EXTRAHOP

7 Mistakes That Stall Agentic SOC Programs

How to avoid common issues in context quality, data architecture, and governance for AI agents

Table of Contents

What many agentic SOC implementations miss

Agentic AI is transforming the security operations center. By deploying autonomous AI agents that can reason, plan, and take action across complex environments, security teams can dramatically increase SOC velocity and efficacy. That's especially critical as the latest frontier models enable automated vulnerability discovery and exploitation at unprecedented speed and scale.

But there's more to agentic SOC implementation than picking the right model and writing the right prompts. Without the right data and architecture, AI agents won't know what's normal for your environment, what's on your network, or what's happening inside it. That can lead to waves of false positives that undermine the promised efficiency of agentic AI—as well as missed detections that allow attacks to continue. Your organization can suffer both successful breaches and the governance, risk, and compliance (GRC) exposure of not being able to explain what your agent did or why.

Fortunately, these are solvable problems. In this ebook, we'll discuss seven common mistakes in agentic SOC implementation, the problems they cause, why they happen, and practical ways to avoid them.

1.

Starving the agent for context

WHY IT'S A PROBLEM

Errors and hallucination

Scenario: After months of engineering effort, agentic SOC workflows are ready for production. As AI agents receive alerts, they query the available data and produce structured triage summaries, just as intended. But in the weeks that follow, it becomes apparent that these summaries are hallucinated. Their veneer of intelligence has led to false confidence in their accuracy, but they've actually degraded SOC performance.

THE CAUSE

Inadequate data

SOC implementations often focus primarily on LLM selection and orchestration frameworks, viewing the context layer—the data infrastructure the agent actually reasons over—as a secondary concern. Often, this means simply routing alerts through a SIEM pipeline. But SIEMs don't understand what assets mean, how systems relate, or what behavioral normal looks like for a specific device. Without these essential inputs, the agentic SOC platform falls back on invention.

HOW TO AVOID IT

Build the context layer first

An agent is only as accurate as the data it reasons over, and no amount of model capability closes a context gap. Your network detection and response (NDR) platform can serve as a continuously updated source of truth for asset inventory, live behavioral baselines, and lateral movement indicators. By querying this data via a real-time context API, the agent can fully understand the nature and criticality of each alert to execute SOC workflows accurately.

2.

Treating CMDB as ground truth

WHY IT'S A PROBLEM

Inaccurate alert criticality and prioritization

Scenario: To investigate an alert on an internal IP, an AI agent turns to the CMDB, which indicates that it belongs to a decommissioned test server with low criticality. The agent de-prioritizes the alert accordingly—unaware that the IP was reassigned six months ago to a production database holding customer data. You can't blame the agent; it made the right call based on the available context. But that data is now at risk.

THE CAUSE

Out-of-date or erroneous CMDB data

CMDBs capture what IT intended the environment to look like at one point, not its real-time state. Out-of-date or erroneous tags, shadow IT, unmanaged devices, transient cloud workloads, and IoT assets can make the current reality quite different from the CMDB version agents reason over.

HOW TO AVOID IT

Make your network detection platform the runtime source of truth

Your NDR platform can capture the living ground truth about your environment to provide the real-time context agents need. Every communicating device on the network leaves a trace the platform can passively classify, without manual entry, coverage gaps, or staleness windows. This includes communication mapping so agents understand how systems interact and what normal connectivity looks like across the environment. With the right context, agents can make the right decisions.

3.

Skipping hardcoded guardrails

WHY IT'S A PROBLEM

Damaging actions by autonomous agents

Scenario: To execute end-to-end SOC workflows, a team gives autonomous agents the ability to isolate production endpoints, modify firewall rules, or disable executive accounts. This enables a fast response to live threats—but mistakes can have serious consequences. After a series of operational self-disruptions, the team is forced to pull back the agents.

THE CAUSE

Prompt-level constraints

Organizations understand that autonomous agents need constraints, but these are often implemented as prompt-level instructions: "do not isolate production systems," "always require human approval for account lockout," and so on. But prompts can be overridden due to complex reasoning chains, adversarial input, or hallucination over poor data. To be reliable, guardrails need to be built more strongly.

HOW TO AVOID IT

Enforce a strict action taxonomy at the orchestration layer

A prompt instruction is only a suggestion. Real constraints have to be enforced architecturally. A three-tier action taxonomy hard-coded at the orchestration layer can help prevent disruptive errors:

Low-risk (autonomous)

Data enrichment, ticket creation, notifications—no human gate for in-policy actions; approval required for out-of-policy actions

Medium-risk (conditional)

Host quarantine or temporary firewall blocks—quick analyst confirmation

High-risk (strict gate)

Account lockout, segment isolation, production system changes—senior human approval required

Network telemetry can verify compliance with real-time visibility into agent activity and alerts on any changes made.

4.

Neglecting explainability

WHY IT'S A PROBLEM

A lack of trust and auditability

Scenario: An organization's autonomous agents function as designed, closing alerts, recommending containment, and escalating cases. But their opaque reasoning process can't be traced or verified, eroding trust and leading experienced analysts to reject some decisions on instinct. This lack of auditability also exposes the organization to compliance risk; regulators such as the SEC and EU increasingly expect organizations to be able to reconstruct exactly how and why an AI system acted as it did during a cybersecurity event, with a complete audit trail of every relevant interaction.

THE CAUSE

Black-box tools

The infrastructure, model inference, and orchestration logs produced by most AI agent deployments record the actions taken by agents, but not the decisions behind them. A conclusion without a reasoning chain can't be audited, contested, learned from, or defended.

HOW TO AVOID IT

Require transparent reasoning

Agents should be required to output clear, structured reasoning alongside their conclusions. Every triage verdict and containment recommendation must be supported by a structured chain-of-evidence log: what data the agent queried, what each source returned, and what threshold triggered the routing decision. Network telemetry anchors this record by preserving an unalterable account of what the agent saw at the moment it acted, even if its internal reasoning is later disputed.

5.

Forgetting that adversaries adapt

WHY IT'S A PROBLEM

Missed novel attacks

Scenario: An agentic SOC platform uses threat intelligence feeds to detect potential signs of attack. Seeing nothing that matches historical attack templates, it confidently maintains an all-clear posture. But it's wrong—attackers using novel techniques have already breached defenses, and a false sense of security is extending dwell time.

THE CAUSE

Detection logic that doesn't evolve

Adversaries study automated detection systems and alter their tactics to evade them—something the latest LLMs make easier than ever. Threat intelligence identifies known bad actors, but it doesn't update the way the agent reasons about behavior it hasn't seen categorized before. Detection logic, behavioral models, and reasoning prompts are typically calibrated at launch, validated against known attack patterns, and left to run. When attackers adapt but agents don't, a breach is only a matter of time.

HOW TO AVOID IT

Implement a continuous feedback loop

An agentic SOC is a living strategy, not a static deployment.
To keep its reasoning up-to-date, SOC teams should implement:

- **Targeted red team exercises** to probe agent detection logic and detect gaps
- **Regular reviews of all false negatives** to see what agents are missing and why
- **Continuous injections of fresh threat intelligence** and behavioral models into the platform

Behavioral baselines built from network traffic are especially valuable. Attackers must use protocols to move through the environment, and unlike the endpoint artifacts or indicators they can delete or spoof, the network record is far harder to fake or erase.

6.

Using PCAPs where records suffice

WHY IT'S A PROBLEM

High token consumption

Scenario: Seeking maximum efficacy, a security team opts for full packet capture as the data source for its AI agents. But from the agent's perspective, PCAP data is verbose, unstructured, and largely redundant, leading it to spend tokens parsing format rather than reasoning over content. As both investigation latency and cost increase, the economics of agentic triage start to break down.

THE CAUSE

Unnecessary full-packet inspection

In human analyst workflows, full-packet inspection is sometimes necessary for forensic confirmation. But the vast majority of data an investigative agent needs to reason over is already captured in structured wire data records, including source and destination, protocol, duration, and byte counts, as well as deeper details such as HTTP URIs, DNS queries, SSL certificate metadata, and application-layer operations. These records are already parsed, structured, and immediately interpretable by a model, at a fraction of the token consumption and cost of PCAPs.

HOW TO AVOID IT

Use records for logic and PCAPs for evidence

Agents should follow the same design principle as human analysts: let records power the logic of investigation, and turn to PCAPs only when a confirmed finding needs forensic proof. With this approach, triage agents never touch packets, while investigation agents work from records and examine packets only when they need payload-level confirmation of malware extraction, exfiltration content, or legal chain-of-custody.

7.

Defaulting to cloud LLMs when local inference fits

WHY IT'S A PROBLEM

Cloud cost and risk issues

Scenario: To turbocharge its AI agents, an organization defaults to the most capable available cloud LLM for all agentic tasks, regardless of complexity. But this power comes at a cost, as token consumption scales with alert volume in ways that weren't modeled in the business case. Meanwhile, network telemetry containing sensitive internal asset and behavioral data is transmitted to external cloud APIs, raising data residency, AI governance, security, and dependency concerns among GRC and legal teams.

THE CAUSE

Architectural overkill

Not every task requires or benefits from frontier-level capability. Many basic logic and reasoning tasks such as triage can be handled perfectly well with a small model with localized inference. Going with a high-end cloud LLM for the full spectrum of agentic operations increases both cost and GRC exposure without a corresponding improvement in performance.

HOW TO AVOID IT

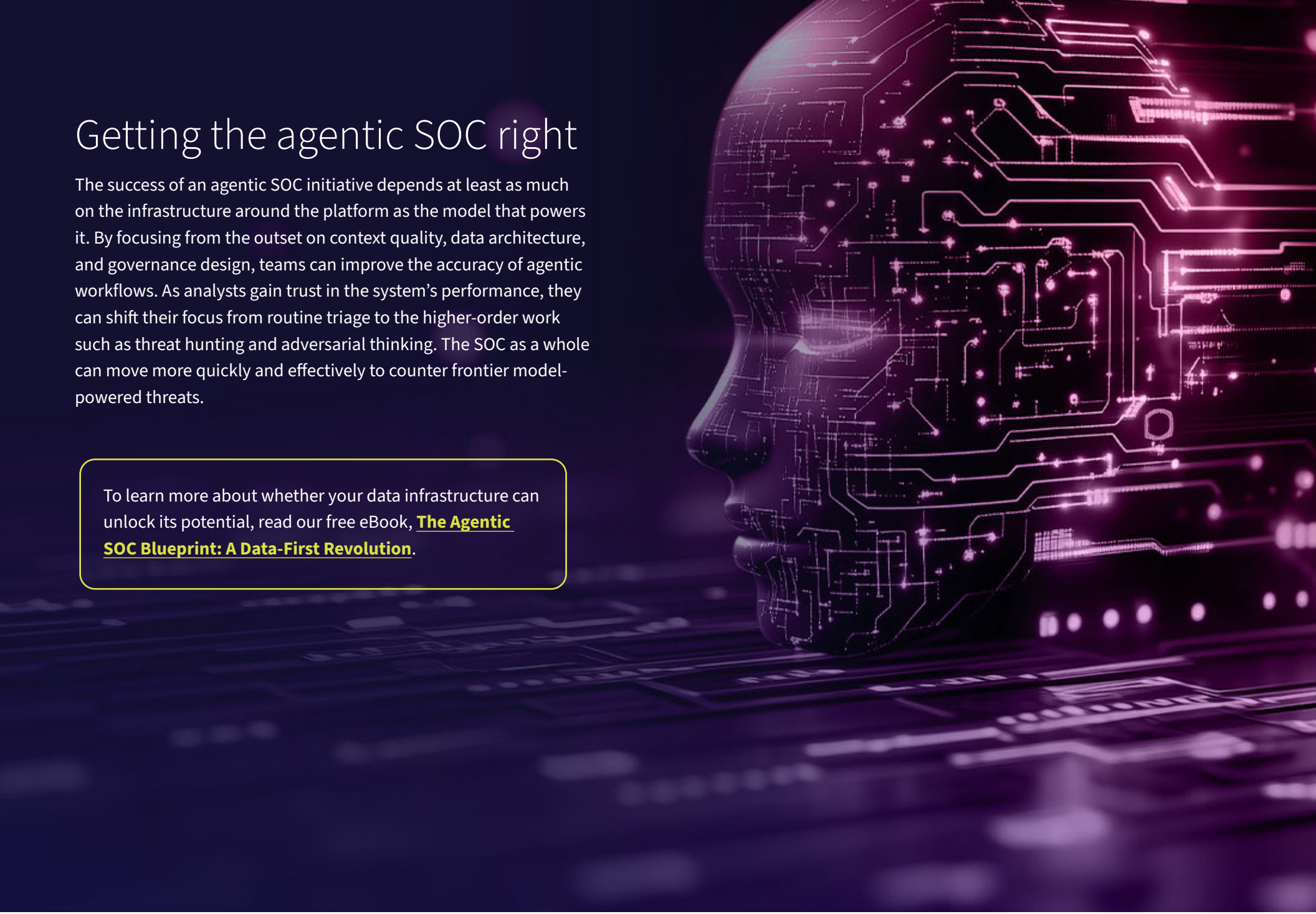
Match the model to the task

Architectural decisions should be made strategically. Triage agents and routine enrichment can run on smaller, locally deployed models where inference cost is low and latency is minimal. Data kept inside the perimeter remains fully under the organization's control, reducing the attack surface and third-party dependency. Cloud LLMs should be reserved for tasks where their power can make a material difference, such as investigation synthesis, hypothesis generation, and complex multi-source correlation.

Getting the agentic SOC right

The success of an agentic SOC initiative depends at least as much on the infrastructure around the platform as the model that powers it. By focusing from the outset on context quality, data architecture, and governance design, teams can improve the accuracy of agentic workflows. As analysts gain trust in the system's performance, they can shift their focus from routine triage to the higher-order work such as threat hunting and adversarial thinking. The SOC as a whole can move more quickly and effectively to counter frontier model-powered threats.

To learn more about whether your data infrastructure can unlock its potential, read our free eBook, **The Agentic SOC Blueprint: A Data-First Revolution**.



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**.

EXTRAHOP®