MCP SERVER | NO CODE | CLOUD HOSTED

Rollbar MCP

Instant bug diagnosis and deployment tracking.

Rollbar connects your AI client directly to a continuous code error tracking platform. Use this MCP to monitor live application health, identify active bugs, review full stack traces, check deployment history, and manage the lifecycle of critical errors—all from chat.

A+ Quality Score 100/100

error-tracking | bug-reporting | stack-trace | site-reliability | deployment-monitoring | incident-resolution

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

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.

Rollbar MCP

10 tools available
Cloud-hosted on Vinkius

This integration turns your conversational agent into an instant site reliability engineer. You stop jumping between dashboards and start talking to your code's health status. Your AI client can now list active bugs or immediately pull up the top issues affecting users, pinpointing exactly where things broke. Want deep detail? Just give it an error ID, and the tool pulls the full stack trace logs so you know precisely which line of code failed. You can also report a new deployment using a specific Git commit tag, keeping your monitoring dashboard current automatically. Need to close out a fix? Tell your agent to change the status of the bug ticket. By connecting this MCP through Vinkius, you get complete visibility into your application's health and deployment history without leaving your chat window.

Core Capabilities

01 — Check for current bugs

The agent can list all currently tracked error items or quickly show the most frequent issues impacting users right now.

02 — View detailed failure logs

Provide a specific error ID to pull up every recorded instance and get the complete stack trace showing where the code failed.

03 — Track deployments

The agent lists all past code deployments or reports a brand new release tied to a Git revision.

04 — Change bug status

Update an error item's state, marking it as resolved or muting the alert if necessary.

One Click on Vinkius — From Prompt to Execution

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

01 First, launch the Rollbar tool inside your MCP environment.

02 Next, you sign in to your Rollbar developer dashboard and generate a read/write access token.

03 Finally, paste that specific Rollbar Access Token into the connection form. You're ready to chat.

The bottom line is, after connecting the token, you talk directly to Rollbar about your code's health.

Built For

This MCP is built for engineers who hate context switching. It helps the Ops Engineer tired of clicking through multiple dashboards at 2 AM. It’s perfect for anyone whose job involves knowing, quickly and accurately, why a production release might have broken.

Site Reliability Engineer (SRE)

You use this to detect if a new commit introduced bugs by asking the agent to compare deployment logs against active errors.

Software Developer

You pull complete stack traces into your chat window, letting you debug production issues without leaving your IDE or communication tool.

Engineering Manager

You monitor overall deployment frequencies and manually resolve fixed bug tickets just by typing the command; no complex filter setup required.

What Changes When You Connect

01 You stop manually cross-referencing Git tags with error dashboards. Simply ask your agent to report a new release using the `report_deploy` tool, automatically linking future bugs to that specific commit.

02 Instead of wading through hundreds of dashboard entries, use `top_active_items` to get an immediate list of the most frequent and high-impact issues affecting your user base right now.

03 Debugging complex failures is faster. If you have a bad error ID, passing it to `get_occurrence` pulls the full stack trace directly into your chat, letting you see exactly where in the code the failure happened.

04 Managing bug lifecycles becomes conversational. After fixing an issue, tell your agent to use `update_item` and change its status; no manual dashboard clicks are required.

05 You can get a bird's-eye view of historical issues by running `list_items`, giving you a clean summary of every distinct error type the application has seen.

Real-World Applications

The sudden production outage

A developer notices an alert about a broken endpoint. Instead of logging into Rollbar, they ask their agent to run `get_occurrence` for the latest ID. The agent immediately pulls the full stack trace showing a critical data type mismatch at Line 42 in the payments module.

Post-deployment sanity check

An SRE just merged code and needs to verify monitoring is active. They tell their agent to run `report_deploy` for 'production' with the new commit ID, ensuring all subsequent errors are correctly linked.

Cleaning up old bugs

An Engineering Manager knows a bug was fixed last week but the ticket is still marked active. They instruct their agent to use `update_item` on that specific error ID, changing its status to 'resolved' and clearing the board.

Finding persistent pain points

A developer wants to know what's breaking the most often. They ask their agent for `top_active_items`, getting a ranked list of bugs, allowing them to prioritize fixes based on real user impact.

Patterns to Avoid

Manually checking logs

✗ AVOID

Logging into Rollbar's web UI, filtering by environment, then manually scrolling through dates and commit IDs trying to correlate the error with the deployment that caused it.

✓ INSTEAD

Use the `report_deploy` tool to log your new release first. Then, if an issue pops up, ask your agent to check for active errors using `list_items`; this automatically links the failure back to the correct deployment context.

Forgetting old bugs

✗ AVOID

A bug is fixed and deployed, but someone forgets to update the ticket status in the web UI, leaving it marked as active forever.

✓ INSTEAD

After implementing a fix, tell your agent to use `update_item` on that error ID. It handles the necessary state change instantly, keeping your records clean.

Searching for general errors

✗ AVOID

Asking an AI assistant simply to 'show me bugs' without specifying if they are currently active or related to a certain deployment.

✓ INSTEAD

To get the most actionable data, use `top_active_items`. This tool filters down the noise and gives you only the highest-impact errors that need attention immediately.

The Right Fit

Use this MCP if your primary pain point is the friction between monitoring code health and communicating about it. If you're an engineer who spends more time copying error IDs into Slack than actually fixing them, this tool helps. You must be working with a centralized error tracking platform like Rollbar to use it effectively. Don't use this MCP if all you need is to list team members or manage project invitations; those are separate tools. If your only goal is to track who has access, look for an identity management connector instead.

The pain of context switching when debugging production code.

Today, tracking down a critical bug means opening Rollbar, finding the error ID, checking the deployment history to see what changed last night, then maybe jumping into Git or Jira just to verify the fix status. It's a minimum of five tabs and three different logins before you even start reading stack traces.

With this MCP connected through Vinkius, that entire process collapses into one conversation. You simply ask your agent, 'What broke after the last deployment?' and it executes the necessary checks—from listing deployments to finding the top active errors—and gives you a single answer.

Rollbar MCP Gives You Full Visibility Into Code Failures

You no longer have to copy-paste an error ID into multiple systems or wait for someone else to run the report. The agent handles the heavy lifting of calling `get_item` and then fetching all associated details via `list_occurrences`, presenting it all in plain text.

The result is immediate, actionable knowledge. You get deep, precise visibility into your code's health—right where you work—without ever leaving the chat window.

Rollbar MCP – 10 Tools

These tools let you list error types, check specific bug occurrences, report new deployments, and manage the status of active code issues.

#	TOOL	DESCRIPTION
01	`get_deploy`	Pulls specific details for one code deployment record.
02	`get_item`	Retrieves the main information for a single error item.
03	`get_occurrence`	Shows all deep logs and stack traces for one specific recorded instance of an error.
04	`list_deploys`	Provides a list of every code deployment that has been registered.
05	`list_invites`	Lists any pending team invitations for the project.
06	`list_items`	Gets a list of all error types and items currently tracked by Rollbar.
07	`list_occurrences`	Lists individual instances of an error that belong to a specific item.
08	`report_deploy`	Reports and logs a new code deployment using specified environment and Git revision details.
09	`top_active_items`	Retrieves the list of error items that are currently happening most frequently across users.
10	`update_item`	Changes an error item's status to active, resolved, or muted.

See It in Action

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

U Check the dashboard for any listed error items recorded recently.

I checked the primary error repository running `list_items`. I found 4 distinct error tickets currently listed as active issues. Let me know if you would like me to isolate the most frequent ones using `top_active_items` or examine a specific ID in depth.

U Get the stack trace and precise occurrence detail for occurrence ID '1234abcd'.

I requested the deep logs calling `get_occurrence` specifically mapped to '1234abcd'. The system presented the raw stack trace revealing an uncaught `TypeError` exception breaking at `Line 42` inside `payments_module.js`. Let me print the full code output block mapped by Rollbar for you below to review.

U Report a new deployment in Rollbar for the 'production' environment using the commit revision '8a3bd41'.

Action logged successfully! I triggered `report_deploy` and recorded the newest application launch for the 'production' infrastructure in Rollbar tagged dynamically with the tracking code revision '8a3bd41'. All subsequent exceptions identified via Rollbar modules will now be bound closely correlated automatically to this specific commit update marker.

Frequently Asked Questions

01 **How do I check for recent errors using Rollbar MCP?**

You can list all current issues by asking the agent to run `list_items`. For a more targeted view, use `top_active_items` to see only the most frequent bugs.

02 **Can I update an error status with Rollbar MCP?**

Yes, you can tell your agent to use `update_item`. This changes the bug's official state in Rollbar to active, resolved, or muted.

03 **What if I need to report a new release using Rollbar MCP?**

Use the `report_deploy` tool. You just need to provide the environment name (like 'production') and the specific Git commit revision number for the deployment.

04 **Does Rollbar MCP help with stack traces?**

Absolutely. If you have an error ID, the agent can run `get_occurrence` to retrieve the complete raw stack trace logs, showing exactly where your application failed.

05 **Is this tool for managing user accounts?**

No. This MCP is solely designed for code health and deployment monitoring within Rollbar; it cannot manage users or invitations (though `list_invites` can check for pending invites).

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 `"rollbar": { "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

Rollbar 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 Rollbar. 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	Rollbar MCP
Server ID	`019d75ff-fd2a-732f-b62d-b79182ab004f`
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/rollbar.