MCP SERVER | NO CODE | CLOUD HOSTED

Guance Cloud MCP

Monitor infrastructure health through conversation.

Guance Cloud / 观测云 gives your AI agent full access to complex system monitoring data. Stop clicking through dashboards and log tabs. This MCP lets you ask natural language questions about infrastructure, retrieving monitor statuses, reviewing real-time events, and running deep data queries (DQL) without touching the Guance console.

A+ Quality Score 100/100

system-monitoring | incident-response | dashboarding | application-performance | infrastructure-metrics | real-time-alerts

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/guance-cloud

SCAN TO VIEW

The infrastructure that powers AI agents in the real world.

Vinkius connects AI to the world's software through secure, enterprise-grade infrastructure — enabling real-world execution at scale, built on the Model Context Protocol (MCP).

Your AI Connections Run Through Vinkius Cloud

The world's largest managed MCP catalog

Vinkius is the cloud infrastructure where AI agents connect 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.

Guance Cloud / 观测云 MCP

10 tools available
Cloud-hosted on Vinkius

Guance Cloud connects your AI agent directly to an entire observability stack. Forget logging into a dashboard just to check if a service is down or what caused the spike last night. This MCP lets you manage complex system monitoring, log analysis, and incident response simply by talking to your agent. You can ask it to list all monitors, pull detailed configuration reports on dashboards, or browse specific system events in real time. If you need deep metrics, you don't build a query; you just tell the agent what average CPU usage you want over the last hour. Your agent acts like an always-on site reliability assistant, keeping your infrastructure data accurate and giving you instant answers whether you're troubleshooting or auditing resource usage. Connecting Guance Cloud to Vinkius means you get this power alongside thousands of other industry tools through one place.

Core Capabilities

01 — Check system health status

The agent retrieves the current metadata and alert status for your entire monitoring workspace.

02 — List and check monitor details

You can ask the agent to list every configured system monitor and get its detailed setup information.

03 — Browse historical events

The agent pulls a stream of real-time observability records, including errors, alerts, and system changes.

04 — Run custom data queries

You tell the agent what metrics you need, and it executes powerful DQL statements to fetch specific logging or performance data.

05 — Audit infrastructure keys and billing

The agent provides visibility into your organizational API access keys and current usage billing records.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP and provide your Guance Cloud API Key (DF-API-KEY).

02 Connect the credentials to your preferred AI client, like Cursor or Claude.

03 Ask your agent a question—for example, 'What was the average latency last night?'—and it handles the data retrieval.

The bottom line is you get instant answers about complex infrastructure metrics by talking to your agent instead of navigating web consoles.

Built For

This is for Ops Engineers and SREs who spend too much time clicking through multiple dashboards just to find one metric. If you're tired of logging in at 2 AM just to figure out why the system failed, this MCP saves your sanity.

SRE Engineer

You use the agent to automate incident response, asking it to list monitors or check events when a production outage happens.

DevOps Engineer

You ask for specific data using DQL queries and retrieve dashboard configurations without opening the GUI.

Infrastructure Lead

You coordinate monitoring strategies by having the agent audit resource usage or check billing details.

What Changes When You Connect

01 Instead of manually running `list_monitors` to see what's broken, you simply ask your agent for the status. You get immediate visibility into every alert rule without leaving your chat window.

02 Need to check past activity? Use the agent to browse real-time observability events via `list_events`. This means instantly knowing if a system change caused an error, which is way faster than digging through logs.

03 The biggest time saver: running complex queries. Instead of building DQL in a separate tool and pasting it somewhere else, you just ask the agent to perform data querying using `query_data` directly.

04 You don't have to guess what resources exist. Running `list_dashboards` gives you an immediate overview of your monitoring setup, letting you audit coverage instantly.

05 It keeps everything connected: You can check the overall status with `get_workspace`, then drill down to specific alerts using `get_monitor`. It's a complete workflow built into one conversation.

Real-World Applications

Troubleshooting an unexpected service dip

An SRE gets paged about high latency. Instead of jumping between the metrics dashboard and the event log, they ask their agent to list monitors and then execute a DQL query for average latency across the cluster last night. The agent answers with a clear number and identifies the time window where the issue started.

Investigating a data loss incident

A technical analyst suspects log data is missing. They ask the agent to list log sources, then browse recent observability events using `list_events` to see if any deletion alerts fired. The agent pinpoints the exact time and source of the gap.

Auditing resource access

An Infrastructure Lead needs to know which services are using which keys. They ask their agent to list access keys, check billing usage with `get_billing`, and confirm that all required dashboards are available via `list_dashboards`. It compiles a full compliance report instantly.

Patterns to Avoid

Treating it like a simple search engine

✗ AVOID

Trying to ask, 'What is my CPU usage?' without specifying the metric or time range. The agent gets confused because you didn't provide enough context.

✓ INSTEAD

Always be specific. Instead of a vague query, use `query_data` and tell your agent: 'Give me the average CPU usage for the Production cluster over the last 15 minutes.' Specificity is key.

Manually copying configuration details

✗ AVOID

Finding a critical monitor setting in the console, copy-pasting it into a document just to share it with a teammate later.

✓ INSTEAD

Use `get_monitor` to have your agent pull the exact JSON structure and status of that monitor directly into your chat for easy sharing or logging.

Ignoring system metadata

✗ AVOID

Only looking at error logs without understanding if the underlying workspace itself was configured correctly.

✓ INSTEAD

Start by calling `get_workspace` to get a high-level status overview. This gives you context before diving into specific errors or metrics.

The Right Fit

Use this MCP when your core task involves monitoring, debugging, or auditing live system health and performance data from Guance Cloud. If you need to know the *state* of a complex technical system—like 'What happened with my API latency last night?' or 'List all active alerts'—this is perfect.

Don't use it if your goal is simple record creation (you can't create monitors here) or if you only need basic data retrieval that doesn't involve multiple steps, like fetching a single user name. For those simpler tasks, other specialized tools might be better.

If you are analyzing metrics and events, the combination of `list_monitors`, `get_monitor`, and `query_data` is your core workflow.

The Manual Chore of System Health Checks

Right now, checking if a system is healthy means opening the Guance console. You navigate to the dashboards tab, then you open the monitor list, check status codes one by one, and finally, you have to manually build a query using DQL just to get average CPU usage for a specific time window. It's clicking through five different tabs just to answer two simple questions.

With this MCP connected via Vinkius, those clicks disappear. You simply talk to your agent: 'Check the status of the high-CPU monitor and give me the 15-minute average.' The agent executes `get_monitor` and `query_data` in the background and spits out a clean answer instantly.

Guance Cloud / 观测云 MCP: Instant Observability Insights

You no longer have to copy-paste an API key into a script or manually reconcile dashboard configurations across different teams. You can ask the agent for `list_access_keys` and instantly get a full, auditable list of every credential in one chat prompt.

Your AI client acts as a dedicated SRE co-pilot, handling all the complex data orchestration—from listing events to running deep DQL queries—so you stay focused on solving the problem, not clicking through the interface.

Guance Cloud / 观测云: 10 Tools for Observability

Use these tools to check system health, list monitors, run DQL queries, and audit billing data through natural conversation.

#	TOOL	DESCRIPTION
01	`get_billing`	Retrieves your current cloud service usage and cost information.
02	`get_event`	Fetches the detailed record for a specific observability event.
03	`get_monitor`	Gets all configuration details for a single system monitor.
04	`get_workspace`	Retrieves high-level metadata and status information about your monitoring workspace.
05	`list_access_keys`	Shows a list of all API access keys configured in the workspace for auditing purposes.
06	`list_dashboards`	Returns a comprehensive list of every dashboard configured in the system.
07	`list_events`	Pulls a list of recent observability events, including alerts and errors.
08	`list_log_sources`	Provides an inventory of all available log data sources for analysis.
09	`list_monitors`	Lists every active and configured system monitor in the environment.
10	`query_data`	Runs powerful Data Query Language (DQL) statements to fetch specific metrics or log data.

See It in Action

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

U List all active monitors in Guance Cloud.

I've retrieved your active monitors. You have 15 monitors configured, including 'High CPU Alert', 'API Latency', and 'Database Connection'. Would you like to check the status of the alerts?

U Show me recent events from the last hour.

I've listed 5 events from the last hour. There was 1 Critical alert for 'Storage Full' and 4 Information events regarding system updates. Should I retrieve the details for the Critical alert?

U Query average CPU usage using DQL.

I've executed the DQL query. The average CPU usage across your 'Production' cluster over the last 15 minutes is 42%. Would you like a breakdown by individual host?

Frequently Asked Questions

01 **How do I list monitors using Guance Cloud / 观测云 MCP?**

You ask your agent to execute `list_monitors`. The agent will return a full catalog of every monitor configured in the workspace, letting you know exactly what is being watched.

02 **Can I run custom metrics queries with Guance Cloud / 观测云 MCP?**

Yes. You use the `query_data` tool. Just tell your agent which metric and time window you want, and it runs the necessary DQL statement for you.

03 **What is the difference between list_events and get_event?**

`list_events` gives you a feed of recent occurrences (errors, alerts). `get_event` allows you to deep-dive into one specific event ID to see all its associated details.

04 **Do I need to manually find the API key for Guance Cloud / 观测云 MCP?**

The agent can help. You use `list_access_keys` to get a list of existing keys and confirm which credentials are available for your workspace.

05 **How do I check billing usage with Guance Cloud / 观测云 MCP?**

You ask the agent about billing, and it executes `get_billing`. You get an immediate breakdown of your current service consumption and costs.

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 `"guance-cloud": { "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

Guance Cloud / 观测云 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 Guance Cloud / 观测云. 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	Guance Cloud / 观测云 MCP
Server ID	`019d8444-c688-70d0-84d6-0cc533614025`
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/guance-cloud.