MCP SERVER | NO CODE | CLOUD HOSTED

Hetzner MCP

Control Servers, Networks, and Storage via Conversational AI

Hetzner MCP lets your AI agent manage your entire cloud infrastructure through conversation. You can list servers, configure firewalls, create load balancers, spin up new volumes, and cycle power states—all using natural language commands from any compatible client.

A+ Quality Score 100/100

cloud-hosting | server-management | infrastructure-as-code | firewall | cloud-compute

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

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.

Hetzner MCP

38 tools available
Cloud-hosted on Vinkius

Managing a cloud environment used to mean logging into the console and clicking through endless menus just to check if a server was running or adjusting a firewall rule. Now, you can connect your Hetzner Cloud account directly to your agent. You simply tell your AI what needs doing—like 'I need a new web front-end' or 'Check the status of the staging database.' It handles the complex steps: listing all resources, checking for available primary IPs, and executing the necessary changes. This MCP gives your agent full control over everything from creating network zones to deleting old load balancers. When you connect this through Vinkius, it becomes one single point of access, letting any compatible client use your infrastructure knowledge base without needing multiple integrations.

Core Capabilities

01 — Server and Resource Lifecycle Management

Create, delete, get details on, or power cycle specific servers and storage boxes.

02 — Advanced Networking Configuration

Manage network security by creating firewalls, configuring floating IPs, and setting up load balancers.

03 — Storage Provisioning and Snapshots

Create new volumes, manage storage box subaccounts, or manually take snapshots of data containers.

04 — DNS Zone and IP Management

List existing datacenters, create private networks, and handle DNS zones for your infrastructure.

05 — Credential and Asset Tracking

Generate new SSH keys or list all certificates associated with your cloud accounts.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP on Vinkius.

02 Provide your Hetzner Cloud API Token when prompted.

03 Ask your AI client a question or give it a command, like 'Power up the dev server' or 'List all firewall rules.'

The bottom line is, you get full cloud control without ever leaving your chat window.

Built For

This MCP is for technical roles—DevOps Engineers, System Administrators, and Software Developers. You're the person who gets frustrated when a simple network change requires jumping between five different web dashboards just to verify connectivity or spin up a test environment.

DevOps Engineer

You use this MCP to audit server states, create load balancers, and automate the deployment of new infrastructure components directly from your terminal.

System Administrator

You manage complex network rules by listing firewalls or running a complete power cycle on a cluster when troubleshooting connectivity issues.

Software Developer

You spin up temporary development environments, create new volumes for testing data, and generate SSH keys so you can immediately start coding against the fresh setup.

What Changes When You Connect

01 Stop clicking through dashboards. Instead of manually checking status or creating a firewall rule set across multiple tabs, you simply ask your agent to 'Secure the web server with a new firewall' and it handles all the necessary steps using `create_firewall`.

02 Instant visibility into your fleet. If you need to audit every active resource, use `list_servers` or `get_server`. You get immediate data on status, IP addresses, and running OS details without navigating away from your workflow.

03 Automate disaster recovery steps. Instead of manually setting up a backup process, tell your agent to 'Snapshot the production database,' triggering `create_storage_box_snapshot` instantly.

04 Efficient capacity planning. Need more resources? You can list all available server types using `list_server_types` and launch new instances with a single command, bypassing manual resource allocation screens.

05 Simplify complex networking. Forget juggling IPs and load balancers. Your agent handles the flow: first listing primary IPs (`list_primary_ips`), then creating an IP (`create_floating_ip`), and finally attaching it to a `create_load_balancer`.

Real-World Applications

Troubleshooting connectivity for production traffic

A sysadmin notices external users complain about slow connections. They ask their agent, 'Check the firewall and IP status for the web tier.' The agent executes `list_firewalls` and checks `list_floating_ips`, immediately pointing out that a recent rule change blocked necessary outbound traffic.

Rapidly spinning up a testing environment

A developer needs a sandbox for a new feature. They instruct their agent: 'Create a new server instance, give it 50GB of storage, and use the latest Ubuntu image.' The agent handles `create_server`, `list_images` (to find the right one), and `create_volume` in sequence.

Scaling up backend capacity after a traffic spike

The DevOps team sees high load. They tell their agent, 'Add another front-end server behind the existing load balancer.' The agent uses `list_load_balancers` to verify the target group and executes `create_server`, ensuring smooth scaling.

Archiving old data safely

A team needs to archive logs from an old application. They ask their agent, 'Take a snapshot of the production log storage.' The agent immediately runs `list_storage_boxes` to find the correct box and executes `create_storage_box_snapshot`, securing the data.

Patterns to Avoid

Trying to manually check status updates

✗ AVOID

A user has to open 4 different tabs—one for servers, one for load balancers, one for firewalls, and one for IPs—just to get a basic health overview.

✓ INSTEAD

Instead of clicking through dashboards, ask your agent to 'Give me the overall status of the web tier.' The agent aggregates data from `list_servers`, `list_firewalls`, and `list_load_balancers` into one readable summary.

Confusing server type with capability

✗ AVOID

A user thinks that just knowing the 'CX21' means they can provision it, but forgets to specify *where* or *how many*.

✓ INSTEAD

Always use `list_server_types` first to confirm availability and then instruct your agent: 'Provision two CX21 servers in the Frankfurt datacenter.' This ensures resource constraints are met.

Overlooking dependencies before deployment

✗ AVOID

A user tries to create a floating IP address without checking if the required network zone exists, resulting in an API failure.

✓ INSTEAD

First, run `list_zones` to verify the DNS structure. Then, instruct your agent: 'Create a new private network and link it to this zone.' This builds dependencies correctly.

The Right Fit

Use this MCP if you need deep, multi-step control over the full lifecycle of an infrastructure stack—provisioning servers, configuring networking rules (firewalls, load balancers), and managing associated data storage. It’s for people who treat their

cloud environment like a single machine they can talk to.

Don't use this if your only goal is basic billing information or viewing general usage metrics; you need an accounting-specific tool for that. Similarly, if you only manage user permissions without touching network rules, you might find another credential management API more focused. This MCP excels at the 'build and configure' phase of operations.

The Dashboard Overload

Today, managing a cloud environment means logging into multiple web consoles. You open the server tab to check status, then switch to the network tab to audit firewalls. If you need to change an IP address, you jump to another page just for IPs, and if that changes requires updating DNS, you click over yet again. It’s a constant cycle of switching tabs, verifying IDs, and copy-pasting data across five different screens.

With this MCP, all those steps vanish. You simply tell your agent, 'I need to patch the web tier.' The agent handles listing servers (`list_servers`), checking firewall rules (`list_firewalls`), updating necessary IPs (`create_floating_ip`), and confirming the change—all in one conversation. You get immediate confirmation that the infrastructure is updated.

Hetzner MCP: Full Cloud Control

You eliminate the need to manually create every single resource, from listing datacenters (`list_datacenters`) to setting up a new storage box subaccount. You never have to worry about forgetting which API endpoint handles load balancers versus firewalls.

What’s different now is that your AI acts like an expert system administrator who has read every piece of Hetzner documentation and remembers the exact sequence of steps needed, delivering the result in plain language.

Hetzner MCP: 38 Tools Available

These tools give your AI agent direct programmatic access to every major function within the Hetzner Cloud platform, allowing you to execute complex tasks in sequence.

#	TOOL	DESCRIPTION
01	`list_certificates`	Retrieves a list of all existing security certificates.
02	`create_certificate`	Generates and applies a new security certificate to your infrastructure.
03	`get_storage_box`	Gets the current settings and status of a specific Storage Box.
04	`list_images`	Lists all available operating system or machine images for deployment.
05	`create_firewall`	Sets up a brand new firewall rule set for network traffic control.
06	`create_floating_ip`	Allocates and creates a public, floating IP address for your services.
07	`create_load_balancer`	Configures and launches a new load balancer to distribute incoming traffic.
08	`create_network`	Establishes an entirely new private network for resource isolation.
09	`create_server`	Provisions and launches a brand new virtual server instance.
10	`create_ssh_key`	Generates and registers a secure SSH key pair for access control.
11	`create_storage_box_snapshot`	Takes a manual, point-in-time snapshot of the data in a Storage Box.
12	`create_storage_box_subaccount`	Creates an isolated subaccount within an existing Storage Box for segmented access.
13	`create_volume`	Allocates and creates a new, detached block storage volume.
14	`create_zone`	Establishes a new DNS zone for domain name resolution.
15	`list_datacenters`	Displays all physical datacenter locations where resources can be deployed.
16	`delete_load_balancer`	Permanently removes an existing load balancer configuration.

#	TOOL	DESCRIPTION
17	delete_server	Terminates and deletes a virtual server instance from your account.
18	list_firewalls	Displays all existing firewall rulesets you have created.
19	list_floating_ips	Retrieves a list of all allocated public floating IP addresses.
20	get_action	Fetches specific details about an infrastructure action or event.
21	get_server	Retrieves detailed operational information for a single server instance.
22	list_load_balancers	Retrieves detailed information on all active load balancer configurations.
23	list_locations	Shows a list of available physical deployment locations for your resources.
24	list_networks	Retrieves details about all private networks you have established.
25	list_placement_groups	Displays groups that help manage physical placement and availability zones.
26	poweron_server	Immediately sends a command to power on a previously powered-off server.
27	list_primary_ips	Displays all primary, static IP addresses assigned to your account.
28	rebuild_server	Restores a virtual server instance using a specified operating system image or ISO file.
29	list_server_types	Shows all available types of servers (e.g., CX11, CX21) you can provision.
30	list_servers	Lists the operational status and details for every server in your account.
31	list_ssh_keys	Displays all SSH keys currently stored or associated with your accounts.
32	list_storage_box_snapshots	Retrieves a list of historical snapshots available for a Storage Box.
33	list_storage_box_subaccounts	Displays all subaccounts currently configured within a specific Storage Box.
34	list_storage_boxes	Lists the status and existence of all your data storage boxes.

#	TOOL	DESCRIPTION
35	`update_storage_box`	Modifies settings on a Storage Box, like enabling or disabling specific protocols (e.g., SMB).
36	`list_volumes`	Retrieves a list of all currently allocated block storage volumes.
37	`list_zones`	Displays the existing DNS zones configured for your domain names.
38	`list_isos`	Displays a list of supported ISO files for server rebuilding.

See It in Action

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

U List all my active servers in the Hetzner Cloud.

I've found 3 servers: 'web-prod' (ID: 12345, running), 'db-master' (ID: 12346, running), and 'staging-app' (ID: 12347, off).

U Get the full details for server ID 12345.

Fetching details for server 12345... It is a CX21 instance located in fsn1-dc14, currently running Ubuntu 22.04 with IP 1.2.3.4.

U Power on the server with ID 12347.

Sending power-on command to server 12347... The action has been initiated successfully.

Frequently Asked Questions

01 **How do I use the Hetzner MCP to check my current servers?**

You run `list_servers` or ask your agent to 'List all active servers.' This retrieves the operational status and details for every server in your account, showing you if they are running or stopped.

02 **Can I use Hetzner MCP to set up a new network?**

Yes. You start by asking the agent to create a private network using `create_network`. This establishes the isolated environment needed before you can deploy resources within it.

03 **What if I need a public IP address? Do I use Hetzner MCP?**

Yes, for external access. You must first check available IPs using `list_primary_ips`, then allocate one with `create_floating_ip` to ensure your service is reachable.

04 **Is hetzner mcp safe to use in production?**

Yes, it allows you to manage core infrastructure functions like creating firewalls (`create_firewall`) and deleting servers. Always test complex changes on a staging environment first.

05 **How does Hetzner MCP handle storage backups?**

You use the `list_storage_boxes` tool to find your data containers, then issue a command for `create_storage_box_snapshot` to secure the current state of the data.

06 **Can I see the status of all my cloud servers at once?**

Yes, the `list_servers` tool retrieves a complete list of your instances, including their IDs, status, and labels.

07 **Is it possible to turn on a server that is currently powered off?**

Absolutely. Use the `poweron_server` tool with the specific Server ID to initiate the power-on sequence.

08 **Can I manage security rules through this integration?**

Yes, you can use `list_firewalls` to audit existing rules and `create_firewall` to set up new protection layers for your infrastructure.

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 `"hetzner": { "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

Hetzner 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 Hetzner. 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	Hetzner MCP
Server ID	`019e38a7-ffa2-7345-ab62-7ddf53fedb27`
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/hetzner.