MCP SERVER | NO CODE | CLOUD HOSTED

Svix MCP

Manage Webhook Delivery Via Conversation

Svix lets you manage your entire webhook infrastructure using natural conversation with your AI client. You'll take full control of complex event-driven workflows, configuring applications and endpoints without touching a dashboard. Need to check why an invoice notification failed? Your agent handles the debugging by listing message attempts or retrieving delivery status for specific payloads. It's like having a dedicated DevOps engineer running live right inside your chat window.

A+ Quality Score 100/100

webhooks | api-infrastructure | event-delivery | message-queues | endpoint-management | real-time-sync

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

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.

Svix MCP

15 tools available
Cloud-hosted on Vinkius

Managing webhooks usually means jumping between dashboards, digging through obscure logs, and manually updating configuration files just to test one single endpoint. This MCP changes that. You connect Svix and give your AI client the power to handle all of your event delivery needs conversationally. Instead of guessing whether a message got delivered or why an application failed to send data, you simply ask your agent to check it. It lets you group related services by listing applications, set up new receiving URLs via endpoint configuration, and even manually trigger test messages for verification. Because Vinkius hosts this MCP, you get direct, natural language access to all these core event management functions—from creating a whole application down to deleting old endpoints. Your AI client becomes the central hub that keeps your entire webhook lifecycle running smoothly.

Core Capabilities

01 — Configure Webhook Applications

Group and organize related services by listing, creating, or updating specific applications.

02 — Manage Destination Endpoints

Create new receiving URLs, modify existing endpoints, or delete stale ones for a given application.

03 — Monitor Message Payloads and Delivery Status

Send test messages to verify routes, list message attempts, and retrieve detailed status for any event payload.

04 — Diagnose Failures and History

List all endpoint delivery attempts or review historical message failures to pinpoint exactly where an integration broke.

One Click on Vinkius — From Prompt to Execution

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

01 First, subscribe to this MCP and provide your Svix API Secret Key within your AI client.

02 Next, tell your agent what you need: 'List all my applications,' or 'Create a new endpoint for X.'

03 The agent executes the command against the service, retrieving real-time data like delivery status, application details, or lists of failed attempts.

The bottom line is that your AI client treats webhook management as a natural conversation rather than a series of clicks and forms.

Built For

This MCP is for the backend engineer who's tired of spending an hour debugging logs at 2 am. It's also for DevOps teams that need to verify event routing in real-time, or product leads managing customer integrations without leaving their chat interface.

Backend Engineer

Needs to quickly create and test new webhook destinations (endpoints) and trigger sample messages from the IDE before committing code.

DevOps Engineer

Uses this toolset to monitor delivery attempts across dozens of endpoints, troubleshooting failing connections using natural language queries.

Product Integration Lead

Must verify that customer-facing webhook applications are receiving and processing messages correctly after a deployment.

What Changes When You Connect

01 You skip the dashboard. Instead of manually checking logs, you ask your agent to list endpoint attempts or get message details immediately.

02 Debugging is faster because you don't have to copy/paste IDs. You just tell your agent to check a specific application and it pulls up all its associated endpoints.

03 Setup and teardown are instant. Need to delete old webhook destinations? Use the delete endpoint tool instead of navigating deep into settings menus.

04 Testing is built-in. Instead of writing a script, you simply ask your agent to create a message and check delivery status for immediate verification.

05 Complete lifecycle control is available. You can use list applications to see everything currently running, or update application details without leaving your chat window.

Real-World Applications

Validating Customer Payment Flows

A product integration lead needs to confirm that a payment webhook hit the correct destination. Instead of logging in to the portal, they ask their agent to get message details for the specific transaction ID, verifying both delivery status and the payload content.

Troubleshooting Failed API Calls

A backend engineer sees a failure notification. They use list endpoint attempts on their agent, telling it which application failed, immediately identifying the broken URL and its last known error code.

Onboarding New Microservices

A DevOps team needs to connect three new services. They ask their agent to create three separate applications and configure corresponding endpoints for each service in a single, structured conversation.

Auditing Webhook Activity

An operations manager needs an audit trail of all messages sent last week. They use list messages and then follow up by listing message attempts to ensure every record was successfully processed.

Patterns to Avoid

Manual Log Diving

✗ AVOID

Copying a failure ID from an email, logging into the web portal, finding the correct application tab, and then searching through chronological logs to see what went wrong.

✓ INSTEAD

Use your agent to list message attempts or get details of a specific message. This bypasses manual navigation entirely.

Configuration Drift

✗ AVOID

Having endpoints that were supposed to be temporary, but they remain active and cost resources because no one remembers to delete them from the dashboard.

✓ INSTEAD

Use the delete endpoint tool or the delete application tool. It's an instant cleanup command.

Testing Blindly

✗ AVOID

Triggering a message but not knowing if it hit all expected endpoints, or if the payload was corrupted.

✓ INSTEAD

First, list endpoints to confirm targets. Then use create message and follow up with list message attempts to verify delivery across the board.

The Right Fit

Use this MCP if your pain point is managing event-driven architectures where webhooks are core to your business logic. Specifically, you need a conversational way to monitor, troubleshoot, or modify endpoint configurations and message flows. Don't use it if you only need simple data storage; for that, a standard database connector works better. Also, don't use this if all your event routing happens within a single application boundary without external services—if the scope is internal, you might not need webhook management at all. However, if you are dealing with multiple microservices or third-party integrations that trigger events to your system, this MCP gives you the centralized control needed.

The Pain of Webhook Debugging Today

Today, verifying a webhook delivery is a nightmare. You get an alert saying 'Failure,' so you have to copy that failure ID and navigate through multiple tabs—the applications list, the endpoints list, then finally the message logs. It takes constant switching between your terminal, your dashboard, and your debugger.

With this MCP, all of that manual clicking disappears. You simply tell your agent: 'Check delivery status for last night's invoice.' Your AI client handles the journey through the infrastructure, giving you the exact details without ever leaving your chat window.

Svix MCP Gives You Full Webhook Management

Manual tasks like listing all applications and endpoints are replaced by simple commands. Need to verify a payload? Instead of setting up a complex test script, you ask your agent to create message and instantly review the list message attempts.

It's not just reading logs; it's active control. You can update application details or delete stale components directly through conversation. Your entire event infrastructure is managed by natural language.

Svix: 15 Tools for Webhook Management

These tools let you interact with every aspect of your webhooks infrastructure, allowing you to create, read, update, and delete applications, messages, and endpoints via natural language.

#	TOOL	DESCRIPTION
01	`create_application`	Builds a new, logical grouping for your webhooks.
02	`create_endpoint`	Sets up a new destination URL that will receive webhook events.
03	`create_message`	Sends an immediate test message to simulate a real event payload.
04	`delete_application`	Removes an entire set of webhooks and associated endpoints.
05	`delete_endpoint`	Takes down a specific receiving URL when it's no longer needed.
06	`get_application`	Retrieves all the details for one specific webhook application by its ID.
07	`get_endpoint`	Shows the full configuration and status of a single endpoint URL.
08	`get_message`	Fetches specific details about one test message, including its event type.
09	`list_applications`	Shows an overview of every webhook application you have set up.
10	`list_endpoint_attempts`	Retrieves a list of past delivery attempts and their status for a given endpoint.
11	`list_endpoints`	Lists all the active endpoints associated with a specific application.
12	`list_message_attempts`	Shows records of past message deliveries and their corresponding attempt statuses.
13	`list_messages`	Provides an overview list of all messages that were sent under a specific application.
14	`update_application`	Modifies the settings or name of an existing webhook application.
15	`update_endpoint`	Changes the URL, filters, or status of an already configured endpoint.

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 Svix applications.

I've retrieved your applications. You have 2 active apps: 'Main API' (ID: app_2X...) and 'Billing Service' (ID: app_9Y...). Which one would you like to inspect?

U Create a new endpoint for app_2X... with URL https://webhook.site/test and subscribe to 'user.created' events.

Endpoint created successfully! The new endpoint ID is ep_3Z... and it is now configured to receive 'user.created' events at the specified URL.

U Show me the details for message msg_4W...

Fetching message details... Message msg_4W... was sent with event type 'invoice.paid'. It has been successfully delivered to 1 endpoint and is currently pending on another.

Frequently Asked Questions

01 **How do I list all my webhooks applications using Svix MCP?**

You use the list_applications tool. This instantly pulls up an overview of every webhook application you have set up in your account.

02 **Can I test a message delivery with the Svix MCP?**

Yes, you can trigger a test event using create_message. After sending it, use list message attempts to confirm where it delivered successfully.

03 **Which tool do I use if an endpoint URL is broken?**

You should start by listing endpoint attempts for that specific application. This helps pinpoint the exact failure and whether the issue is with the payload or the destination itself.

04 **Does Svix MCP let me update endpoints in natural language?**

Absolutely. You use the update_endpoint tool to change a URL, modify its filters, or change its status without having to manually edit the configuration dashboard.

05 **I need to clean up old webhooks; what should I do?**

Use delete application to remove entire groups of endpoints, or use delete_endpoint if you only want to retire a single, specific receiving URL.

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 `"svix": { "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

Svix 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 Svix. 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	Svix MCP
Server ID	`019e38f6-9f4d-731f-88f1-12eff9826843`
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/svix.