MCP SERVER | NO CODE | CLOUD HOSTED

Fastly MCP

Control your entire CDN stack conversationally.

Fastly MCP manages your edge cloud delivery and CDN infrastructure entirely through natural conversation. You can deploy instant security patches by promoting service versions, audit backend origins (AWS/GCP), purge global cache instantly via a simple chat command, or map specific domains to verify routing rules—all without touching a dashboard.

A+ Quality Score 98.33/100

cdn | edge-computing | waf | cache-purging | web-security | network-optimization

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

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.

Fastly MCP

12 tools available
Cloud-hosted on Vinkius

Think of this MCP as giving your AI client direct, conversational access to the core controls of your CDN. You stop logging into multiple consoles and start talking to your infrastructure instead. Your agent can look up every service you run, tell you which version is actually serving live traffic right now, or check if a specific domain name points to the correct edge location.

It handles everything from coordinating complex deployments—like moving a drafted configuration to active status—to emergency maintenance tasks, such as issuing an absolute HTTP PURGE across all global data centers. If you already use Vinkius for other integrations, adding Fastly means one place controls your entire network stack. It lets you manage backend origins and verify port constraints while keeping track of which architectural instance is controlling live traffic today.

Core Capabilities

01 — Deploy Service Versions

You tell the MCP to promote a configuration draft or activate an existing version for immediate service rollout.

02 — Manage CDN Services

You can create, delete, and get detailed status information on every Fastly service running in your account.

03 — Purge Global Cache

The MCP issues a command to wipe all cached content for a specific service globally, forcing fresh data retrieval from the source.

04 — Audit Network Details

You can list and verify domains attached to your services or check the physical upstream origins mapped within configurations.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP and provide your Fastly API token from your account control panel.

02 Connect your AI client (Claude, Cursor, etc.) to Vinkius and activate the Fastly toolset.

03 Start by asking your agent a direct question, like 'What services need cache purging?' or 'List all domains attached to service X.'

The bottom line is that you use natural language to execute complex infrastructure commands without needing developer credentials or manual dashboard navigation.

Built For

DevOps and SRE teams who spend hours clicking through dashboards trying to verify a production rollout. Cloud Architects who need quick, comprehensive views of domain mappings and backend origins. Security Engineers needing instant proof of patch deployment.

Site Reliability Engineer (SRE)

Manages global cache purges after deploying new content or verifies service versions to ensure the correct configuration is live across all points-of-presence.

Cloud Architect

Audits backend origin mappings and extracts FQDN apex domains attached to services without having to manually navigate through the console settings.

DevOps Engineer

Uses the MCP to deploy instant patches by promoting drafted versions, moving from a development state directly into production traffic.

What Changes When You Connect

01 Instantly deploy patches. Use the `activate_service_version` tool to promote a draft configuration, moving changes from test environments straight into production traffic.

02 Never worry about stale content again. A single prompt using `purge_all_cache` issues an absolute HTTP PURGE across all global points of presence for immediate data freshness.

03 Audit your network easily. You can run `list_version_domains` to verify every FQDN apex domain terminating at the edge, or use `list_services` to get a full inventory count.

04 Verify your infrastructure depth. Use `list_version_backends` to pinpoint the physical upstream origins (AWS/GCP) mapped inside any service version, ensuring no load balancer is missed.

05 Manage entire services lifecycle. You can list all running assets with `list_services`, get performance metrics via `get_service_stats`, and even clean up old infrastructure using `delete_service`.

Real-World Applications

Rolling out an emergency security patch.

A Security Engineer needs to deploy a zero-day fix. Instead of manually logging into the console and promoting drafts, they simply ask their agent: 'Activate version 12 for service X.' The MCP executes `activate_service_version` instantly, making the changes live globally.

Troubleshooting a broken website image.

A Frontend Developer notices an old logo is showing up. They instruct their agent to 'Purge cache for service Y.' The MCP runs `purge_all_cache`, clearing the global CDN and forcing the network to grab the fresh version from the origin.

Verifying complex routing rules.

A Cloud Architect needs to know which domains are attached to a staging environment. They prompt their agent to run `list_version_domains` against the specific service version, getting an immediate list of all FQDNs terminating there.

Checking backend dependencies before deletion.

An SRE needs to decommission a service but must first confirm its connection points. They use `list_version_backends` on the target service version, verifying every mapped origin (AWS/GCP) is accounted for before running `delete_service`.

Patterns to Avoid

Confusing cache purging with deployment.

✗ AVOID

Manually purges the cache and then tries to deploy a fix, assuming the purge solved the issue. This wastes time because the actual code change needs explicit activation.

✓ INSTEAD

If content is stale, first run `purge_all_cache`. If the underlying configuration or patch is wrong, you must use `activate_service_version` *before* purging.

Assuming all services are active.

✗ AVOID

Attempting to manage a service using generic commands without first checking its status. This can lead to errors when trying to list versions or origins.

✓ INSTEAD

Always start by running `list_services` to confirm the exact name and existence of the target service before attempting any other operation.

Deleting a service without auditing backends.

✗ AVOID

Running `delete_service` on a crucial asset because its function is unknown. This might take down an entire backend origin (AWS/GCP) that another service relies upon.

✓ INSTEAD

Before deleting, use `list_version_backends` and `get_service` to fully map out all dependencies attached to the service.

The Right Fit

Use this MCP if your job requires constantly managing complex edge deployments. Specifically, if you need to deploy instant patches by promoting drafts (`activate_service_version`), or if cache invalidation is a core part of your workflow (`purge_all_cache`). You also need the ability to audit physical network details like domain names (`list_version_domains`) and backend origins (`list_version_backends`).

Don't use this MCP if you only need basic content storage or simple DNS record management. If your toolset is limited to just reading general usage metrics, `get_service_stats` handles that fine, but it won't help you deploy code or audit origins. This MCP is for hands-on infrastructure control; it's not a read-only reporting tool.

The headache of manual CDN maintenance

Today, deploying a simple patch requires jumping between three different dashboards: the service console to check versions, the domain manager to verify FQDNs, and then hitting the cache invalidation tool. You copy-paste IDs, click through confirmation modals, and spend twenty minutes just confirming that every single origin is correctly mapped.

With this MCP, you just talk to your agent. Instead of clicking six buttons across three tabs, you ask one question: 'I need to patch service X.' Your agent handles the version promotion (`activate_service_version`), verifies domain names, and issues the global purge—all in a single conversation.

Control your entire deployment lifecycle with Fastly MCP

You no longer have to wait for a human Ops team member to execute a cache wipe. You can proactively request the `purge_all_cache` command, or use `list_service_versions` to confirm exactly which draft needs promoting via `activate_service_version`.

It moves you from being reactive—fixing things after they break—to being proactive. You gain immediate control over your edge layer's state and history.

Fastly: 12 Tools for Edge Cloud Management

These tools give you direct command access to every core function of your Fastly account—from service creation to cache purging.

#	TOOL	DESCRIPTION
01	activate_service_version	Promotes a specific draft configuration version of a service to active status immediately.
02	create_service	Initializes and creates an entirely new Fastly content delivery network service in your account.
03	delete_service	Permanently removes a specified Fastly service from your cloud infrastructure.
04	get_me	Retrieves basic information about the API token identity linked to the MCP connection.
05	get_service	Pulls all core details and status information for a named Fastly service.
06	get_service_stats	Retrieves usage statistics, including performance metrics, for a specific running service.
07	get_service_version	Gets detailed information about an individual configuration version of a service.
08	list_version_backends	Lists all physical upstream origins (AWS, GCP) mapped within a specific service configuration version.
09	list_version_domains	Retrieves a list of domain names associated with a particular service version.
10	list_services	Provides an inventory of all existing Fastly services managed by your account.
11	list_service_versions	Fetches a complete list of every configuration version draft and active state for one service.
12	purge_all_cache	Issues an absolute HTTP PURGE command, invalidating all cached content globally for a target service.

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 Fastly services

Retrieving services... I found 3 CDN services: 'Prod-Main-CDN' (ID: 1a2b), 'Staging-App' (ID: 3c4d), and 'Media-Assets' (ID: 5e6f). All are currently terminating traffic across the edge network. Which one would you like to inspect?

U Activate version 15 for service 'Prod-Main-CDN'

Version 15 promoted! I've successfully activated the drafted configuration for 'Prod-Main-CDN'. The changes are now live across all global points-of-presence. I'll monitor the traffic health for you.

U Purge all cache for service '1a2b'

Cache purge initiated! I've issued a global HTTP PURGE for service 1a2b. All static content has been invalidated, and the network will fetch fresh data from your origins immediately.

Frequently Asked Questions

01 **How do I purge the cache using Fastly MCP?**

You use the `purge_all_cache` tool by specifying the service ID. This sends a global HTTP PURGE command, invalidating all static content immediately.

02 **Can I see what domains are attached to my services with Fastly MCP?**

Yes, you can use `list_version_domains` to get an exact list of FQDN apex domains tied to a specific service version for auditing.

03 **What is the difference between creating and activating a service version with Fastly MCP?**

Creating (`create_service`) builds a new, draft configuration. Activating (`activate_service_version`) promotes that existing, tested draft to live traffic status.

04 **Does Fastly MCP help me find my backend origins?**

Absolutely. Use `list_version_backends` to list all physical upstream Origins (AWS/GCP) associated with a given service version, helping you verify connectivity points.

05 **What if I need to delete a Fastly service? Which tool should I use?**

Use the `delete_service` tool. However, always run `list_version_backends` first to ensure no other services rely on that 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 `"fastly": { "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

Fastly 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 Fastly. 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	Fastly MCP
Server ID	`019d7596-de20-73fd-ad95-3294a43662ce`
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/fastly.