MCP SERVER | NO CODE | CLOUD HOSTED

QuickNode MCP

Manage Web3 Data Streams and RPC Calls via AI Agent

QuickNode lets your AI agent manage complex Web3 infrastructure directly from natural language prompts. You can create real-time data streams for historical and live blockchain ingestion, deploy event webhooks based on templates (like EVM wallet filters), and execute fundamental RPC calls to read core network data like the latest block number.

A+ Quality Score 100/100

web3 | ethereum | rpc | data-streams | webhooks

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

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.

QuickNode MCP

18 tools available
Cloud-hosted on Vinkius

Connecting QuickNode to your agent gives you command over high-performance blockchain data. Instead of jumping between a developer terminal, a dashboard, and an API key vault, your AI client handles all the complexity. You can build full Web3 data workflows right where you're writing code or asking questions. Want to monitor specific contract events? Your agent deploys webhooks using predefined templates, piping real-time alerts directly to your endpoint. Need to keep track of historical transactions? You configure and maintain streaming pipelines for live or archived blockchain ingestion. This capability means that the full power of QuickNode's infrastructure catalog is accessible through Vinkius, letting you manage everything from setting up a key-value store to checking the latest block number—all in one chat session.

Core Capabilities

01 — Manage Data Streams

Create, list, and update continuous data feeds for monitoring historical or live blockchain activity.

02 — Set up Event Webhooks

Deploy automated webhooks that listen for specific contract events (like wallet changes) and send real-time alerts to your service.

03 — Query Key-Value Data

Create, retrieve, and modify simple data points to power advanced filtering logic within your streaming pipelines.

04 — Access Core RPC Data

Run direct calls against the blockchain node to fetch fundamental information, such as the most recent block number.

One Click on Vinkius — From Prompt to Execution

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

01 First, you subscribe to this MCP and provide your QuickNode API Key along with your RPC URL.

02 Second, tell your AI agent exactly what infrastructure change you need—for instance, 'I need a stream for Polygon data.'

03 The system executes the request via the appropriate tool, manages the configuration on the backend, and confirms success or failure back to your chat.

The bottom line is that you manage complex Web3 infrastructure by simply telling your agent what you want done.

Built For

This MCP is for the data engineer who gets tired of manually configuring streams and webhooks across three different dashboards. It's for the smart contract developer who needs to prototype complex, real-time monitoring logic without touching a command line.

Web3 Developer

Prototypes data ingestion pipelines by using natural language to create and manage streams and test core RPC calls.

Data Engineer

Orchestrates complex, multi-stage blockchain data workflows, setting up key-value storage logic and webhooks for filtering.

DevOps Team

Monitors and updates production infrastructure configurations, streams, and webhooks on the fly using conversational commands.

What Changes When You Connect

01 Instead of manually checking a dashboard to see if your data stream is running, you ask your agent to list all active streams. It runs the `list_streams` tool and gives you an instant status update.

02 Stop building complex filtering logic in separate databases. You can use the system's key-value store—creating lists with `create_kv_list` or updating them with `update_kv_list` —to power your data pipelines directly.

03 Need to know what happened on Ethereum? Use `rpc_eth_blocknumber` to get the latest block number immediately, or use `rpc_eth_getlogs` to filter for specific events without writing a single query.

04 When a wallet changes its address, you don't need separate monitoring services. Your agent uses `create_webhook` with an EVM template to instantly capture and route that event data.

05 You can perform advanced reads using `rpc_eth_call`, which executes a message call without the overhead of creating a full transaction, saving time and gas.

06 If you need to pause or change how your pipeline works, simply use `update_stream` instead of logging into the web console to modify the existing data source.

Real-World Applications

Real-time monitoring of smart contract actions.

A compliance analyst needs to track every time a specific governance contract emits an event. The agent uses `create_webhook` with the appropriate template, directing all future events to a logging endpoint. They then use `list_webhooks` later to confirm the listener is active.

Analyzing transaction failures.

A developer needs to check why a recent deployment failed. The agent uses `rpc_eth_gettransactionreceipt` with the hash and provides a full receipt, allowing the developer to pinpoint exactly where the transaction failed.

Building a data lookup service.

A backend service needs temporary storage for mapping user IDs to contract addresses. The agent uses `create_kv_set` to store the pairs and later retrieves them using `get_kv_set`, making the data immediately available for stream processing.

Debugging historical data flow.

A data engineer suspects a gap in their archived records. They ask the agent to run `rpc_eth_getlogs` with specific filters and time ranges, getting an array of all matching log events for validation.

Patterns to Avoid

Manually checking every stream status.

✗ AVOID

A user manually opens the QuickNode console to check if 'Mainnet Ingest' is active, then checks a second dashboard for webhook status. This takes minutes and requires logging in multiple times.

✓ INSTEAD

Just prompt your agent: 'List all my streams and webhooks.' The system runs `list_streams` and `list_webhooks`, providing one consolidated list of everything that's running.

Assuming data exists in the KV store.

✗ AVOID

A developer tries to read a key-value pair by hardcoding the name, only to find an empty result because they forgot to update it first. This causes brittle code that breaks on change.

✓ INSTEAD

First, use `get_kv_list` or `get_kv_set` to confirm the data exists and is correct. If you need to modify it, run `update_kv_list` before reading.

Forgetting a stream needs configuration updates.

✗ AVOID

The blockchain network changes its default gas fee structure, but the developer forgets to update the streaming pipeline's settings in the console. The data feed stops silently.

✓ INSTEAD

Use `update_stream` to make necessary adjustments to an existing stream's parameters without having to recreate or delete the entire setup.

The Right Fit

You should use this MCP if your job involves monitoring, processing, or querying complex, real-time blockchain data. Specifically, if you need to build automated pipelines that react to contract events (using `create_webhook`), manage continuous data feeds (`create_stream`), or perform quick reads against the network state (`rpc_eth_blocknumber` or `rpc_eth_call`). Don't use this if your goal is simply reading static, non-blockchain documents. For that, you need a standard file storage MCP. Also, don't try to use it for writing smart contract code—it manages infrastructure, not the source code itself.

The Web3 data flow used to require three different consoles.

Today, if you needed to see what was happening on a blockchain, you'd jump through hoops. You open your terminal for RPC calls, switch to the streaming dashboard to manage continuous feeds, and then log into a separate webhook service just to check event triggers. Copying keys, refreshing tabs, and jumping between UIs is exhausting.

With this MCP, all of that complexity disappears. Your agent handles it in plain language. You simply ask it to monitor activity or retrieve specific logs. The outcome is a single, clear answer without you ever leaving your primary work environment.

QuickNode lets you manage your entire data infrastructure.

Before this MCP, updating an active stream required deleting the old configuration and manually rebuilding it. Changing a webhook meant clicking through multiple template selectors and submitting forms in sequence.

Now, tell your agent to update or create the resource. It uses tools like `update_stream` or `create_webhook`, making infrastructure management as simple as sending a natural language command.

QuickNode: 18 Tools for Web3 Data Operations

These tools let you manage the entire lifecycle of your blockchain infrastructure, from creating data streams to executing complex read-only RPC calls.

#	TOOL	DESCRIPTION
01	`create_kv_list`	Builds a new list structure for storing key-value data pairs.
02	`create_kv_set`	Creates a single, specific key-value pair in the storage system.
03	`create_stream`	Sets up a new continuous data stream to monitor blockchain activity.
04	`create_webhook`	Deploys an automated webhook listener based on a defined event template.
05	`delete_kv_set`	Removes a specific key-value pair from the storage system.
06	`delete_stream`	Shuts down and removes an existing data stream configuration.
07	`delete_webhook`	Removes a previously deployed webhook listener.
08	`get_kv_list`	Retrieves all the items currently stored in a key-value list structure.
09	`get_kv_set`	Fetches the value associated with a specific key from the storage sets.
10	`get_stream`	Retrieves full details about one specific, existing data stream.
11	`list_streams`	Shows a list of all currently active blockchain data streams.
12	`list_webhooks`	Retrieves the details for every deployed webhook listener.
13	`rpc_eth_blocknumber`	Checks and returns the current number of the most recent block on the network.
14	`rpc_eth_call`	Executes a read-only message call directly against the blockchain without making a transaction.
15	`rpc_eth_getlogs`	Finds and returns an array of all logs that match a specific filter object.
16	`rpc_eth_gettransactionreceipt`	Gets the complete receipt details for any transaction using its unique hash.

#	TOOL	DESCRIPTION
17	update_kv_list	Adds or modifies items within an existing key-value list structure.
18	update_stream	Makes changes to the configuration settings of a live data stream.

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 QuickNode streams.

I've retrieved your streams. You have 2 active streams: 'Mainnet-Ingest' (ID: st_123) and 'Polygon-Archive' (ID: st_456). Would you like the details for one of them?

U Create a new webhook for EVM wallet filtering using the template 'evmWalletFilter'.

I'm creating the webhook with the 'evmWalletFilter' template. Please provide the configuration JSON for the addresses you want to monitor.

U What is the current block number on the network?

The most recent block number on the connected network is 18,452,910.

Frequently Asked Questions

01 **How do I check if my data stream is working with QuickNode MCP?**

To see what's running, ask your agent to list all active streams. It runs `list_streams` and provides a summary of every configured pipeline.

02 **Can I use the Webhook tools in QuickNode MCP for testing?**

Yes. You can create webhooks using templates like 'evmWalletFilter' via `create_webhook`. Your agent will help you deploy these listeners and later, `list_webhooks` shows what you set up.

03 **What is the difference between rpc_eth_call and other RPC tools in QuickNode MCP?**

The `rpc_eth_call` tool executes a message call immediately, letting you read data without paying for a transaction. It's ideal for non-state-changing reads.

04 **How do I delete old resources in the QuickNode MCP?**

You use specific tools like `delete_stream` or `delete_webhook`. Just tell your agent which resource to remove, and it handles the cleanup for you.

05 **Can I query data using the KV Store in QuickNode MCP?**

Yes. You use tools like `create_kv_list` or `get_kv_set` to manage key-value pairs, which powers advanced filtering logic for your streams.

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 `"quicknode": { "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

QuickNode 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 QuickNode. 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	QuickNode MCP
Server ID	`019e38df-0c22-7271-ad60-59630aab09e1`
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/quicknode.