MCP SERVER | NO CODE | CLOUD HOSTED

Better Stack MCP for AI Agents

Manage uptime, track incidents, and control on-call scheduling

Better Stack automates incident response and uptime monitoring. Connect your Better Stack account to any AI client so it can act as a Level 1 SRE, diagnosing downtime, checking monitor status, and managing escalations through natural conversation.

A+ Quality Score 100/100

incident-management | on-call-scheduling | downtime-alerts | escalation-chains | sre-automation

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/better-stack-1

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.

Better Stack MCP

10 tools available
Cloud-hosted on Vinkius

This MCP lets your AI agent function like an on-call engineer, letting you manage critical system alerts without leaving your chat window. Instead of jumping between monitoring dashboards and terminal sessions during an outage, your agent handles the initial triage. It can check which monitors are failing—whether it's a simple HTTP endpoint ping or a complex DNS probe. If there's an active issue, the AI finds all firing incidents, checks their full timelines, and even lets you acknowledge them to stop the paging cascade. The agent also tracks who is currently scheduled on call so you know exactly who to talk to next. Connecting this through Vinkius means your preferred AI client can access hundreds of other tools too, giving you a complete operational picture right where you work.

Core Capabilities

01 — List all active monitors

Retrieves a comprehensive list of every uptime monitor configured in Better Stack.

02 — Get specific monitor details

Fetches the full technical definition and status for any single monitoring check, like HTTP pings or latency rules.

03 — List current system incidents

Provides a list of all known or active outages currently recorded in Better Stack.

04 — Inspect incident timelines

Pulls the full, detailed history and root cause payload for a specific reported outage.

05 — Acknowledge an ongoing alert

Flags an active system incident as acknowledged in Better Stack, temporarily stopping automated paging alerts.

06 — Force resolve an issue

Manually sets a specific reported outage to a resolved state within the monitoring platform.

07 — Check scheduled workers

Lists all passive heartbeats and background worker checks to ensure core system processes are running correctly.

08 — List status pages

Reads the configuration details for public-facing status dashboards across your global infrastructure.

09 — View on-call schedules

Retrieves current active shifts and team rotation calendars to identify who is responsible right now.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/better-stack-1** — connect your AI agent in three steps.

01 Connect your Better Stack API token through Vinkius, granting your AI client the necessary permissions.

02 Tell your agent what you need: for instance, 'Show me all active incidents' or 'Who is on call today?'

03 The agent runs the appropriate tool and delivers the actionable data directly in conversation.

The bottom line is that it gives your AI client a single, conversational point of entry to manage complex operational alerts across multiple systems.

Built For

DevOps engineers and SREs who spend their nights jumping between dashboards desperately need this. If you're the ops engineer tired of manually checking ten different monitoring tools at 2 AM, this MCP saves your sanity.

Site Reliability Engineer (SRE)

Uses this to audit on-call matrices and orchestrate initial debugging workflows by pulling incident timelines or resolving dormant alerts without leaving the chat.

DevOps Engineer

Runs checks like listing heartbeats and monitoring status pages to rapidly debug failing background workers while simultaneously patching code in a controlled environment.

Engineering Manager

Reviews recent downtime timelines or acknowledges critical alerts quickly so they can provide accurate updates to stakeholders without deep technical knowledge.

What Changes When You Connect

01 Stop switching context during an outage. Your agent handles incident triage—listing active incidents or inspecting monitor details—all without you leaving the chat.

02 Quickly determine who needs to be paged. You can pull active shifts and team schedules using the list_on_call tool, so you instantly know who's responsible for a failing system component.

03 Reduce alert fatigue and noise. By acknowledging an incident (acknowledge_incident) or forcing it resolved (resolve_incident), your agent controls the severity of notifications automatically.

04 Deep dive into failures without logging in. Need to audit why something broke? Use get_incident to pull the full technical timeline payload for any reported issue.

05 Verify background services immediately. The list_heartbeats tool lets you check passive tracking endpoints, ensuring critical workers haven't silently failed.

06 Maintain public transparency while debugging privately. You can read configured status pages (list_status_pages) to ensure customers see accurate information regardless of the current incident.

Real-World Applications

The API endpoint is throwing 502 errors.

A developer asks their agent, 'Why is our main API failing?' The agent responds by listing monitors and uses get_monitor to pinpoint the exact HTTP endpoint ping that's failing. It then checks the incident timeline (get_incident) to find the root cause in minutes.

It's 3 AM and nothing is working.

The on-call engineer asks, 'Who should I talk to about this downtime?' The agent uses list_on_call to confirm John Doe is currently paged in Level 1. This saves them from wasting time checking old rotation schedules.

We need to update the incident report.

An engineering manager asks, 'What happened with Incident #8012?' The agent retrieves the detailed timeline payload using get_incident, providing all necessary logs and status changes for a comprehensive post-mortem.

We have background workers that might be failing.

A backend developer asks, 'Check our cron jobs.' The agent calls list_heartbeats, verifying that all passive tracking endpoints are reporting green. If one is down, they get the specific details via get_heartbeat to start patching.

Patterns to Avoid

Checking status manually

✗ AVOID

Copying and pasting URLs into ten separate monitoring dashboards or opening a dozen tabs during an outage. This wastes time, causes context switching, and misses the full picture.

✓ INSTEAD

Ask your agent to list_monitors first to see everything at once. Then use get_monitor on any specific service you need details for. Better Stack manages all of this in one chat.

Ignoring scheduled rotations

✗ AVOID

Calling multiple team members sequentially when an incident happens, wasting time determining who is actually responsible at that exact moment.

✓ INSTEAD

Use the list_on_call tool. This instantly tells you which team member or rotation calendar dictates who has primary ownership right now.

Confusing alerts with actual incidents

✗ AVOID

Treating a simple warning notification as a major outage, leading to unnecessary escalation and false alarms.

✓ INSTEAD

Always start by listing_incidents. This separates minor warnings from confirmed, active outages that require immediate action.

The Right Fit

Use this MCP if your primary pain point is context switching during system outages or needing a single source of truth for uptime status. If you frequently find yourself opening multiple tabs—one for monitoring, one for schedules, and another for incident timelines—this tool is essential because it groups all that data into conversational tools like list_monitors and get_incident.

Don't use this if your team needs to manage ticketing system workflows (like Jira or Zendesk). For those, you need a dedicated CRM integration MCP. If your goal is just running scheduled reports without real-time incident response, consider an API wrapper tool instead of the full Better Stack suite.

Better Stack MCP for AI Agents: Automated Incident Triage and Uptime Monitoring

Right now, when a critical service goes down, your workflow is manual hell. You open the monitoring dashboard to check pings, then switch to an incident management tool to see if it’s already recorded. Then you might jump to a calendar system just to figure out who is on call—all while trying not to lose track of which tab showed the real root cause. It's slow, and context switching kills your focus.

With this MCP, your agent handles all that cross-platform legwork for you. You simply ask your AI client what’s wrong, and it uses tools like list_incidents and get_incident to pull together a complete diagnosis from multiple sources instantly. You get an actionable summary, not just links to dashboards.

Better Stack MCP for AI Agents: Controlling On-Call Schedules and Escalations

Before this, determining who was supposed to be on call often involved checking a separate Google Sheet or an internal wiki page. If the sheet wasn't updated, you wasted time calling the wrong people, delaying resolution.

Now, your agent uses list_on_call to check the official routing calendars directly. This eliminates guesswork and ensures that when an alert fires, everyone knows precisely who is primary responder right now.

Better Stack: 10 Tools for Incident Management & Uptime Monitoring

Use these specific tools to list monitors, retrieve incident timelines, acknowledge alerts, view on-call schedules, and check background worker heartbeats.

#	TOOL	DESCRIPTION
01	`list_monitors`	Lists all uptime monitors configured in Better Stack.
02	`get_monitor`	Retrieves specific details for one monitor, like its ping type or latency rules.
03	`list_incidents`	Gets a list of all explicitly reported system incidents.
04	`get_incident`	Fetches the full technical payload and timeline for a specific incident.
05	`acknowledge_incident`	Changes the status of an ongoing incident to 'acknowledged' in Better Stack, which stops automated paging notifications.
06	`resolve_incident`	Forces a specific reported outage into a resolved state within the monitoring system.
07	`list_heartbeats`	Lists all passive tracking endpoints and cron jobs used to validate background worker status.
08	`get_heartbeat`	Pulls explicit technical details about a single, passive heartbeat check.
09	`list_status_pages`	Lists all configured public-facing status pages for your global services.
10	`list_on_call`	Retrieves the current on-call team schedules and routing calendars.

See It in Action

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

U I'm seeing a lot of alerts today. What active incidents are going on?

System Incident Report

- **ID #651239:** Critical API Down (Started 4 minutes ago)
* Status: Active, Escalating via Phone Call.
* Action: Would you like me to acknowledge this alert locally?
- **ID #7801:** Database Connection Fluctuation
* Status: Resolved. Acknowledged at 9/3/24.

Frequently Asked Questions

01 **How can I use the Better Stack MCP for AI Agents to check if my service is down?**

You ask your agent to list all monitors. It will give you a clean breakdown of every single uptime check, showing which HTTP endpoints or DNS probes are currently failing so you know exactly where the problem lies.

02 **Can I use the Better Stack MCP for AI Agents to find out who is on call right now?**

Yes. Your agent uses the list_on_call tool to pull up the active rotation calendars. You'll get a clear name and role, so you don't waste time calling people outside their shift.

03 **What if I want to know why an old incident happened?**

You can ask your agent about specific incidents using the tool that retrieves the full timeline payload. It pulls all the historical data, including error codes and timestamps, so you have everything for a post-mortem report.

04 **Does the Better Stack MCP for AI Agents help me stop constant notifications?**

Absolutely. If an incident is confirmed or if it's already been addressed, your agent can acknowledge_incident or resolve_incident to manage the alert status and prevent unnecessary paging.

05 **Does this MCP work for multiple services? Like my front end AND back end?**

Yes. It manages all monitors across your entire fleet, whether they are tracking HTTP endpoints or background cron heartbeats. You get one view of everything connected to Better Stack.

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 `"better-stack-1": { "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

Better Stack 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 →

INDEPENDENT PLATFORM DISCLAIMER

Vinkius is an independent platform and is not affiliated with, endorsed by, sponsored by, verified by, or otherwise authorized by Better Stack. 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	Better Stack MCP
Server ID	`019d755b-724f-73b1-9dbc-ee8184a178fe`
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/better-stack-1.