

EXTRAHOP®

A CISO's Guide to
Agentic AI
Adoption



Executive Summary

Security teams are evaluating AI tools fast, but market innovation is moving even faster. A security model selected today can become obsolete before procurement finishes assessing it. Attackers are speeding up too: CrowdStrike's 2026 Global Threat Report puts average eCrime breakout time, from initial access to lateral movement, at 29 minutes, with the fastest observed at 27 seconds.

The architecture beneath the tools determines how quickly a security team can adopt what comes next, and whether that requires a quick swap or a total rebuild.

The Agentic SOC Alliance, launched by ExtraHop with 14 other security and AI companies, has standardized a three-layer architecture of Context, Harness, and Model, designed to keep underlying security capabilities stable as AI models evolve and new capabilities emerge. Context decides how well any model reasons, the harness decides what it may do, and the model is the part you should expect to replace.

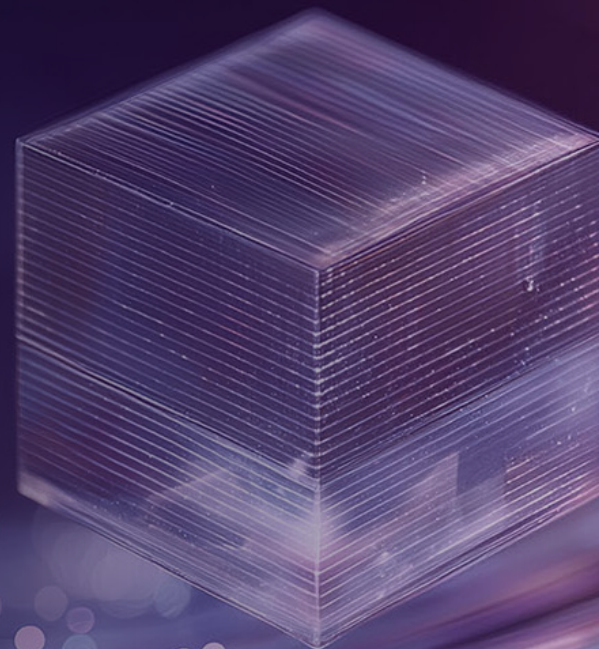


Table of Contents

Building for AI Adoption.	4
The Limits of Asking “Which AI Tool?”	5
Putting Context, Harness, and Model to Work in the SOC	6
Proof From the Field: Context in Action	8
The Architectural Risk of Model Lock-In	9
Why Model Evaluation Isn’t a One-Time Decision	10
When Agents Take on SOC Work, What Happens to Analysts?	12
7 Best Practices to Move Security Agents from Pilot to Production.	13
Agentic AI Security Comes Down to Architecture Decisions	14

Building for AI Adoption

Artificial intelligence (AI) has moved into security operations faster than most organizations have built the architecture and governance to support it. Agents are already triaging alerts, investigating incidents, correlating evidence, and taking bounded response actions.

As agents take on more of the security operations center (SOC) workload and deliver early gains in efficiency, many security leaders are knee-deep sorting through the ever-expanding market of models, agents, and platforms; each promising greater speed, accuracy, or efficiency than the next.

Yet, evaluating AI tools on a tool-by-tool basis introduces friction. Each standalone deployment not only forces teams to go through lengthy procurement cycles, but also saddles teams with building, testing, and maintaining new data pipelines, integrations, interfaces, permission structures, and isolated controls.

Security leaders need a way to absorb the next AI capability without burning cycles and going through a rebuild, thereby keeping analysts on task, moving fast, and applying human judgment where it matters as automated capabilities continue to scale.



The Limits of Asking “Which AI Tool?”

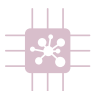
In a fast-moving market, picking a state-of-the-art AI model is a short-term fix to a long-term engineering challenge. Rather than optimizing for today’s top-performing model, security leaders should build an architecture that can accommodate whatever comes next.

Building long-term adaptability is contingent upon maintaining three independent layers:



Context

Gives agents the current, structured security evidence and environmental context they need to reason effectively. It sets the quality of every answer: a strong model reasoning over thin or stale evidence still gets it wrong.



Harness

Governs what agents can access, which tools they can use, and what actions they are authorized to take.



Model

Provides the reasoning capability that can be changed to best meet the needs of the task at hand and as better models emerge.

Keeping the first two layers independent from the Model turns a new capability into a quick evaluation and deployment instead of an elaborate integration project. Security teams can swap the reasoning layer whenever a stronger model emerges, leaving the rest of the architecture completely untouched.



Keeping the context, harness, and model elements independent means that upgrading an AI model becomes relatively simple: the new piece plugs right into your existing system without disrupting the architecture around it.

Why the Network Belongs in the Context Layer

The network sees every device that communicates, managed or not, including the ones no agent was ever installed on. Attackers can blind an endpoint agent, but they still have to authenticate, move between systems, and send data out across the network. And network evidence tells an agent what actually happened, not just what was possible.

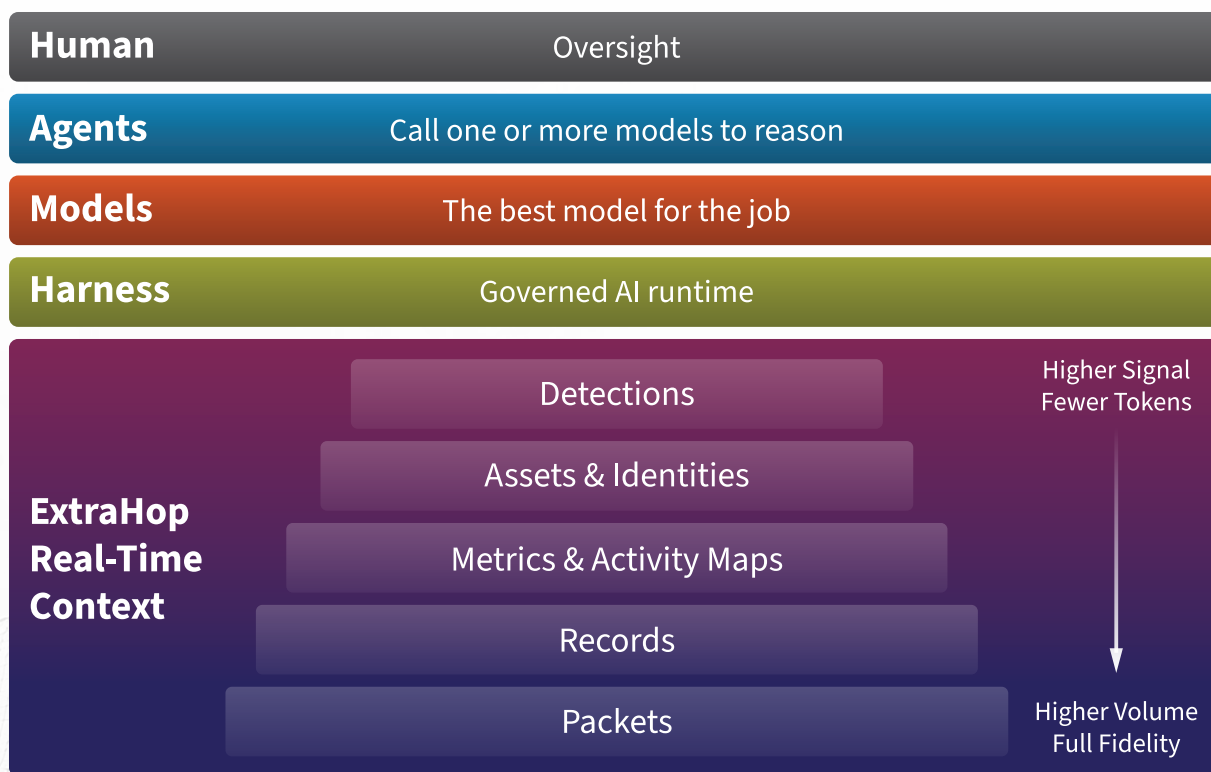
Putting Context, Harness, and Model to Work in the SOC

Each layer of Context, Harness, and Model takes a concrete, testable form once running in the SOC: context connects across the sources the team already trusts, agent actions follow strict written policy, and every action is checked against that policy before it executes.

Operationalizing the Context, Harness, Model framework begins with **connected context**. Each source (network, identity, endpoint, and threat intelligence) stays authoritative where it lives and exposes its evidence through open interfaces, so an agent can join across them when a question is asked instead of hunting down evidence across separate consoles. Copying everything into one store adds delay and strips away the detail each source does best.

Grounding the model in connected, current evidence reduces blind spots, cuts speculative tool calls, and lowers the risk of hallucinations caused by partial visibility. Agents do their best work bound to high-fidelity detections, risk-scored and tied to specific devices and identities, so every alert gets investigated, not just the ones analysts reach.

Once agents can reach the evidence they need, **bounded autonomy** governs every subsequent action, giving the agent full latitude to investigate threats while restricting its authority to execute remediation steps.



Bounded Autonomy, Not Blind Autonomy

The Agentic SOC Alliance architecture draws a clear line between what an agent can act on alone and what requires human sign-off, in concrete operational terms:

- **What the agent can act on alone:** An agent can autonomously handle low-risk, reversible containment steps that policy has pre-approved, with a strictly bounded scope (such as blocking a malicious IP or isolating a user workstation). It earns each class of action by matching or beating analysts on the same cases in testing, and keeps that authority only while it continues to.
- **What requires human sign-off:** An agent must escalate irreversible, high-impact, or business-disruptive actions, such as taking down production infrastructure, rotating a credential that services depend on, or modifying core network policies, for review. Upon identifying a required action that falls outside a pre-approved scope, the agent must halt, attach its evidence chain, and route the request to a security engineer.

For accountability purposes, having a human signature still matters. Any out-of-bounds action should involve a review of the evidence, assessment of the context and risk, along with explicit approval of an agent's action.

Reasoning runs on top of that connected context: a model reads the evidence, then produces a verdict, a confidence score, and a declared blast radius spelling out which entities the action would touch. The harness then checks that verdict against written policy and an evidence threshold it enforces itself, rather than taking the model's confidence at its word.

A team missing the context, the data layer, hands a model incomplete evidence. A team missing the policy layer hands the model unchecked authority. Only a team that's already established both the data layer and the policy layer can treat model selection as the low-stakes decision it should be.

“The good and bad of AI is that models will do exactly what you tell them to do and reason their way to a solution — which may or may not be what the user intended.”

Chad LeMaire
CISO
ExtraHop

Proof From the Field: Context in Action

A major property and casualty insurer operates an advanced agentic security operations center. The engineering team designed an architecture that separates underlying security data sources, execution tools, and models. The design allows the insurer to automate threat hunting across the entire enterprise by drawing on telemetry from multiple domains while keeping its AI models independently swappable by task, tiering lightweight models for routine queries and more capable models for complex investigations.

To connect the distinct layers, the platform team deployed Model Context Protocol (MCP) servers linking an orchestration framework built on BlinkOps directly to network detection telemetry in ExtraHop, endpoint data in CrowdStrike, and event logs in Google SecOps. When threat intelligence arrives, the orchestrator parses indicators of compromise and queries ExtraHop to verify whether specific network traffic occurred. Validating network activity upfront turns a possible match into a confirmed one, or rules it out, which helps eliminate false positives, even during sweeping checks across 1,800 IP addresses at once.

The separated data layer also gives the insurer the flexibility to route tasks to different reasoning engines based on complexity. The security operations team is piloting a tiered model approach, routing complex investigation synthesis to a higher-capability Claude model, while offloading routine data queries to lower-cost models or on-premise, open-weight models running on local hardware. Using specialized ExtraHop skills to fetch targeted network summaries, instead of dumping raw log files, reduced API token costs from \$20 per hunt to much less.

Real-World Impact

By decoupling network evidence from the model, the enterprise built an automated threat hunting pipeline that parses incoming threat feeds, runs retrospective sweeps across historical network data, creates detection rules, and alerts analysts in Microsoft Teams in minutes — all while reducing token expenses and avoiding reliance on any single model vendor.

The Architectural Risk of Model Lock-In

Whether a SOC runs one AI model or several, depending on a fixed proprietary model or set of closed models ties the SOC to vendor release calendars. If a vendor's models fall behind emerging threat patterns, the security team cannot upgrade without disrupting the underlying architecture.



Open-weight models avoid vendor lock-in, as teams can adapt and update them directly.

Rebuilding around a new model can take months, extending the time between a new threat's emergence and a team's ability to respond. Lock-in also happens one layer down, when a platform requires copying your evidence into its own store and ties every agent to its schema. Open, documented interfaces at the context layer, such as REST APIs and MCP, keep a model change from turning into a data migration.

Vendor lock-in becomes even riskier under strict regulatory or data-residency mandates. When compliance requires models to run on-premise or on self-managed infrastructure, security teams need open-weight options, a requirement many hosted-only platforms can't meet.

An interchangeable Model layer breaks the lock-in cycle. Whether adopting a cutting-edge cloud LLM or deploying an open-weight model on-premise for compliance, security teams can swap reasoning engines as needs evolve, without rebuilding the surrounding architecture.

An independent model layer also gives organizations the flexibility to run multiple models across different SOC workflows — matching lightweight, low-cost models to routine queries and high-capability reasoning engines when deep reasoning is required.

A Composite Example

A regional bank had already built its investigation platform so that case data — alerts, evidence, context — lived independently of any single AI model. The model only reasoned over the independent case data, never storing or structuring it.

The separation meant that model connections didn't require rebuilds, only a plug-in, turning what was once a recurring engineering cost into a negligible one.

Why Model Evaluation Isn't a One-Time Decision

Every model claims top performance, but evaluating true utility requires testing how a model operates both before it's used in production and during an actual investigation. Model selection is an exercise in balancing competing operational trade-offs.

Testing candidate models against real investigation tasks exposes the relationship between reasoning accuracy, execution speed, and resource cost. A triage agent screening for false positives needs a different model than an agent conducting a full investigation, and standardizing on a single vendor's platform means that organizations may be overlooking that distinction.

In lab testing, each model receives the same prompt twice: once as an analyst naturally drafts it, and once rewritten with added detail and specificity. Comparing both runs reveals how prompt precision alters an agent's reasoning path, a variable standard procurement ignores.



ExtraHop runs a given investigation task through multiple models and prompt structures within its own lab environment, testing scenarios like certificate-abuse investigations, and measures the operational trade-offs directly. One model required 78 tool calls and 223 seconds to score 90 out of 100 on ExtraHop's rubric. A rewritten prompt finished the same task in 24 calls and 43 seconds but scored 68 out of 100. The right choice depends on the job: a triage agent may take the faster run, while an investigation agent shouldn't trade accuracy for speed.

New models enter the market constantly. A standing evaluation framework replaces one-off procurement decisions with continuous assessment, allowing security leaders to assign specific models and prompt structures to the right SOC operational tasks.

Three principles make that evaluation meaningful. Compare models against your own analysts on the same cases, not just against each other. Treat the winner as temporary and re-test on a schedule, because the best available model changes within months. And never let speed or cost make up for missing an accuracy or safety bar. Run all of it in an isolated copy of your environment, not on live production.


ollama
out 15,549 78 calls 223.6s

> tool call #1 search_detections empty; search_devices found ca.acmelegal.lab (172.16.204.12) CA host

> tool call #2 Mapped assets; certsrv HTTP searches by server address returned empty

> tool call #3 ESC8 detection-type metadata absent; port-80 flow search to CA

> tool call #4 Detection types esc8/adcs not found; found CA flow records

> tool call #5 certsrv/HTTP searches to CA still empty

> tool call #6 CA MAC HTTP records all outbound Windows Update, not server

> tool call #7 CA host HTTP empty; metric query confirms zero server requests

> tool call #8 Confirmed CA has zero HTTP server requests in 7 days

> tool call #9 Proxy (172.16.206.50) only significant HTTP server; broadening detection search

> tool call #10 search_detections error on relative timeframe

> tool call #11 Retried with absolute range; found detections including plaintext auth and C2

> tool call #12 Detection activity: plaintext auth involving proxy and workstation .159

> tool call #13 HTTP records show proxy is Squid forward proxy port 3128

> tool call #14 Workstation .159 traffic to proxy on 3128 examined

> tool call #15 Found curl to evil-ftp via proxy; no Kerberos HTTP records

> tool call #16 Searched certificate metric catalog; only TLS metrics, no ADCS

> tool call #17 Considered enrollment via RPC/DCOM; no port-80 or RPC traffic

> tool call #18 Net metric query revealed /certsrv/ POST/GET traffic from 172.16.206.22

> tool call #19 clientAddr search found device VMware BB666E at 172.16.206.22

> tool call #20 Identified attacker device KALI using curl/8.17.0 against /certsrv/

> tool call #21 Retrieved all KALI certsrv records to CA

> tool call #22 Fetched additional KALI records to locate POST and template

> tool call #23 Distinguished browser NTLM enrollments from curl POST to certfnsh.asp

> tool call #24 PCAP download returned no packets; timestamps noted

> tool call #25 Retried PCAP with broader BPF; still no packets

> tool call #26 PCAP unavailable; examining record bodies for template

> tool call #27 Flow shows 401 NTLM challenge then 200 success (cert issued)

> tool call #28 KALI DNS lookups for CA; no Kerberos, curl NTLM directly

> tool call #29 Clamped record search confirmed no Kerberos, NTLM-only auth

> tool call #30 NTLM record search returned auth exchange details

> tool call #31 NTLM records reveal user ian.lindsay, domain acmelegal.lab, ESC8 confirmed

> tool call #32 Final report: ESC8 abuse, KALI relayed ian.lindsay NTLM, cert issued

> answer #33 # AD CS Certificate-Services Abuse Against acmelegal.lab — Investigation Report ## Root Ca


ollama
out 4,168 24 calls 43.8s

> tool call #1 I'll investigate this using the ExtraHop network-side evidence I can actually reach. Note

> tool call #2 I see hits against '172.16.204.12/certsrv/' (AD CS web enrollment). One source stands out:

> tool call #3 search_devices({"active_from": "-7d", "filter": {"field": "ipaddr", "operand": "172.16.

> tool call #4 This is the ESC8 smoking gun. I have the complete enrollment timeline on the web endpoint

> tool call #5 search_devices({"filter": {"field": "ipaddr", "operand": "172.16.204.12", "operator": "

> tool call #6 search_devices({"filter": {"field": "ipaddr", "operand": "172.16.204.12", "operator": "

> tool call #7 Interesting — '192.168.204.2' (the Edge user-agent) is also hammering the listener at '172

> answer #8 I have enough to write the report. Note for the user: the prompt asked for "Event ID 4886/

When Agents Take on SOC Work, What Happens to Analysts?

Testing tells you which model, using which prompt, handles a given SOC task well. But once the work moves over to an agent, the next question is: what happens to the analyst?

Take the most common case — Tier-1 triage. When agents clear the alert queue, analysts spend less time triaging. Without clear direction from leadership, a shrinking queue can suggest a gradual role phaseout.

The clearest message a leader can send is that a redesign of SOC work is underway. Analysts will take on higher-level work, determining what agents should investigate, what actions they should be permitted to take, and when human judgment is needed. Agents take the shift; people close the gaps and perfect the context agents reason on.



Automation is best rolled out after leadership defines new, higher-level analyst responsibilities and before agents take over Tier-1 activities.

How Analyst Responsibilities Evolve

A Tier-1 analyst who once triaged alerts can become a detection engineer, applying their knowledge of what good triage looks like to refine the logic and prompts that shape agent behavior.

A SOC analyst managing manual triage transitions into a context engineer, ensuring that agents receive the relevant telemetry before an investigation begins.

A shift lead managing queue volume assumes responsibility for agent accuracy, tool-call efficiency, and policy boundaries.

Senior analysts decide when an agent has earned more authority, moving from reviewing agent work to writing the policies that govern it.

Analysts take on direct responsibility for designing, supervising, and refining the systems doing the work.

7 Best Practices to Move Security Agents from Pilot to Production

1. Give each agent its own login. If agents share one login, you can't discern which agent took which actions.
2. Validate everything an agent ingests before an agent relies on it. Bad data translates to bad decisions for every agent that reads it.
3. Let agents escalate to a person. Two agents left to sort out a disagreement on their own is not a safe default.
4. Limit the actions that one agent can take against another. No agent should be able to shut down or take over a different agent's task.
5. Start new agents on small, low-risk jobs and keep assessing their work against policies.
6. Record every agent's reasoning, evidence, and actions in your system of record, so any decision can be reconstructed later.
7. Check an agent's conclusion against independent evidence, such as what the network actually saw, before it acts.



Agentic AI Security Comes Down to Architecture Decisions

As agents take over triage, the analyst role moves into detection engineering and system oversight, the work that increasingly defines seniority in the SOC as humans take responsibility for how systems reason and act.

A security team needs context connected across the sources it already trusts, action governed by written policy, and reasoning independent of any one model. With that architecture in place, a new model can be swapped in without rebuilding the surrounding systems, freeing up analyst time that was once spent on integration for a new kind of work that requires human judgment. ExtraHop supplies the network evidence at the base of that context layer: what actually happened, observed in real time and kept for lookback.

See just how much manual effort bogs down investigations industry-wide in [**ExtraHop's 2026 Global Threat Landscape Report**](#).

For more information about the architecture built to support the agentic SOC, read [**Open Models Need Open Context**](#), by ExtraHop's CEO Greg Clark.



About ExtraHop

ExtraHop is a leader in real-time network intelligence, empowering organizations with the high-fidelity context they need to power security and IT AI automation, detect risks faster, and respond with confidence.

ExtraHop anchors AI agents with the real-time, ground-truth foundation they need to operate reliably in the SOC and NOC. For security, that means surfacing threats and providing definitive evidence to investigate and respond to novel AI attacks and unsanctioned usage at machine speed. For IT operations, it means identifying and resolving performance issues in seconds to keep critical systems running.

Engineered for unmatched scale, the ExtraHop RevealX platform delivers a complete, unified view of the network for the largest enterprises in the world. Capturing and analyzing network data across data centers, campus, branch, cloud, and AI environments, ExtraHop combines behavioral analysis with deep visibility into encrypted traffic to uncover what others miss.

To learn more, visit extrahop.com or follow us on [LinkedIn](#).

