MCP SERVER | NO CODE | CLOUD HOSTED

Rancher MCP

Query Cluster Status & Manage Namespaces via AI

Rancher MCP gives your AI agent direct access to manage complex Kubernetes environments through the Rancher platform. You can query cluster health, list namespaces across multiple clusters, and diagnose individual pod failures—all from a simple chat prompt. It eliminates tedious context switching between CLI commands and UI dashboards.

A+ Quality Score 100/100

kubernetes | container-orchestration | cluster-management | devops | containers | infrastructure-monitoring

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

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.

Rancher MCP

10 tools available
Cloud-hosted on Vinkius

Managing container infrastructure usually means juggling command-line interfaces and web portals for every single cluster you own. This MCP changes that. It lets your AI agent interact with all the Kubernetes clusters managed under your Rancher control plane, turning complex operations into simple conversations. Need to know if a specific microservice deployment is running correctly? Just ask. The AI can check everything from listing available projects and namespaces to verifying the operational health of any workload or pod. All the data you need—cluster status, node lists, user accounts—is instantly accessible through your preferred client on Vinkius. You stop debugging in terminals; you start asking questions.

Core Capabilities

01 — Identify Managed Clusters

Get a full inventory of every Kubernetes cluster connected to the Rancher platform.

02 — View Cluster Components

List all nodes and available logical projects within your managed clusters.

03 — Diagnose Pod Status

Check the current operational state of specific pods, including identifying crash loops or health issues.

04 — Audit Namespaces and Workloads

Explore logical partitions (namespaces) and list all deployed applications and workloads within a project.

05 — Check User Access

Retrieve a complete list of user accounts managed by the Rancher platform.

One Click on Vinkius — From Prompt to Execution

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

01 Enable this MCP and configure it using your specific Rancher Server URL and the required Bearer Token for API access.

02 Instruct your AI client to perform a diagnostic task, such as listing all available clusters or checking a pod's status.

03 The AI executes the query against the Rancher platform and returns structured data detailing the cluster state, namespaces, or workload metrics.

The bottom line is: you talk to your agent, it talks to Rancher, and you get a clear, summarized answer without writing any code.

Built For

This MCP is built for the infrastructure team. It's for the ops engineer who spends too much time switching between dashboards just to find out why one service is failing. If your job involves debugging cluster issues or verifying deployment topologies, this saves you hours of context-switching.

DevOps Engineer

Debugging live cluster statuses by querying pod metrics and listing deployments without opening a terminal.

Kubernetes Administrator

Running automated queries to verify policies, check node lists, and confirm project boundaries across environments.

Backend Developer

Ensuring microservices are running smoothly on target clusters and verifying namespace health without local setup or manual CLI work.

What Changes When You Connect

01 Stop context switching between dashboards. You can list all managed clusters and view their status instantly, bypassing the need to navigate through multiple UIs.

02 Diagnose specific service failures fast. Use `list_workloads` or check pod metrics to pinpoint exactly which microservice is crashing within a namespace.

03 Gain full visibility into your infrastructure. By running `list_nodes`, you can get an immediate inventory of every machine acting as a worker in any cluster.

04 Understand project structure easily. Need to see what's deployed? Use `list_apps` or check the list from `list_namespaces` to map out service boundaries.

05 Manage user access without logging in. Running `list_users` gives you an immediate roster of who has access to which part of your environment.

Real-World Applications

Troubleshooting a new deployment failure

A developer notices high error rates for the 'billing-api' service. Instead of manually checking logs and running `kubectl get pods`, they ask their agent to check the pod status using `list_pods`. The agent immediately reports that the pod is in a crash loop, pointing them straight to the failing container.

Auditing resource sprawl

The administrator needs to know if a new team created unauthorized environments. They ask the agent to run `list_clusters` and then use `list_projects`. The system quickly returns all active project IDs, allowing them to flag any unexpected or unmanaged resources.

Verifying service scope

A backend developer needs to know if a specific feature lives in the 'staging' environment. They query `list_namespaces` for the target cluster ID, confirming the existence of the necessary development partitions before starting their work.

Onboarding new team members

A manager needs to give a new engineer an overview of who can access which resources. They ask the agent to run `list_users` and receive a clean, actionable list of all authenticated accounts in the platform.

Patterns to Avoid

Using SSH/CLI for status checks

✗ AVOID

Logging into multiple servers via SSH, running `kubectl get pods -n <namespace>` repeatedly, and manually aggregating results to figure out which service is down.

✓ INSTEAD

Instead, use the agent. Ask it to run `list_pods` on your target cluster ID. It aggregates all the status checks for you in one go.

Manual UI navigation

✗ AVOID

Clicking through 15 different menus and dashboards just to confirm if a specific application (Helm app) is running in the 'production' project.

✓ INSTEAD

Directly ask your agent to `list_apps` for that project. It cuts out all the clicks and gives you the status immediately.

Assuming cluster scope

✗ AVOID

Running general commands without specifying a target environment or cluster ID, leading to ambiguous results or hitting the wrong production system.

✓ INSTEAD

Always start by asking for an inventory using `list_clusters` first. This confirms you're working on the right infrastructure before drilling down.

The Right Fit

Use this MCP if your primary need is *read-only diagnostic inspection* across multiple, complex container environments managed by Rancher. You want to know: 'What state are we in?' or 'Why did it break?'. It's perfect for troubleshooting and auditing (e.g., checking `list_namespaces` or `get_cluster`).

Don't use this if your goal is *deployment, configuration change,* or *security policy enforcement*. If you need to create a new namespace, modify a deployment definition, or delete resources, you need dedicated CI/CD tooling (like ArgoCD) or direct API calls. This MCP focuses on giving your agent the data it needs to make informed decisions, but it doesn't execute state-changing operations itself.

The Pain of Context Switching

Today, figuring out a cluster issue means jumping between tools. You open the UI dashboard for Cluster A, check its pods there. Then you switch to your terminal and run `kubectl` against Cluster B. If you need to verify which team owns the 'payments' namespace, you have to manually dig through projects, then namespaces, then workloads—it’s a painful cycle of copy-pasting IDs.

With this MCP, all that manual effort disappears. You just tell your agent, 'Show me every project and every pod in the staging environment.' It runs `list_projects` and `list_workloads` for you, compiling a clean report instantly. Your AI client is doing the heavy lifting so you can actually focus on fixing things.

Rancher MCP Gives You Complete Visibility

The process of auditing who has access or what services exist used to require running half a dozen different commands: `list_users` for accounts, then `list_clusters` for boundaries, and finally manually checking every project's associated

Now you ask your agent to 'Audit the environment.' The MCP runs all those checks—from listing users to checking cluster health with `get_cluster`, providing a single, unified status report. You get

workloads. It was slow, error-prone, and exhausting.

the full picture without ever leaving your chat window.

Rancher: 10 Tools for Cluster Management

Use these tools to query every aspect of your infrastructure, from listing user accounts to diagnosing specific worker node failures.

#	TOOL	DESCRIPTION
01	`get_cluster`	Retrieves specific operational details for one Kubernetes cluster.
02	`get_project`	Gets the detailed information for a chosen Rancher project.
03	`list_apps`	Lists all Helm applications that are installed within a specific project.
04	`list_catalogs`	Retrieves a list of available Helm chart repositories (Catalogs).
05	`list_clusters`	Generates an inventory listing all Kubernetes clusters managed by Rancher.
06	`list_namespaces`	Lists the specific logical partitions (namespaces) tied to a project.
07	`list_nodes`	Provides an inventory of every worker node within a specified cluster.
08	`list_projects`	Lists the logical projects available inside a given cluster.
09	`list_users`	Generates a list of all user accounts defined in the Rancher platform.
10	`list_workloads`	Lists all Kubernetes workloads, such as Deployments and StatefulSets, within a project.

See It in Action

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

U List all Kubernetes clusters managed by my Rancher instance.

Through `list_clusters`, I have enumerated your environment. You have 2 visible instances: 'production-us-east' (State: Active) and 'staging-development' (State: Provisioning).

U Query the namespaces available inside cluster 'c-8xk9z'.

I requested `list_namespaces` for the cluster ID 'c-8xk9z'. It contains several namespaces, including: 'default', 'kube-system', 'monitoring-setup', and 'frontend-tier'.

U Check the status of the 'auth-service' pod located in the 'backend-production' namespace on cluster 'c-lq4×2'.

Checking pod metrics... Using `list_pods`, the 'auth-service' pod is marked 'Running', executing in the requested namespace. No crash loops were detected in its recent container history.

Frequently Asked Questions

01 **How do I check if my pods are running using Rancher MCP?**

You use the `list_pods` tool to diagnose the status of any pod. This query tells you if a container is 'Running', or if it's stuck in a crash loop, saving you manual CLI checks.

02 **What is the difference between list_clusters and list_projects?**

`list_clusters` gives you an inventory of all physical clusters managed by Rancher. `list_projects` works within a single cluster to show logical groupings or boundaries for resources.

03 **Does Rancher MCP help me find the right namespace?**

Yes, running `list_namespaces` will enumerate all the logical partitions (namespaces) available inside your target project. This helps you confirm where a specific service is deployed.

04 **Can I list applications installed in a project?**

You use the `list_apps` tool to see every Helm application deployed within a given project boundary, giving you a quick inventory of your services.

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 `"rancher": { "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

Rancher 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 Rancher. 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	Rancher MCP
Server ID	`019d75fc-7db1-7135-b70f-b2516fd8bf3f`
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/rancher.