MCP SERVER | NO CODE | CLOUD HOSTED

SigNoz MCP

Manage alerts and rules via natural conversation.

SigNoz MCP lets you manage observability alerts and rules directly through your AI agent. This open-source alternative handles infrastructure monitoring, allowing you to list, create, and update alert thresholds without touching a complex web console.

A+ Quality Score 100/100

alerting | infrastructure-monitoring | open-source | telemetry | incident-response

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/signoz-datadog-alternative

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.

SigNoz (Datadog Alternative) MCP

5 tools available
Cloud-hosted on Vinkius

Connecting SigNoz to your AI agent gives you full control over your monitoring stack using natural language. If you rely on observability data but hate navigating dozens of dashboards just to check an alert threshold, this is for you. You can tell the agent exactly what rules exist, update a condition on a high-priority metric, or delete old alerts that are no longer relevant. It's about treating your entire monitoring infrastructure like a conversation with your team member—fast, direct, and actionable. When you connect SigNoz via Vinkius, you gain instant access to this powerful set of tools alongside thousands of others, letting you keep all your operational data centralized right from your code editor.

Core Capabilities

01 — Check account status

Verify your current service account details and API key permissions instantly.

02 — List existing rules

Retrieve a comprehensive list of all alert configurations currently in use.

03 — Create new alerts

Build entirely new alert rules, setting specific conditions and thresholds for monitoring.

04 — Modify existing rules

Update the thresholds or trigger conditions of any rule that's already running.

05 — Remove unused alerts

Quickly delete alert rules to keep your monitoring environment clean and focused.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/signoz-datadog-alternative** — connect your AI agent in three steps.

01 Subscribe to this MCP in Vinkius, then enter your SigNoz Instance URL and API Key.

02 Call the agent and tell it what you need; for example, 'list all alert rules.'

03 The agent executes the command, retrieves the data, and presents the actionable results back to you.

The bottom line is that you manage your entire monitoring stack using plain conversation, not clicking through complex web UIs.

Built For

This MCP is built for the ops engineer who's tired of spending 20 minutes navigating dense dashboards just to check if a critical alert threshold needs adjusting. It targets anyone whose job involves maintaining uptime and ensuring observability configurations are accurate.

Site Reliability Engineer (SRE)

Audits existing alert thresholds and conditions across multiple services without leaving the code editor.

DevOps Engineer

Automates the creation of new alert rules as part of deployment or scaling pipelines.

Platform Team Manager

Manages observability configurations across staging and production environments via simple chat commands.

What Changes When You Connect

01 Audit your entire stack instantly. Instead of clicking through complex UIs to find all alert configurations, you simply tell the agent to list all configured rules.

02 Automate rule changes during deployment. Use the `create_rule` tool within your CI/CD process to ensure new services have monitoring coverage immediately.

03 Keep things clean. Run a check for stale or redundant alerts and use `delete_rule` to remove old configurations, keeping your dashboard noise low.

04 Validate connectivity instantly. Before writing complex code, run `get_service_account_me` to confirm your API key is valid and you have the correct permissions.

05 Adjust thresholds on the fly. If a service's normal operating range shifts, use `update_rule` to change its specific alert conditions without manual console access.

Real-World Applications

Debugging an unexpected outage

A developer notices high latency spikes but isn't sure which monitoring rule is failing. They ask their agent, 'List all rules related to database performance.' The agent uses `list_rules` and points them directly to the relevant alert ID, saving a deep dive into multiple dashboards.

Onboarding a new microservice

A platform team needs to monitor a freshly deployed service. They use their agent to execute `create_rule`, defining the specific metrics and thresholds needed for the new component, ensuring zero monitoring gaps from day one.

Compliance cleanup

An SRE is tasked with auditing old alerts that haven't been touched in six months. The agent uses `list_rules`, allows the engineer to review the list, and then executes `delete_rule` for every flagged obsolete alert.

Validating credentials before a release

Before pushing code that relies on monitoring data, an engineer first calls `get_service_account_me`. The agent confirms the API key is valid and shows current Read/Write permissions, preventing deployment failures later.

Patterns to Avoid

Manually clicking through dashboards

✗ AVOID

Spending 20 minutes navigating to the Alerts section, then filtering by service name, and finally manually copying rule IDs into a spreadsheet.

✓ INSTEAD

Just tell your agent to `list_rules`. It collects all configurations for you. If you need to change one, use `update_rule` directly through chat.

Guessing if an alert is active

✗ AVOID

Assuming a rule that was retired or renamed still exists and will trigger alerts when the service fails.

✓ INSTEAD

Always run `list_rules` first. This confirms what's actually configured in SigNoz, so you don't get false positives.

Forgetting to test a new rule

✗ AVOID

Writing code that relies on monitoring data without ever confirming the service account has write access to create or modify rules.

✓ INSTEAD

Always start with `get_service_account_me` to verify permissions before attempting any creation or modification.

The Right Fit

Use this MCP if your primary pain point is managing complex infrastructure alerting rules, and you find yourself spending too much time clicking through web UIs. This tool excels at the operational tasks: listing, creating, modifying, and deleting specific alert conditions. Don't use it if you just need to read raw metrics or analyze historical trends; for that, a general observability data retrieval MCP is better. Only use this when your goal is explicitly changing the *definition* of an alert (e.g., raising the CPU threshold from 80% to 90%). If you only need to check if a metric crossed a line once, simply querying the raw metrics is sufficient; don't bother with `list_rules`.

Checking monitoring alerts used to involve too many clicks.

Today, figuring out what alerts are active requires jumping between multiple tabs. You open the dashboard, click into the 'Alerts' section, then filter by service, and finally scroll through dozens of rules just to confirm if a threshold is set correctly. It’s slow, and you often miss something critical.

With this MCP, you simply ask your agent what alerts are active for that service. The system uses `list_rules` and gives you the full list and status immediately in text form. You get immediate clarity without opening a single web console.

Manage Alert Rules with SigNoz MCP

Updating an alert condition used to mean logging into the monitoring system, finding the specific rule ID, navigating to its settings page, and manually adjusting a slider or input box. It was repetitive, error-prone work.

Now, you just tell your agent, 'Raise the latency threshold for Service B.' The MCP executes `update_rule` instantly, making changes reliably through conversation. It's about getting precise control over your system definitions without leaving your terminal.

SigNoz (Datadog Alternative) with 5 Tools

Use these tools to manage everything about your infrastructure monitoring stack: from checking credentials to creating brand new alert conditions.

#	TOOL	DESCRIPTION
01	`create_rule`	Creates a brand new alert rule using specified details like the metric, condition, and thresholds.
02	`delete_rule`	Removes an entire alert rule from the monitoring system by its identifier.
03	`get_service_account_me`	Confirms your current service account credentials and associated API key permissions.
04	`list_rules`	Retrieves a comprehensive list of all currently configured alert rules in the system.
05	`update_rule`	Modifies an existing alert rule, allowing you to change its thresholds or triggering conditions.

See It in Action

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

U List all alert rules in my SigNoz instance.

I've retrieved 8 alert rules. Some active rules include 'High CPU Usage' (ID: 1), 'Database Latency' (ID: 2), and 'Service 5xx Errors' (ID: 3). Would you like to see the details for any specific rule?

U Show me the details of my current service account.

Your API key is valid. You are connected as 'Monitoring-Admin' with the following permissions: [Alerts: Read/Write, Dashboards: Read]. Your account ID is 'sa-99283'.

U Delete the alert rule with ID 'alert-123'.

The alert rule 'alert-123' has been successfully deleted from your SigNoz instance.

Frequently Asked Questions

01 **How does the SigNoz MCP handle different alert metrics?**

The MCP uses specific JSON payloads to define which metric, condition, and thresholds are needed for a new rule. You don't need to worry about the payload structure; just tell your agent what you want monitored.

02 **Can I use SigNoz MCP to delete an alert rule?**

Yes. If you know which rule needs removing, you can use `delete_rule` through your agent. It handles the necessary cleanup in SigNoz for you.

03 **Do I need special permissions to run list_rules?**

The MCP first runs `get_service_account_me` to validate your connection. If your service account lacks read access, the agent won't be able to retrieve the rule list.

04 **Is SigNoz MCP only for new alerts?**

Not at all. You can use `update_rule` to modify any existing alert—you don't have to recreate it just because you want a slightly different threshold.

05 **What happens if I try to create a rule with bad data?**

The agent will catch the error before sending it. It reads the tool documentation and tells you exactly what format the metric, condition, or thresholds must be in, so you can correct your request.

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 `"signoz-datadog-alternative": { "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

SigNoz (Datadog Alternative) 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 SigNoz (Datadog Alternative). 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	SigNoz (Datadog Alternative) MCP
Server ID	`019e38ed-0a30-730e-ae73-04fc43d68cf9`
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/signoz-datadog-alternative.