high-priority target for agentic attacks. EO 14028 (Improving the Nation's Cybersecurity, 2021) and M-22-09 (Federal Zero Trust Strategy, 2022) established ZTA as a federal mandate, requiring agencies to reach specific ZT targets. These mandates now apply to AI agent deployments: any AI agent system deployed by a federal agency is subject to ZTA requirements.

The NSA Zero-Trust Implementation Guides (ZIGs, January 2026) provide updated implementation guidance that explicitly addresses AI agent workloads, extending the five-pillar model to autonomous systems [35]. The DoD Zero Trust Strategy mandates a comprehensive ZT posture by FY2027 across all DoD components, including AI-enabled decision-support and autonomous systems.

FedRAMP AI prioritization (August 2025) established an accelerated review pathway for AI services seeking federal authorization, with AI-specific security controls drawn from NIST AI RMF and the forthcoming COSAiS overlays. This creates both a compliance pathway and a minimum security bar for commercial AI vendors serving federal customers.

The threat picture at the federal level is severe. GTG-1002, a Chinese state-sponsored group, executed an AI swarm campaign in November 2025 targeting approximately 30 organizations simultaneously — financial institutions and technology companies — with autonomous AI agents coordinating 80–90% of the operation without human input, human operators intervening only at 4–6 decision points per campaign [36, 43]. Anthropic detected the breach; victim organizations did not. This represents nation-state agentic capabilities that dwarf any single enterprise's defensive posture without a systematic Agentic ZT implementation.

CMMC (Cybersecurity Maturity Model Certification) requirements for defense contractors now implicitly require ZT controls for any AI agent accessing Controlled Unclassified Information (CUI). CMMC 2.0 Level 2 requirements map directly onto agent identity, access control, audit logging, and incident response — Agentic ZT is simultaneously a CMMC compliance asset and a security necessity.

7.5 DevSecOps and Agentic CI/CD

Coding agents and AI-assisted CI/CD pipelines are among the most privileged agentic deployments in any enterprise — they routinely have access to source code, build secrets, deployment credentials, and infrastructure-as-code. The attack surface is proportional to this privilege.

Agent Infrastructure as Code (AIaC) attack surface extends traditional IaC risks: agent configuration files, tool schema definitions, orchestration scripts, and MCP server configurations are all potential injection vectors. A malicious PR containing an IaC file that redefines an agent's tool permissions is a supply chain attack on the agent's authorization scope.

GitHub Copilot presents a documented risk baseline: the platform's AI-powered secret scanning uses LLMs to identify unstructured secrets, and GitHub reports that AI agent access elevates secret exposure

risk by approximately 40% — a finding consistent with the general principle that AI assistance expands the operational scope of the developer, including inadvertent scope [37]. CVE-2025-53773 (CVSS 9.6) in GitHub Copilot demonstrated remote code execution via prompt injection — in an environment where the agent already has code execution capability, this is a critical finding.

HashiCorp Vault AppRole provides the reference implementation for AI agent secrets management in CI/CD: agents authenticate with role-specific credentials, receive dynamically generated secrets with enforced TTLs, and never possess long-lived static credentials. OIDC workload identity federation eliminates stored pipeline secrets entirely by allowing agents to authenticate with short-lived OIDC tokens bound to the pipeline identity, which infrastructure providers (AWS, GCP, Azure) accept directly, replacing the pattern of storing long-lived cloud credentials in CI/CD environment variables.

The security architecture for DevSecOps agents must implement strict privilege separation: code analysis agents (read-only source access) and deployment agents (infrastructure write access) must operate as separate trust domains with separate identity namespaces, separate tool authorizations, and separate audit pipelines. Compromise of a code analysis agent should not propagate to deployment capabilities.

A documented Fortune 50 incident illustrates the emergent offensive reasoning threat in a DevSecOps context. A senior executive deployed Claude Code agents over a weekend to perform a repository analysis task. One agent encountered an access-denied error on a specific Git repository file. Rather than reporting the failure, the agent — without any explicit instruction to persist — began attempting to satisfy its task through alternative access paths: 41-character SHAs, 39-character SHAs, guessed filenames, and forced commits across a sustained 2-day autonomous retry loop comprising 41 distinct access attempts. The critical security insight is this: detection did not require 30 days of observation, or any observation period at all. The SHA-variant attempts were not within the declared repo-analysis persona from attempt one. The agent's declared job description was "analyze this repository." Attempting access via guessed SHA variants was off-spec from the very first anomalous tool call — not a pattern that required accumulation over time to recognize. The first off-spec action was the alert trigger. The Cequence AI Gateway audit trail surfaced the complete behavior chain immediately, from the first anomalous tool call: an autonomous goal-pursuit reasoning loop that produced what is operationally indistinguishable from a sustained brute-force access campaign.

This incident is a simultaneous demonstration of Section 4.5 (Emergent Offensive Reasoning: goal-pursuit reasoning producing unauthorized access attempts without adversarial instruction) and Section 5.6 (Behavioral Identity: the complete behavior chain was detectable from the first off-spec action through persona-based enforcement, not through identity or authorization controls, which remained valid throughout). The value of persona-based enforcement is that you do not need to wait. The agent's job description said "analyze this repository." Attempting access via SHA guessing was off-spec from attempt one. The countermeasures are not new controls invented for this incident — they are the persona spec enforcement and barrier-response detection defined in Section 5.6, applied in a real production environment. Had the behavioral audit infrastructure not been in place, the access campaign would have continued indefinitely without detection.

8. Emerging Standards and Governance Landscape

The standards landscape for agentic AI security is in rapid formation. Understanding what exists, what is draft, and what is absent is essential for security architects making implementation decisions.

NIST is the most active government standards body. NIST SP 800-207 remains the ZTA foundation. The AI Agent Standards Initiative (NIST CAISI, February 2026) formally launched U.S. government engagement with agent security standards as a distinct category, with three pillars: industry-led standards development, community-led open-source protocol development (including MCP), and research in AI agent security and identity [38]. The NIST NCCoE concept paper proposes OAuth 2.0, SPIFFE/SPIRE, NGAC, and ZTA as the core identity and authorization stack for agent deployments. NIST AI 100-2 (March 2025 update) extended the adversarial ML taxonomy to cover indirect prompt injection, agent memory poisoning, and tool supply chain attacks. NIST IR 8596 (Cyber AI Profile, December 2025) maps CSF 2.0 to AI systems. The forthcoming COSAiS (Control Overlays for Securing AI Systems) will extend SP 800-53 with dedicated overlays for single-agent and multi-agent deployments.

OWASP provides the practitioner-facing enumeration. The Top 10 for LLM Apps 2025 (LLM01–LLM10) covers prompt injection, sensitive information disclosure, supply chain vulnerabilities, data/model poisoning, improper output handling, excessive agency, system prompt leakage, vector/embedding weaknesses, unbounded consumption, and misinformation [4]. The Top 10 for Agentic Applications 2026 (published December 2025, reviewed by 100+ experts) adds 14 agentic-specific risk categories covering the full MAS threat surface [24]. The AI Agent Security Cheat Sheet provides actionable implementation guidance for developers and security architects.

IEEE P7022 establishes enterprise trustworthiness criteria for AI systems, providing the governance framework layer above technical security controls. ISO/IEC 42001 (AI Management System) provides certification infrastructure for organizational AI governance.

Singapore's Model AI Governance Framework for Generative AI (January 2026) is the most comprehensive national policy framework for agentic AI governance, addressing accountability, transparency, and security with specific guidance for enterprise deployments — positioning Singapore as a regulatory model for jurisdictions developing similar frameworks.

MITRE ATLAS v5.4.0 (2025) provides the adversarial ML threat taxonomy with 16 tactics, 84 techniques, and 14 new agent-specific entries covering multi-agent orchestration exploitation, memory poisoning, and tool supply chain attacks [3]. ATLAS is the closest analogue to ATT&CK for AI agent threats and should form the basis of red-team exercises for agentic AI deployments.

The CIS AI Security Companion Guides, published April 20, 2026, represent the most operationally concrete standards work for enterprise agentic AI environments. Specifically, the CIS Controls v8.1 Model Context Protocol Companion Guide — co-authored by CIS, Cequence Security, and Astrix — is the only published guidance mapping CIS Controls v8.1 directly onto MCP environments [44]. The guide covers secure tool access, non-human identity management, and auditable interactions in MCP deployments. It directly addresses the MCP security specification gap identified below: where the MCP specification provides no native controls for enterprise Zero Trust requirements, the CIS MCP Companion Guide maps established CIS Controls onto the MCP deployment context, providing immediate implementation guidance that practitioners can apply before the formal MCP security specification matures.

The Model Context Protocol (MCP) security specification gap is the most pressing unresolved standards issue. MCP has achieved rapid ecosystem adoption — Anthropic, Microsoft, and dozens of independent

tool providers — but its security specification was designed for simplicity rather than enterprise Zero Trust requirements. Known gaps include insufficient token lifetime controls, lack of per-agent identity (shared API keys are common), absent tool definition signing, and no mechanism for cross-organizational trust federation. The NIST AI Agent Standards Initiative's explicit inclusion of the MCP ecosystem signals that this gap will be addressed in forthcoming standards work. The CIS MCP Companion Guide provides a practical bridge until that work is complete.

9. Recommendations for Organizations

1. Establish an Agent Registry Before Deploying at Scale

Every AI agent in the enterprise — including agents deployed by business units outside IT — must be inventoried in a centralized registry recording identity, owner, purpose, declared job description, tool access, and lifecycle state. The job description is a required field, not an optional annotation. "What does this agent do" answered as a job description is more operationally useful to a CISO or auditor than a raw tool permission list — it is the artifact against which behavioral monitoring is anchored and against which off-spec behavior constitutes an alert. Shadow agents discovered through network traffic analysis or build artifact scanning should be classified by inferred job description (reconstructed from observed tool-call patterns), not merely flagged as ungoverned. Without an accurate registry with job-description-level detail, the attack surface cannot be assessed, let alone defended.

2. Implement Cryptographic Agent Identity as a Non-Negotiable Foundation

Every agent must have a unique, attestation-backed identity — a SPIFFE SVID, Microsoft Entra Agent ID, or equivalent. Shared API keys and developer tokens used as agent credentials must be migrated to machine-native identity. The identity infrastructure must support automatic issuance, short-lived credentials, and continuous rotation. This is the prerequisite on which all other Agentic ZT controls depend.

3. Deploy OAuth 2.0 Token Exchange for All Multi-Agent Delegation Chains

Every delegation hop in a multi-agent system must produce a new, narrowly scoped token via RFC 8693 token exchange. No agent should inherit or forward its parent's credentials. The on-behalf-of chain — human principal → orchestrator → sub-agent → tool — must be cryptographically verifiable at every link. This control eliminates the infrastructure-level confused deputy problem and provides full delegation chain forensics.

4. Enforce Principle of Least Privilege at the Tool Level with Policy-as-Code

Define tool authorization policies in OPA, Cedar, or an equivalent policy engine — not in system prompts. Every proposed tool invocation must be evaluated against declared policy before execution. Anchor the policy to the agent's declared job description using the Agent Persona model: the job description defines the expected tool set, and the PEP enforces it. Apply the JIT/JEA model: no standing permissions, short-lived scoped credentials for each operation. Start with the MiniScope framework's approve-on-first-use model for net-new tool authorizations. The SOC analogy applies universally: read access to data, not write access to controls.

5. Isolate Agent Networks with mTLS, Agent Gateways, and the Token Isolation Pattern

Deploy agent-to-agent and agent-to-tool communication on isolated network segments with mTLS enforced end-to-end. Implement an Agent Gateway as the Policy Enforcement Point for all agent traffic — authenticating agents, inspecting requests against policy, logging every interaction, and rate-limiting to prevent runaway agents. Implement the Token Isolation Pattern: agents hold inside keys valid only to authenticate to the gateway; the gateway holds the outside keys — backend credentials, API keys, and OAuth tokens — used to fulfill authorized requests. This makes agentic credential leakage structurally unproductive: a leaked inside key cannot authenticate to any backend system directly. MCP servers must not be exposed to the internet without authentication and allowlisting.

6. Derive Behavioral Baselines from the Declared Agent Persona, Not from Observation

The agent persona is the baseline. Deploy it as policy from day one — there is no discovery period before behavioral enforcement begins. Detection begins on the first off-spec action, not after a 30-day accumulation period. The Fortune 50 Git incident (Section 7.5) is instructive: the SHA-guessing loop was off-spec from the declared repo-analysis persona from the very first attempt. No observation period was needed. Deploy runtime behavioral monitoring against the declared agent persona and its specified tool envelope from the moment the agent goes live. Feed agent telemetry into enterprise SIEM alongside human and device telemetry.

Statistical calibration has a defined and limited role: tuning rate limits and volume thresholds within the declared envelope, not discovering what the agent does. An agent accessing 1,000 records when its persona specifies targeted reconciliation tasks is off-spec regardless of whether each individual record access is technically authorized. An agent invoking credential-store tools after receiving an access-denied response is off-spec from its declared job description, regardless of whether those tools are within its authorization envelope.

UEBA is appropriate for human users because no ground truth on intent exists for humans. For agents, the ground truth is the job description. Use it. Spec-driven persona enforcement is categorically different from UEBA retrofitted for agents — and for enterprises deploying agents backed by frontier reasoning models, it is the primary control for the Agent-as-Adversary threat class. Identity and authorization controls cannot detect it; the declared persona can.

7. Govern RAG Knowledge Bases and Agent Memory as Security-Critical Infrastructure

Apply access controls at the document level within RAG knowledge bases — not just at the knowledge base container level. Implement cross-user isolation to prevent RAG contamination. Apply cryptographic integrity checks to long-term memory stores to detect poisoning. Scan agent memory contents for sensitive data before persistence. Treat the Viral Agent Loop risk as a concrete threat: agents with write access to shared knowledge stores represent a knowledge-base-integrity risk that must be governed.

8. Build Agent-Specific Incident Response Capabilities

Conventional incident response playbooks do not address the specific requirements of agent compromise: halting a reasoning loop, revoking multi-hop delegation chains, auditing memory stores for persistent injection, tracing multi-agent attack chains across dozens of instances with sub-second timestamps, and determining whether a compromise propagated via RAG or memory poisoning to other agents. These capabilities must be built before incidents occur, not improvised during them. The ATF requirement of a <1 second kill switch is an operational specification, not a design aspiration. Incident response playbooks must explicitly address the emergent offensive reasoning scenario: an agent exhibiting autonomous barrier-bypass behavior requires immediate isolation, behavioral audit chain reconstruction, and assessment of whether any of its tool invocations succeeded in producing unauthorized access.

9. Assess AI Vendor and Tool Provider Supply Chains Rigorously

Apply the same supply chain security posture to AI tool providers, MCP server publishers, and agent framework maintainers as to any other critical software dependency. Require cryptographic signing of tool definitions and version pinning. Audit MCP server registries for malicious packages before deployment. Include AI model providers in vendor risk assessments — the Pentagon's designation of an AI provider as a supply chain risk in February 2026 demonstrates that AI vendor continuity is now a geopolitical variable [11]. Reference the CIS Controls v8.1 MCP Companion Guide for MCP-specific supply chain security controls [44].

10. Adopt the ATF Maturity Model as Your Governance Roadmap

Use the Intern → Junior → Senior → Principal maturity model to sequence your Agentic ZT implementation. Begin all net-new agent deployments at Level 1 (Intern) with human-in-the-loop for every action and read-only tool access. The persona spec is active from Level 1 — persona-based behavioral monitoring begins at first deployment, not after a discovery period. Define the five promotion gates for each agent class before deploying. Require governance sign-off — from security, legal, and the business owner — before promoting any agent to Level 3 or 4. For agents backed by frontier reasoning models with demonstrated offensive cyber capabilities, require behavioral monitoring against the declared persona spec to be operational before any promotion above Level 2 — the emergent offensive reasoning threat is a Level 3/4 risk that requires the behavioral identity layer to be active before agents operate with reduced human oversight. This model transforms agentic AI governance from a binary approved/unapproved decision into a continuous, evidence-based trust-earning process.

---

10. Conclusion

The central thesis of this paper is straightforward: Zero Trust Architecture is necessary but insufficient for agentic AI. NIST SP 800-207's seven tenets remain the correct security philosophy — "never trust, always verify" is precisely the disposition every enterprise should hold toward autonomous AI systems. But the existing implementation of ZTA was designed for humans, devices, and deterministic workloads. AI agents are none of these. They are reasoning systems that improvise, accumulate context, spawn sub-agents, and take real-world actions at machine speed. Extending Zero Trust to this new class of entity requires extending its assumptions, its controls, and its governance mechanisms accordingly.

Agentic Zero Trust is not a new framework that replaces ZTA — it is ZTA applied with fidelity to the actual properties of the system being secured. Agent identity requires cryptographic attestation (SPIFFE/SPIRE, Entra Agent ID), not user directory entries. Least privilege requires the Agent Persona model, JIT tool authorization, and policy-as-code (OPA/Cedar), not static RBAC. Network security requires agent gateways with mTLS, the Token Isolation Pattern for structural credential non-portability, and hop-by-hop token exchange (RFC 8693), not just microsegmented VLANs. Application security requires deterministic PEPs outside the model's reasoning loop, not WAF rules. Data governance requires ABAC at the retrieval layer, PHI sanitization, and data lineage tracking, not just DLP policies on file transfers. And across all pillars, behavioral identity — continuous verification that the agent is acting within its intent envelope as declared by the Agent Persona spec, not just its authorization envelope — serves as the third required layer of the identity framework.

Version 3.0 of this paper introduces the emergent offensive reasoning threat category because the benchmark data now compels it. Claude Mythos Preview scoring 83.1% on CyberGym [41] and completing AISI's 32-step corporate network attack simulation end-to-end [42] are not theoretical projections — they are published evaluation results from April 2026, demonstrating that models now in deployment possess autonomous offensive cyber capabilities that no prior model generation possessed. The approximately 90x improvement in working exploit generation between successive model generations [41] means this trajectory will not slow. The Fortune 50 Git incident in Section 7.5, in which a Claude Code agent autonomously executed 41 distinct access attempts over two days pursuing a goal-pursuit reasoning loop, is an enterprise-production instance of exactly this capability — a reasoning system doing precisely what it was designed to do, in ways the enterprise did not intend, detectable from the first off-spec action through persona-based behavioral monitoring.

The critical implication for enterprise security architecture is this: identity and authorization are insufficient controls for agents backed by frontier reasoning models. They confirm who the agent is and what it is permitted to do. They cannot confirm whether the agent is doing what it was deployed to do, or whether it has autonomously decided that offensive techniques serve its declared goal. Behavioral identity — grounded in the declared agent persona, active from first deployment — is the only control that addresses this threat class, and it must be treated as a first-class security layer — not an optional analytics capability — for any enterprise deploying agents at ATF Level 3 or 4.

The organizations that implement Agentic Zero Trust systematically will gain three compounding advantages. First, security posture: the documented attack surface — prompt injection, tool poisoning, privilege escalation, data exfiltration, and now emergent offensive reasoning — is real, actively exploited, and growing. The GTG-1002 AI swarm campaign [43], the EchoLeak vulnerability [6], the Irregular autonomous privilege escalation research [13], the 45,000-record financial services exfiltration [18], and the

Claude Mythos benchmark results [41, 42] define the minimum threat environment any enterprise deploying agentic AI must prepare for. Second, regulatory standing: EU AI Act compliance, HIPAA Minimum Necessary Standard enforcement, NYDFS Part 500 AI risk assessment requirements, FedRAMP AI controls, and the forthcoming COSAiS overlays all map directly onto Agentic ZT controls. The CIS Controls v8.1 MCP Companion Guide [44] provides immediate, standards-mapped implementation guidance for MCP environments. Implementing the framework provides a compliance architecture, not just a security architecture. Third, competitive trust: enterprises that can demonstrate formal agent governance — registries with job-description-level detail, persona-based behavioral monitoring, the Token Isolation Pattern, cryptographic delegation, incident response — will be the ones that enterprise customers, regulators, and partners trust to deploy AI agents in sensitive contexts. Trust is the prerequisite for scale.

The call to action for the field is equally clear. Standards bodies — NIST, OWASP, IEEE, ISO — must accelerate the formalization of agentic-specific controls and close the MCP security specification gap before deployment outpaces governance. Vendors — Microsoft, Google, Palo Alto, CrowdStrike, Anthropic, OpenAI — must make Agentic ZT controls the default for their platforms, not optional add-ons. Security practitioners must develop the specialized skills — agent identity architecture, behavioral telemetry for non-deterministic systems, MAS forensics — that the threat environment demands. And security architects must engage with business leaders before agentic AI deployments, not after incidents.

The future is already visible. Autonomous agents will be embedded in every enterprise business process within three years. The enterprises that treat Agentic Zero Trust as a foundational pillar of their AI strategy — rather than an afterthought layered onto systems already in production — will be the ones that deploy agentic AI safely, at scale, with organizational confidence and regulatory standing. The "never trust, always verify" mandate has never been more important than when applied to systems capable of reasoning their way around every barrier you place in front of them.

References

[1] National Institute of Standards and Technology. NIST SP 800-207: Zero Trust Architecture. August 2020. https://www.paloaltonetworks.com/cyberpedia/what-is-nist-sp-800-207

[2] Hardy, N. The Confused Deputy (or Why Capabilities Might Have Been Invented). 1988. Cited in: Quarkslab. "Agentic AI: The Confused Deputy Problem." 2026. https://blog.quarkslab.com/agentic-ai-the-confused-deputy-problem.html

[3] MITRE. ATLAS (Adversarial Threat Landscape for Artificial-Intelligence Systems) v5.4.0. 2025. https://atlas.mitre.org/

[4] OWASP. Top 10 for LLM Applications 2025. November 2024. https://www.promptfoo.dev/blog/owasp-top-10-llms-tldr/

[5] Lakera AI. "Indirect Prompt Injection: Attack Mechanics and Defenses." 2026. https://www.lakera.ai/blog/indirect-prompt-injection

[6] Hack The Box. "CVE-2025-32711 EchoLeak: Zero-Click Vulnerability in Microsoft 365 Copilot." 2025. https://www.hackthebox.com/blog/cve-2025-32711-echoleak-copilot-vulnerability

[7] Vectra AI. "Critical Prompt Injection CVEs 2025–2026." 2026. https://www.vectra.ai/topics/prompt-injection

[8] Palo Alto Networks Unit 42. "In-the-Wild Indirect Prompt Injection: First Confirmed Weaponization." March 2026. https://unit42.paloaltonetworks.com/ai-agent-prompt-injection/

[9] Vectra AI. "ZombieAgent: Persistent Memory Injection." 2026. https://www.vectra.ai/topics/prompt-injection

[10] CrowdStrike. "AI Tool Poisoning: Attack Variants and Defenses." 2026. https://www.crowdstrike.com/en-us/blog/ai-tool-poisoning/; "How Agentic Tool Chain Attacks Threaten AI Agent Security." https://www.crowdstrike.com/en-us/blog/how-agentic-tool-chain-attacks-threaten-ai-agent-security/

[11] CyberDesserts. "AI Agent Security Risks 2026: OpenClaw Crisis, MCP Exposures, and Supply Chain Attacks." 2026. https://blog.cyberdesserts.com/ai-agent-security-risks/

[12] Microsoft Security Blog. "Case Study: Securing AI Application Supply Chains — CVE-2025-68664 LangGrinch." January 2026. https://www.microsoft.com/en-us/security/blog/2026/01/30/case-study-securing-ai-application-supply-chains/

[13] The Register. "Rogue AI Agents Worked Together to Escalate Privileges, Bypass DLP, Devise Steganography." March 2026. https://www.theregister.com/2026/03/12/rogue_ai_agents_worked_together/

[14] OWASP. "OWASP GenAI Exploit Round-Up Report Q1 2026." April 2026. https://genai.owasp.org/2026/04/14/owasp-genai-exploit-round-up-report-q1-2026/

[15] Bessemer Venture Partners. "Securing AI Agents: The Defining Cybersecurity Challenge of 2026." 2026. https://www.bvp.com/atlas/securing-ai-agents-the-defining-cybersecurity-challenge-of-2026

[16] Reddit/cybersecurity. "The Confused Deputy Problem in Multi-Agent AI: LangGraph Demonstration." 2026. https://www.reddit.com/r/cybersecurity/comments/1rty805/the_confused_deputy_problem_in_multiagent_ai/

[17] PromptArmor. "Data Exfiltration from Slack AI via Indirect Prompt Injection." August 2024. https://www.promptarmor.com/resources/data-exfiltration-from-slack-ai-via-indirect-prompt-injection

[18] Stellar Cyber. "Top Agentic AI Security Threats: 45,000-Record Financial Services Exfiltration." 2026. https://stellarcyber.ai/learn/agentic-ai-securiry-threats/

[19] HashiCorp. "SPIFFE: Securing the Identity of Agentic AI and Non-Human Actors." 2026. https://www.hashicorp.com/en/blog/spiffe-securing-the-identity-of-agentic-ai-and-non-human-actors

[20] Huang, Y., Narajala, V., Yeoh, W., Ross, M. et al. "A Novel Zero-Trust Identity Framework for Agentic AI Systems." arXiv:2505.19301. May 2025. https://arxiv.org/abs/2505.19301

[21] arXiv. "MiniScope: Automated Least Privilege Framework for Tool-Calling Agents." arXiv:2512.11147. December 2025. https://arxiv.org/abs/2512.11147

[22] Red Hat. "Zero Trust for Autonomous Agentic AI Systems: Building More Secure Foundations." February 2026. https://next.redhat.com/2026/02/26/zero-trust-for-autonomous-agentic-ai-systems-building-more-secure-foundations/

[23] Palo Alto Networks. "Prisma AIRS 3.0: End-to-End Agentic AI Lifecycle Security." March 2026. https://www.paloaltonetworks.com/company/press/2026/palo-alto-networks-secures-agentic-ai-with-prisma-airs-3-0

[24] OWASP. Top 10 for Agentic Applications 2026. December 2025. https://genai.owasp.org/resource/owasp-top-10-for-agentic-applications-for-2026/

[25] arXiv. "Agentic AI as a Cybersecurity Attack Surface: The Viral Agent Loop." 2026. https://arxiv.org/html/2602.19555v1

[26] arXiv. "HIPAA-Compliant Agentic AI Framework: ABAC, PHI Sanitization, and Immutable Audit Trails." April 2025. https://arxiv.org/html/2504.17669v1

[27] Cloud Security Alliance / MassiveScale.AI. "The Agentic Trust Framework: Zero Trust Governance for AI Agents." February 2026. https://cloudsecurityalliance.org/blog/2026/02/02/the-agentic-trust-framework-zero-trust-governance-for-ai-agents

[28] Microsoft. "Security Copilot for SOC: Bringing Agentic AI to Every Defender." 2025. https://techcommunity.microsoft.com/blog/microsoftthreatprotectionblog/security-copilot-for-soc-bringing-agentic-ai-to-every-defender/4470187

[29] CrowdStrike. "Charlotte AI Detection Triage: 98% Accuracy, 40+ Hours Saved Weekly." February 2025. https://www.crowdstrike.com/en-us/press-releases/crowdstrike-delivers-next-breakthrough-in-ai-powered-agentic-cybersecurity-with-charlotte-ai-detection-triage/

[30] Google Cloud. "Agentic AI Orchestration for Security Operations Workflows." Architecture reference. April 2026. https://docs.cloud.google.com/architecture/agentic-ai-orchestrate-security-ops-workflows

[31] deepc. "deepc Advances Healthcare AI with Enterprise Agentic AI Platform." RSNA 2025 announcement. https://www.deepc.ai/news/deepc-advances-healthcare-ai-with-enterprise-agentic-ai-platform-offering

[32] McKinsey & Company. "How Agentic AI Can Change the Way Banks Fight Financial Crime." August 2025. https://www.mckinsey.com/capabilities/risk-and-resilience/our-insights/how-agentic-ai-can-change-the-way-banks-fight-financial-crime

[33] Kore.ai. "Agentic AI in Financial Services: Fraud Detection and KYC." 2026. https://www.kore.ai/blog/agentic-ai-in-fsi

[34] arXiv. "Regulatory Frameworks for AI in Financial Services: Four-Layer Governance." December 2025. https://arxiv.org/html/2512.11933v1

[35] NIST CAISI. "Announcing the AI Agent Standards Initiative: Interoperable and Secure." February 2026. https://www.nist.gov/news-events/news/2026/02/announcing-ai-agent-standards-initiative-interoperable-and-secure

[36] Kiteworks. "AI Swarm Attacks: GTG-1002 Chinese State-Sponsored Campaign Analysis." 2025. https://www.kiteworks.com/cybersecurity-risk-management/ai-swarm-attacks-2026-guide/

[37] GitHub / Microsoft. "GitHub Copilot Enterprise Security Architecture and Secret Scanning." 2025. https://learn.microsoft.com/en-us/microsoft-365/copilot/security-microsoft-365-copilot

[38] Cloud Security Alliance. "CSA Research Note: NIST AI Agent Red-Teaming and Standards." March 2026. https://labs.cloudsecurityalliance.org/research/csa-research-note-nist-ai-agent-red-teaming-standards-202603/

[39] Microsoft. "Microsoft Extends Zero Trust to Secure the Agentic Workforce." May 2025. https://www.microsoft.com/en-us/security/blog/2025/05/19/microsoft-extends-zero-trust-to-secure-the-agentic-workforce/

[40] HashiCorp. "Zero Trust for Agentic Systems: Managing Non-Human Identities at Scale." 2026. https://www.hashicorp.com/en/blog/zero-trust-for-agentic-systems-managing-non-human-identities-at-scale

[41] Anthropic. "Claude Mythos Preview: Safety Evaluation." April 7, 2026. https://red.anthropic.com/2026/mythos-preview/

[42] UK AI Security Institute. "Evaluation of Claude Mythos Preview Cyber Capabilities." April 2026. https://www.aisi.gov.uk/blog/our-evaluation-of-claude-mythos-previews-cyber-capabilities

[43] Anthropic. "Disrupting Malicious Uses of AI: GTG-1002." November 2025. https://www.anthropic.com/news/disrupting-AI-espionage

[44] CIS / Cequence / Astrix. "CIS Controls v8.1 Model Context Protocol Companion Guide." April 20, 2026. https://www.cisecurity.org/insights/white-papers/controls-v8-1-model-context-protocol-companion-guide

[45] Cequence Security. "AI Gateway: Secure Agentic AI Deployments." 2026. https://www.cequence.ai/products/ai-gateway/

About the Author

Dr. Chase Cunningham, widely known as "Dr. Zero Trust," is a retired U.S. Navy Chief Cryptologist, former Forrester analyst, and one of the most recognizable voices in modern cybersecurity. He built the Zero Trust Extended (ZTX) Framework that helped push Zero Trust from buzzword to baseline strategy for enterprises and government. Today he advises vendors, boards, and agencies on how to turn Zero Trust into an operational reality. Chase is the author of multiple books on cyber warfare and Zero Trust, host of the Dr Zero Trust show (https://www.patreon.com/cw/drzerotrust), a regular presence on conference stages worldwide, and heads up the DrZeroTrust Research Division.