MCP SERVER | NO CODE | CLOUD HOSTED

StackHawk MCP

Automate vulnerability scans and triage alerts.

StackHawk connects your AI client to the StackHawk DAST platform. This MCP lets you run automated security scans, find vulnerabilities, and manage alerts without leaving your chat interface. It turns complex security protocols into simple natural language commands for effortless risk assessment.

A+ Quality Score 100/100

dast | application-security | security-testing | vulnerability-management | ci-cd-pipeline | automated-scanning

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

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.

StackHawk MCP

10 tools available
Cloud-hosted on Vinkius

Security testing shouldn't mean juggling dashboards and running command-line tools just to check for basic vulnerabilities. This connector gives your AI client direct access to StackHawk's dynamic application security testing (DAST) capabilities. You can ask your agent to assess a live environment, list all registered applications, or get the full details of a specific scan run using simple conversation.

When you need to check for threats, you don't have to manually navigate through multiple reports. Simply instruct your AI client to find critical alerts from a recent test and then classify them—say, marking a false positive or accepting the risk. This capability accelerates remediation across modern CI/CD pipelines. All this power is accessible through Vinkius, making it one place for all your connected services. Your agent handles the complex authentication and data retrieval so you just get actionable security insights.

Core Capabilities

01 — Running Automated Security Scans

Start comprehensive DAST audits against specific environments or halt running scans when they are finished.

02 — Auditing Application Assets

Retrieve a complete list of all monitored applications and the different operational environments (like Staging or Production) for any given app.

03 — Retrieving Vulnerability Reports

Fetch detailed metadata about past scans, or download individual security alerts to understand exactly what was found.

04 — Managing and Classifying Alerts

Instruct the system to review a specific vulnerability alert and assign it a status like 'false positive' or 'risk accepted'.

One Click on Vinkius — From Prompt to Execution

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

01 First, authenticate your connection by using the `login` tool to get a valid bearer token for StackHawk.

02 Next, ask your AI client to list applications (`list_applications`) or environments (`list_environments`) to scope out what needs scanning.

03 Finally, instruct the agent with specific commands—for example, 'Run a scan on Production and then triage any high-risk alerts'—to execute actions.

The bottom line is that your AI client handles the complex API calls and data parsing, allowing you to manage advanced security operations using natural language only.

Built For

This MCP is essential for DevSecOps Engineers and Backend Developers who are tired of switching between multiple dashboards (StackHawk, Jira, CI/CD logs) just to get a clear picture of application risk. It's built for people who need security checks integrated directly into their workflow.

DevSecOps Engineer

Uses the MCP to programmatically initiate scans, check organizational compliance metrics via `list_api_keys`, and automatically classify high-priority alerts using `triage_alert`.

Backend Developer

Connects with the agent to quickly review security regression findings after a microservice deployment, parsing results directly into their terminal window.

Engineering Leader

Uses the MCP to audit cross-application threat landscapes by listing all monitored applications (`list_applications`) and reviewing overall compliance status.

What Changes When You Connect

01 You don't have to jump between dashboards. Your AI client lets you start a scan, review the `list_scans` results, and then immediately pull detailed findings using `get_scan_alerts`, all without switching tabs.

02 Stop spending time manually classifying risks. After a scan, simply tell your agent to review critical alerts and use `triage_alert` to mark known false positives or accept the risk on high-priority items.

03 Gain visibility into every part of your stack. Use `list_applications` and `list_environments` to get a complete inventory of every service you're monitoring, ensuring nothing gets overlooked in compliance checks.

04 Audit credentials easily. The `list_api_keys` tool lets you check which API tokens are active across the organization, improving overall security hygiene without manual database lookups.

05 Keep your development flow going. Instead of pausing work to run a scan, you can instruct the agent to initiate it using `run_scan`, and then track its progress using `list_scans` while continuing other tasks.

06 Get deep data instantly. If you need full metadata on what was found, use `get_scan_results`. This tool provides more detail than just an alert count, giving the engineering team actionable context.

Real-World Applications

Responding to a Major Incident

An engineer notices unusual behavior on Production. They instruct their agent: 'Check for all scans run against Production in the last 24 hours, get the alerts, and flag anything that looks like an SQL injection.' The agent uses `list_scans`, then `get_scan_alerts` to compile a risk report instantly.

Onboarding New Services

A developer has deployed a new microservice. They ask their agent to 'Register this new service and run a baseline scan.' The agent first uses `get_application_details` to check configuration, then initiates the test via `run_scan`, ensuring immediate coverage.

Reducing Alert Fatigue

The security team receives hundreds of alerts weekly. They ask their agent: 'Review all high-risk findings from the last scan and classify any known false positives.' The agent uses `get_scan_alerts` followed by `triage_alert`, cutting down manual cleanup time.

Pre-Deployment Checklist

Before deploying to Staging, a team lead asks the agent: 'List all active applications and confirm we have an environment configured for testing.' The agent uses `list_applications` and then `list_environments`, confirming readiness before code merge.

Patterns to Avoid

Relying on Manual Dashboards

✗ AVOID

The engineer has to log into the StackHawk web portal, find the correct application ID, manually select 'Production' environment, and then click the 'Run Scan' button. This takes five minutes of clicks just to start the test.

✓ INSTEAD

Instead, they tell their agent: 'Using this MCP, run a scan against Production for our main gateway.' The agent handles all authentication (`login`) and initiation (`run_scan`), getting them started in seconds.

Forgetting to Contextualize Alerts

✗ AVOID

The developer runs the scan and gets 50 alerts. They then have to manually download each alert report, one by one, to see if it's a false positive.

✓ INSTEAD

They instruct their agent: 'Show me all critical alerts from the last scan.' The agent uses `get_scan_alerts` and then immediately suggests using `triage_alert` on questionable findings.

Ignoring Scope Creep

✗ AVOID

The team thinks they only need to check one application, but later realize three others are running in the same org that weren't included in the initial scan run.

✓ INSTEAD

They use the MCP to first call `list_applications` to get a full inventory of all services monitored by StackHawk before writing any scanning commands. This ensures comprehensive coverage.

The Right Fit

Use this if your primary pain point is translating complex, multi-step security operations—like running scans, checking logs, and triaging findings—into conversational actions within your chat window. You need to move from 'I need to run a scan' to 'Run the scan on Production and classify the results.' Don't use this if you just need simple API key management; for that, other dedicated credential tools work better. Furthermore, don't expect it to fix coding bugs itself; it only points out vulnerabilities using `get_scan_alerts`. If your goal is pure compliance reporting against a specific standard (like PCI-DSS), you might prefer an MCP built specifically around mapping findings to regulatory frameworks, rather than relying solely on the raw output of `get_scan_results`.

Security audits are painful. They require too much switching and clicking.

Today, running a basic security assessment feels like juggling five different dashboards. You start in the main portal to find your application ID. Then you switch to an environment manager tab to select 'Staging.' After that, you hit the scan button, wait for it to finish, and then jump into a separate alerts panel just to count how many high-risk findings showed up. It’s time-consuming clicking.

With this MCP, all those steps disappear. You simply tell your agent: 'Run a full DAST audit on Staging.' The system handles the authentication, targets the correct environment using `list_environments`, runs the scan via `run_scan`, and reports back the results directly in text. You get security findings without the dashboard clutter.

Using StackHawk MCP gives you immediate control over vulnerability triage.

Manual remediation involves downloading dozens of individual alerts from different scans and then manually reviewing each one to see if it’s a genuine threat or just a false positive. This is where the process grinds to a halt.

Now, you can instruct your agent: 'Review the latest findings and classify any known false positives.' The system uses `get_scan_alerts` and then executes `triage_alert`, immediately updating the status of those vulnerabilities right from your chat. It's instant risk management.

StackHawk: 10 Tools for Security Management

These tools allow you to manage scan executions, view application assets, fetch detailed results, and triage security alerts directly through your AI agent.

#	TOOL	DESCRIPTION
01	`get_application_details`	Retrieves detailed configuration information for a specific StackHawk application.
02	`get_organization_details`	Gets overall details about your StackHawk organization, including subscription tier status.
03	`get_scan_alerts`	Downloads specific security alerts discovered during a DAST scan run.
04	`get_scan_results`	Provides detailed results and metadata for any given DAST scan execution.
05	`list_applications`	Lists all registered DAST applications within your entire StackHawk organization.
06	`list_environments`	Shows all configured environments (like Development, Staging, Production) available for an application scan.
07	`list_api_keys`	Lists all API keys set up in the organization, useful for auditing access hygiene.
08	`list_scans`	Displays a list of all DAST scan executions, including high-level alert counts.
09	`login`	Authenticates with StackHawk and obtains the necessary bearer access token for subsequent tool use.
10	`triage_alert`	Changes the status of a specific DAST security alert to 'false positive', 'risk accepted', or 'in progress'.

See It in Action

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

U Log in with my API token, list my projects and environments, then show the critical vulnerabilities from the latest scan.

Authentication successful. 🔒
Applications (`list_applications`):
- `Payment-Gateway-Core`
- `Internal-Admin-Dashboard`

Environments (`list_environments`):
- `Production`
- `Staging`

Critical Alerts (`get_alerts`):
- SQL Injection — Risk: High — Endpoint: /api/auth

U Run a new scan against the Production application.

Scan initiated via `run_scan`. ✅
The DAST audit is running asynchronously. Track progress with `list_scans` using the returned UUID.

Frequently Asked Questions

01 **How do I start a new scan using StackHawk MCP?**

You use the agent to call the `run_scan` tool, specifying which application and environment you want to audit. The system handles the initiation process for you.

02 **Can StackHawk MCP help me check my API token status?**

Yes, you can use `list_api_keys` to retrieve a list of all configured API keys for your organization, helping you audit which credentials are active and needed.

03 **What if I find an alert that is false positive? How do I handle it?**

You use the `triage_alert` tool. You simply tell the agent to review a specific vulnerability finding and change its status to 'false positive' or 'risk accepted'.

04 **How many applications can StackHawk MCP list?**

The `list_applications` tool retrieves all DAST applications registered in your organization, providing you with a complete inventory of monitored services.

05 **Is the data from get_scan_results always up to date?**

Yes. The results come directly from StackHawk's live scan records, giving you detailed metadata and ensuring you're working with current information regarding a specific DAST run.

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 `"stackhawk": { "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

StackHawk 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 StackHawk. 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	StackHawk MCP
Server ID	`019d760c-df45-716e-9823-f90ebe3681f4`
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/stackhawk.