

Inmitten des AI Vulnerability Storm – Open Source Security RELOADED



Stephan Kraft
Open Source Community Advocate &
Business Development, Red Hat

The 'Vulnpocalypse': Why experts fear AI could tip the scales toward hackers



 Dario Amodei, co-founder and chief executive officer of Anthropic, during the World Economic Forum in Davos, Switzerland, in January. (Krisztian Bocsi / Bloomberg via Getty Images)
© Krisztian Bocsi

As AI grows more capable of identifying software vulnerabilities, experts are increasingly warning of a potential disaster scenario: the so-called "Vulnpocalypse." Hackers could quickly turbocharge their attacks with AI technology designed to identify holes in cyber defenses, security researchers warn. This week, that scenario started to feel less theoretical.

Claude Mythos Preview identifies 27-year-old bug, finds 'thousands' of zero-days in weeks

April 9, 2026



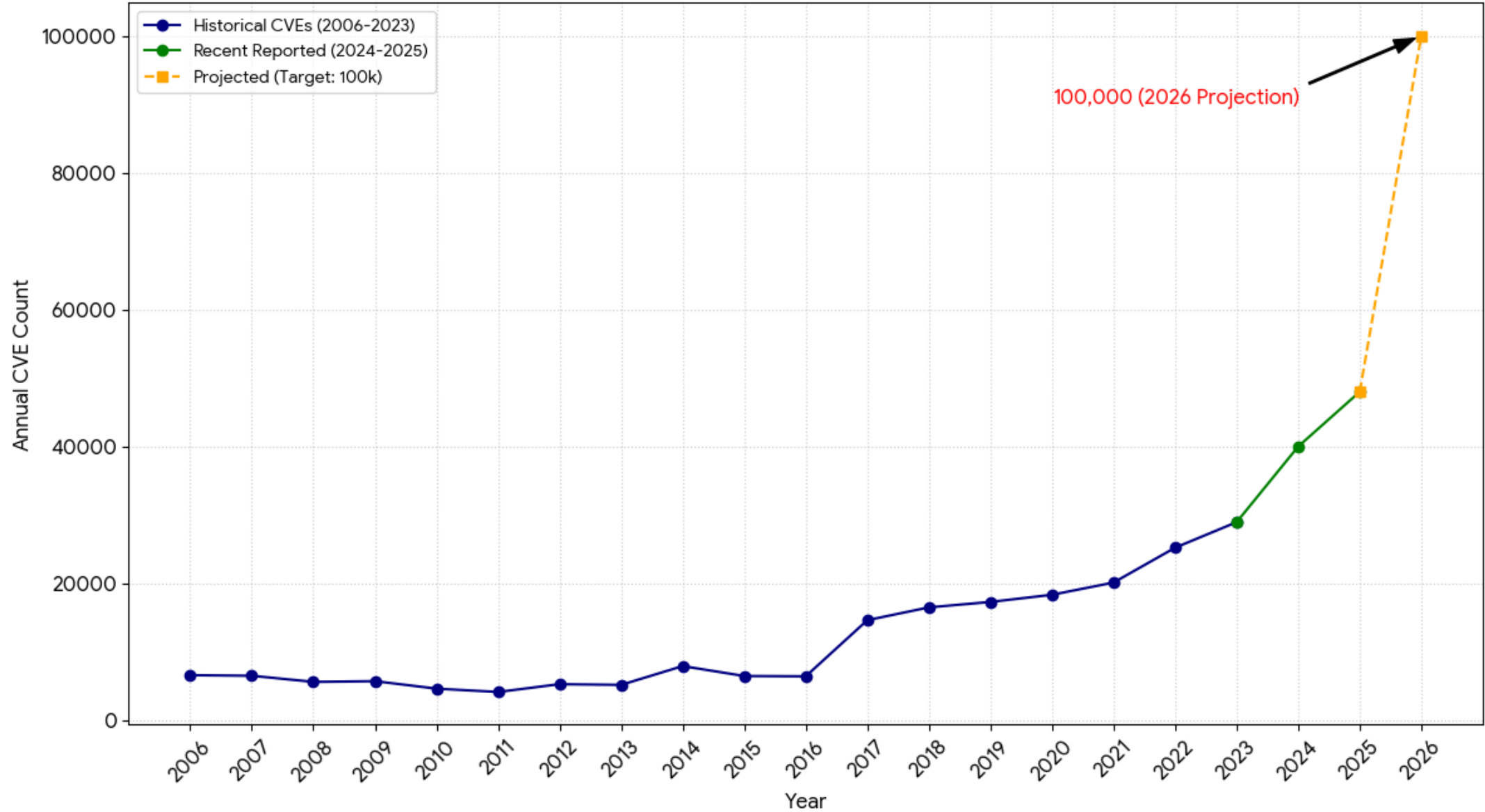
Anthropic said Feb. 7 that Claude Mythos Preview, its new large language model (LLM), discovered "thousands" of severe zero-day vulnerabilities in both open and closed-source software in a matter of weeks.

The company's Red Team said what sets Mythos Preview apart is its ability to develop novel exploits for vulnerabilities with little to no human intervention, sometimes even "overnight."

Along with dramatic speed increases, Mythos Preview identified decades-old bugs, including a 27-year-old integer overflow flaw in OpenBSD and a 16-year-old out-of-bounds write flaw in FFmpeg, demonstrating the model's ability to find new flaws in code that was heavily scrutinized for years.

Over 99% of the vulnerabilities Claude Mythos has found are currently unpatched.

CVE Growth Trend: Reaching 100,000 by 2026

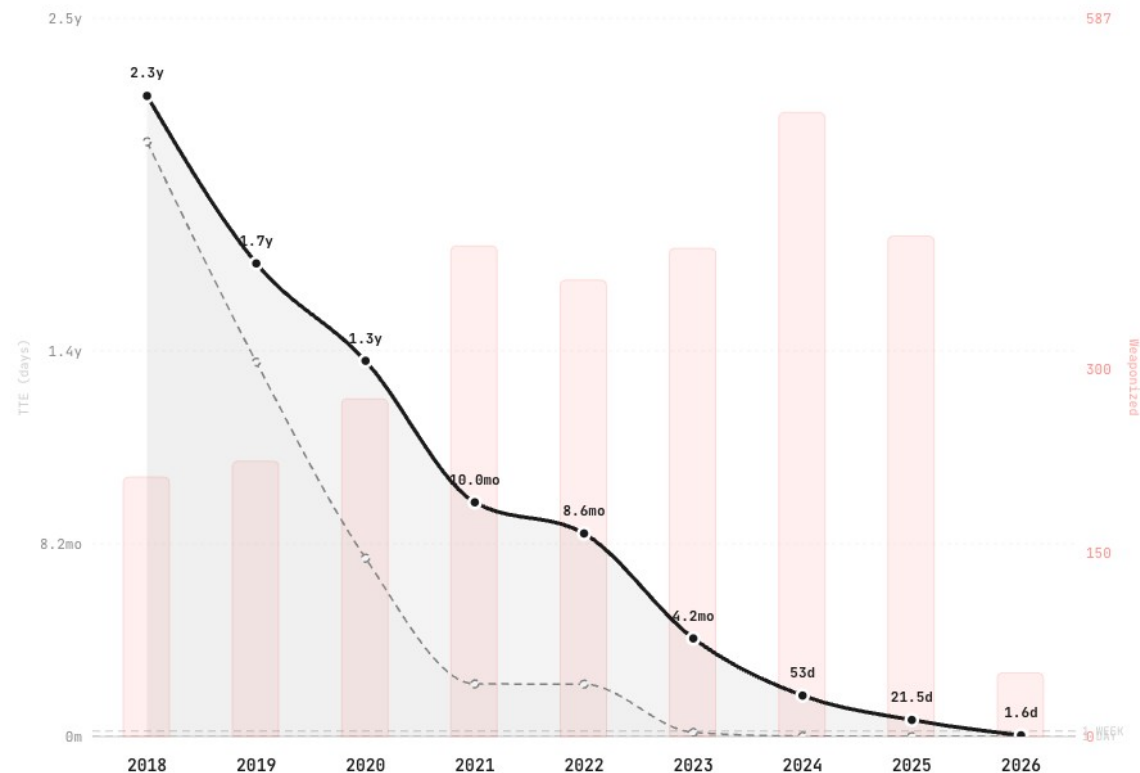


Cybersecurity in the age of AI is changing

From Vulnerability to Exploitation

TTE measures the gap between CVE public disclosure and first confirmed in-the-wild exploitation. Zero = same-day.

— Mean TTE (10% trimmed, days) --- Median TTE (days) ■ Weaponized Exploits (count)



Based on 3,500+ confirmed-exploited CVEs (CISA KEV + VulnCheck KEV, with VulnCheck XDB timestamps for early-year CVEs)

• zerodayclock.com

<https://www.anthropic.com/glasswing>



Discovery is exponential. Remediation is linear.

45%

of vulnerabilities in large organizations remain
unpatched after 12 months.

Source: Centre for Emerging Technology and Security (Alan Turing Institute), 2025 –
cited in CETAS analysis, April 2026.

Should We Go Back to Closed Source?

A fair question in the AI era – but the data says the answer is no

None of this requires the source repository:

**Compiled
binaries &
firmware**

**APIs & protocol
behavior**

**Runtime state &
memory
structures**

**Network traffic
patterns**

Source: Red Hat – “Why proprietary software isn’t a retrospective trend but a trap for failure”



Should We Go Back to Closed Source?

A fair question in the AI era – but the data says the answer is no

MTTD – Mean Time to Detect

Limited external review of closed code means vulnerabilities sit undiscovered – until an AI model finds them for someone else.

MTTU – Mean Time to Understand

Undocumented, unreviewed legacy code takes longer to investigate once a flaw is found – proprietary or not.

MTTR – Mean Time to Remediate

Vendor-controlled patch cycles delay the fix. Closed code doesn't make the fix ship faster.



Should We Go Back to Closed Source?

A fair question in the AI era – but the data says the answer is no

Resilience isn't a function of code visibility. It's a product of detection speed, automation, and remediation velocity.



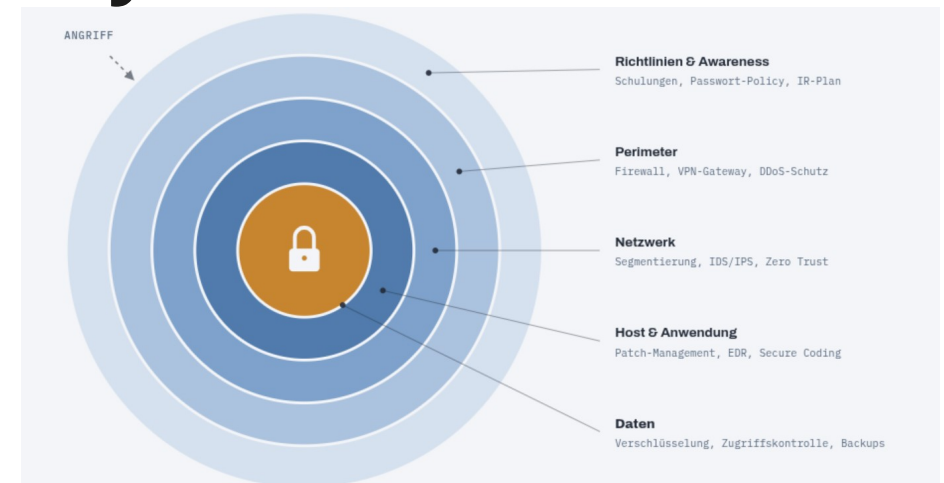
The Operational Reality:

Human patching cycles cannot keep pace with autonomous, recursive AI exploitation engines.

What Is Defense in Depth?

Information security strategy integrating people, technology, and operations capabilities to **establish variable barriers across multiple layers** and missions of the organization.

Source: NIST - https://csrc.nist.gov/glossary/term/defense_in_depth



The security mandates – and what Red Hat brings to the table

5 Automate security at scale

4 Implement guardrails for AI agents & models

3 Protect the pipeline (Supply Chain)

2 Build on trusted sources

1 Start with a strong foundation (Metal/OS)

The security mandates - and what Red Hat brings to the table

5 Automate security at scale

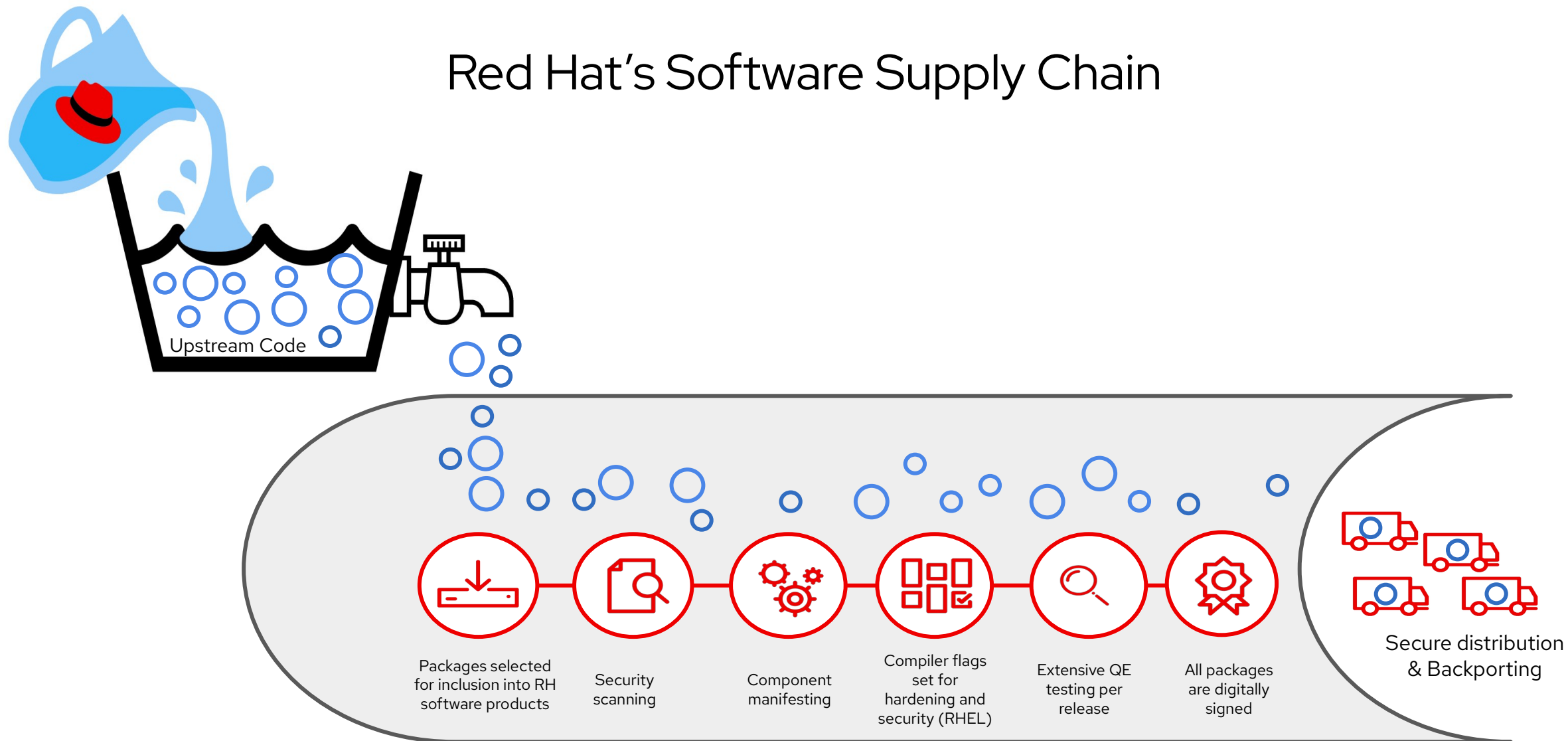
4 Implement guardrails for AI agents & models

3 Protect the pipeline (Supply Chain)

2 Build on trusted sources

1 Start with a strong foundation (Metal/OS)

Red Hat's Software Supply Chain



The security mandates – and what Red Hat brings to the table

5 Automate security at scale

4 Implement guardrails for AI agents & models

3 Protect the pipeline (Supply Chain)

2 **Build on trusted sources**

1 Start with a strong foundation (Metal/OS)

Start with trusted sources



You are responsible for the **Business Applications**.



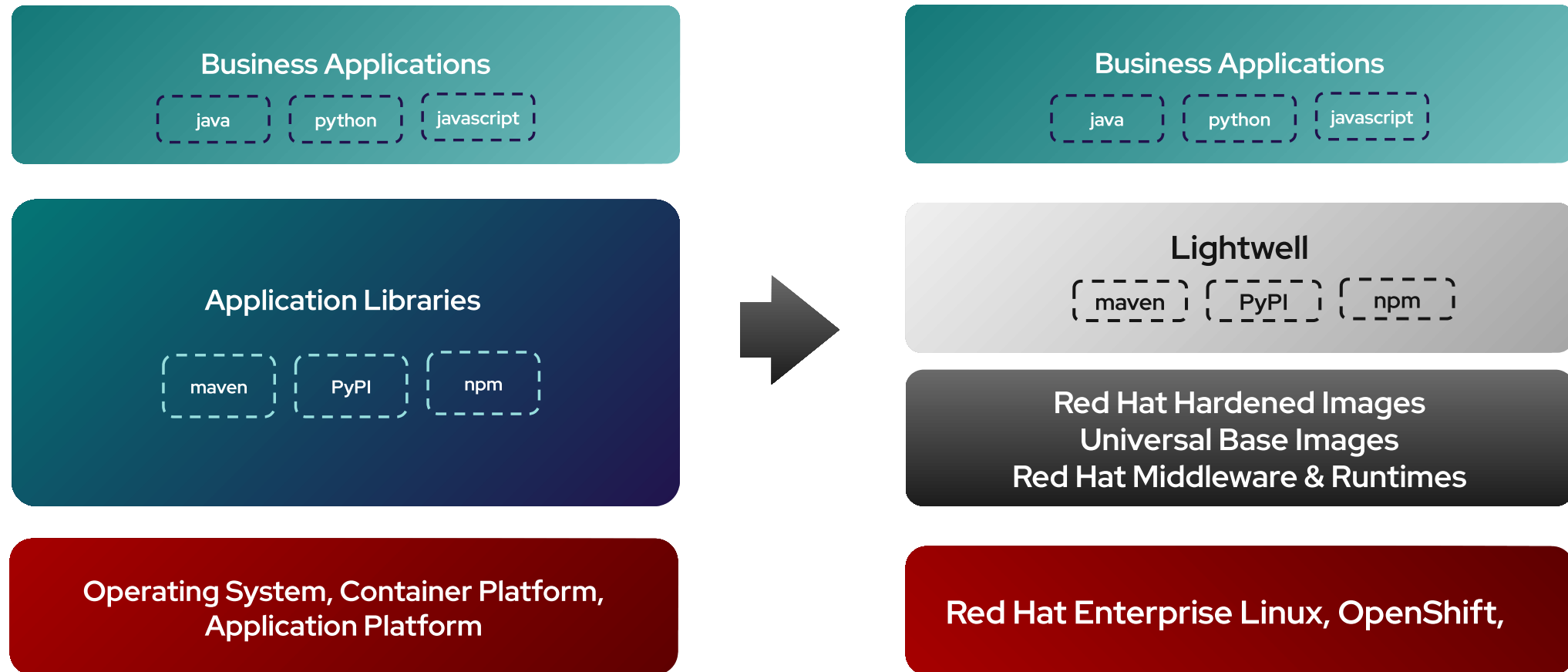
Who is responsible for the **Open Source libraries** in your applications?



Who is responsible for the code that provides the **platform** you applications run on?



Start with trusted sources – What does Red Hat provide?



The security mandates – and what Red Hat brings to the table

5 Automate security at scale

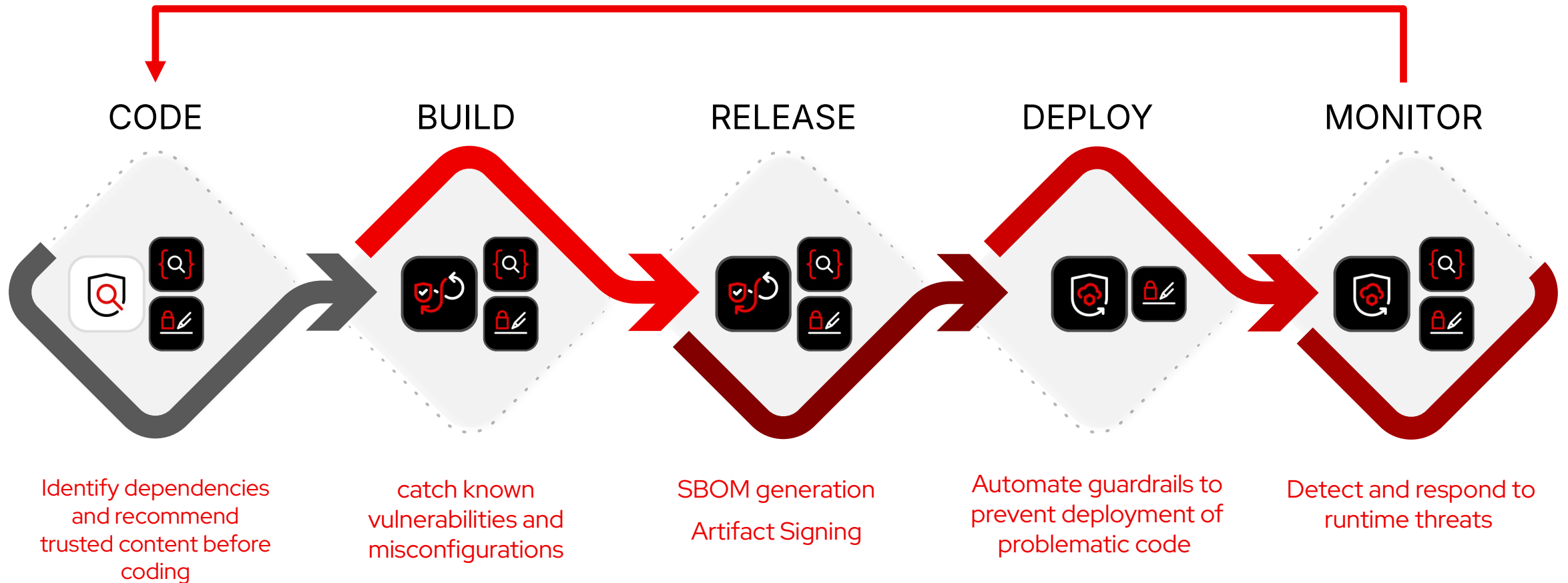
4 Implement guardrails for AI agents & models

3 **Protect the pipeline (Supply Chain)**

2 Build on trusted sources

1 Start with a strong foundation (Metal/OS)

Secure Your Software Supply Chain Security



The security mandates – and what Red Hat brings to the table

5 Automate security at scale

4 Implement guardrails for AI agents & models

3 Protect the pipeline (Supply Chain)

2 Build on trusted sources

1 Start with a strong foundation (Metal/OS)

Why Now: The Agentic Security Imperative

45-90x

non-human identities for every human identity

Entro Labs, 2026

92%

of orgs not confident legacy IAM can manage AI agents

Cloud Security Alliance, 2026

81%

success rate in task-hijacking attacks on agents

NIST AI Agent Red Team

57%

of employees using personal GenAI without IT oversight

Gartner, 2026

#4

Agentic AI as top cybersecurity trend for 2026

Gartner Cybersecurity Trends

EU AI Act

high-risk AI system requirements now in effect

European Commission, 2026



1

Start with a hardened platform

OpenShift ships locked down by default. Keep restricted-v2 SCCs, SELinux enforcing, and add DNS-based egress filtering.

Red Hat OpenShift

SCCs (restricted-v2)

SELinux

EgressFirewall

Without this: Weakened SCCs let agent pods run as root, mount service account tokens, and reach any IP

Without this layer:

Weakened SCCs let agent pods run as root, mount service account tokens, and reach any IP

2

Isolate the runtime

If an agent triggers a container escape, the attacker lands in a throwaway VM, not the host node.

OpenShift sandboxed containers

Kata Containers

agent-sandbox (SIG Apps)

Without this: Container escape = host node compromise. Attacker accesses all pods on that node.

Without this layer:

Container escape = host node compromise. Attacker accesses all pods on that node.

3

Sandbox the process

Control what individual binaries can do inside the container. Per-binary network policy, filesystem allowlisting, credential-free inference.

NVIDIA OpenShell

Landlock LSM

Seccomp BPF

OPA / Rego

Without this: node and curl have the same permissions. Prompt injection spawns curl to exfiltrate data.

Without this layer:

node and curl have the same permissions. Prompt injection spawns curl to exfiltrate data.

4

Guard the conversation

Intercept prompt injections before the model. Filter unsafe outputs. Gate every tool call.

NeMo Guardrails

TrustyAI Operator

MCP Gateway (Kuardant)

Without this: Prompt injections reach the model unchecked, agent calls any tool without authorization

Without this layer:

Prompt injections reach the model unchecked, agent calls any tool without authorization

5

Attack yourself first

Proactive vulnerability scanning. Run in CI. Feed findings back into guardrails.

Garak (120+ probes)

Chatterbox Labs

Without this: Vulnerabilities discovered by adversaries first, guardrails never improve from real attacks

Without this layer:

Vulnerabilities discovered by adversaries first, guardrails never improve from real attacks

6

Deploy with identity and observability

Every agent gets a cryptographic identity. Every tool call is traced with full lineage.

AgentRuntime CRD

SPIFFE / SPIRE

OpenTelemetry

Kagenti

Without this: Shared API keys, no audit trail, no way to trace which agent made which call

Without this layer:

Shared API keys, no audit trail, no way to trace which agent made which call

The one thing to remember: Start with the assumption that the agent will be compromised. Not might. Will. The question is not whether, but whether your stack limits the impact when it does.

The security mandates – and what Red Hat brings to the table

5 Automate security at scale

4 Implement guardrails for AI agents & models

3 Protect the pipeline (Supply Chain)

2 Build on trusted sources

1 Start with a strong foundation (Metal/OS)

Vulnerability Detection to Remediation with Automation

Vulnerability
lands.
Apply virtual
patching.

Patch with
confidence,
when ready.

Prove it
worked.
Every time.

Key Takeaways

1

Nothing is perfect

Nor humans, nor software, nor AI models.

2

Defense in depth matters more than ever

Patching is only one layer of defense.

3

AI is required for defenders

Whether AI shifts the balance toward defenders or attackers is a philosophical debate – but that defenders need AI to keep up is not up for debate.

4

The floor can be reached

Vulnerabilities are finite. The more Open Source software is interrogated and used at scale, the higher the chance of finding them all.

5

Generic and directed measures both matter

Think of it like washing your hands (generic) and taking medicine (directed)



Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

 linkedin.com/company/red-hat

 youtube.com/user/RedHatVideos

 facebook.com/redhatinc

 twitter.com/RedHat