SALT

CISO Guide: 10 Questions to Ask Before Deploying Agentic AI

How to Assess Risk, Readiness, and Responsibility in the Age of Autonomous Systems

SALT

Introduction

Agentic AI is no longer theoretical. Organizations are already integrating autonomous agents into internal workflows, customer-facing services, and complex decision-making processes. These systems don't just generate responses, they reason, retain memory, trigger actions, and connect with sensitive systems via APIs.

For CISOs, this presents a major inflection point. Traditional controls built for human users, static applications, or passive models are not sufficient. Securing agentic AI requires a new mindset, a new risk framework, and a sharper focus on the systems that enable autonomy. This is especially true for the Model Context Protocol (MCP) API broker and the APIs agents rely on.

This guide presents 10 critical questions every CISO should ask before approving, deploying, or scaling agentic AI inside the enterprise.

1. What decisions will the agent make without human oversight?

Understanding autonomy boundaries is essential. Will the agent initiate actions, update records, send messages, or make approvals on its own? If so, what controls and auditing mechanisms are in place?

2. How is the agent's context defined, updated, and secured?

Agent behavior is driven by its context around goals, permissions, memory, tools, and environment. This is typically passed through the Model Context Protocol. Who controls this context, and how is it validated to prevent manipulation?

3. What APIs will the agent be allowed to access?

Autonomous agents act through APIs. Have you mapped which APIs are exposed to the agent and under what conditions? Can the agent chain calls across services? Are those APIs permissioned, rate-limited, and monitored?

4. Can agent behavior be audited over time, not just per request?

Because agents operate continuously and adapt based on feedback, their risk profile evolves. Do you have the ability to trace API behavior, context shifts, and execution history across sessions?

SALT

5. What happens if an agent is given malicious or misleading input?

Agents that rely on user prompts, documents, or system-generated context are vulnerable to prompt injection, context poisoning, and memory manipulation. How does your system detect and respond to malicious influence?

6. Who governs agent roles, capabilities, and escalation paths?

Agents often have roles with predefined tools or scopes. Is there a governance framework for who creates agents, assigns tools, defines escalation logic, or restricts risky operations? Is access reviewed regularly?

7. How do we test and red team agent behavior before production?

Do you have a framework to simulate adversarial use cases like recursive agent loops, context hijacking, or tool misuse? Traditional penetration testing is not sufficient for agentic behavior.

8. Can we detect when an agent drifts from its intended behavior?

Agents may gradually evolve due to memory, user input, or unclear boundaries. Are you monitoring for behavioral drift or emergent patterns that differ from the agent's original goal?

9. How are agent-triggered actions reviewed, revoked, or rolled back?

In the event of a mistake or misuse, is there a clear way to reverse actions or revoke access? Can you quarantine an agent session, block further API use, or suspend its operation in real time?

10. What visibility does security have into agent-API interactions?

API security is your strongest enforcement layer. Can your team monitor which APIs the agent is calling, what data is involved, and whether the behavior is consistent with its defined context?

Why API Security Is Central to All 10

Every agentic action, from sending a message to modifying a record, is executed via an API.

That makes API security not just a supporting control, but the core enforcement layer for agent governance.

Salt Security provides the visibility, anomaly detection, and context correlation needed to:

- Understand what your agents are doing
- Detect when they drift from expected behavior
- Protect the APIs that connect your agents to the real world

Conclusion

Agentic AI has incredible potential — but it also brings a new kind of operational risk. The agents you deploy are not just software. They are actors making decisions, accessing systems, and triggering workflows that impact your business in real time.

As a CISO, your responsibility is not just to secure the endpoints and the network, but to understand and govern the logic that drives autonomous systems.

At the heart of that logic is the Model Context Protocol, and at the edge of every decision is an API.

Make sure you can see, control, and secure both.

To learn more, request a free demo here: Free Agentic AI Security Demo