MCP SERVER | NO CODE | CLOUD HOSTED

Semgrep MCP

Govern code security and compliance directly from chat.

Semgrep lets your AI client read and write directly to Semgrep's security platform. It gives you the ability to audit code vulnerabilities, analyze specific flaws, mark findings as fixed or false positives, and deploy custom semantic rules without leaving your chat window.

A+ Quality Score 100/100

sast | sca | code-security | vulnerability-management | static-analysis | devsecops

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/semgrep

SCAN TO VIEW

The connectivity layer between AI and the world's software.

Vinkius sits between AI and every application. All communication passes through Vinkius Cloud via the Model Context Protocol (MCP) — with governance, observability, and security at every layer.

Your AI Connections Run Through Vinkius Cloud

The world's largest managed MCP catalog

Vinkius is the connectivity layer where AI connects to the software your business already runs. We handle the hosting, the security, the credentials, the uptime — you get agents that actually do things.

We operate the world's largest managed MCP catalog. Major SaaS platforms, CRMs, databases, and cloud providers — running, monitored, production-ready. This MCP server is hosted and maintained by the Vinkius Cloud for AI Agents.

> *The agent doesn't manage credentials, doesn't manage uptime, doesn't manage security. Vinkius does.*
>
> — Architecture principle

Four Pillars of the Vinkius Runtime

01 — Security by design

Credentials stay encrypted at rest via AES-256. The AI agent never touches raw keys — they're injected into a sandboxed V8 isolate at runtime. Actions are logged, and connections have an emergency kill switch.

02 — Built on MCP Fusion

This MCP server was built with **MCP Fusion**, the open-source framework (Apache 2.0) that powers the entire Vinkius catalog. Schema-as-firewall strips undeclared fields, compiled PII redaction runs at zero overhead, and cryptographic lockfiles produce git-diffable audit trails.

03 — Deterministic observability

Eight immutable metrics per endpoint: request volume, p95 latency, error rate, active connections, cost attribution. A live payload feed logs every tool call with mutation detection.

04 — Autonomous operations

Servers are deployed, monitored, and patched autonomously. New capabilities and security patches ship weekly. Zero-downtime deployments ensure continuous availability across all managed MCP servers.

AES-256	Ed25519	24h TTL	V8 Isolate
Encryption at rest	PKI vault signatures	Ephemeral session keys	Sandboxed execution

One Token. Instant Access.

Every MCP server on Vinkius is accessed through a **Connection Token**. Tokens are generated in the cloud dashboard and produce a unique MCP endpoint URL. Paste this URL into any MCP-compatible client — no SDK required.

A single token can serve **multiple AI clients simultaneously**, or you can issue separate tokens per client for granular access control. Each token tracks its own request count, last activity timestamp, and can be individually enabled or revoked.

MCP ENDPOINT

```
https://edge.vinkius.com/{token}/mcp
```

Claude · Cursor · VS Code · Windsurf · Grok · Gemini

Security Is the Architecture

Security in Vinkius is not a feature — it's the foundation of the runtime. The gateway enforces multiple independent protection layers between AI agents and third-party APIs.

01 — Ed25519 PKI Vault

Every workspace has an Ed25519 Master Key. Session keys are generated ephemerally (24h TTL) and signed by the Master Key. Credentials never leave the vault boundary.

02 — V8 Isolate Sandboxing

Tool code runs inside isolated-vm V8 isolates with 64 MB memory caps and per-request timeouts. No filesystem access, no network access except through the SSRF-guarded fetch bridge.

03 — SSRF Guard

All outbound HTTP requests are DNS-resolved and validated before execution. Private IP ranges (10.x, 172.16-31.x, 192.168.x, AWS metadata 169.254.x) are blocked at the network layer.

04 — DLP & PII Redaction

A ResponseGuard pipeline intercepts every tool response. Configurable redaction patterns strip sensitive fields (emails, SSNs, card numbers) before data reaches the AI agent.

05 — Cryptographic Audit Trail

Every request is signed into a SHA-256 hash chain with Ed25519 signatures. Events form a tamper-proof, SIEM-exportable forensic record.

06 — Honeytoken Trap System

Phantom credentials are injected into isolated environments. If a honeytoken is used outside Vinkius infrastructure, the server is quarantined instantly.

Emergency Kill Switch

EU AI Act Art. 14(1) Compliant

The kill switch is an **emergency halt** mechanism — not a simple toggle. When triggered, it executes three actions atomically:

01 — Server deactivated

The MCP server is immediately taken offline across the entire cluster.

02 — All tokens revoked

Every connection token is invalidated. Total lockout — reconnection blocked until new tokens are issued.

03 — WebSocket connections killed

Active connections terminated via Redis pubsub broadcast. Propagates to every runtime node in the cluster.

Full Visibility. Zero Guesswork.

The Vinkius cloud dashboard includes a full MCP Governance suite — real-time analytics and security controls for production AI operations.

Control Plane

KPI dashboard with request volume, latency, success rate, token consumption, and AI-generated operational briefings.

Agent Activity

Which AI clients are connecting, how often, and what they're doing — real-time session tracking.

FinOps

Cost tracking per tool, payload compression savings, budget optimization signals, and consumption trends.

Tool Health

Slowest and most error-prone tools, with actionable root-cause insights and performance baselines.

Firewall & DLP

PII redaction activity, sensitive data protection counters, and security event timeline.

Incident Log

Error trends, failure rates, status-code breakdowns, and forensic audit trail access.

> Get started at **cloud.vinkius.com** — connect your AI agent in under 60 seconds.

Semgrep MCP

10 tools available
Cloud-hosted on Vinkius

Managing code security shouldn't mean abandoning your IDE for a web dashboard. This MCP connects your AI agent directly to Semgrep's AppSec platform, letting you audit security findings right where you work. Instead of copying vulnerability details into a ticket and waiting for a human to triage it, your agent can pull the latest CI scan results, analyze the bad code snippet, and instantly update its status—whether that means marking it as fixed or confirming it's a false positive. You can also use it to enforce custom security standards by having your AI client write and deploy new semantic rules across all your repositories. By connecting this MCP via Vinkius, you give your agent access to the full catalog of code quality tools, accelerating compliance auditing instantly.

Core Capabilities

01 — Update finding status

Mark specific security findings as fixed, ignored, false positives, or mitigated directly from your chat.

02 — Deploy custom rules

Create and deploy new semantic security rules to forbid newly discovered bad coding patterns across the entire organization.

03 — Delete existing rules

Remove obsolete or unnecessary custom security rules from your active deployment set.

04 — Fetch all security findings

Retrieve a global list of static analysis vulnerabilities, including file lines and severity levels, for any given project deployment.

05 — Get flaw details

Analyze an individual vulnerability to see the exact malicious code block, suggested fixes, and associated CVE data.

06 — List monitored projects

See a list of all repositories and projects currently being scanned by Semgrep within your organization's deployment scope.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/semgrep** — connect your AI agent in three steps.

01 Enable the Semgrep MCP through Vinkius, then supply an API token from your Semgrep Dashboard settings.

02 Engage your AI agent in any MCP-compatible client and prompt it to analyze a security report or check compliance metrics.

03 The agent executes specific actions—like updating findings status or listing rules—and returns actionable data directly into the chat thread.

The bottom line is you get real-time, auditable control over your code's security posture without ever leaving your AI interface.

Built For

Security Engineers and DevOps staff who are tired of context switching between multiple dashboards. It’s for the developer needing to stop clicking through tickets just to verify a fix.

AppSec Engineer

Uses this MCP to quickly delete obsolete security rules or audit compliance metrics across all deployments in one chat command.

DevOps Engineer

Retrieves global AppSec performance metrics and pipeline fix rates, compiling executive summaries for stakeholders instantly.

Software Developer

Lets the agent fetch a specific finding ID blocking a pull request, explains the vulnerability meaning, and drafts the precise semantic code fix needed to pass the scan.

What Changes When You Connect

01 Stop context switching. Instead of hopping between the IDE, Semgrep dashboard, and Jira, your AI agent manages everything—from fetching findings to updating their status with a single command.

02 Accelerate triage dramatically. Use `list_findings` followed by `get_finding_details` so you don't have to copy-paste raw vulnerability data; the details appear right in the conversation.

03 Enforce policy on demand. Need to block a new, bad coding practice? You can use `create_rule` to write and deploy a custom semantic rule instantly across all your repositories.

04 Simplify compliance reporting. Run `get_metrics` to pull fix rates and overall AppSec statistics, then pipe that data directly into an executive summary report without manual export/import steps.

05 Clear up the backlog fast. If you confirm a vulnerability is irrelevant or already patched, use `update_finding_status` to change its state permanently, cleaning up developer queues.

Real-World Applications

A PR blocks because of an unknown dependency flaw.

The agent fetches the findings list and uses get_finding_details on a specific ID. It explains to the developer exactly why the vulnerability exists, links to CVE data, and suggests the precise code change needed for the fix.

Quarterly compliance audit requires proof of patch rates.

The DevOps user asks the agent to get_metrics. The AI client returns a detailed report showing the overall Fix Rate and the median time-to-resolve critical vulnerabilities, which is perfect for an executive meeting.

An old, unused security rule needs removal.

The AppSec Engineer instructs the agent to list_rules first. After identifying the obsolete pattern, they use delete_rule to take it out of service globally without logging into the web interface.

Need to quickly confirm a reported bug is actually harmless.

The developer runs an initial check using list_findings. They then use update_finding_status on the specific finding ID, marking it as 'false positive' and logging the action for audit purposes.

Patterns to Avoid

Manual Dashboard Hunting

✗ AVOID

A developer has to manually copy vulnerability details from the Semgrep UI into a ticket, then wait for an engineer to log back in and manually update the status.

✓ INSTEAD

Instead, tell your agent to list_findings first. Then use get_finding_details on the specific ID you care about. Finally, use update_finding_status right away to mark it as 'false positive' or 'fixed'.

Ignoring Scope Limitations

✗ AVOID

Trying to run a rule check without specifying which deployment is affected, leading to ambiguous results and errors.

✓ INSTEAD

Always start by using list_deployments to identify the correct scope. All subsequent operations (like creating rules or listing findings) must be scoped correctly.

Writing Rules in Code Comments

✗ AVOID

A developer writes a security requirement as a comment, hoping it gets noticed by auditors later.

✓ INSTEAD

For actual enforcement, use create_rule. This tool deploys a semantic rule that automatically checks and enforces the pattern across your entire codebase.

The Right Fit

Use this MCP if your core pain point is context switching related to code security. You need an AI agent to interact with Semgrep's complex data—fetching findings, modifying statuses, or deploying rules—all through a chat interface. This is ideal for teams that value speed and auditability over manual dashboard navigation.

Don't use this if you just need general API access to pull raw data

dumps; the MCP provides structured, actionable workflows (e.g., `get_metrics`, which aggregates performance). If your only goal is to read a list of available rules without taking action, simply listing them is enough. But if you need to *act* on that information—like marking a finding or creating a new rule—this MCP is what you need.

Dealing with Security Findings Feels Like Juggling Tabs

Right now, dealing with security vulnerabilities means context switching: you open your IDE to write code. When the build fails, you jump to the Semgrep dashboard to read findings. Then, if a team member asks about it, you copy data into Jira or Slack. You spend more time managing the workflow than actually fixing the bug.

With this MCP connected via Vinkius, that process collapses. Your AI agent pulls the raw finding details directly from Semgrep and gives you an analysis right in your chat. You can then use `update_finding_status` to mark it as fixed without ever leaving your conversation.

Deploying Rules with Semgrep MCP

Manually deploying a new security rule requires navigating the platform, writing YAML, testing against a staging environment, and then finally pushing it to production. It's slow and prone to human error.

Now, you can have your agent draft or adjust the semantic rule definition using `create_rule`. You get immediate feedback and deployment control from chat. The enforcement is instant.

Semgrep with 10 Tools

These tools allow you to audit security issues, view compliance metrics, create new enforcement rules, and mark vulnerability findings directly through your AI agent.

#	TOOL	DESCRIPTION
01	`create_rule`	Deploys a customized security rule that forbids specific bad coding patterns across your enterprise repositories.
02	`delete_rule`	Removes an existing custom Semgrep security rule from the deployment scope entirely.
03	`list_deployments`	Lists all defined organizational deployments, providing necessary slugs to define the scope for other security operations.
04	`list_findings`	Fetches a comprehensive list of global static analysis vulnerabilities within a specified deployment's code base.
05	`get_finding_details`	Retrieves deep, atomic information on a specific flaw, including malicious code blocks, suggested fixes, and CVE links.
06	`get_project`	Searches for an exact Semgrep project using its precise repository name to scope security queries.
07	`list_rules`	Displays all current semantic rules that are actively deployed globally across your codebase.
08	`get_metrics`	Gathers AppSec metrics and compliance statistics, ideal for generating high-level executive security dashboards.
09	`list_projects`	Lists all monitored repositories or projects within a defined deployment scope over time.
10	`update_finding_status`	Marks the state of a specific vulnerability finding, changing it to 'fixed', 'false positive', or 'ignored'.

See It in Action

Real prompts you can use once this MCP is connected to your AI agent through Vinkius Cloud.

U List the most severe unmitigated findings currently breaking our CI/CD pipeline on the 'vinkius/cloud' repository.

I've fetched the findings list scoped to 'vinkius/cloud'. Currently, there are 2 blockers. 1) Exposed hardcoded AWS Secret Access Key in `auth.py:L45`. 2) Outdated dependency `lodash@4.17.15` in `package.json` vulnerable to prototype pollution. Would you like me to resolve the Lodash one by generating an update commit script?

U Mark vulnerability issue ID #58032 as a 'false_positive' using the update finding tool.

The operation is complete. I've sent the mutation `update_finding_status` for issue 58032 within your deployment. Semgrep now permanently recognizes this event as `false_positive`, and subsequent scans will not flag this specific semantic line match again.

U Review the company's Semgrep performance metrics focusing on fix rate.

I've analyzed your AppSec timeline with `get_metrics`. Your overall Fix Rate for the trailing 30 days is hovering around 83%. The median 'time-to-resolve' for critical SAST vulnerabilities is 3.4 days. This shows strong engagement natively within developer pull requests before merge.

Frequently Asked Questions

01 **How do I use Semgrep MCP to check my overall security health?**

You run the list_findings tool, specifying the target deployment slug. This retrieves a comprehensive report of all vulnerabilities found across the code base in one go.

02 **Can I update finding status using Semgrep MCP?**

Yes, you use the update_finding_status tool. You just need to provide the specific finding ID and the desired status (e.g., 'false_positive').

03 **What is the best way to review compliance data with Semgrep MCP?**

To get high-level stats, use the get_metrics tool. This returns AppSec performance metrics and overall compliance statistics for executive reporting.

04 **Does Semgrep MCP help me write new security rules?**

Yes, you can use create_rule to deploy a custom semantic rule. You define the pattern once, and it enforces that rule across all your repositories.

05 **Which tool do I use to find out which projects Semgrep is monitoring?**

Use list_projects. This tool reads the monitored repository list for a specific deployment scope, giving you visibility into your entire security footprint.

Go Live in 60 Seconds

Get your connection token from **cloud.vinkius.com**, then paste the endpoint URL into any MCP-compatible client.

YOUR MCP ENDPOINT

```
https://edge.vinkius.com/[TOKEN]/mcp
```

CLIENT	WHERE TO CONFIGURE
Claude AI	Profile → Customize → Connectors → "+" → Add custom connector → Paste endpoint
Cursor	Settings → Features → MCP Servers → "+ Add New MCP Server" → Type: SSE → Paste endpoint
VS Code	Ctrl/Cmd+Shift+P → "MCP: Add Server" → add `"semgrep": { "url": "…" }`
Windsurf	MCP Settings → `mcp_settings.json` → Add endpoint URL
ChatGPT	Settings → Tools & plugins → Add MCP server → Paste endpoint
Gemini	Extensions → Add MCP Server → Paste endpoint URL

ASK AN AI ABOUT THIS

Let your preferred AI explain this MCP server

- Ask ChatGPT
- Ask Claude
- Ask Perplexity
- Ask Gemini
- Ask Grok

READY TO CONNECT

Semgrep is live on Vinkius Cloud.

Get your connection token, paste it into your AI agent, and start building. No SDK. No deployment. Just results.

Start at cloud.vinkius.com →

vinkius.com · support@vinkius.com

INDEPENDENT PLATFORM DISCLAIMER

Vinkius is an independent platform and is not affiliated with, endorsed by, sponsored by, verified by, or otherwise authorized by Semgrep. All third-party trademarks, logos, and brand names are the property of their respective owners. Their use in this document is strictly for informational purposes to identify service compatibility and interoperability.

DOCUMENT INFORMATION

Generated	June 2026
MCP Server	Semgrep MCP
Server ID	`019d7605-b00e-71f5-ab3f-aa8a74304cf7`
Platform	Vinkius Cloud for AI Agents
Endpoint	`https://edge.vinkius.com/{token}/mcp`

LICENSE & USAGE

This document is generated automatically by the Vinkius PDF Engine. Content reflects the MCP server configuration at the time of generation and may change as updates are deployed. For the most current information, visit vinkius.com/mcp/semgrep.