SANS | Research Program

2026 EDITION

Agentic Security:

Detection and Response at Machine Speed

Written by:
- Gee Rittenhouse
- Eric Johnson
- Ashley Nelson
- Matt Meck
- Kyle Shields

In partnership with:

aws

August 2026

About the Authors

Gee Rittenhouse
Vice President of Agentic Security at AWS

Gee Rittenhouse is the Vice President of Security Services at AWS, overseeing key services including Security Hub, GuardDuty, and Inspector. He holds a PhD from MIT and brings extensive leadership experience across enterprise security and cloud. He previously served as CEO of Skyhigh Security and Senior Vice President and General Manager of Cisco's Security Business Group, where he was responsible for Cisco's worldwide cybersecurity business.

Eric Johnson
Fellow at SANS Institute; Principal Security Engineer at Puma Security

Eric Johnson brings a background in software development, cloud automation, and applied security into the classroom. He is a Fellow and Principal Security Engineer at Puma Security, and the lead author and instructor for SEC540: Cloud Native Security and DevSecOps Automation at the SANS Institute, where he focuses on securing cloud-native and DevSecOps environments in practice. He also co-authors and teaches SEC549: Cloud Security Architecture and SEC510: Cloud Security Engineering and Controls, offering a full spectrum of cloud security education from design to controls to continuous automation.

Ashley Nelson
Senior Worldwide Security Specialist at AWS

Ashley Nelson is a Senior Worldwide Security Specialist at AWS. She works directly with enterprise security leaders to accelerate cloud security posture and build the operational muscle to secure AI workloads in production.

Matt Meck
Worldwide Security Specialist at AWS

Matt Meck is a Worldwide Security Specialist at AWS. He works across enterprise customers and product teams to build detection and response capabilities that keep pace with rapidly evolving threats.

Kyle Shields
Security Specialist Solutions Architect at AWS

Kyle Shields is a Security Specialist Solutions Architect focused on threat detection and incident response at AWS. Today, he's focused on helping enterprise AWS customers adopt and operationalize AWS Security Incident Response and improve their security posture.

Introduction

Organizations across every industry are evaluating and adopting autonomous AI agents. These agents authenticate on behalf of users or with preauthorized user consent, execute multistep workflows, and make decisions across infrastructure, often without waiting for human approval. This requires organizations to rethink security posture, threat detection, and incident response. Threats now move at machine speed, and security operations must match that pace.

This chapter provides a framework for security leaders navigating this transition, whether your organization is just evaluating agentic AI, running a pilot, or operating at scale. It covers agent identity and governance, how detection needs to evolve, how to design responses that balance speed with precision, and how your existing security foundations extend to this new world.

The good news is that agentic security is not a blank slate. It builds on the same principles security teams already know, including identity governance, least privilege, defense in depth, and backup and recovery, and adapts them for workloads that are autonomous and probabilistic. You are not starting over; you are extending what already works.

The Rise of Agentic Workloads

AI workloads have evolved through four distinct stages, each introducing cumulative security requirements, meaning controls from earlier stages still apply and new controls layer on top (see Figure 1):

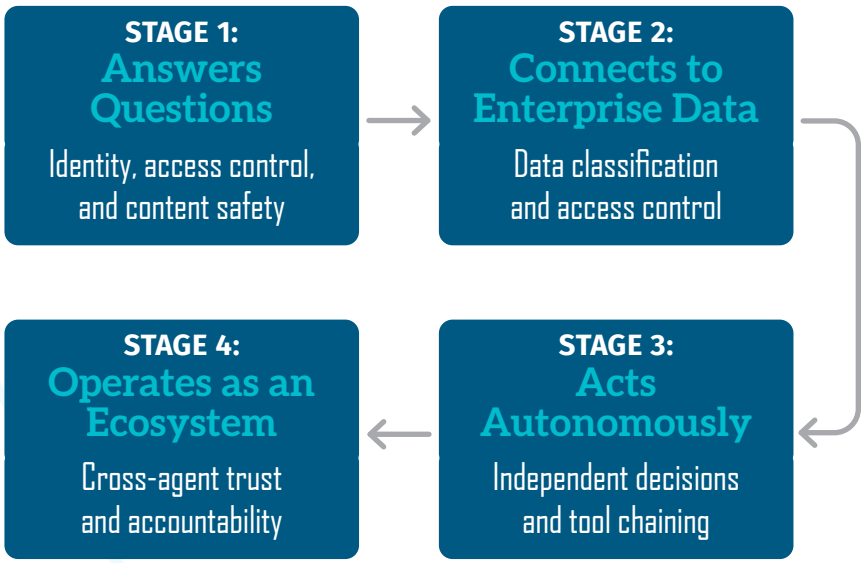

Figure 1. Four Stages of Progression

1. **Answers questions—**Chat agents and summarizers generate responses from foundation models with no external data connections. The security focus here centers on identity, access control, content safety, and monitoring. Even at this stage, prompt injection is the top threat to AI applications according to the OWASP Top 10 for Large Language Model Applications.[1]
2. **Connects to enterprise data—**Retrieval-augmented generation (RAG) systems and knowledge bases access documents, databases, and APIs to generate grounded responses. Every query becomes an implicit access request against the data estate. The security focus expands to data classification, fine-grained access control, and output validation.
3. **Acts autonomously—**Agents take actions on behalf of users: processing transactions, modifying records, executing code, and coordinating across systems. They make independent decisions, chain actions together, and operate across multiple tools and data sources to complete complex objectives.
4. **Operates as an ecosystem—**Multiple specialized agents collaborate on shared objectives, delegating subtasks to one another, negotiating access, and coordinating across organizational boundaries. No single human orchestrates the workflow.

[1] https://owasp.org/www-project-top-10-for-large-language-model-applications

Agents discover other agents, negotiate trust, and compose into teams dynamically. This raises new security questions about how organizations govern trust between agents built by different teams or across organizational boundaries, how they trace accountability when a chain of 10 agents produces an outcome, and how they prevent privilege accumulation as agents delegate to one another. These questions are not hypothetical, and early versions of multiagent systems are already running in production today.

Agents are becoming more autonomous, more connected, and more capable, which means what you design today needs to hold up not just for single agents but for the multiagent ecosystems that are already forming.

Why This Matters Now

Agents are being built by an expanding population of builders, including nontraditional developers using low-code tools, creating governance challenges that your existing security programs can extend to address. The industry response has been uneven. Although 80% of organizations have adopted AI, only 10% govern it.[2] The gap between adoption velocity and security maturity is widening. For additional context on securing and governing AI at the right layers, refer to the AWS AI Security Framework blog post.[3]

Each stage of agentic evolution inherits every security requirement that came before it, so organizations securing today's simple chat agents are already laying the foundation for tomorrow's multiagent ecosystems.

[2] "The state of AI in early 2024: Gen AI adoption spikes and starts to generate value," May 2024, www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai-2024

[3] "The AWS AI Security Framework: Securing AI with the right controls, at the right layers, at the right phases," May 2026, https://aws.amazon.com/blogs/security/the-aws-ai-security-framework-securing-ai-with-the-right-controls-at-the-right-layers-at-the-right-phases

Understanding Agentic Behavior and Identity

Securing agentic workloads starts with understanding how they behave, because the properties that make agents powerful are the same ones that create new security exposure. Agents are probabilistic, adaptive, and autonomous, and each of those characteristics requires a different security response than traditional workloads demand.

Probabilistic, Adaptive, and Autonomous

Traditional workloads are deterministic, but agentic workloads operate on fundamentally different principles (see Figure 2):

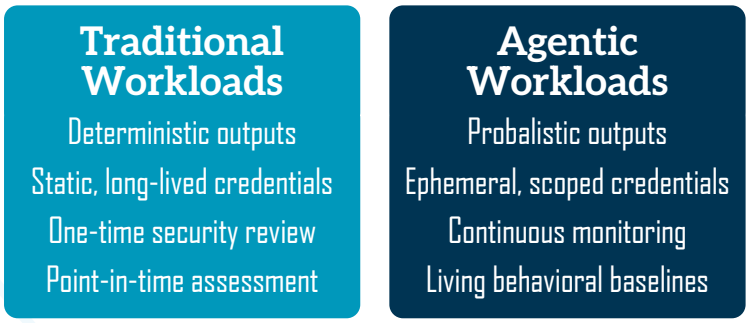

Figure 2. Traditional vs. Agentic Workloads

1. **Same input produces different outcomes.** The same prompt can produce a compliant response on one request and a policy-violating response on the next. This requires continuous monitoring rather than point-in-time assessment.
2. **Prompts contain both user input and instructions.** A prompt is natural-language text that instructs the model to perform a task, which makes it possible to craft misleading or malicious prompts that manipulate the model's behavior to bypass safety filters and execute unintended instructions. Prompt injection exploits this by embedding hidden instructions in user input. Mitigating this requires input validation, content classification, and output validation at every AI endpoint.
3. **Agents adapt over time.** Agents learn and modify their behavior as they interact with users, data, and tools over time, so a one-time security review at launch is necessary but not sufficient. Continuous monitoring and behavioral baselines are required to detect modified and learned behaviors adopted by agents as they interact with users, data, and tools over time.

Agentic workloads break the assumptions traditional security was built on: Because outcomes are probabilistic and identities are nonhuman, security controls need to operate continuously and at machine speed rather than as one-time gates.

4. **Agents have autonomy and agency.** They connect to APIs, tools, and data, making independent decisions. Every agent must be scoped with least-privilege permissions determined by what the user the agent is acting on behalf of is allowed to access, with authorization enforced independently of the model.

Existing threat models were built for deterministic systems and likely don't account for probabilistic outputs, prompt injection, or the autonomous decision-making that defines agentic workloads, which is why organizations need to revisit their threat modeling assumptions before deploying agents at scale.

Nonhuman Identity as a First-Class Consideration

Every agent needs its own identity with temporary, scoped credentials with scoped access, rather than sharing a credential that allows broad access. This allows organizations to extend Zero Trust principles to AI agents. Every request must be authenticated and authorized independently, and every action needs a traceable authorization chain. Meeting this standard requires:

- **Ephemeral credentials—**Agents receive temporary credentials with scoped access rather than persistent access. Identity life cycle management must operate at machine speed.
- **Independent authorization—**Authorization is enforced independently of the model and scoped to least privilege.
- **Context-aware permissions—**An agent acts on behalf of a user, so its permissions should be scoped to only the data that user can access—adapting further based on the task, the data being accessed, and the calling context. For example, an agent handling a routine data lookup for an authorized user operates with read-only access to a single data source. If the same agent is asked to modify records or access a different data source, the policy engine evaluates the new request independently, requiring additional authorization or escalating to a human checkpoint based on the sensitivity of the action.
- **Multitenant awareness—**When an agent serves multiple users or teams, security teams need to know who each agent serves so they can scope the roles it assumes per user. This helps ensure the agent never operates with broader access than the current user is entitled to.

The High-Risk Trifecta

When a single agent combines access to sensitive data, the ability to communicate externally, and exposure to untrusted content, the risk profile changes significantly. That convergence makes the agent a potential vector for unintended data exposure or manipulation. Design patterns that prevent any single component from combining all three go a long way toward reducing that risk (see Figure 3).

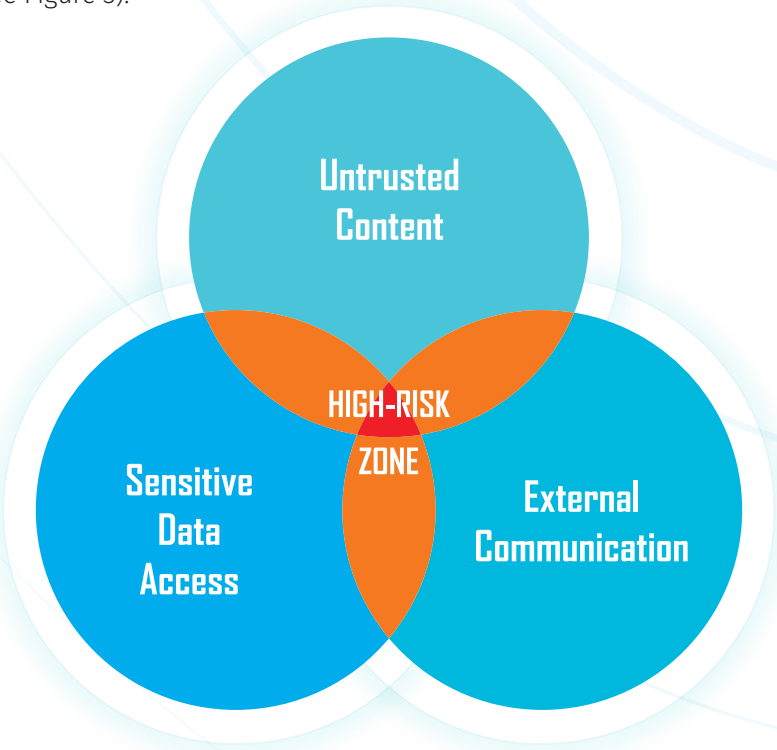

Figure 3. The High-Risk Trifecta

Evolving Detection for Machine-Speed Workloads

Agentic workloads move faster than human-reviewed alerts can track, which means detection has to evolve from periodic assessment toward continuous, instrumented observation. That shift affects how telemetry is collected, how anomalies are defined, and how security teams measure their own effectiveness against an expanding attack surface.

Converging Security and Application Telemetry

Security teams and platform teams must share a common data plane to see agent behavior end-to-end. In agentic environments, the boundary between application telemetry and security telemetry dissolves. An API call that looks routine from an operations perspective may represent a scope violation from a security perspective, and without unified logging and shared context between teams, that distinction disappears. Instrumentation needs to ship with every agentic workload by default, not be added after the fact.

Agentic security isn't about catching bad actors. It's about measuring, in minutes, how long an autonomous system can operate outside its boundaries before anyone notices.

Attack Surface Minutes as a New Metric

Traditional security metrics measure whether a vulnerability exists, but agentic security demands measuring *how long* a vulnerability is exploitable before detection and response close the gap. The concept is analogous to dwell time in traditional security, applied here to the window of exploitability rather than attacker presence. Attack surface minutes quantify this window, capturing the time between when a vulnerability becomes exploitable and when controls contain it (see Figure 4). That measurement becomes the primary metric for agentic-era detection because it drives architectural decisions that matter most, including when to shift from periodic scanning to continuous monitoring, when batch alerting needs to become streaming detection, and when manual triage should evolve into automated containment.

Figure 4. Attack Surface Minutes

Behavioral Baselines for Autonomous Workloads

Existing user behavior analytics do not transfer to agentic workloads. Agent traffic patterns, API call sequences, and resource access cadences require purpose-built models. Key considerations include:

- **Legitimate anomalies—**AI coding tools generate rapid, multiprocess patterns that appear anomalous to legacy detection systems. Baselines must account for legitimate agentic activity.
- **Dynamic behavior—**Agents adapt and evolve, so baselines must be living models rather than static thresholds.
- **Context-dependent normalcy—**An agent's "normal" behavior varies by task, time, and calling context.
- **Minimum baseline period—**Instrument highest-risk agents first and collect at least 30 days of baseline data before tuning detection rules.

Detection in agentic environments isn't about catching a bad actor. It's about recognizing when an autonomous system has exceeded its operational boundaries, regardless of whether the cause is compromise, misconfiguration, or emergent behavior.

Automated Response: Speed with Precision

When detection operates at machine speed, response must match it. Human-reviewed alerts and manual containment workflows were designed for a pace that agentic workloads have already exceeded. Response architecture must be deliberate, predefined, and calibrated to act before the window of impact widens.

The Response Continuum

Agentic-era response operates on a continuum calibrated by confidence thresholds and potential impact, moving between containment, escalation, and closing the loop depending on what the situation needs. Traditional approval gates fail in this context because, unlike a human user, an agent will not pause to reconsider when presented with a warning. Circuit-breaker models replace that assumption, triggering automatic suspension when threshold violations occur rather than waiting for a human to confirm the decision.

Confidence-Calibrated Escalation

Confidence-calibrated escalation matches human involvement to the actual risk of the decision. Table 1 maps three representative scenarios to the appropriate response, from full autonomy at the low-risk end to mandatory human checkpoints and automatic containment where confidence is low or impact is high.

Not every anomaly requires intervention, and not every legitimate action needs a human in the loop (see Figure 5).

Scenario	Action
Low-impact action + high confidence of legitimacy	Full autonomy (no human required)
High-impact action + any confidence level	Mandatory human checkpoint before execution
Behavioral anomaly + low confidence	Automatic containment, human review before release

Table 1. Matching Escalation Level to Action Risk and Confidence

Figure 5. The Response Continuum

The right response isn't always escalation, and it isn't always containment. It's matching the two to actual risk, automatically, before a human could react in time.

Defense in Depth for Agent Response

Effective agent response doesn't rely on a single control. When a threat is detected, four independent layers activate in parallel, each targeting a different dimension of the agent's operational footprint:

1. **Identity layer—**Revoke agent credentials, suspend sessions
2. **Network layer—**Block egress traffic to unauthorized destinations, isolate the workload
3. **Application layer—**Disable tool access, freeze state
4. **Data layer—**Restrict data access, enable enhanced logging

Builders should design for the "pit of success" principle: Where the secure path is the default path, detection feeds directly into automated containment, recovery paths are predefined and tested, and the system degrades gracefully when humans are unavailable (see Figure 6).

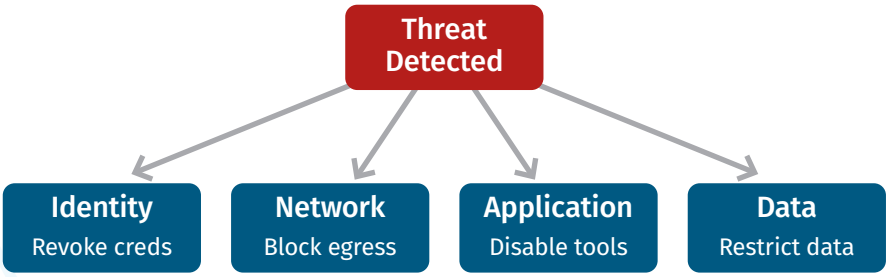

Figure 6. Defense-In-Depth Agent Response

Regulatory Observability

Complete audit trails of every agent action, decision, and escalation must be architected from day one and stored in integrity-checked, persistent, secured storage. Retrofitted for compliance after the fact is not a viable strategy. That means organizations need to be able to reconstruct the full decision chain from triggering prompt to action taken, demonstrate who authorized the agent's permissions, demonstrate that controls were operating at the time of any incident, and show that response was proportionate and timely. Immutable audit logging across all layers is a baseline requirement, and agents should be held to the same logging and traceability rigor as any identity or principal in the ecosystem.

Practical Security Considerations

This is not a wholesale departure from what security teams already know. Least privilege, defense in depth, data protection, and continuous monitoring all still apply. What changes is how you apply those principles when the actor is an autonomous system instead of a person.

The following are the seven questions we hear most often from security leaders along with their answers:

- **How do I prevent the agent from surfacing PII in its responses?** Output validation guardrails are the primary control, and they work independently of the model. Every agent response should pass through content filters that catch and redact PII before it reaches the user, covering names, email addresses, phone numbers, financial identifiers, and any data classified as sensitive. Even if the model generates PII in its response, the guardrail catches it. The model should never be the sole control between sensitive data and the user. A deterministic validation layer enforces that boundary consistently regardless of what the model produces on any given request.

- **How do I make sure users only see information they're authorized to access?** Authorization must be enforced at the data layer, not the prompt layer. Telling an agent in its system prompt to "only show data the user has permission to see" is not a security control. Prompts can be bypassed, ignored, or overridden, which means the underlying data query must be scoped to each user's permissions at retrieval time. If a user cannot see a record through the normal app interface, the agent should not be able to pull it either. That requires integrating the agent's data access into an existing RBAC or ABAC system and filtering results before they ever enter the model's context window.

- **Agents can now delete data. How do I protect against that?** When an agent can write to or delete from production systems, unintended data loss is a real possibility, whether from a bad prompt, adversarial input, or unexpected behavior. The answer is not to avoid giving agents write access (that undermines their usefulness) but to reduce the likelihood that a single failure causes irreversible damage. Immutable backups should live in storage the agent's credentials cannot reach, destructive operations should require explicit policy approval (so nothing deletes without matching allow rule), and high-impact actions should route to a human checkpoint no matter what. If one control fails, the others still hold.

> **The model is never the control. Whether it's redacting PII, scoping data access, or approving a tool call, every safeguard here has to work whether or not the agent cooperates.**

- **How do I contain the damage if an agent is compromised or misconfigured?** Each agent should run in its own isolated environment with private subnets that have no internet access, a dedicated identity scoped to its current task, and access limited to only the data stores it needs. That isolation helps prevent a compromised agent from reaching other agents, other data, or other systems, keeping the scope of any single failure bounded and predictable. Stateless containers that spin up fresh every invocation further limit what an attacker can access or persist across sessions.

- **How do I know if an agent is doing something it shouldn't?** Build behavioral baselines and set circuit breakers. Traditional monitoring watches for known-bad patterns, but agent monitoring also must watch for deviation from known-good ones. That means establishing what "normal" looks like for each agent and covering which tools it calls, how frequently, in what sequences, and against what data, then alerting when behavior deviates from that baseline. For high-risk anomalies, automated circuit breakers suspend the agent immediately rather than waiting for a human to review an alert queue. The agent's full session history (every prompt, every tool call, every response) should be archived in immutable storage so that any incident can be reconstructed forensically after the fact.

Telling an agent to only show authorized data in its system prompt isn't a security control. It's a suggestion, and prompts can be bypassed, ignored, or overridden, which is why authorization has to be enforced at the data layer instead.

- **What about the tools and APIs my agent connects to?** How do I establish trust across those connections? Treat every external tool as an untrusted input surface. Agents that call external APIs, MCP servers, or third-party tools inherit the security posture of those dependencies, and a compromised tool can feed malicious instructions back to the agent. Maintaining a registry of approved tools and servers, validating outputs from external tools before the agent acts on them, and using signed artifacts with version pinning help verify that the tool an agent calls today is the same one that was evaluated and approved. Each tool's permissions also should be scoped narrowly, so a tool that only needs read access never gets write access simply because it was easier to configure that way.
- **Do I need a different approach for every tool call an agent makes?** Yes. In traditional applications, an authorized user can generally perform any action within their role. In agentic systems, each individual tool call should be evaluated against a policy engine. The pattern is analogous to cloud IAM, where nothing is allowed unless explicitly permitted. The policy considers what tool is being called, what data it will access or modify, whether the action is consistent with the agent's current objective, and the potential impact if the action goes wrong. Default-deny at the tool invocation layer is the most critical architectural pattern for agentic security. Frameworks like Cedar[4] and Open Policy Agent[5] make it practical to implement at scale.

The Underlying Principle

Across all these considerations, the same pattern emerges. Agentic security extends what security teams already know, and the practices that protect against data loss, unauthorized access, and insider threats in traditional environments still apply. What changes is the speed at which agents act (milliseconds rather than the minutes implied by attack surface minutes) the scale at which they operate (thousands of actions per session), and the need for deterministic controls that don't rely on the model's cooperation. Layered defenses, the assumption that any single control can fail, and a default-secure architecture are the constants that hold as the agentic landscape evolves.

[4] https://cedarpolicy.com/en
[5] www.openpolicyagent.org

Building Security That Matches the Speed of Innovation

Organizations evaluating or deploying agentic workloads should assess their readiness across five domains (see Figure 7):

1. **Instrumentation—**Every agentic workload ships with observability by default: traces, logs, and behavioral telemetry. Start by extending existing logging to cover agent API calls and tool invocations.
2. **Identity governance—**Nonhuman identities receive ephemeral credentials and are subject to automated life cycle management. Start by inventorying all agents and mapping their credential models.
3. **Behavioral baselines—**Purpose-built detection models flag anomalous agent activity. Start by instrumenting highest-risk agents and collecting 30 days of baseline data.
4. **Response playbooks—**Automated containment systems trigger confidence-calibrated escalation. Start by identifying agents' highest-consequence actions and building circuit breakers.
5. **Supply chain governance—**Every tool, API, and agent-to-agent interaction is monitored and governed. Start by auditing external connections and registering agents in a central registry.

Figure 7. The Five Readiness Domains

The Compounding Principle

Security controls compound as workloads evolve from prototype to production to scale. And, in agentic environments, thousands of actions can execute in seconds so the cost of getting that wrong is not hypothetical. Organizations with a high level of ungoverned "shadow AI" paid an additional $670,000 per breach on average[6]. Investing in controls early costs far less than remediating after an incident.

Organizations begin by extending existing controls to AI through configuration changes, then layering in threat detection and AI-specific monitoring, and ultimately automating governance and incident response at scale. Each phase builds on the previous one rather than replacing it.

Security has to move at the same speed as the workloads it protects, across five domains that compound as agents scale from prototype to production.

Evolving Team Structures

Security must become as adaptive and fast as the workloads it protects. This means rethinking team structures so security and platform teams share responsibility for agent observability, shifting to continuous detection over periodic assessment, evolving metrics from static vulnerability counts to attack surface minutes, and developing analysts who understand probabilistic systems. The opportunity is to redesign processes around what agentic security enables.

[6] "Cost of a Data Breach Report 2025," IBM Security, www.ibm.com/think/x-force/2025-cost-of-a-data-breach-navigating-ai

A Note on Open Research

Several challenges in agentic security are under active research across the industry. This is expected in a rapidly evolving field and does not mean organizations should delay adoption. It means organizations should implement defense in depth (so no single unsolved problem leads to compromise), design architectures that can incorporate new controls as they emerge, and stay connected to the communities doing this work. Adopting a common security telemetry schema, such as the Open Cybersecurity Schema Framework (OCSF), makes agent activity portable across tools and ready for AI-driven security operations.[7]

For the most current details on open research areas and emerging mitigations, stay connected with organizations such as:

- Coalition for Secure AI (CoSAI) (coalitionforsecureai.org)[8]
- OWASP Agentic AI Security (genai.owasp.org)[9]
- Frontier Model Forum (frontiermodelforum.org)[10]
- MITRE ATLAS (atlas.mitre.org)[11]

These resources evolve faster than any single publication can. Bookmark them, revisit them quarterly, and use them to inform your security program as the field matures.

[7] "How OCSF Powers Agentic Security Operations," March 2026, https://aws.amazon.com/blogs/opensource/ocsf-achieves-itu-support-powering-ai-ready-security-operations
[8] www.coalitionforsecureai.org
[9] https://genai.owasp.org
[10] www.frontiermodelforum.org
[11] https://atlas.mitre.org

Recommended Next Steps

Regardless of where an organization sits on the agentic AI adoption curve, the following actions provide a clear starting path.

Understand the Landscape

The following resources provide ongoing guidance:

- Coalition for Secure AI (CoSAI) publishes governance frameworks and landscape papers that define the security challenges unique to agentic systems.
- OWASP Agentic AI Threat Model provides a comprehensive taxonomy of attack vectors and mitigations specific to autonomous agents.
- Frontier Model Forum drives cross-industry collaboration on safe development of frontier AI.
- MITRE ATLAS maintains a knowledge base of adversary tactics targeting AI systems, analogous to ATT&CK for traditional threats.

These resources provide teams with a shared vocabulary that enables better decision-making.

Join the Community

Agentic security is evolving rapidly, and no single organization has all the answers. Engage with CoSAI working groups, follow OWASP AI publications, and participate in industry forums where practitioners share lessons learned. The challenges described in the Open Research Area section are being actively worked by these communities. Staying connected means security programs benefit from collective progress rather than solving everything alone.

Understand What Is Available

Familiarize teams with the tools and frameworks that already exist. The AWS Strands Agents SDK[12] is an open source framework for building agents with built-in observability and guardrails integration. Cedar and Open Policy Agent offer policy-as-code frameworks for implementing default-deny authorization at the tool invocation layer. Cloud-native services for guardrails, content filtering, and agent observability exist today and plug into your existing security operations.

For Organizations with Workloads Already Deployed

Organizations with agents already running or under active development should begin by taking inventory of what exists, what data those agents access, and what actions they can take. The highest-risk agents can be identified by looking for the convergence of sensitive data access, external communication, and untrusted input exposure. Existing IAM controls should be extended to cover those agents, which in most cases means configuration changes rather than architecture changes. Output validation guardrails should be added to agents with the broadest access, incident response playbooks should be updated to include AI-specific scenarios, and policy-based authorization with implicit deny and explicit allow should be implemented for tool invocations on the most critical workloads first.

AI is being embedded into every layer of infrastructure, every application, and every enterprise workflow. Security must lead the charge, not follow. Organizations that build this foundation now are building the security architecture for whatever comes next. Security teams that enable AI adoption do not say no to AI. They say: *This is how you do it securely.*

The goal isn't to say no to AI.
It's to say: This is how you do it securely.

---

[12] "Introducing Strands Agents, an Open Source AI Agents SDK," May 2025, https://aws.amazon.com/blogs/opensource/introducing-strands-agents-an-open-source-ai-agents-sdk

SANS | Research Program

About the SANS Research Program

The SANS Research Program is a key initiative by the SANS Institute and a premier global provider of cybersecurity research and information. SANS Research Program is designed to provide cybersecurity practitioners and leaders with data-driven insights, thought leadership, and solutions that help them better understand and respond to evolving security challenges. All content is authored by SANS instructor experts from around the world who apply their years of experience from hands-on practitioner work in the field, advisory roles, and the classroom to provide education, guidance, and actionable insights that help make the cyber world a safer place.

To learn about sponsorship opportunities for research, content, and in-person or virtual events, email us at **Sponsorships@sans.org** or go to **www.sans.org/sponsorship**.

Free Resources

Access more free educational content from SANS at:

SANS Reading Room
www.sans.org/white-papers

Check out upcoming and on-demand webcasts:

X
x.com/SANSInstitute

LinkedIn
www.linkedin.com/company/sans-institute

YouTube
www.youtube.com/@SANSInstitute