CLOUD SECURITY PARTNERS

Securing AI WITH AI

Review our extensive blog articles on each chapter for more information.

cloudsecuritypartners.com

cloud-security-partners

Table of Contents

"**Skills** are verbs.

Agents are roles."

CH. 3, Skills & Agents Page 16

"**AI writes insecure code by default.** Here are the 5 vulnerabilities it ships **every time.**"

CH. 4, Generating Secure Code Page 21

BONUS

AI Claude skill reviewer
Free on GitHub

38 Page

"13.4% of audited AI skills ship a critical security flaw."
AI Skills Supply Chain Page 34

BONUS

AI Security Maturity Self-Assessment
Level 0 to 3

47 Page

BONUS

The 30-point AI Security Checklist
Exclusively at the end of this eBook

54 Page

CLOUD SECURITY PARTNERS

Introduction

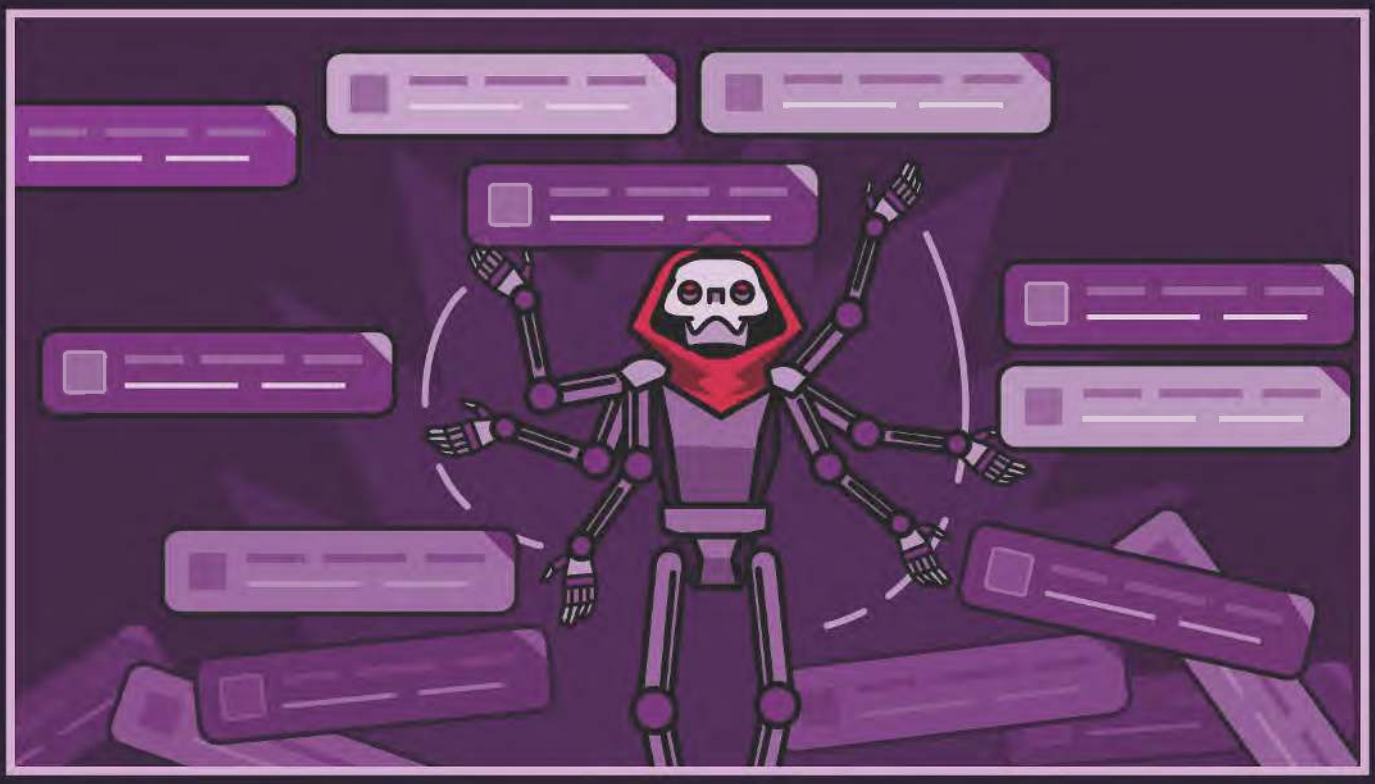
AI-powered development has changed everything in software engineering. Code creation is no longer a bottleneck in deploying software; ideas can go from concept to production within minutes rather than weeks. However, the same speed that transforms development makes it challenging to secure. Traditional security tools and processes were built for deterministic systems and human-paced workflows. Nondeterministic AI agents write code, make decisions, and execute actions at an unprecedented scale, requiring new tools and processes to secure them.

At Cloud Security Partners, we've worked with organizations across every industry that are navigating this shift. From startups to established businesses, we've seen the same pattern; AI adoption is outpacing the security programs meant to govern it:

- **Engineers are using AI assistants** without any formal security policy
- **AI skills marketplaces are proliferating** faster than anyone can vet them
- **Code is being generated faster** than traditional review cycles can keep up

This ebook is a practical guide for security teams preparing for the AI-powered future. We cover the current state of AI security tooling, how skills and agents are reshaping security workflows, the emerging risks of agents and skills, how to generate secure code, and how to build a comprehensive AI security program from the ground up.

Whether your organization is vibe coding everywhere or just beginning to experiment with AI assistants, the principles and frameworks in this ebook will help you move fast without leaving security behind.



Illustrated by Jeff Prymowicz

Chapter 1

The AI Security PARADOX

CLOUD SECURITY PARTNERS

Written by
Mike McCabe

View Original Blog Post

“It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness,”

Charles Dickens' 1859 novel
A Tale of Two Cities

Engineering and security are in an unprecedented time; we're at a crossroads of incredible promise but immense uncertainty. There are tools that can turn ideas into code immediately, compressing release timelines from weeks to minutes. There are tools that can audit huge codebases, compressing multi-hour code reviews to seconds. Code creation is no longer the limiting factor for new platforms and products.

However, this transformation has many consequences for security. The security community has long struggled to build tools and processes for deterministic human systems, with mixed results and developer pushback. Now, we are adding another challenge:

> **nondeterministic systems that make decisions and execute actions at a superhuman speed.**

Even shifting left into CI/CD pipelines is too slow and too late. This requires an entire paradigm shift.

Security must be integrated into the code creation process like never before.

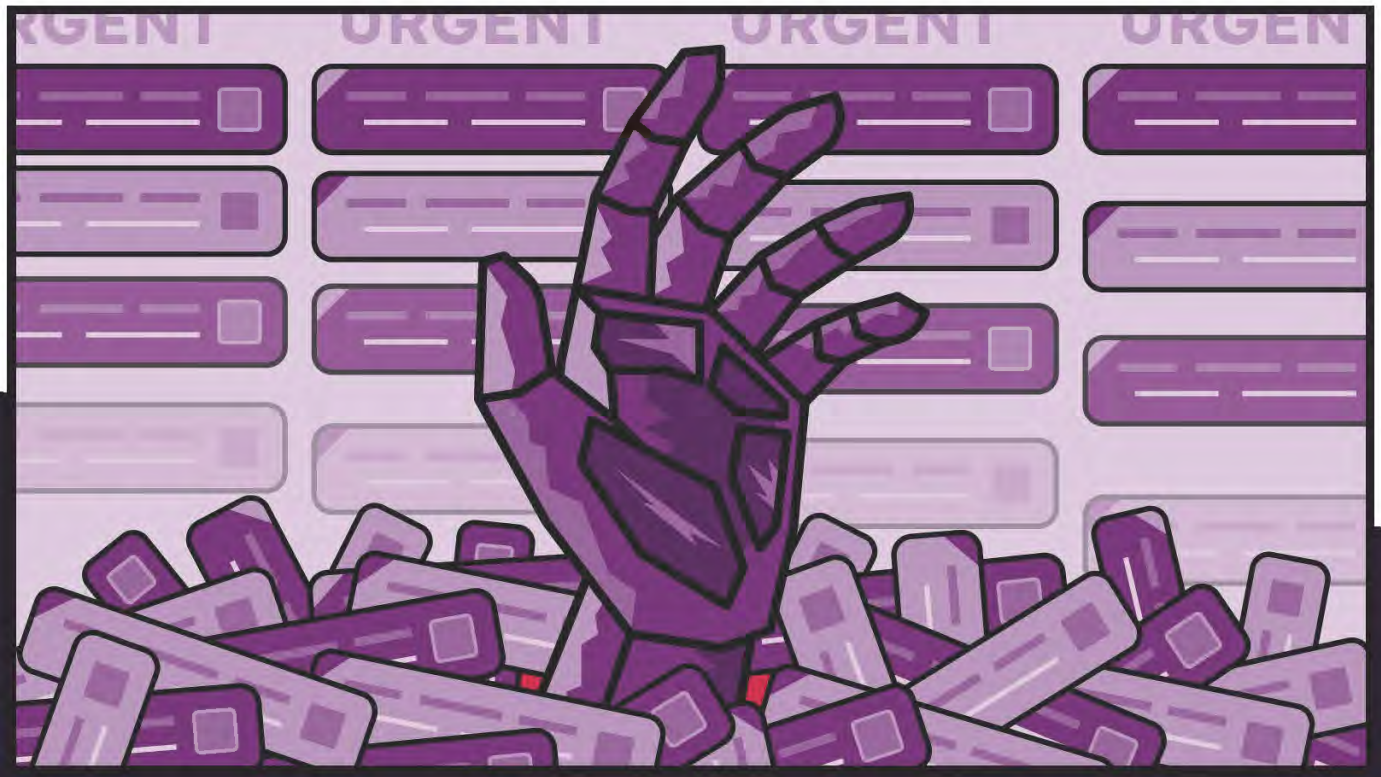
For security teams, this moment cuts two ways:

1. Teams can shape their own tooling ecosystems.

Security teams can leverage AI and agents to do more work. AI lets security teams build and scale. Tools that previously had to be bought can now be built in minutes. Teams can also scale their expertise by deploying agent teams to perform reviews, create documentation, and triage alerts.

2

Teams must reason about an entirely new tooling, workflows, and content that the business creates and uses.

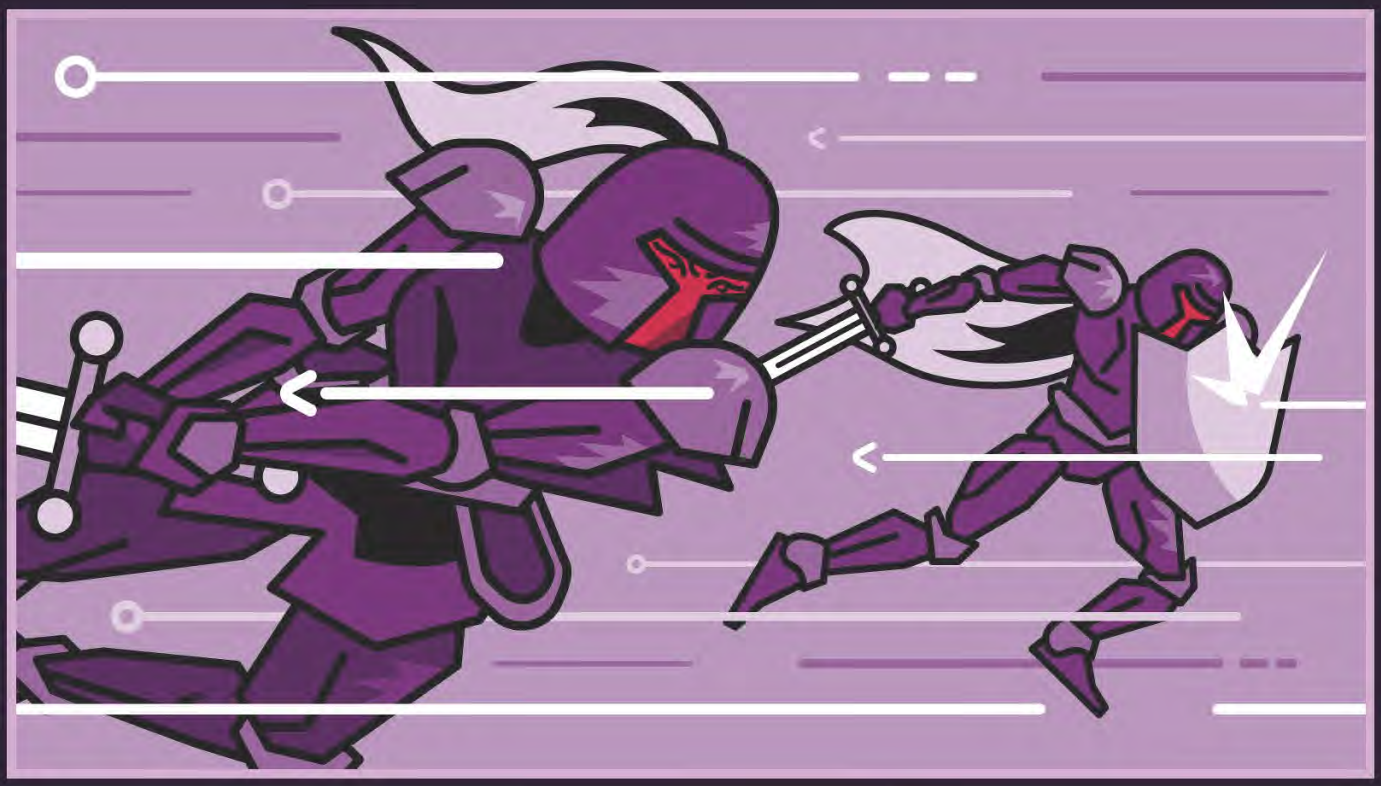


Security teams are being asked to do more, with more, at a much higher speed than ever. The speed of content generation, tool adoption, and workflows has increased to a point where security teams must define security models and protocols before the business moves past them.

Teams must secure not only applications, networks, and cloud environments, but also secure AI infrastructure and agents themselves.

The path forward is not simply adopting AI and absorbing the consequences.

It is being intentional: defining where agents augment your team versus where they add invisible overhead, and setting security models before the business has already moved past you.



Security has always been a discipline of thinking several moves ahead. Being deliberate now means understanding what tools are at your disposal, how they actually work, and where they fall short

—before betting your program on them.

Chapter 2

The AI Security Tooling LANDSCAPE

CLOUD SECURITY PARTNERS

Written by
Alex Lynch

View Original Blog Post

The same technology that is expanding attack surfaces is also giving security teams new ways to analyze code, reason about risk, and scale their work. Attackers are already using AI to develop exploits, defenders need to be empowered by the same class of tools, not left relying on processes designed for a slower era.

Today, there are three different types of defensive AI security tooling:

- **General-purpose coding agents** like Codex and Claude Code that were not built for security, but are being used for security work. These tools assist human analysis.
- **AI-augmented security tooling** that layers an AI model into an existing detection workflow. These tools combine AI with traditional signals like static analysis, rule engines, or policy enforcement. They integrate with existing security tools by explaining findings, prioritizing, and reasoning about issues that fall outside standard pattern matching.
- **Agentic security systems** that attempt more autonomous work. These tools are purpose-built for running multi-step code review, live application assessment, and even long-running analysis workflows. These tools push toward fully autonomous security agents.

While AI tooling is being developed for many security solutions, our research indicates that, currently, **certain use cases are best suited to AI:**

Code review

Many models are proficient at reading code, tracking intent across files, explaining suspicious behavior, and helping reviewers investigate issues that do not reduce neatly to a single known pattern.

Threat modeling assistance

AI is well-suited to first-draft work: enumerating abuse cases, surfacing trust boundaries, proposing architectural questions, and helping teams turn vague concerns into something reviewable.

Policy and documentation drafting

Security teams spend a surprising amount of time converting expertise into artifacts other people can consume: secure coding guidance, review checklists, internal standards, decision logs, and explanatory notes. LLMs are strong because the task is less about perfect precision than about quickly producing a credible first draft that an expert can refine.

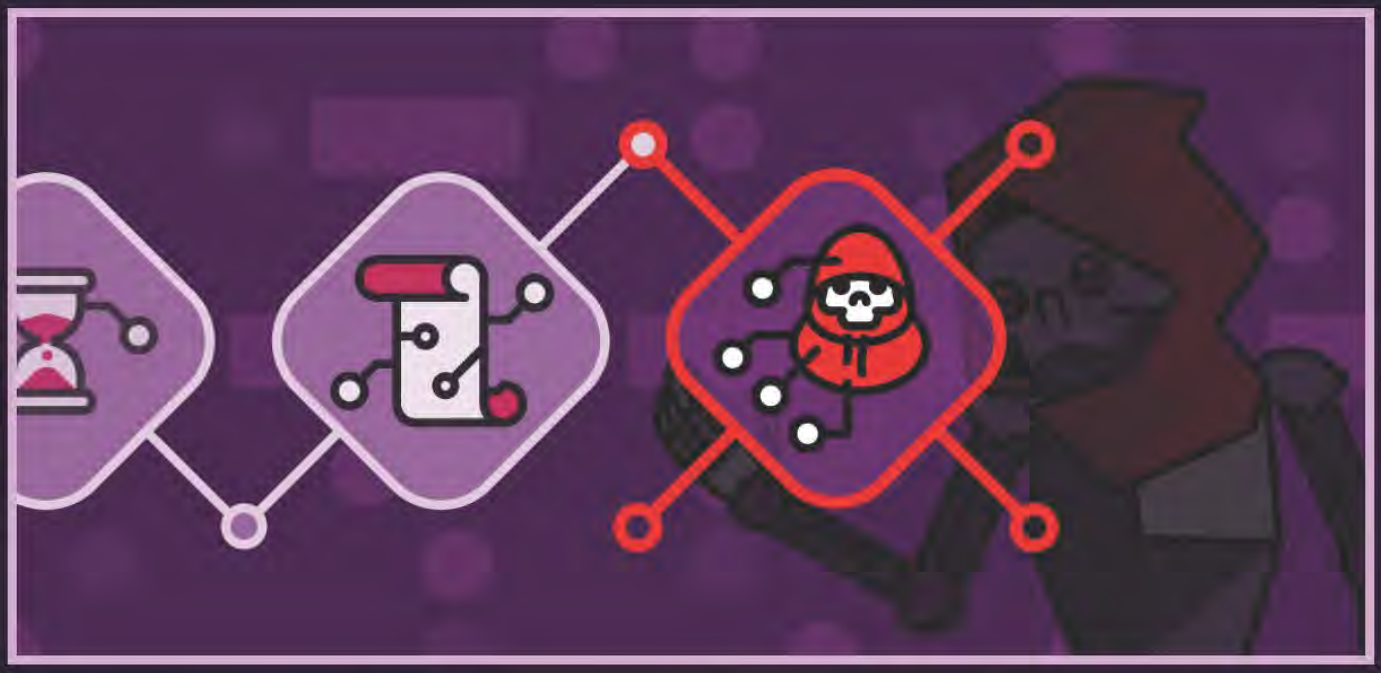
Incident triage support

AI can summarize evidence, connect findings across sources, help frame likely impact, and generate useful next questions.

Current AI models are best at reasoning over code and text, synthesizing context, and accelerating expert interpretation.

Consider a pull request that adds a new tenant-level authorization check to a SaaS application: traditional SAST may catch obvious issues like unsafe query construction or hardcoded secrets, while an AI-augmented security tool can review the

changed files, explain the intended control flow, and flag suspicious patterns such as missing tenant scoping or inconsistent permission checks. A general-purpose coding agent can then help the reviewer investigate related files, summarize how authorization works elsewhere in the application, and draft focused questions for the developer.



Fully autonomous agentic security systems remain less mature solutions, as they may not be reliable enough to trust as an outright security control.

How all of these tools compare with deterministic tooling is a question that we address later in this book.

Chapter 3

Skills & Agents, the New Security

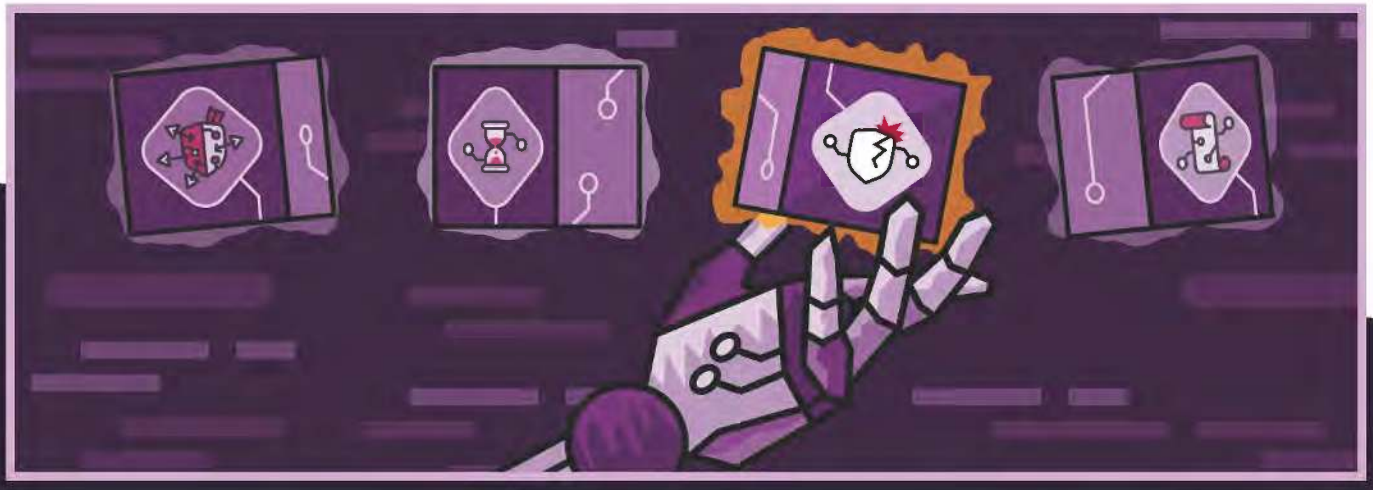
WORKFORCE

CLOUD SECURITY PARTNERS

Written by
Brian Henderson

View Original Blog Post

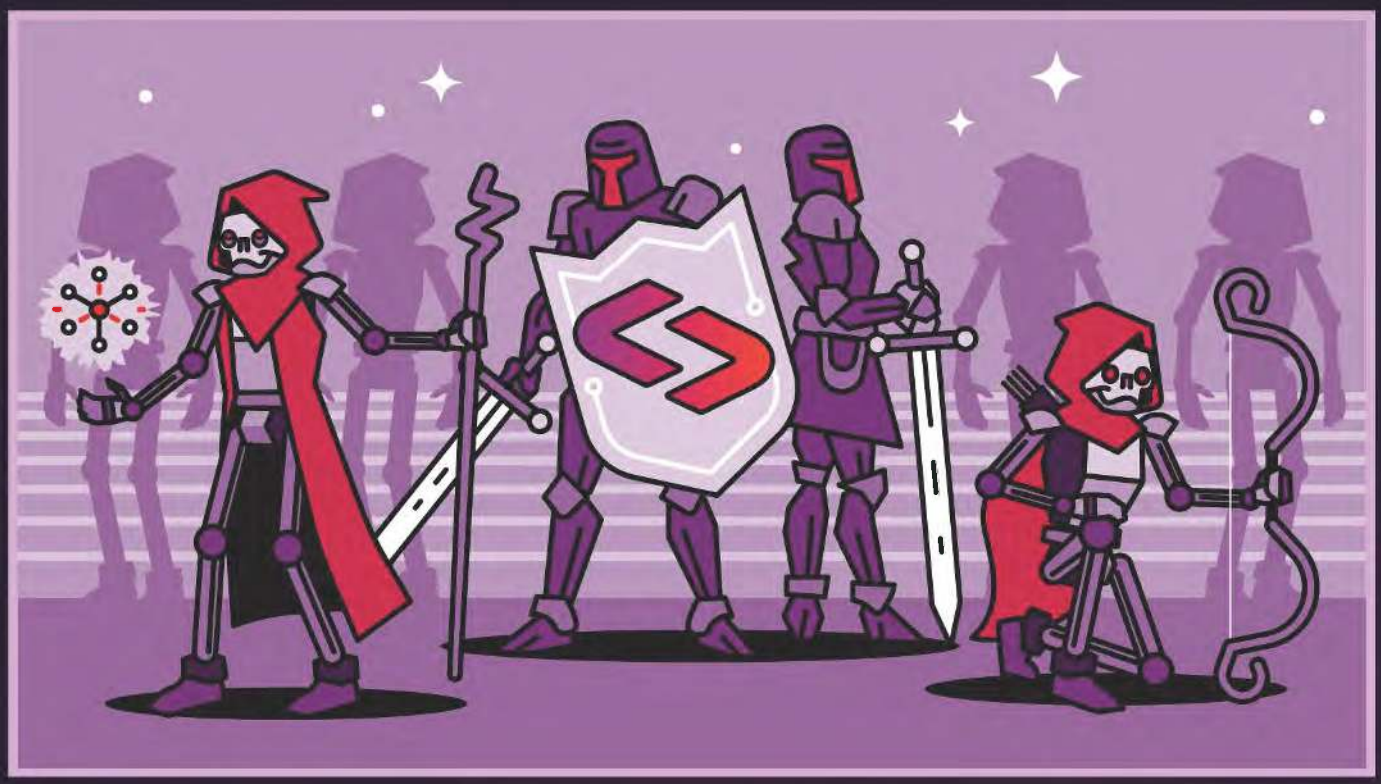
Much of security is spent stitching together tools: pulling logs, cross-referencing CVEs, running indicators through threat intel logs, and gradually building a picture of what happened. **It is tedious, repetitive, and formulaic, which is the exact kind of work that skills and agents are designed to solve.**



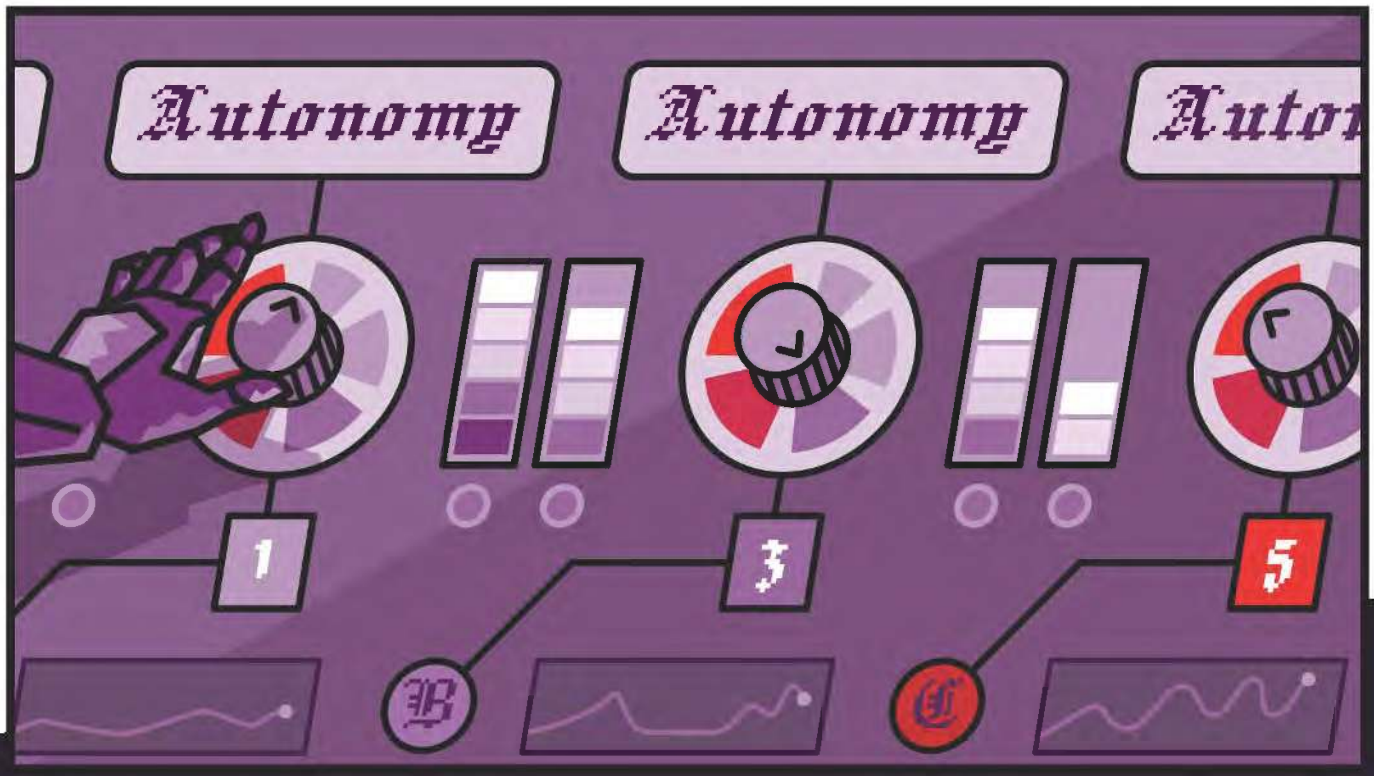
Skills and agents are the building blocks for new agentic tasks. **Skills are reusable, task-focused instructions you define for an AI system. Agents are the autonomous drivers that decide when to apply them. Composed with tools and orchestration, they form agentic workflows.**

Skills package the instructions, context, and scripts needed to do one well-defined task, like looking up threat intel on an indicator, parsing a specific log format, or scoring an alert. Each skill is narrow by design, so any one of them in isolation looks unremarkable. The value comes from composition: an agent that reaches for the right skill at the right moment, and chains several together, turns a set of small, reliable tasks into something that handles real work.

Agents are what drive that composition. They plan, reason through a problem, pull in the relevant skills and tools to execute the smaller steps, and loop until the task is complete or until they hit a point where they should hand back to a human.



Agents are also how you scale a security team without scaling headcount in lockstep. You don't need full AI "coworkers" to start; copilots are a practical entry point. The autonomy you grant an agent is a dial, not a switch. Some agents work alongside analysts — surfacing context and suggesting next steps while a human stays in control. Others run independently on routine, well-scoped tasks.



When designing skills and agents, make sure you're addressing the following:

- **Are my skills atomic?** Skills work best when they do one thing well. If a skill is doing several things, split it into smaller ones.
- **What's the escalation path?** Agents will get things wrong. Decide where a human reviews automated actions before they take effect, and let that judgment inform which tasks you're willing to run fully autonomously.
- **How do I know it's working?** Define how you'll measure whether a skill or agent is performing correctly with eval cases, logging, and spot-checks. Without this, errors surface as incidents instead of metrics.

You also don't need to build everything at once. To build your AI workforce progressively, follow this timeline:

1

Augment with skills.

Skills help analysts on focused tasks like enrichment, summarization, and triage scoring, while humans stay in control. Publishing an internal skills registry lets your team share skills and spread what works.

2

Automate with agents.

Agents can handle routine, well-defined processes end-to-end. Tasks like alert triage, ticket creation, and basic containment with clear criteria can be automated, with analysts supervising rather than executing.

3

Orchestrate multiple agents.

Agents coordinate subagents and tools across complex, multi-domain investigations. This phase requires real trust, solid governance, and clarity about where human judgment stays essential.

Chapter 4

GENERATING Secure Code with AI

CLOUD SECURITY PARTNERS

Written by
Sean Lyford

View Original Blog Post

AI writes production code at an incredible pace, often faster than the speed of human review. **"Vibe coding"** is an engineering practice that embraces this, where engineers fully generate and deploy code generated by AIs. To secure code being generated so quickly, it is important to ensure that the AIs are **generating secure code.**

AI often generates insecure code by default. **Cloud Security Partners** finds that sample applications generated with OpenAI Codex frequently contain the following vulnerabilities:

- **Incomplete CSRF protections** across multiple frameworks
- **IDOR vulnerabilities** in Rails applications
- **Weak, hardcoded secret keys** in Flask when environment variables were missing
- **Weak or missing cookie security flags** in ExpressJS applications
- **Hallucinated dependencies,** also known as typosquatting, across multiple frameworks

LLMs tend to find common security patterns, like parameterized SQL queries and ORM usage, **but miss context-specific configurations** like CSRF middleware setup, scoped database queries, and secure cookie flags.

The solution is better prompt engineering. Specific instructions and examples of each vulnerability class help agents learn insecure patterns and thus help them avoid writing bad code. The best way to teach the model these examples is by creating **skills** with concise, explicit examples. **OpenAI's Security Best Practices skill** is a great starting point for security-specific skills.

Generic statements, unfortunately, do not provide any measurable value, due to the highly contextual nature of most software applications. This includes instructions like **"review this code for security issues".**

A permissions change may be valid in some situations, but not others, which may be difficult for an AI to understand.

They may even be actively harmful, as they eat up an AI's context window, degrading the AI's performance.

When generating code with AI, follow these principles:

Be specific

Be explicit with what you want any agentic task to do. Include framework-specific examples with skills and name libraries explicitly in your prompts. Avoid generic statements, and leave little room for interpretation.

Iterate

Iterate on your examples and skills to ensure that code is securely being generated. Models are updated, outputs vary, and new issues will emerge as model behavior changes.

Review downstream

Prompt engineering improves the baseline, but it does not eliminate the need for human analysis and pipeline tooling. Traditional SAST, AI SAST, and human review are always needed.

Chapter 5

AI vs Deterministic Tools: An Honest In-Depth

COMPARISON

CLOUD SECURITY PARTNERS

Written by
Jordan Darrah

View Original Blog Post

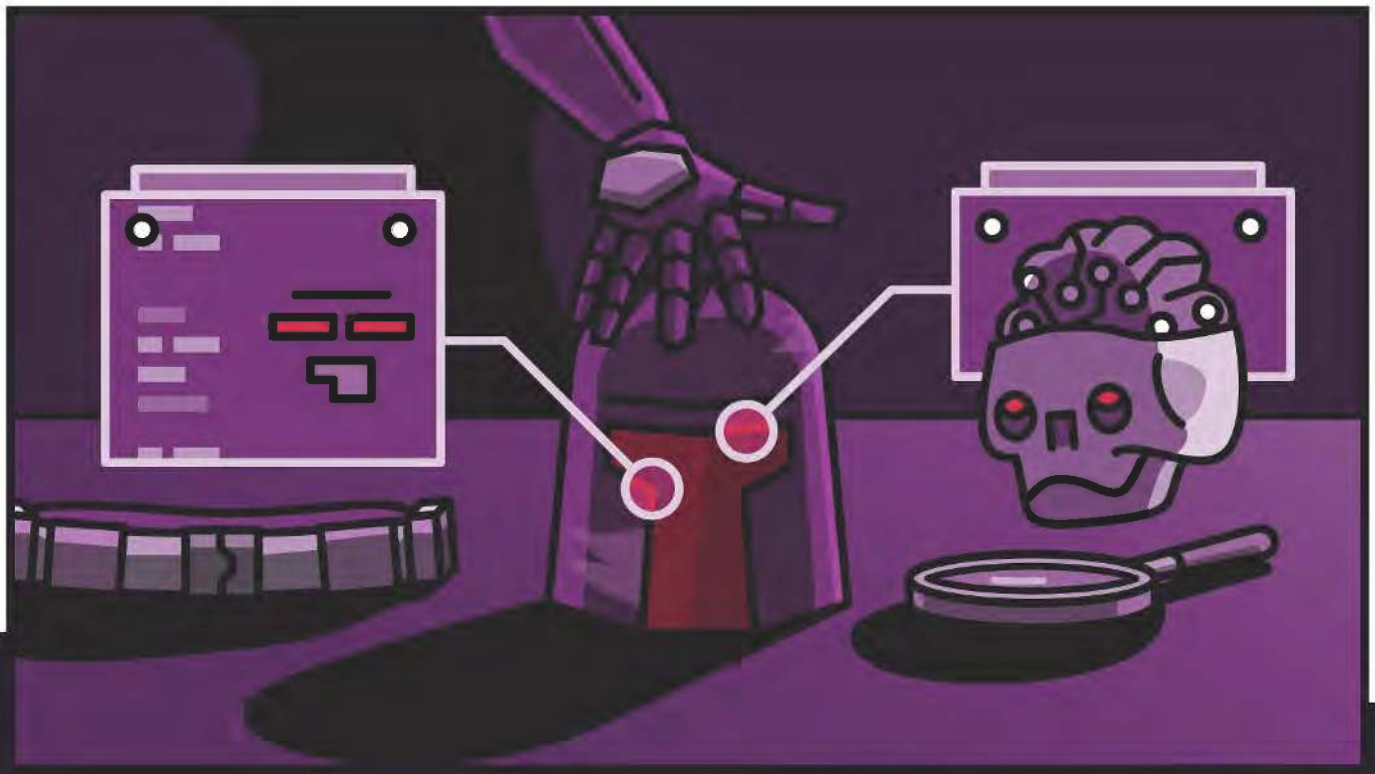
Security teams already rely on deterministic SAST tools in their toolkit. **Does AI make this obsolete?** We tested three types of security tools against manual testing.

1. **Traditional SAST tools** like Semgrep and Brakeman rely on predefined patterns matching syntax across a codebase. They excel in being unfailingly consistent: fast, inexpensive, and with an auditable evidence trail that compliance teams require. **However,** they have no contextual understanding of what code is supposed to do.

2. **AI security reviews,** like Claude Code's `/security-review`, work fundamentally differently. Instead of matching patterns against syntax, they match intent across a codebase. They are able to analyze patterns that are dynamically generated based on context. Due to context window limits and the size of codebases, they tend to operate on diff hunks, analyzing changed files.

3. **Full repository AI scanners** are an emerging category that map data flows, trust boundaries, and authorization logic across an entire codebase before running AI analysis.

Our testing shows that →



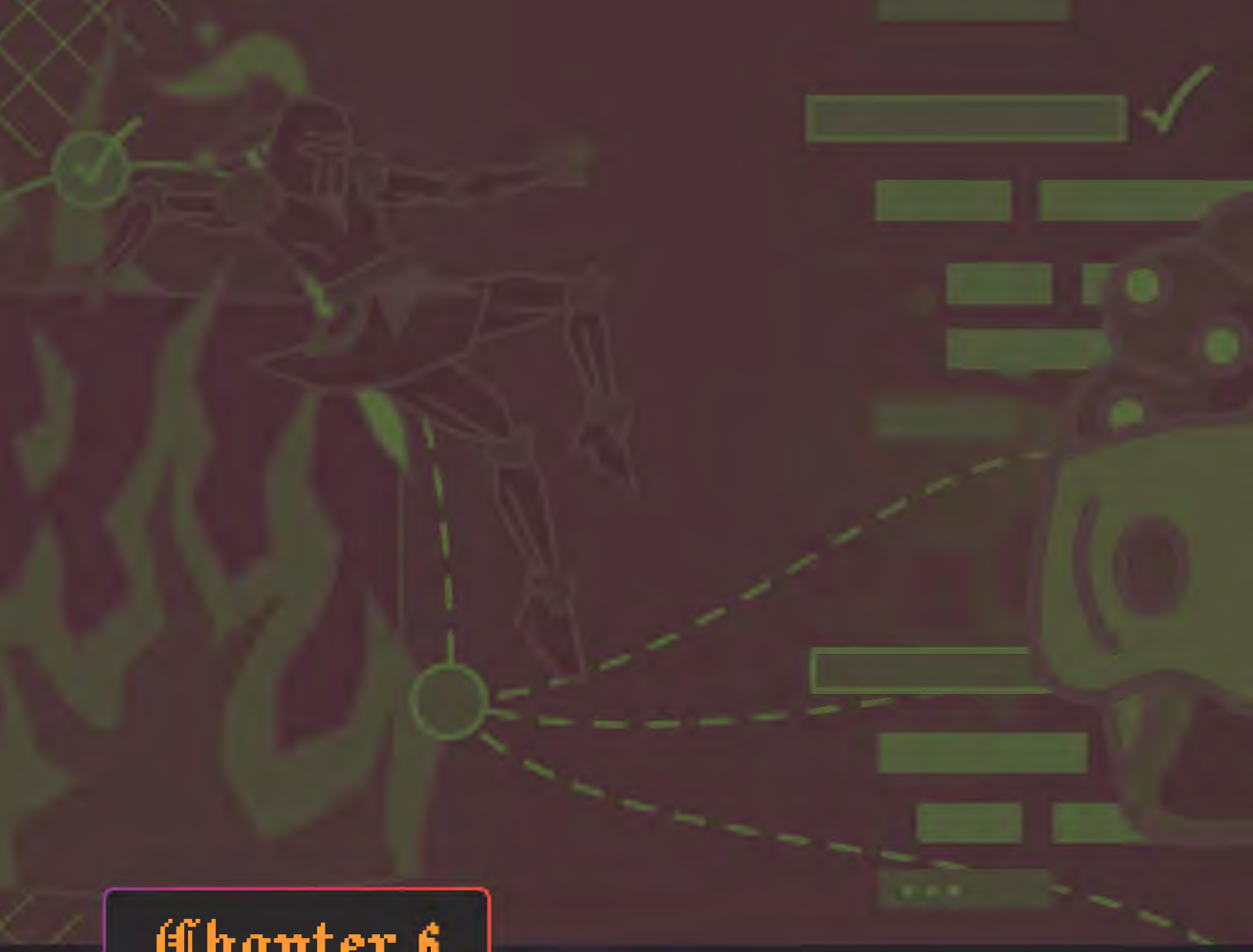
- **Manual review** finds the most vulnerabilities, **but** takes the longest amount of time.
- **Traditional SAST** is the quickest, **but** does not identify as many vulnerabilities as the AI tools or a human. It is highly consistent, **but** requires human validation and misses all business logic flaws.
- **Diff-aware AI review** finds vulnerabilities reasonably quickly, and catches some business logic issues, including authorization bypasses, **but** misses critical findings and marks valid vulnerabilities as false positives, with no evidence trail.
- **Full repo AI scanning** finds the most vulnerabilities of any tooling, and includes audit artifacts with architectural context. **However,** it is slower than traditional SAST.

The AI security tools also carry a risk of **prompt injection,** especially on untrusted code. Untrusted code, like a Pull Request from a malicious developer, could social engineer an LLM with misleading comments, tricking it into hiding valid findings.

The takeaway is that **no single tool wins.**

AI SAST tools complement deterministic security tooling. The most effective security programs use all of them:

- **Use traditional SAST** for speed, consistency, and compliance, and for scanning code from untrusted developers.
- **Use Diff-aware AI** for contextual feedback during development.
- **Use Full-repo AI** for deep architectural reviews.
- **Use manual human review** for high-risk changes and custom business logic that no automated tool can fully evaluate.



Chapter 6

AI-Assisted Network Penetration

TESTING

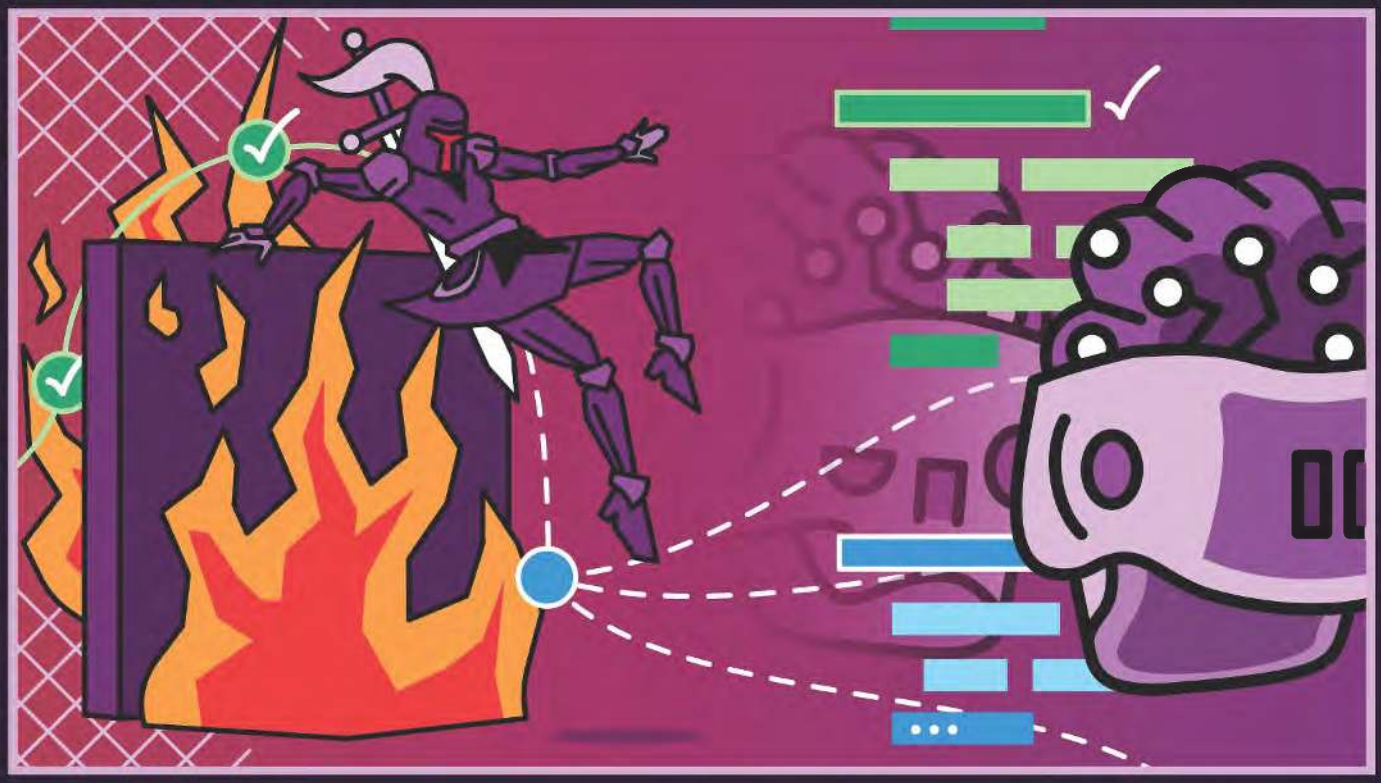
CLOUD SECURITY PARTNERS

Written by
Robert Mikołajski

View Original Blog Post

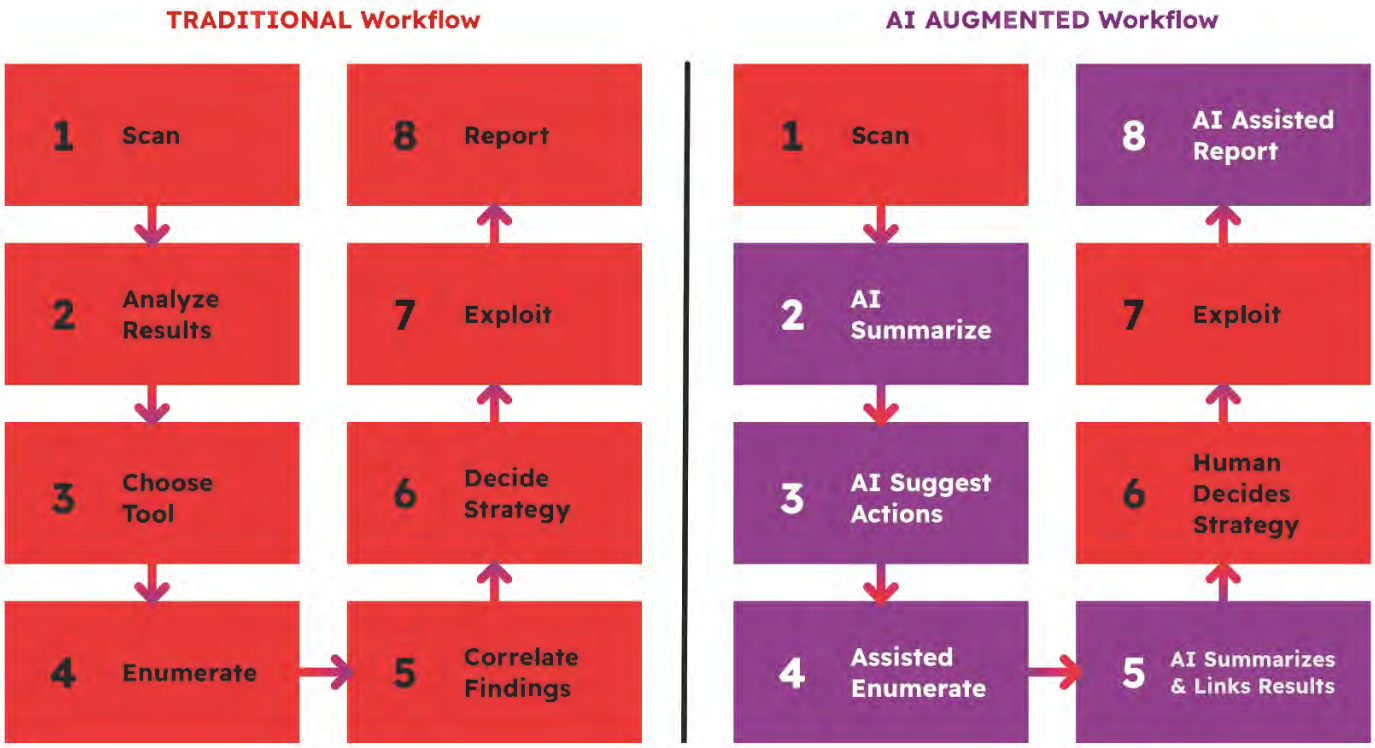
AI tools are most often used to improve code security. **However,** another key use case is assisting in network penetration tests.

Network penetration tests have historically been a deeply manual process. Even with tools like Bloodhound, Nessus or Nmap, the process requires significant manual effort from a human tester to select tools, review large outputs, identify leads and then chain together weaknesses to demonstrate real impact.



AI systems can help automate and augment large parts of this process, as they excel at correlation and large scale data analysis. The best ways they help are by **ingesting tool output, suggesting follow-up actions,** and **constructing potential attack paths across the network.**

As an example, in the diagram below, traditional manual network penetration testing looks like the left side. However, with the aid of an AI, this workflow can be improved to automate shown on the right side:



Most AI-driven penetration testing systems are not scanners or exploit frameworks.

Tools like PentestGPT and PentAGI act mainly as orchestration layers, sitting on top of traditional security tooling and using an LLM to assist with reasoning and workflow decisions.

There are some tools like RapidPen which attempt to automate the entire penetration testing process. However, due to current model limitations, **we would advise treating their results as supervised automation rather than fully independent testing.** Fully autonomous penetration testing can damage the test's effectiveness due to a few key factors:

- **Stealth and Detection:** Automated systems often scan freely, but aggressive activity triggers IDS, endpoint monitoring, or account lockouts. Human testers routinely adjust tactics in ways automated workflows currently cannot.
- **Pivoting and Network Segmentation:** Enterprise networks are rarely flat. Navigating firewalls, VLANs, and jump hosts through multi-hop attack paths remains difficult for AI-driven systems.
- **Authentication Complexity:** Advanced pentest techniques like Kerberos delegation abuse rely on implementation details that vary between environments. LLMs can reason about these mechanisms, but reliably exploiting them across diverse systems remains challenging.
- **Risk of Service Disruption:** Penetration testing activities can cause instability or trigger lockouts in the environment. Human testers pace their activity and assess impact before each action. Automated systems may execute commands without understanding their operational consequences.

When using AI tools to assist in penetration testing, **make sure to be careful about sending artifacts to external AI services.** Assessments frequently uncover credential material and other confidential data, and sending these to non-compliant AI services may break your compliance obligations to your customers, as discussed in Chapter 8.

For these reasons, **AI should be used as an assistant** when performing penetration tests, to add a human in the loop appropriately and prevent destructive outcomes.

Chapter 7

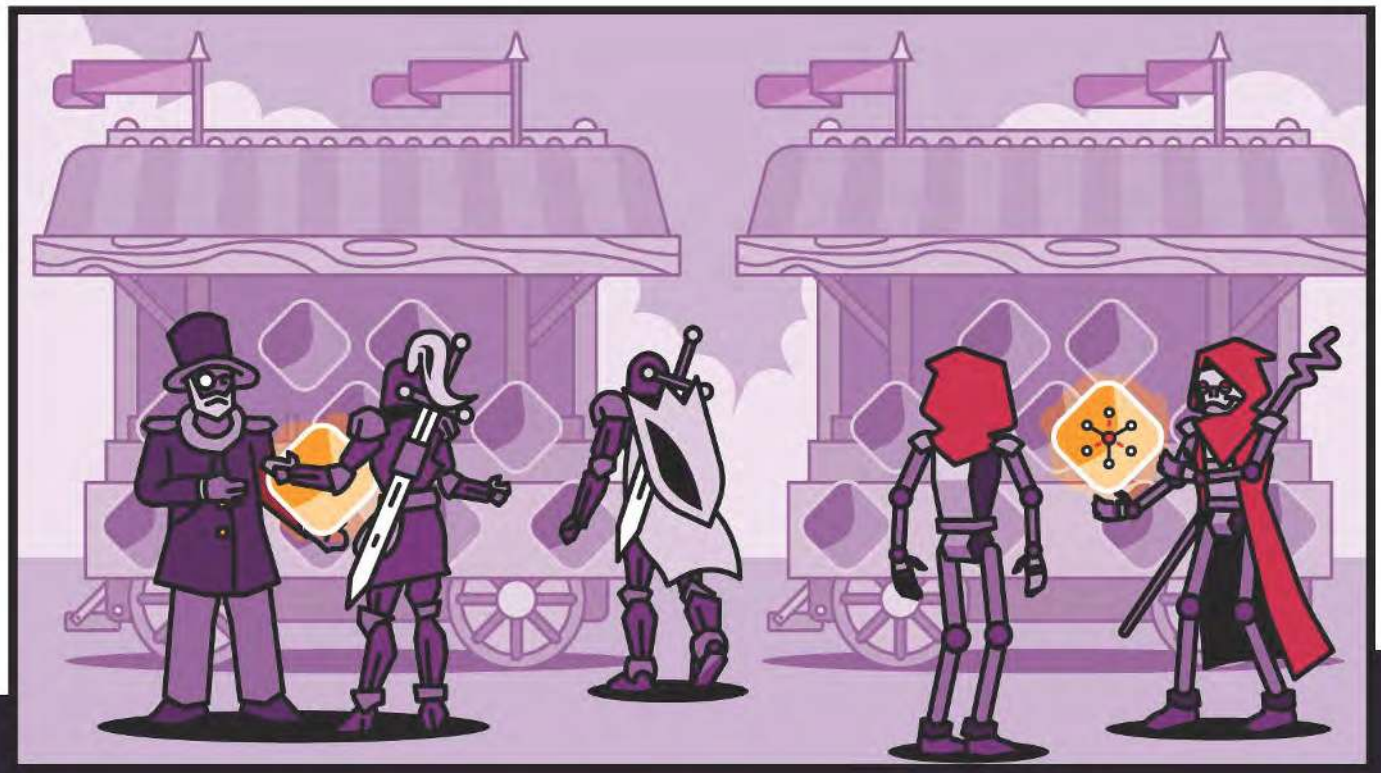
Securing the AI Skills SUPPLY CHAIN

CLOUD SECURITY PARTNERS

Written by
Mike McCabe

View Original Blog Post

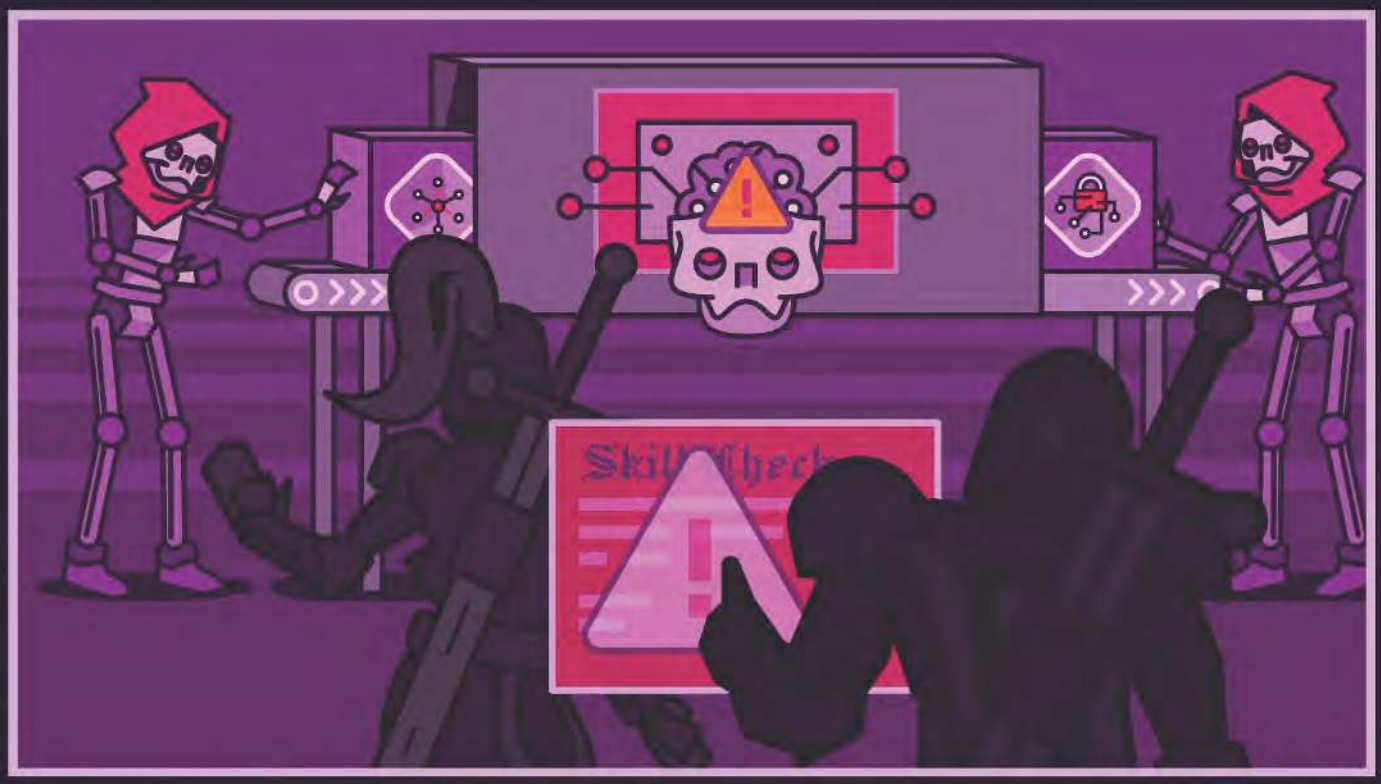
Whenever a new package ecosystem takes off, security is an afterthought — **until it isn't.** The AI skills ecosystem is repeating many old mistakes, and unfortunately, the stakes could be higher.



Claude Code, Codex, and other AI development tools have rapidly growing skills ecosystems. As discussed previously, **skills are markdown files that contain instructions, scripts, and commands, and installing one is as simple as a single command.**

With that one command, you have added third-party instructions into your AI session. Some may require approval to execute. **Others do not.**

The security data is already concerning. A Snyk audit found that **13.4% of audited skills contained at least one critical severity security issue,** such as malware distribution, prompt injection, or exposed secrets. **36.82% of audited skills contained a vulnerability of any severity.**

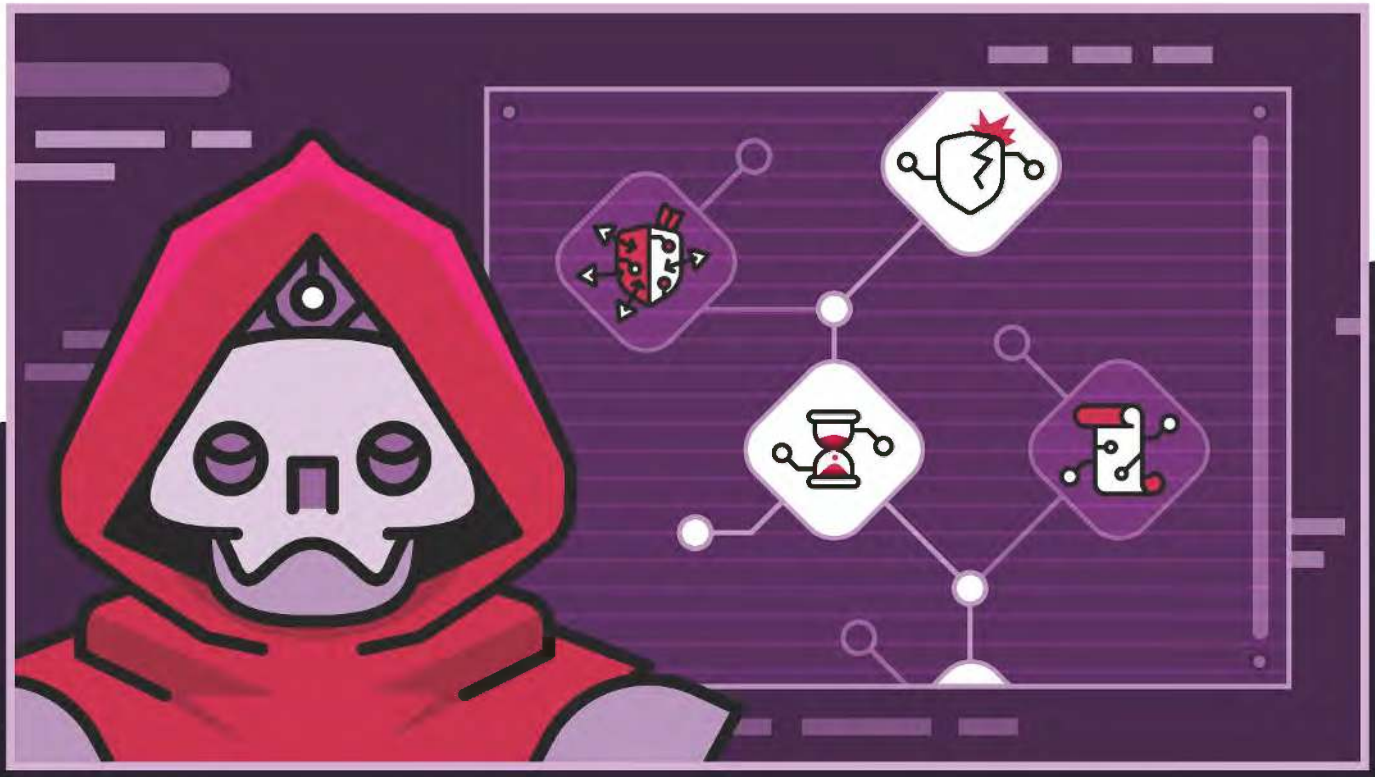


AI skills are different from traditional packages. Instead of pulling in code, they pull in **instructions** that shape how your agent behaves, what it runs, and what it should access.

Hence, the blast radius is everything that your agent can access, which could lead to critical vulnerabilities.

If you are using AI skills, **treat them like any third-party dependency:**

- **Review** the skill, where it came from, and who maintains it
- **Inspect** what data it touches, and whether it induces any external network calls
- **Verify** that the permissions it requests are actually justified
- **Test** in a non-production environment before deploying to live systems
- **Log** invocations and outputs so you can audit what is happening



We have open-sourced a **Claude skill** to help review other skills for potentially malicious commands, and evaluate source repository trust using the **OSS Scorecard** to help automate some of this manual review.

We've open-sourced it at:
https://github.com/CloudSecurityPartners/skills

Chapter 8

AI and COMPLIANCE

CLOUD SECURITY PARTNERS

Written by
Alexandria Poulin

View Original Blog Post

Existing compliance standards like SOC 2, PCI DSS, HIPAA, and GDPR were not designed with generative AI in mind, but their obligations still apply. AI systems process personal data, handle health records, touch payment information, and connect directly to production systems.

Despite the lack of AI-specific standards, your customers and auditors still expect the same compliance obligations.

Understanding compliance and security obligations is critically important. According to IBM's research, **one in five organizations has already reported a breach caused by shadow AI, and only 37% have any policy to detect it.** Two major challenges contribute to this issue:

- **Shadow AI.** Many organizations adopt AI tools without a security review, vendor assessment, or formal vetting. Using unmonitored, uncompliant tools on privileged data can lead to severe compliance risks.
- **Lack of vendor knowledge.** Many organizations rush to adopt AI without fully vetting vendors. It is of paramount importance to understand a vendor's training data, retention policies, and downstream data access. Auditors need to understand provenance chains, so you need to as well.

When evaluating compliance, treat AI no differently from any other tool. Understanding the **intent** of each compliance standard and adhering to it is key to getting ahead of regulators and auditors. For example:

- SOC 2 change management controls may require that a human review all pull requests.
- HIPAA may restrict how patient data can flow through AI systems.
- Customer contracts may include data handling provisions that AI workflows could breach.

Within your program, build your compliance posture around AI before regulators force your hand:

- **Inventory every AI tool.** You can't manage what you can't see.
- **Set guardrails.** Decide what data can and cannot be entered into AI tools. Put it in writing.
- **Classify the data each tool can access.** Map AI workflows to the regulatory frameworks that govern that data.
- **Vet AI vendors.** AI vendors should go through the same security and compliance scrutiny as any other third party. Evaluate data retention, model training policies, and access controls.
- **Assign clear ownership.** Someone should be accountable for AI risk oversight, vendor coordination, monitoring, and executive visibility.
- **Document all decisions.** Compliance requires evidence. Record what tools are approved, what data they access, what policies govern their use, and why.
- **Continuously evolve.** AI is evolving quickly. The structure you build today should be reviewed and refined regularly. Governance is ongoing.

Chapter 9

Building your AI Security PROGRAM

CLOUD SECURITY PARTNERS

Written by
CSP Team

View Original Blog Post

Solving every challenge in this book is dependent on one thing:

building a robust AI security program that manages risk proactively.

Most organizations adopt AI tooling much faster than security programs can react, so building a robust program is all about focused, deliberate effort across three areas: **risk assessment, training,** and **metric generation.**

Assessing risk: AI adoption carries inherent risks around data exfiltration and excessive autonomy. Before rolling out any new AI tooling, have a process for an AI risk assessment that covers the following:

- **What data will the AI access?** Customer documentation, source code, logs, and operational data all carry different sensitivities.
- **What are your compliance obligations?** As stated in the last chapter, your customers and your auditors may have different requirements you must follow, which can impact which tools can be adopted.
- **What technical safeguards can you apply?** Can you create agentic sandboxes, firewalls, or token filtering to reduce AI risks?

Train your team: AI writes much of the code at many mature companies, making code review the most important software skill today. Humans are the last line of defense before AI-generated pull requests reach production.

Your developers should be trained on:

- reviewing code for common vulnerabilities from the OWASP Top 10; and
- securely configuring local agents, including sandboxing, data redaction, and filesystem access controls.

Measure success with metrics: Success can only be measured with metrics. Three key metrics, with baselines collected before AI adoption, will help target your adoption further:

- **Developer Adoption.** Low adoption signals that a tool needs improvement or replacement.
- **Detection rates.** Continuously compare AI security tool vulnerability detection against traditional SAST tools like Semgrep and manual review to gain confidence over time.
- **False positive ratios.** High false positives indicate tooling that needs further tuning.

AI Security programs should be built incrementally. When building and assessing your organization's readiness, use the following **Maturity Model:**

Level 0

Ad Hoc: No AI security standards exist, even though employees may already be using AI on corporate data. Shadow AI usage is likely present and untracked.

Level 1

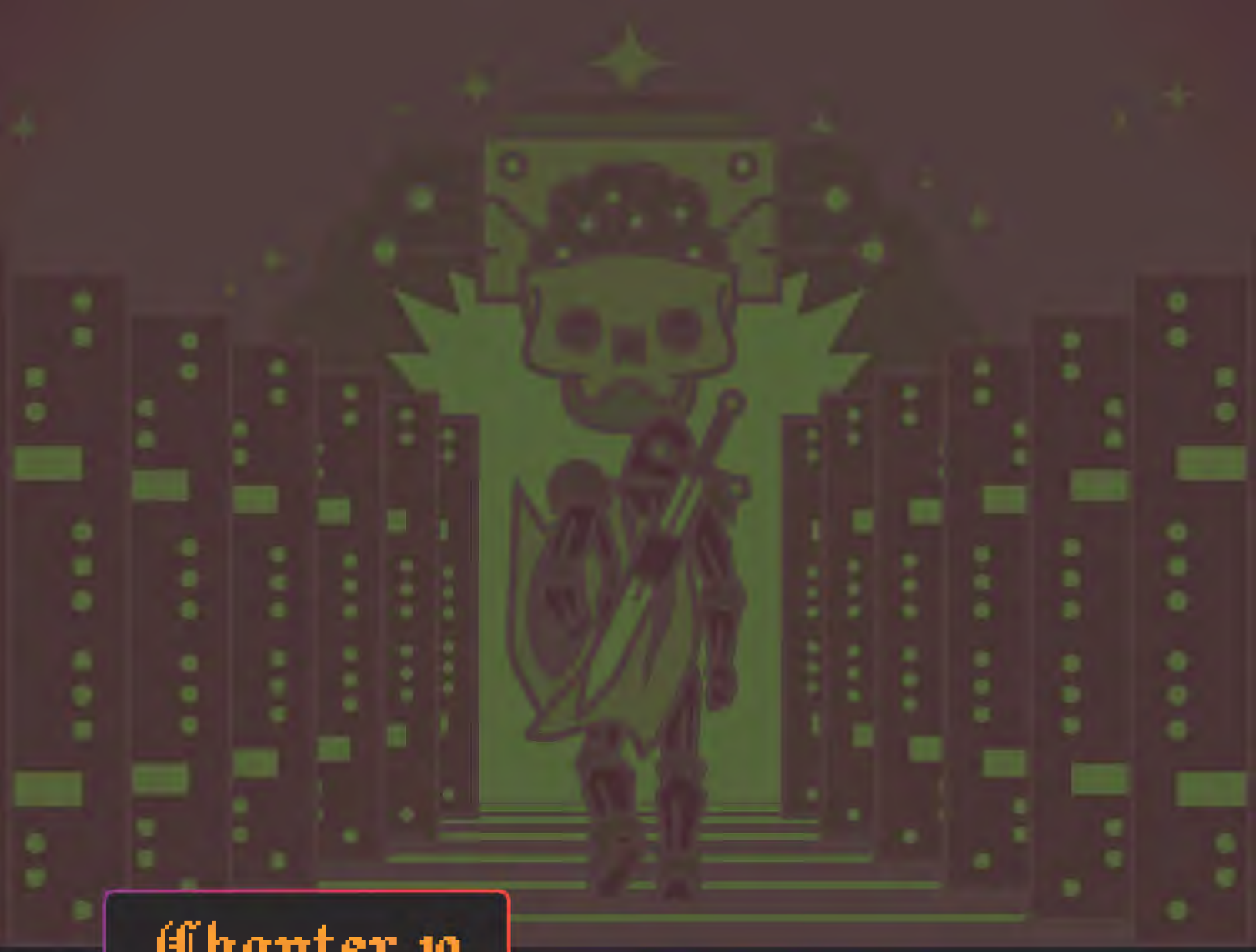
Experimenting: Risk is understood, a draft security policy is in place, and data classifications are in place for all data accessible to AI agents. AI tools are inventoried and mapped to relevant compliance obligations.

Level 2

Defined: There is a comprehensive security policy that covers risk, data handling, and developer responsibility. Metrics are baselined and collected consistently. Secure AI training is a part of onboarding. AI vendors are vetted through the same compliance scrutiny as any third party, with documented decisions.

Level 3

Mature: Metrics drive continuous improvement. Detection ratios are regularly evaluated, new tools are assessed before adoption, and strong technical safeguards for AI agents like agentic sandboxes are in place. Compliance posture is auditable end-to-end.



Chapter 10

What Comes NEXT

CLOUD SECURITY PARTNERS

Written by
Rinaldi Rampen

View Original Blog Post

At the start of this ebook, we introduced a paradox: **unprecedented capability paired with unprecedented uncertainty.** That paradox has not been resolved. If anything, it has deepened. The world of AI moves incredibly quickly, with new model releases, new security capabilities and new attack surfaces being introduced almost weekly.

What comes next is not just more AI. It is AI with **more autonomy, broader reach, and higher stakes,** amplifying all the risks and problems we've discussed earlier.

Agent autonomy is expanding. Today's AI systems are still largely supervised, with humans in the loop at key decision points. As agents evolve, they will operate across longer time horizons, interact with more systems, and execute with broader permissions and less direct oversight. While fully autonomous execution is impractical for modern agents, this is not a permanent ceiling, but merely a product of the current capabilities.

With increased autonomy, comes a greater **agentic blast radius.** When agents operate against internal systems, third party services and external APIs, and with one another, traditional trust boundaries break down. A single misconfigured permission, flawed prompt, or compromised workflow can cascade through an entire chain of automated decisions before anyone notices. Traditional security controls, like the maker-checker model[1], may not translate well to AI-enabled organizations.

Attackers are moving faster. Offensive capabilities are advancing faster than defensive ones. AI-accelerated reconnaissance is compressing attacker timelines from days to hours. Prompt injection will mature into a primary attack vector as attackers learn to reliably manipulate agent behavior mid-task. AI-generated social engineering will continue to be a threat vector, operating at a scale and speed most organizations are not yet built to handle.

1 - The maker-checker principle is formally codified as Separation of Duties in NIST SP 800-53 Rev 5 (Control AC-5), which requires that organizations separate duties of individuals to reduce the risk of malevolent activity. ISACA's COBIT 2019 framework addresses the same concept under governance and management objectives for information and technology. See: https://csrc.nist.gov/publications/detail/sp/800-53/rev-5/final and https://www.isaca.org/resources/cobit

Supply chains and regulation are converging. While supply chains and regulations have not yet kept up with the pace of AI development, the industry will eventually catch up. What usually forces that shift is an incident significant enough to make the status quo indefensible. That incident has not happened at scale yet, but the conditions are already in place.

So what do all of these evolutions mean for your team?

As AI supply chains and regulation catch up, AI **SBOMs** will become requirements. Organizations will need to account for not just the code they run, but the models, skills, and external AI services their systems depend on. Shadow AI will stop being a risk talking point and start showing up as a formal audit finding.

Throughout this ebook, we have argued that AI is becoming an extension of the security team, not a replacement for practitioners. That will continue, **but** the role of the human operator is changing. The most valuable security professionals in the next phase will be people that combine deep domain expertise with the ability to build, evaluate and govern AI systems. Modern security teams must be built with AI native tools and individuals to keep up with the speed of attackers.

New specializations will emerge from this shift. AI-specific red teaming will become a distinct discipline. More broadly, AI-assisted red teaming will redefine how even traditional environments are tested, with both defenders and attackers using AI to increase coverage, speed, and creativity in ways manual-only approaches cannot match.

Moving forward is all about embracing the ways that AI enables security while acknowledging the challenges it provides. AI is creating real advantages for organizations that know how to use it, and real exposure for those without comprehensive plans. The path through it is workable. Policies, governance structures, training programs, supply chain controls, and tooling must be redesigned to move fast without constantly having to clean up afterwards.

AI security is not about slowing down adoption. It is about ensuring that as capability scales, control scales with it.

AI security is not a destination. It is a discipline. And the teams building it deliberately today are the ones writing the playbook everyone else will follow tomorrow.

Appendix: Your AI Security Checklist

Secure Your AI-Generated Code

- [ ] Create framework-specific AI skills with explicit secure coding examples.
- [ ] Include examples for common vulnerability classes (CSRF, IDOR, hardcoded secrets, cookie flags).
- [ ] Avoid generic instructions like "review for security issues" — be specific.
- [ ] Iterate on skills as models update and new issues emerge.
- [ ] Run traditional SAST on all AI-generated code before it reaches production.
- [ ] Require human review of all AI-generated pull requests.

Evaluate Your Security Tooling

- [ ] Inventory all the AI security tools your teams are using (general-purpose agents, AI-augmented SAST, agentic systems).
- [ ] Use traditional SAST for speed, consistency, compliance, and untrusted code.
- [ ] Use diff-aware AI review for contextual feedback during development.
- [ ] Use full-repo AI scanning for deep architectural reviews ahead of major releases.
- [ ] Use manual human review for high-risk changes and custom business logic.
- [ ] Do not trust any single tool — layer deterministic and AI tools together.

Vet Your AI Skills Supply Chain

- [ ] Review every third-party skill before installation: source, maintainer, and purpose.
- [ ] Inspect what data each skill touches and whether it makes external network calls.
- [ ] Verify that the permissions each skill requests are justified.
- [ ] Test all skills in a non-production environment first.
- [ ] Log skill invocations and outputs for audit.
- [ ] Use automated skill review tooling (e.g., CSP's open-source skill reviewer + OSS Scorecard).

Address AI Compliance Obligations

- [ ] Inventory every AI tool in your environment.
- [ ] Classify the data each tool can access and map to regulatory frameworks (SOC 2, HIPAA, PCI DSS, GDPR).
- [ ] Identify contractual obligations that AI workflows may violate.
- [ ] Vet AI vendors with the same scrutiny as any third party — evaluate data retention, training policies, and access controls.
- [ ] Assign clear ownership for AI risk oversight, vendor coordination, and audit readiness.
- [ ] Document all AI tool approvals, data access decisions, and policy rationale.
- [ ] Monitor for shadow AI adoption across all departments.

Build Your AI Security Program

- [ ] Conduct an AI risk assessment before rolling out new tools.
  - [ ] What data will the AI access?
  - [ ] What are your compliance obligations?
  - [ ] What technical safeguards can you apply?
- [ ] Train developers on secure code review (OWASP Top 10) and secure agent configuration (sandboxing, data redaction, filesystem controls).
- [ ] Collect baselines for key metrics before AI adoption.
  - [ ] Developer adoption rates.
  - [ ] Vulnerability detection rates (AI vs. SAST vs. manual).
  - [ ] False positive ratios.
- [ ] Assess your maturity level and target the next level.
  - [ ] Level 0: Inventory shadow AI and draft initial policy.
  - [ ] Level 1: Classify data, map compliance obligations, and inventory tools.
  - [ ] Level 2: Formalize policy, baseline metrics, add AI training to onboarding, vet vendors.
  - [ ] Level 3: Drive continuous improvement, audit end-to-end, and enforce technical safeguards.
- [ ] Review and refine your AI security program regularly — governance is ongoing.

We work with teams building for what comes next.

Whether you're assessing your program, training your team, or navigating what AI means for your compliance posture, Cloud Security Partners helps organizations build deliberately.

Reach out to us at Cloud Security Partners to start the conversation.

THRIVE SECURELY

cloudsecuritypartners.com

cloud-security-partners

CLOUD SECURITY PARTNERS