oktsec

The Complete Guide to AI Agent Security

Executive Brief — March 2026

Gustavo Aragón · Oktsec · March 2026

1. The problem: 3 million agents, almost no security

If your team uses Cursor, Claude Code, Copilot, or any AI coding assistant, you are running AI agents in production. Those agents read files, execute shell commands, call external APIs, and modify local configuration. Most do this without security review.

An estimated 3 million AI agents operate within US and UK enterprises today (Gravitee, 2026). Only 14.4% of organizations report their agents going live with full security approval. The other 85.6% deploy without formal review or runtime enforcement. 88% of organizations experienced at least one AI-related security incident in the past year (CSA, 2026).

This is not a future risk. It is a current one.

AI agents break the assumptions traditional security depends on. Dependencies are resolved at runtime, not build time: an agent decides which tools to call based on natural language reasoning, and the same prompt can trigger different tool sequences on consecutive runs. These dependencies never appear in any manifest or lockfile. Compromised agents spread through language, not code: a poisoned message to one agent becomes instructions for the next. No exploit code needed. The attack propagates at API speed through shared documents, emails, and calendar invites. Tool descriptions are the new unsigned, unversioned, mutable dependencies. An attacker who changes a tool description redirects agent behavior without touching any code.

Most teams treat prompt injection as the problem. It is Stage 1 of 7. Analysis of 36 documented incidents shows that 21 attacks crossed four or more stages of a structured kill chain (Schneier et al., arXiv: 2601.09625). The attackers escalated, persisted, spread, and completed their mission. If you only defend Stage 1, you have six unprotected stages.

Stage 1: Initial Access. Prompt injection. 93.3% success rate against Cursor in auto-approve mode.

Stage 2: Privilege Escalation. GTG-1002 used persona claims ("we are conducting defensive testing") to bypass Claude's safety training.

Stage 3: Reconnaissance. The agent maps its own infrastructure on request. No dedicated mitigations exist in any production system analyzed.

Stage 4: Persistence. SpAIware: one interaction permanently compromises ChatGPT's behavior via memory poisoning.

Stage 5: C2. ZombAI: ChatGPT instances join a C2 network using memory and web browsing.

Stage 6: Lateral Movement. Morris II: a self-replicating prompt achieves 90%+ replication through 11 hops.

Stage 7: Actions on Objective. GTG-1002: autonomous cyber espionage across 30 organizations. 80-90% of operations handled autonomously.

The kill chain gives defenders seven chances to stop an attack, not one.

2. What we find: 58,000 skills scanned daily

Aguara Watch continuously scans every public AI agent skill registry. 58,000+ skills monitored across 7 registries, using 188 detection rules across 15 threat categories. Scanned 4 times daily.

Severity	Findings
CRITICAL	163
HIGH	792
MEDIUM	752
LOW	15,975+

95.7% of skills score Grade A. That sounds reassuring until you look at the tail: 587 skills score C or below, 81 score F. Each represents a skill that could exfiltrate data or redirect tool calls. In production registries. Right now.

30 MCP CVEs filed in the past 60 days. 38% of scanned MCP servers lack authentication entirely. 87% of AI-generated pull requests contain at least one security vulnerability (DryRun Security, March 2026). Repositories using AI coding assistants show a 40% higher secret leak rate than baseline (GitGuardian). 23.8 million secrets leaked on public GitHub in 2024.

The supply chain is the fastest-growing attack vector. SANDWORM_MODE deployed 19 malicious npm packages that install rogue MCP servers into Claude Code and Cursor, with a 48-hour activation delay designed to evade install-time scanning. The OpenClaw crisis exposed 135,000+ instances with 93% lacking authentication. Snyk found 36.82% of ClawHub skills contain security flaws with 76 confirmed malicious payloads.

3. What to do: defense-in-depth

No single control stops the full kill chain. You need three layers.

Layer 1: Scan before deployment. Scan every MCP configuration, tool description, and skill file before the agent runs. Catch poisoned descriptions, hardcoded credentials, unpinned dependencies.

```
aguara scan --auto --severity high
```

This discovers 17 MCP client configurations and scans them all. One command. Under a minute.

Layer 2: Isolate at runtime. One container per MCP server. Drop all capabilities (`--cap-drop ALL`). Read-only filesystem. Restrict network egress to allowlisted domains. If the agent cannot reach `~/.ssh/` or `169.254.169.254`, those attack paths fail.

Layer 3: Enforce every tool call. Intercept every tool call before execution. Scan against detection rules. Assign a verdict: clean, flag, quarantine, or block. Log everything with cryptographic signatures. The agent does not get to decide if its own actions are safe. External infrastructure makes that determination.

Your Monday morning checklist

Pin your MCP server versions. Open every config, add version numbers. 30 seconds per server.

Run `aguara scan --auto` on your machine. Right now. It finds exposed credentials, unpinned dependencies, and risky tool descriptions.

Check if your MCP servers are exposed to the internet. If they are on a public IP, firewall them.

Install pre-commit hooks for secret scanning. `brew install gitleaks` and `pre-commit install`. Five minutes.

Resources

The full 56-page guide covers the complete kill chain, OWASP Top 10 for Agentic Applications, credential defense playbooks, supply chain case studies, and 7 implementation checklists. Free download at oktsec.com/ai-agent-security/

Aguara Scanner: Open-source static security scanner. 188+ rules, 5 analysis engines. github.com/garagon/aguara

Aguara Watch: Continuous threat observatory. 58,000+ skills across 7 registries. watch.aguarascan.com

Oktsec: Runtime security for AI agents. MCP Gateway with per-agent identity, tool-level policies, tamper-evident audit trail. One command: `oktsec run`. oktsec.com

All data current as of March 15, 2026. Every claim backed by a CVE, academic paper, or named incident. Copyright 2026 Oktsec.