AI Runtime Security Buyer's Guide

How to Evaluate Controls that Protect AI Applications and Agents, Govern Spend, and Keep Data Under Your Control



FORTINET

Introduction

An AI assistant can help employees with many tasks and projects, but without adequate controls, it may also expose sensitive information or act on malicious instructions hidden in retrieved content. As AI applications and agents become part of everyday work, security teams must protect AI interactions without negatively impacting legitimate tasks.

Inadequate AI runtime security can leave interactions unprotected, send sensitive content through an unexpected inspection path, or interrupt work when spending limits are reached. When choosing AI runtime security, buyers need to know where policies apply, what they cover, where data goes, and how costs are controlled.

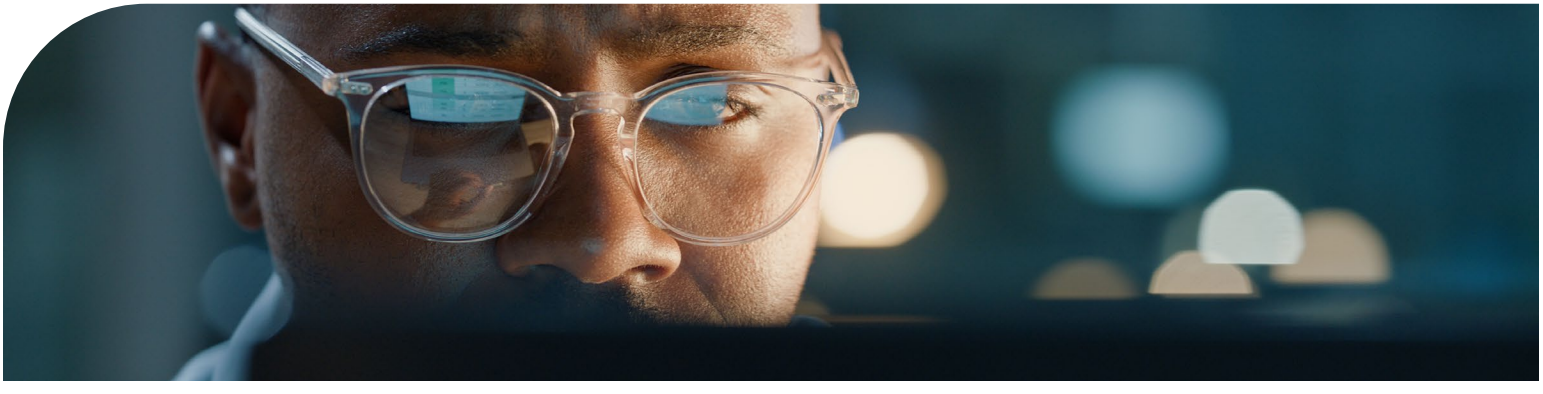


What Is AI Runtime Security?

AI runtime security inspects and enforces policies while connected AI applications and agents exchange information with models and tools. Depending on the solution, inspection may cover prompts, responses, retrieved content, and Model Context Protocol (MCP) tool lists, calls, and responses. These interaction points matter when an agent encounters untrusted instructions or passes sensitive information to another system.

An AI runtime control point can help identify malicious instructions, detect or redact sensitive information, block policy-violating interactions, and record decisions. Protection is not universal: Coverage depends on which applications, agents, models, and interactions are connected or routed through it. External model providers may still receive and process requests.

That distinction matters when evaluating solutions. The question is not simply whether a vendor offers AI guardrails, but whether its controls apply to the workflows, data types, tools, and deployment arrangements an organization actually uses.



Five Key Considerations When Selecting AI Runtime Security

1 Match runtime controls to the workflow

An AI assistant preparing a salesperson for a meeting, for example, may use account history, support cases, and approved product information. The challenge is protecting sensitive content in that workflow without preventing the assistant from doing its job. Runtime controls must apply the right model policies and inspect the information that is exchanged. Permissions to access the underlying CRM, files, or other source systems should remain the responsibility of identity and access management.

When evaluating a solution, it's important to map the models, data types, tools, and MCP servers used by the most important workflows. Some teams may need several approved models, while others need a narrower selection. The solution should be tested on whether policies can vary by team and detect and protect sensitive content, including credentials and API keys, as it passes through connected AI interactions.

An impressive integration list offers little reassurance if the most important workflow cannot be protected.

Key questions to ask the vendor are:

- Which applications can route requests through the control point?
- What changes are required to connect them?
- Does configuration require custom code?

The solution should be able to inspect text, images, audio, video, and code where relevant. Hidden instructions can appear in files or images, which means that an uninspected screenshot can expose a customer record.

2 Inspect agent interactions, not just prompts

The risk changes when an agent moves from reading information to using tools. Untrusted content can influence a tool request, including one that sends data beyond the original task. A runtime control should inspect these exchanges and enforce content-based policies before unsafe requests proceed. Blocking a tool call because it contains sensitive data is not the same as deciding whether the agent is authorized to perform a business action.

For example, if an employee asks an agent to summarize messages, but one email contains hidden instructions to retrieve and send a customer file, the attack originates in retrieved content, not in the employee's request. An effective solution can detect the injected instruction in the tool response and independently inspect the outbound tool call for sensitive data.

A warning that arrives after a sensitive file has been sent does not provide the same protection as stopping the request. The solution should be evaluated for its ability to block or redact requests before they reach a model or tool, including any impact on legitimate workflow latency. A demonstration should show the actual enforcement decision, not only an alert after the event.

3 Make governance enforceable and auditable

A developer assistant, customer-service application, and financial-analysis workflow face different risks. A single set of rigid rules may obstruct useful work, while separately managed controls can make oversight difficult. Teams need to maintain distinct policies, approved models, budgets, and logs within separate environments, with roles controlling who can view, change, or approve them.

Frameworks such as the NIST AI Risk Management Framework, relevant ISO standards, and OWASP guidance can inform requirements; mappings alone do not prove compliance. Organizations should ask:

- How quickly can a team establish a governed baseline using predefined policies?
- Can those policies be customized?
- How are exceptions and violations recorded?

Audit records show how an incident unfolded, but they can also become another place where sensitive information accumulates. Prompts, responses, and logs may contain confidential content. It's important to understand what is retained, where logs reside, who can access them, whether access to original prompt and response content is restricted, and how records reach existing security systems. The goal is to capture useful evidence without creating an uncontrolled second store of sensitive data.

4 Trace data across the entire deployment

A solution described as "self-hosted" does not necessarily keep every AI request inside your environment. Data sovereignty depends on where prompts, responses, logs, and related information are inspected, processed, and stored. A self-hosted runtime control point can keep inspection, policy

management, and logging within an organization's environment; a cloud-delivered guardrail may send interaction content to its vendor for inspection. Neither choice determines where an external model provider processes requests.

It's important to evaluate the inspection path and the model path. Some workflows may permit external models; others require self-hosted processing. Policies should be able to restrict model selection and redact sensitive content before an external request is sent. If the solution routes requests across model groups or fails over to backup models, you should confirm that every model that could serve a request meets your data-handling requirements.

The vendor should be able to provide: a deployment-specific data-flow diagram showing the control, management, and data planes; where logs are stored; where inspection runs; where models process requests; and any third-party services. This will establish which prompts, responses, and logs leave your environment, for what purpose, and under whose control.

5 Govern spending with clear limits and operational trade-offs

A useful AI workflow can become expensive when an agent repeats multistep requests, employees use higher-cost models, or poorly constructed prompts trigger unnecessary processing. Malicious prompts and Denial-of-Wallet attacks can also consume tokens or compute. Buyers need to understand where costs originate and how spending limits can contain unexpected or excessive usage.

Budgets and model-selection policies involve trade-offs. A hard spending limit can prevent further requests once the configured cap is reached, but it may not prevent the individual request that causes an unexpected expense. Budget windows can be as short as 24 hours to help contain runaway spending.

The questions to ask when evaluating governance include:

- Which requests stop when a limit is reached?
- Who can change the budget?
- How are interrupted workflows handled?

Historical token reports are useful, but they cannot prevent unexpected spending or explain why a critical workflow stopped. Organizations should compare whether security, finance, and application owners can attribute spending, enforce limits, and understand the operational impact when a budget cap stops requests.

The Ideal Solution: Six Capabilities to Compare

When evaluating solutions, teams should consider the following six capabilities using the AI workflows they actually operate. The vendor should provide evidence of what each solution inspects, enforces, records, and manages, including its limitations and deployment effort. Then, organizations should consider whether these functions operate under coordinated policies or require separate tools and logs. The right comparison tests actual workflows and the people responsible for running them, not just advertised feature lists.

1 Coverage

Effective inspection requires support for the text, image, audio, video, and code formats relevant to the organization's workflows. Coverage limitations should also be evaluated. Poisoned tool descriptions can influence an agent before any tool call, so a model-integration list alone does not establish coverage.

2 Permissions

Policies can vary across teams or workspaces, including approved models, budgets, and the owners of requests. Evaluators need to distinguish model access policies and content-based tool-call enforcement from source-system permissions, which remain with identity and access management. A runtime control point does not necessarily supply per-agent action authorization or human approval workflows.

3 Enforcement

Effective enforcement acts during the interaction, not only after it. Alerts, denial of unsafe requests, sensitive-data redaction, and, where available, replacement with configurable replacement text that allows legitimate work to continue should be tested. For tool calls, sensitive content must be blocked before transmission. False positives, latency, and detection should be measured as they all affect enforcement.

4 Governance

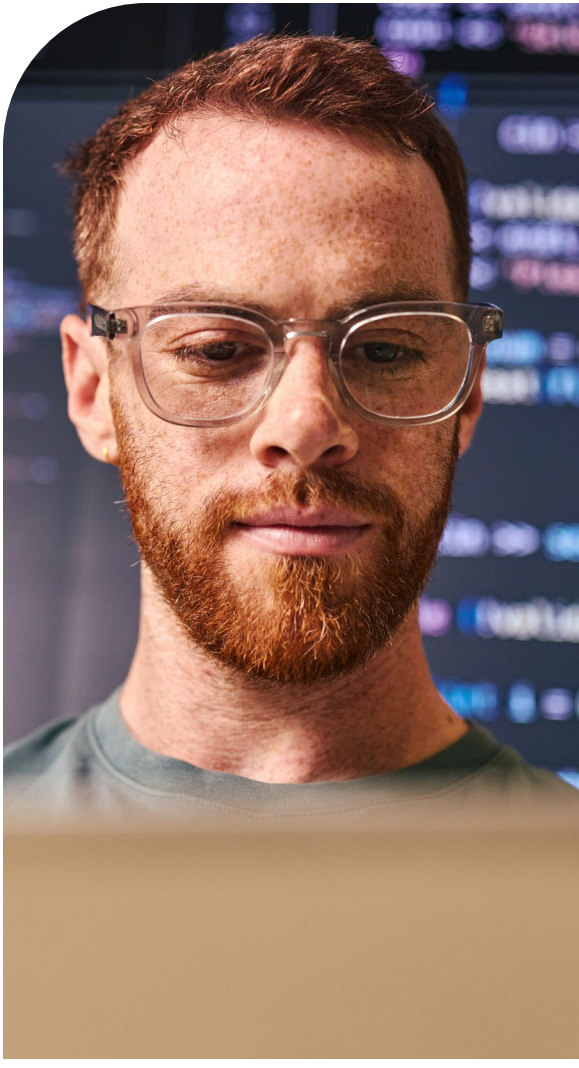
Adaptable policies, separate administrative roles, and auditable decisions need to be available. And predefined governance and security policies should be customized and deployed quickly. To help demonstrate compliance, log retention, access controls, sensitive-content handling, and export to existing security tools need to be built in.

5 Cost

Usage and cost should be able to be attributed to users, teams, projects, applications, or models. Things to test during an evaluation include how spending caps are configured, what happens when a cap is reached, and who can update the limit. Short budget windows should help contain unexpected spending, including costly requests caused by errors or abusive activity. In addition, rate limits should restrict request and token volume. The operational impact of hard spending limits should also be assessed.

6 Visibility

Discovery of AI destinations should be distinguished from inspection of interaction content. It's important to find out how the vendor identifies unapproved AI use and agents, whether network integrations can send detected AI traffic for content inspection, and whether resulting policy decisions can affect future network enforcement. Inventory visibility is not proof that every interaction is inspected.



FORTINET

Five Questions to Ask Vendors and Why They Matter

1. **Which AI applications and agents can you discover, inspect, and protect in practice?**

   AI runtime security vendors should be clear about how they discover AI destinations and agents, and which of their interactions actually reach an enforcement point. When evaluating a solution, teams should test one representative multistep workflow: prompt, retrieved content, MCP tool lists and descriptions, tool calls, and responses. Hidden instructions should be included in a tool response and sensitive information in an outbound call. Evidence of detection, blocking or redaction, false positives, latency, supported formats, uncovered paths, and integration effort should be delivered. Discovery alone is not enforcement.

2. **Can one control point apply security, governance, and spending policies across teams?**

   To determine the answer, evaluators should use two workspaces with different approved models, sensitive-data policies, and budgets. The vendor should demonstrate a permitted request, a policy denial, a sensitive-data tool-call block, and what happens when a spending cap is reached. Further, it's key to determine who can change each policy and whether these controls share administration and evidence or require separate guardrail, DLP, and cost tools. Source-system permissions should remain with identity and access management; content-based blocking is not agent-action authorization.

3. **What evidence will our security and governance teams receive after a policy violation?**

   Teams should ask for a sample record from a blocked interaction, including the policy, timestamp, responsible workspace or application, and available context. This lets them verify who can access the evidence, where it is stored, who can view original prompt and response content, and how records reach existing

security tools. A dashboard screenshot is not a substitute for an investigation trail.

4 Where do our prompts, responses, and logs go with each deployment option?

The vendor should provide a diagram for the proposed deployment, not a generic architecture slide. It should show where inspection runs, where control and management functions reside, where logs are stored, and whether prompts or responses go to the security vendor's cloud. Separately, teams should identify external model processing and any redaction or model restrictions before requests leave your environment.

5 Can you attribute AI spending, enforce limits, and explain what happens at the cap?

Vendors should demonstrate usage and cost by user, team, project, application, and model. A spending cap should be configured to show what happens when it is reached, how an administrator changes the limit, and which workflows are interrupted. The length of the budget window and whether rate limits can restrict excessive usage should be clear. In sum, evaluators need to know whether a costly request can complete before the spending cap prevents further requests.





Choose Controls You Can Validate

When evaluating AI runtime solutions, organizations should start with the AI applications and agents that matter most, map their interaction paths, and use the five preceding questions to compare solutions. Demonstrated inspection and enforcement, practical governance, transparent data flows, and measurable spending controls are more indicative of capabilities than feature lists.

FortiAIGate is a self-hosted AI gateway for large language models, agents, and MCP tools. From one control point, FortiAIGate stops AI attacks and data leaks, enforces usage policy, and caps AI spend. Inspection, policy management, and logs stay in your environment.

Learn more about securing and controlling AI



www.fortinet.com

Copyright © 2026 Fortinet, Inc. All rights reserved. Fortinet®, FortiGate®, FortiCare® and FortiGuard®, and certain other marks are registered trademarks of Fortinet, Inc., and other Fortinet names herein may also be registered and/or common law trademarks of Fortinet. All other product or company names may be trademarks of their respective owners. Performance and other metrics contained herein were attained in internal lab tests under ideal conditions, and actual performance and other results may vary. Network variables, different network environments and other conditions may affect performance results. Nothing herein represents any binding commitment by Fortinet, and Fortinet disclaims all warranties, whether express or implied, except to the extent Fortinet enters a binding written contract, signed by Fortinet's SVP Legal and above, with a purchaser that expressly warrants that the identified product will perform according to certain expressly-identified performance metrics and, in such event, only the specific performance metrics expressly identified in such binding written contract shall be binding on Fortinet. For absolute clarity, any such warranty will be limited to performance in the same ideal conditions as in Fortinet's internal lab tests. Fortinet disclaims in full any covenants, representations, and guarantees pursuant hereto, whether express or implied. Fortinet reserves the right to change, modify, transfer, or otherwise revise this publication without notice, and the most current version of the publication shall be applicable.

October 22, 2026 2:48 PM / MKTG-4482-0-0-EN