

MCP SERVER

NO CODE

CLOUD HOSTED

MTA Transit MCP

Track every bus, train, and rail movement in NYC.

MTA MCP gives your AI client direct access to New York City's entire public transit network. You can track subway trains, buses, and commuter rails in real-time, check for service disruptions across all lines (LIRR, Metro-North), and get accurate arrival predictions instantly. It's the single source of truth for NYC travel planning.

A+ Quality Score 100/100

real-time-transit

gtfs-rt

public-transportation

arrival-predictions

service-alerts

commuter-rail



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.

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.

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.

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

Encryption at rest

Ed25519

PKI vault signatures

24h TTL

Ephemeral session keys

V8 Isolate

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.

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.

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.

06 — Honeypot Trap System

Phantom credentials are injected into isolated environments. If a honeypot 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.

FinOps

Cost tracking per tool, payload compression savings, budget optimization signals, and consumption trends.

Firewall & DLP

PII redaction activity, sensitive data protection counters, and security event timeline.

Agent Activity

Which AI clients are connecting, how often, and what they're doing — real-time session tracking.

Tool Health

Slowest and most error-prone tools, with actionable root-cause insights and performance baselines.

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.

MTA MCP

12 tools available
Cloud-hosted on Vinkius

Your AI client now acts as a dedicated transit analyst for New York City. Instead of juggling multiple websites or opening separate apps to check subway status, bus arrivals, and commuter train delays, you talk to your agent once. This MCP connects your AI client to all the real-time data sources used by MTA—from tracking every active bus vehicle to monitoring major commuter lines like LIRR and Metro-North. It pulls live feed data for all NYC Subway lines, tracks specific buses at stops, and gives you immediate service alerts if anything is wrong with any part of the system. Using this through Vinkius makes your AI client a true trip planner that understands complex rail and bus schedules. You get accurate arrival predictions, even knowing which route (local vs. express) is coming next.

Core Capabilities

01 — Monitor all service alerts

Check for active disruptions across the entire MTA system, including buses, subways, LIRR, and Metro-North.

03 — Predict bus arrivals at a stop

Determine exactly when the next bus is expected to arrive, or track multiple routes approaching one specific spot.

05 — Monitor commuter rail lines

Get real-time status and arrival predictions for both LIRR and Metro-North trains into Manhattan.

02 — Track subway lines in real time

Get live train positions and arrival predictions for every major NYC Subway line.

04 — Track all MTA vehicles system-wide

View the current GPS positions of every active bus on the road across NYC.

06 — Discover station network details

List all NYC Subway stations, including their coordinates and borough information.

One Click on Vinkius — From Prompt to Execution

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

- 01 Subscribe to this MCP on Vinkius and input your MTA API key.
- 02 Connect your AI client (Claude, Cursor, etc.) to the catalog using your credentials.
- 03 Ask your agent a specific transit question—for example, 'What are the delays on the LIRR' or 'When does the next M15 bus arrive at 34th Street?'

The bottom line is you get accurate, immediate answers about NYC transit without needing to open any separate transit app.

Built For

Anyone who relies on predictable movement in New York City—from tourists getting lost at 2 AM to logistics managers planning cross-borough transfers. If your job involves knowing where a vehicle is right now, you need this.

NYC Commuter

Uses the MCP to check service alerts and track LIRR or subway trains before leaving home, ensuring they aren't caught by an unexpected delay.

Tourism Operations Lead

Requests a list of all stations in Manhattan or checks real-time bus predictions for groups needing to navigate the city on foot.

Logistics Coordinator

Monitors specific MTA bus routes and tracks vehicles system-wide to estimate delivery windows near major transit hubs.

What Changes When You Connect

- 01 Avoid manual checks by asking for service alerts; the `get_service_alerts` tool immediately tells you if LIRR or subway lines are experiencing major delays.

-
- 02** Pinpoint exact arrivals: Instead of guessing, use `get_bus_predictions` to know exactly when the next bus is coming to your specific stop.
-
- 03** See the full network picture by using `get_bus_routes` to list every service area and then pinpointing stops with `get_bus_stops` .
-
- 04** Track trains across all modes: Monitor subway movements using `get_subway_feed` , while simultaneously checking commuter rail status via `get_lirr_feed` or `get_metro_north_feed` .
-
- 05** Know where vehicles are right now: Use `get_bus_vehicles` to see the system-wide positions of every active bus, not just those near you.
-

Real-World Applications

Planning a cross-borough commute

A user asks their agent for travel plans from Queens to the Upper West Side. The agent first uses `get_service_alerts` to check for weekend maintenance, then uses `get_subway_feed` to predict arrival times on the 4/5/6 line, and finally checks `get_bus_predictions` if a bus connection is needed.

Diagnosing a service failure

A transit analyst asks, 'Why are buses running slow today?' The agent checks `get_bus_vehicles` to see cluster density and then runs `get_service_alerts` to correlate the delay with known incidents.

Managing a field crew visit

A manager needs to know where several buses are in relation to an event. They run `get_bus_vehicles` and then use `get_bus_stops` to find the closest official boarding points for their staff, ensuring no one is left stranded.

Guiding a tourist through the system

A tourist asks for the nearest train. The agent uses `get_stations` to list nearby stations, checks `get_subway_feed` for real-time status on that specific line, and provides immediate directions.

Patterns to Avoid

Asking only about 'delays'

X AVOID

I just want to know if MTA is delayed today.

✓ INSTEAD

Don't ask generally. Always specify the mode and line by using ``get_service_alerts`` first, then follow up with specific data like running ``get_lirr_feed`` or ``get_subway_feed`` for precise status.

Assuming bus routes are simple

X AVOID

List all buses that go near Times Square.

✓ INSTEAD

First, use ``get_bus_stops`` to find the exact stop ID near Times Square. Then, run ``get_bus_predictions`` with that specific ID for accurate results.

Confusing system scope

X AVOID

Does this cover all trains in NYC?

✓ INSTEAD

This MCP separates data by type: use ``get_subway_feed`` for subway lines, and use ``get_lirr_feed`` or ``get_metro_north_feed`` for the commuter rails. You need to call different tools for different rail types.

The Right Fit

Use this MCP if your goal is real-time operational intelligence about NYC transit movement. Specifically, you need to know *when* a vehicle will arrive or *if* there's an active disruption on a named line. If you only need static geographical data (like the coordinates of every subway station), `get_stations` handles that perfectly. However, if your task is simply 'What are my options?' and not 'When do those options run?', then relying solely on general map or directions APIs might be enough. Don't use this if you just need a list of all MTA bus routes; instead, start by running `get_bus_routes` to get the IDs needed for subsequent queries.

Checking NYC Transit Used To Be a Messy Chore

Today, figuring out your commute means opening half a dozen tabs: checking Google Maps for bus routes, going to the LIRR site for commuter delays, and maybe looking at the MTA website just to see if there's an alert on the subway. You're cross-referencing schedules, manually copy-pasting station names, and spending time figuring out which data source is up to date.

With this MCP, you tell your agent what you need —say, 'How will I get from Brooklyn to Harlem?' The AI client handles the coordination. It checks for service issues using `get_service_alerts`, estimates bus arrival times with `get_bus_predictions`, and even tracks if the train is running late via `get_subway_feed`. You just get one reliable answer.

MTA MCP Gives You Full Visibility into All NYC Transit Movements

You no longer have to manually track down the right data set. Instead of running separate queries for LIRR positions and Metro-North positions, you simply ask your agent to monitor both feeds using `get_lirr_feed` and `get_metro_north_feed`. The system handles the complexity.

The difference is control. You move from passively viewing schedules to actively predicting movements, knowing exactly when a bus will pull into your stop via `get_bus_vehicle_at_stop`.

MTA MCP Catalog (12 Tools)

Use these 12 tools to retrieve specific data points like vehicle positions, arrival predictions, or service disruption details across the MTA network.

#	TOOL	DESCRIPTION
01	<code>get_bus_estimated_arrival</code>	Predicts when buses will arrive at a stop, optionally filtering results by the specific route number.
02	<code>get_bus_predictions</code>	Provides real-time predictions for next bus arrivals at any given bus stop location.
03	<code>get_bus_routes</code>	Lists all active MTA bus routes serving the city, including their names and service areas.
04	<code>get_bus_stops</code>	Retrieves a list of all physical bus stops for any specified route, giving coordinates and sequence order.
05	<code>get_bus_vehicle_at_stop</code>	Tracks buses that are currently at or approaching a specific stop with estimated arrival times.
06	<code>get_bus_vehicles</code>	Gets the real-time GPS coordinates and status of all active MTA bus vehicles across NYC.
07	<code>get_lirr_feed</code>	Retrieves live tracking, delays, and arrival predictions for Long Island Rail Road trains.
08	<code>get_metro_north_feed</code>	Provides real-time tracking and service updates for Metro-North Railroad commuter trains.
09	<code>get_service_alerts</code>	Checks the current status of major disruptions, maintenance, or incidents across all MTA modes.
10	<code>get_stations</code>	Lists detailed information on every NYC Subway station, including its borough and coordinates.
11	<code>get_subway_feed</code>	Gathers live train positions, delays, and arrival data for specific major subway lines (e.g., the 1 train or the A line).
12	<code>get_system_time</code>	Returns the official MTA system timestamp to verify API connection integrity.

See It in Action

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

U Show me the next trains on the 1/2/3 line.



Fetching subway feed for lines 1/2/3/4/5/6/S (feed ID: 1)... Real-time GTFS-RT data shows: 1 Train to Van Cortlandt Park — 3 stops away, arriving in 4 minutes; 2 Train to Wakefield-241st — 5 stops away, arriving in 7 minutes; 3 Train to Harlem-148th St — 8 stops away, arriving in 11 minutes. All trains currently running on or close to scheduled times with no major delays reported. Would you like predictions for a specific station?

U When is the next M15 bus arriving at the stop near 14th Street and 3rd Avenue?



Looking up M15 route stops... Found stop ID 308238 at 14 St/3 Av. Checking real-time predictions: M15 Select Bus Service southbound to South Ferry — arriving in 2 minutes; M15 local southbound — arriving in 8 minutes; M15 SBS northbound to East Harlem — arriving in 12 minutes. All vehicles currently on schedule. The SBS stop requires fare payment at street-level machines before boarding. Would you like me to track the exact GPS positions of these buses?

U Check if there are any service alerts affecting the LIRR right now.



Checking MTA service alerts... Currently 2 active LIRR alerts: 1) Babylon Branch — Trains experiencing 10-15 minute delays due to earlier signal problem at Jamaica. Normal service expected to resume by 6:30 PM. 2) Port Jefferson Branch — Track work in effect between Huntington and Northport, expect single-tracking with 20 minute delays through 11:00 PM tonight. All other LIRR branches running on schedule. Penn Station and Grand Central Madison terminals operating normally. Would you like me to check specific train predictions for your branch?

Frequently Asked Questions

01 Does the MTA MCP cover all NYC Subway lines?

Yes. The ``get_subway_feed`` tool supports major subway lines by their feed ID, providing real-time status for most of the system.

02 How do I find a bus stop ID to use with `get_bus_predictions``?

Use the ``get_bus_stops`` tool. This function lists all stops for any given MTA bus route and provides the required unique Stop IDs.

03 Can this MCP track buses system-wide or just near me?

It handles both. You can check every vehicle's location using ``get_bus_vehicles``, or narrow it down to specific approaching vehicles with ``get_bus_vehicle_at_stop``.

04 Is the LIRR data separate from subway data in MTA MCP?

Yes. The system keeps commuter rail and subway data distinct. Use ``get_lirr_feed`` for Long Island Rail Road status, and ``get_subway_feed`` for underground trains.

05 What if I just need to know the total number of stations?

You can run ``get_stations``. This tool returns a complete list of all NYC Subway station records, giving you the necessary count and geographic details.

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 `"mta": { "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

MTA 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](https://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 MTA. 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	MTA MCP
Server ID	019d75d9-551d-7319-b252-ed7f0b5c23fc
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/mta.