CODEHUNTER®
ZERO TRUST FOR CODE

Behavioral Risk Brief

# AI Data Thieves

July 8, 2026

## The Claim

Governance frameworks that treat marketplace listing or basic scanning as sufficient validation create systemic risk when agentic systems execute functionality without independent behavioral verification. Zero Trust for Code addresses this by enforcing behavioral controls at execution, ensuring that what an AI skill or agent-based artifacts does is verified rather than assumed from where it was found.

## The Incident

ESET's H1 2026 Threat Report found that nearly 900,000 AI agent skills that were analyzed, more than 25,000 were flagged as suspicious and over 3,000 were confirmed malicious, with the malicious count growing from roughly 600 to over 3,000 in a three-month window. These skills carried capabilities including command execution, credential loading, code injection, and obfuscation, the same functions that support legitimate agent tasks. Researchers also found that some skills marketed as security scanners performed only superficial checks, giving operators a false sense of protection. Separately, the report documented a new technique called AI-fix, a variant of the ClickFix pattern that uses fake troubleshooting pages hosted on services associated with Anthropic, OpenAI, and Microsoft to convince users to run malicious commands, contributing to a 108 percent year over year rise in ClickFix detections overall.

## The Governance Failure

The governance failure is not the existence of malicious skills, but the absence of controls that validate what a skill will do once an agent is permitted to invoke it. Organizations evaluating agentic AI tools often rely on marketplace presence, download counts, or a passed scan as indicators of safety. These signals confirm that a skill was reviewed once, not that its behavior remains within acceptable bounds every time it executes.

This challenge comes from how agent frameworks are built. Skills inherit the permissions of the agent that invokes them, and agents frequently operate with broad access to credentials, files, and external systems in order to complete tasks. A skill that behaves as documented during review can still execute commands, access credentials, or move data in ways no one explicitly authorized, because the framework does not distinguish between intended task execution and unauthorized action.

The underlying breakdown is the lack of enforceable policy governing what a skill is allowed to do at the moment it runs. Scanning at intake tells an organization what a skill contains. It does not tell them what the skill will attempt once it has the agent's permissions and a live task in front of it.

As agent adoption accelerates, this distinction becomes the whole of the problem. An organization can document every skill it approved and still have no record of what those skills actually did once deployed. Trust assigned at intake cannot substitute for authorization enforced at execution.

## The Regulatory and Business Exposure

- Unauthorized data access and exfiltration through agent-executed skills.
- Credential theft and malware execution inside agent workflows, bypassing controls built for software installs.
- Social engineering exposure through AI-fix pages abusing trusted AI platform domains.

## What Your Auditor Will Ask

- How do you validate the behavior of an AI skill or plugin before an agent is permitted to invoke it?
- How do you distinguish a skill that performs a genuine security check from one that only appears to?
- What controls detect an agent executing commands, accessing credentials, or transferring data outside its intended task?
- How do you prevent a fake troubleshooting or verification page from inducing a user or an agent into running unauthorized commands?
- What evidence do you maintain that agent-executed skills were evaluated against policy before running, rather than reconstructed after an incident?

A consistent signal is the disconnect between marketplace listing and skill behavior. Skills that pass basic scanning continue to carry actions that were never evaluated against what is consider acceptable execution.



CODEHUNTER

## Zero Trust for Code Value

Zero Trust for Code introduces enforcement at the point where an AI skill or agent-executed artifact actually runs, ensuring that capabilities such as command execution, credential access, or code injection are evaluated against defined policy regardless of whether the skill passed a marketplace scan or carries a security label.

This directly addresses the governance weakness exposed by the growth in malicious AI skills: the assumption that a passed scan or marketplace presence indicates safe behavior going forward. By enforcing behavioral constraints at execution, organizations can prevent a skill from exfiltrating data, loading credentials, or executing unapproved commands even when nothing about it was flagged during initial review.

The result is a governance model where agent behavior is evaluated against policy every time it executes, not inherited from a one-time scan or platform trust, holding a skill's actual execution to the same standard as its claimed function throughout its use.

## Governance Action Brief

- Establish governance controls that validate AI skill behavior independently of marketplace listing or scan status.
- Require continuous verification of command execution, credential access, and data movement performed by agent-executed skills.
- Enforce execution policies for AI agents, browser extensions, and automation tooling that consume third-party skills.
- Monitor social engineering patterns, such as fake troubleshooting pages, that pressure users or agents into running unauthorized commands.
- Treat agentic AI environments as high-value systems requiring behavioral enforcement at execution, not only at admission.

## Sources

Analysis based on Help Net Security reporting (July 8, 2026) on ESET's H1 2026 Threat Report covering malicious AI skills and ClickFix variants, CodeHunter Labs evaluation of governance gaps in agentic AI execution and marketplace trust models



CODEHUNTER