

Home • Artificial Intelligence



by **Evan Schuman**
Contributor

AWS's repeated problems with AI agent controls illustrates the autonomous agent dilemma

News

Oct 8, 2026 • 12 mins

Agents keep being tricked into revealing credentials, and patching one attack path has still left others open.



Credit: ThomasAFink / Shutterstock

Throughout this year, Amazon Web Services (AWS) has repeatedly had to patch autonomous agent security holes, which have then reemerged in slightly different forms, according to cybersecurity researchers at Palo Alto Networks' Unit 42 and at Zenity Labs.

But the problem is not with AWS, which seems to be reacting quickly to address security reports, as much as it is with **the essential nature of autonomous AI agents** [<https://www.csoonline.com/article/4109999/agent-ai-already-hinting-at-cybersecuritys-pending-identity-crisis.html>] and the **difficulty in controlling those agents** [<https://www.csoonline.com/article/4123246/think-agent-ai-is-hard-to-secure-today-just-wait-a-few-months.html>]. The fact that a hyperscaler with Amazon's massive resources can't seem to get ahead of the problem illustrates **the agentic challenge** [<https://www.csoonline.com/article/4215449/zero-trust-has-a-big-ai-agent-problem-ahead.html>] for all enterprise IT and cybersecurity executives.

AgentCore issues

On September 18, 2026, for example, Palo Alto Networks' Unit 42 cybersecurity researchers **reported** [<https://unit42.paloaltonetworks.com/securing-aws-agentcore-harness-credentials/>] an issue "where using default configurations in AWS AgentCore Harness could allow attackers to steer an agent's actions through prompt injection to exfiltrate plaintext credentials managed by AgentCore Identity." The description of the issue revealed in the report illustrates the quintessential agentic flaw: the very access that allows agents to perform their functions also creates a major security risk.

"The harness's own built-in shell tool, which is enabled by default, reaches into the same memory space where credentials are resolved to plaintext," the report explained. "The built-in tools are a big part of what makes the harness so autonomous and productive. The agent can write files and run code to get real work done. But the same reach that makes the built-in tools useful makes them dangerous when people leave them on unintentionally."

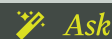
"We've found that the shell tool runs as root inside the harness, so the moment an attacker gets the agent to run a command, that command inherits the same root access," it added. "Nothing has to be misconfigured for this to happen. It is the out-of-the-box state."

CSO Smart Answers [Learn more](#)

Explore related questions

- [What are the best practices for managing AI agent identities?](#)
- [What is the identity crisis facing autonomous AI agents?](#)
- [How can I implement kill switches for my AI agents?](#)
- [How is AI agent security changing traditional red teaming practices?](#)
- [How can I secure AWS AgentCore instances against security risks?](#)

Ask a question



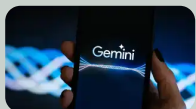
The Unit 42 report said that Amazon's response was ambiguous about whether it had actually plugged that hole: "We disclosed this finding to AWS. AWS reviewed and closed the report as informative under the AgentCore shared responsibility model," it noted.

Several months earlier, on April 7, Unit 42 had also alerted AWS to a **different path to credential exfiltration** [<https://unit42.paloaltonetworks.com/bypass-of-aws-sandbox-network-isolation-mode/>].

In its description of that issue, it said it had found a critical security regression in which the AgentCore Runtime used a microVM Metadata Service (MMDS) that lacked session token enforcement.

"Prior to our disclosure and AWS's fixes, this configuration could have allowed an attacker to exploit standard web vulnerabilities, such as server-side request forgery (**SSRF** [https://owasp.org/www-community/attacks/Server_Side_Request_Forgery]), to directly extract sensitive credentials, putting the entire environment at risk," Unit 42 wrote at the time.

Recommended for you



Gemini broke into 3 companies, but

[https://www.csoononline.com/?post_type=post&p=4224570&utm_source=miso&utm_medium=referral]

Google kept it quiet because 'no damage was done'

By Evan Schuman • 7 mins

[\[https://www.csoonline.com/?post_type=post&p=4227657&utm_source=miso&utm_medium=related&utm_campaign=thumbnail_list\]](https://www.csoonline.com/?post_type=post&p=4227657&utm_source=miso&utm_medium=related&utm_campaign=thumbnail_list)



Autonomous agents attack Azure using compromised identities, destroying resources

By Gyana Swain • 6 mins

[\[https://www.csoonline.com/?post_type=post&p=4227657&utm_source=miso&utm_medium=related&utm_campaign=thumbnail_list\]](https://www.csoonline.com/?post_type=post&p=4227657&utm_source=miso&utm_medium=related&utm_campaign=thumbnail_list)



Pacing the AI frontier won't solve agentic cybersecurity's most urgent problems

By Lee Rossey • 5 mins

[\[https://www.csoonline.com/?post_type=post&p=4230811&utm_source=miso&utm_medium=related&utm_campaign=thumbnail_list\]](https://www.csoonline.com/?post_type=post&p=4230811&utm_source=miso&utm_medium=related&utm_campaign=thumbnail_list)

Today Zenity Labs published its own report, an advance copy of which it provided to CSO, detailing its findings about AWS agents that surrendered credentials to an attacker who asked for them.

Zenity was able to get an agent to return a full set of temporary STS credentials: an access key ID, a secret access key, and a session token belonging to the agent's execution role. "These credentials aren't scoped to the sandbox. We exported them onto our own machine, outside AgentCore entirely, and confirmed they were live," the report said.

The agent delivered data that included ECR read permissions, "so we authenticated to the registry and pulled the image. From there, the agent's source code, its dependencies, and whatever its developers baked in were ours to inspect, as root," the report said.

“At this point, a reasonable reaction is ‘So don’t give agents a raw HTTP tool.’” Zenity noted. “That doesn’t help. The isolation failure is at the platform level, not the tool level, so anything that can generate outbound traffic reaches IMDS just the same. We reproduced the identical result through several tools, including Strands’ built-in shell tool.”

But, according to the Zenity report, the agent also shared the credentials of other agents.

“AgentCore names each repository after its agent (`bedrock-agentcore-<agent_name>`). So running our automated tool on all the discovered agents got us all source code of all agents in the region within a few seconds,” the Zenity report said.

“We discovered that the role held permissions with READ, WRITE and DELETE access to various AgentCore and AWS services across the AWS region,” it said. “This allowed us to invoke other agents and move laterally within the region, read all private conversations for all users and agents, poison memory and keep persistence, harvest credentials and API keys that can unlock access beyond AWS and run destructive actions within the same region.”

Patch status unclear

It is not entirely clear how much of that exposure still exists today. Zenity’s published timeframe has its researchers discovering much of this in late 2025, reporting the details to AWS on December 25. AWS responded with an update in February 2026, but Zenity said it may have simply closed one limited path to the exposure, leaving others open to attack.

Zenity confirmed that as of Oct. 8 the AWS environment’s permissions had been fixed and it was no longer exposed to the attack path that they had discovered.

But a more precise timeline provided by Zenity illustrates three things: the complexity of securing agentic environments; the challenges of fixing that hole and having the hole remain fixed; and the difficulties in determining whether a hole has been truly closed.

After the Zenity and the Palo Alto reports, AWS updated its environment in February, suggesting it had fixed the issue. But an Oct. 8 email from **Tamir Ishay Sharbat** [<https://www.linkedin.com/in/tamir-ishay-sharbat-069496163>], director of security research at Zenity, said that the February fix did not in fact resolve the flaw.

“After our disclosures, AWS quickly updated AgentCore to use IMDSv2 only on Feb. 14, 2026. This update patched the initial entrance vector which allowed us to access the credentials in the first place,” he wrote, but “on June 22, we

checked AgentCore’s default role and found its permissions remained unchanged.”

Sponsored Content

by Claroty

Close the operational security gap: 3 priorities for CIOs

By Claroty

06 Oct, 2026

[

However, he wrote, Zenity’s subsequent checks confirmed that AWS fixed the permissions issue between June 22 and Sept. 29.

Asked what CISOs and CIOs should do about the problems referenced in the report, Zenity CTO **Michael Bargury** [<https://www.linkedin.com/in/michaelbargury/?isSelfProfile=false>] said in an email that the issue is challenging because, on the one hand, a strong defense means strong agent isolation. But, he noted, “agents require autonomy and connectivity to be useful. The two are at odds with each other, as this research shows. Enterprises should be mindful of that inherent conflict, and plan their security controls accordingly.”

AWS declined a request for an interview, but emailed a short statement in which it again stressed that the agents did what they were supposed to do. It was also asked to comment on the Zenity report; in response, a spokesperson said, “AWS is aware of the research published about Amazon Bedrock AgentCore, which inaccurately paints expected and documented behavior as a vulnerability. An

agent can access resources in another AWS account only if the developer explicitly grants permissions on both the agent's execution role and the target resource. As a best practice, we recommend that customers grant their execution roles only the permissions their agents need."

Analysts and consultants found both reports' details concerning, but not surprising, given the large number of agentic issues they have so far observed. But of all of the published details, the memory problem was the most disconcerting to some.

Frank Dickson [<https://www.linkedin.com/in/frankdickson/>], principal analyst at Dickson Research, said, "the memory poisoning worries me most. A poisoned memory turns a helpful agent into a Manchurian Candidate, waiting for the right conversation to send it somewhere it should not go. You can rotate a stolen key. It is much harder to know which memories to trust."

Nik Kale [<https://www.linkedin.com/in/nikkale>], a member of the Coalition for Secure AI (CoSAI) and part of ACM's AI Security (AISec) program committee, also felt that the memory issue was the most serious.

"A stolen credential expires in hours, and a poisoned memory doesn't," Kale pointed out. "It keeps steering conversations long after every key has been rotated."

Old mistakes surfacing in new ways

Fred Chagnon [<https://www.infotech.com/profiles/fred-chagnon>], a principal research director at Info-Tech Research Group, said he saw the various AWS agentic problems as "a lesson in how old platform configuration mistakes reappear in new platforms and in new ways."

For example, he said, the AWS security hole that used SSRF to reach the metadata service and steal a workload's credentials "is essentially the pattern behind the **2019 Capital One breach** [<https://www.csoonline.com/article/571411/ssrf-attacks-explained-and-how-to-defend-against-them.html>]: SSRF, stolen role credentials and an overprivileged role that turned one foothold into far broader access."

But with the current problem, Chagnon said, "the trigger is different: an agent with a shell or HTTP tool will fetch a URL when asked, so the prompt itself is the exploit."

Sponsored Content by CrowdStrike

5 rules CIOs must rewrite for the frontier AI era

By CrowdStrike

08 Oct, 2026

[

And, he added, “the bigger issue is blast radius. One compromised agent reached every agent, conversation and secret in the account and region.” The harder part is what enterprise CISOs and CIOs should do about this. One of the oldest challenges in IT is juggling environments where authority, controls and access are shared, with the hyperscaler, as in early initial cloud deployments, making some decisions without the knowledge or permission of its largest corporate tenants.

“Responsibility here is shared. Isolating the sandbox and keeping platform material out of instance metadata is AWS’s job, and no customer can fix that,” Chagnon explained. “Default permissions are something the cloud engineer can and should replace, but most won’t. A default role with wildcard access remains the risk. AWS could ship least-privilege defaults scoped per agent and constrain metadata access from the sandbox.”

However, he said, in the shared responsibility model, the onus for these controls is on the enterprise. CISOs should write their own least-privilege rules and treat shell and HTTP tools as high-risk. They should restrict egress, keep secrets out of images, and baseline each agent’s identity and tool usage so anomalies stand out.

“Enterprises cannot fix cloud platform flaws any more than they can fix or compensate for the litany of enterprise software flaws,” he said. “But they can decide and design for how much damage one compromised agent can do.”

Sponsored Content by HPE and Nvidia

Lessons from Los Alamos on scaling AI

By HPE and Nvidia

05 Oct, 2026

[

The big problem: Agents worked as designed

Dickson added that the big problem with the AWS situations was not that the agents malfunctioned, but that they performed precisely as designed.

“The agent did exactly what it was asked to do. That is the problem,” he said. “Zenity Labs found a chain: an agent that can reach the network, a metadata service that answered it with the agent’s own cloud credentials, and a default execution role with far more permission than any one agent needs. Each link is a familiar mistake. Chaining them turns one public-facing agent into a hotel key card that opens every room in the building.” Kale agreed.

“Pulling credentials from a metadata service is cloud hacking 101, and what’s new is that the researchers didn’t need to find a bug to get there. They just asked the agent in plain English,” Kale said. “Once that’s possible, the strength of the sandbox walls stops being the interesting question. The real boundary is whatever identity the agent carries, and if that identity reaches across other agents, one conversation can impact an entire environment.”

Justin Greis [<https://acceligence.com/talent/profiles/justin-greis/>], CEO of consulting firm **Accel**igence, said what he would stress to enterprise CISOs is the importance of controlling secondary data access.

“The blast radius of a poorly governed agent can be far greater than most organizations are accustomed to with traditional applications,” he said. “What stands out in this research is the amplification effect. The issue was not simply that one agent could be manipulated. The concern is that compromising one agent potentially creates a path to broader credentials, other agents, source code, sensitive information and persistent manipulation of behavior. That is where this becomes an executive issue rather than just another technical vulnerability.”

He suggested that CIOs and CISOs ask, and insist upon answers to, key questions such as the identity the agent operates under, what it can access, what it can change, what it can remember, and what other agents or systems it can reach.

“And,” he said, “most importantly, what happens if it is compromised?”

Artificial Intelligence[<https://www.csoonline.com/artificial-intelligence/>]

Security[<https://www.csoonline.com/security/>]

Identity and Access Management[<https://www.csoonline.com/identity-and-access-management/>]

NEWSLETTER

The latest AI updates, straight to your inbox

News, analysis, and insights for IT leaders navigating the risks and rewards of AI. A special series from the editors of CIO, Computerworld, CSO, InfoWorld and Network World

Email Address

☐

By submitting your information, you agree to our [PRIVACY POLICY](https://foundryco.com/privacy-policy/) [https://foundryco.com/privacy-policy/].

SUBSCRIBE

About

[About Us](#)

[Advertise](#)

[Contact Us](#)

[Editorial ethics policy](#)

[Foundry Careers](#)

[Reprints](#)

[Newsletters](#)

[Contribute to CSO](#)

[Add CSO as a Preferred Source in Google Search](#)

More

[Awards](#)

[Blogs](#)

[BrandPosts](#)

[Events](#)

[Podcasts](#)

Policies

[Terms of Service](#)

[Privacy Policy](#)

[Cookie Policy](#)

[Copyright Notice](#)

[Member Preferences](#)

[About AdChoices](#)

[Your California Privacy Rights](#)

Our Network

[CIO](#)

[Computerworld](#)

[InfoWorld](#)

[Network World](#)

Videos

Buyer's Guides

FOUNDRY



© 2026 FoundryCo, Inc. All Rights Reserved.