MCP SERVER | NO CODE | CLOUD HOSTED

Telegram Bot MCP

Audit Groups, Send Media, Manage Bots via Chat

Telegram Bot MCP lets your AI agent manage and administer any Telegram bot without touching code or a terminal. Instantly send rich media, audit group membership lists, check chat details, and automate complex messaging workflows using natural conversation through Vinkius.

A+ Quality Score 98.33/100

bot-management | chat-administration | real-time-messaging | webhook | api-integration | notifications

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/telegram-bot-alternative

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.

Telegram Bot MCP

12 tools available
Cloud-hosted on Vinkius

This connector turns tedious bot administration into a conversation with your AI client. Instead of logging into the Bot API documentation to figure out how to do something, you just ask your agent. You can send formatted text or photos directly to users and groups, perfect for support instructions or announcements. Need to know who has admin access in a large group? Your agent runs an audit right away, giving you a list of admins and the current member count. Plus, you can manage the backend setup, like setting up webhooks or deleting them if needed. If your bot needs to leave a supergroup because it's irrelevant, it handles that too. Connecting this MCP through Vinkius means all these advanced features are available instantly within any compatible client, letting your agent act as a real-time bot operator for everything from customer support to community management.

Core Capabilities

01 — Send multimedia messages

Your agent sends formatted text and photos immediately to specific users or entire groups.

02 — Audit chat status

You check the full details of a chat, list all administrators, and retrieve the total number of members in any group.

03 — Manage bot backend settings

The agent can inspect or change webhooks and fetch recent updates without manual API calls.

04 — Handle shared files

You retrieve metadata about media files that were shared with your bot, keeping records organized.

05 — Automate group membership

The agent autonomously leaves a supergroup or chat when its purpose has changed.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/telegram-bot-alternative** — connect your AI agent in three steps.

01 Subscribe to this MCP and provide your unique Telegram Bot Token (obtained from @BotFather).

02 Connect the token to your preferred AI client, like Cursor or Claude.

03 Ask your agent a question, such as 'List all admins in group X,' and it executes the necessary commands automatically.

The bottom line is you talk to your agent how you normally would; it handles the complex chat administration logic behind the scenes.

Built For

This MCP is for community managers, support teams, and ops engineers who spend too much time clicking through multiple dashboards or writing boilerplate code just to keep a group running smoothly. It hands you back your time.

Community Manager

Runs regular audits on large groups to check member counts and identify key administrators without manually checking settings.

Technical Support Agent

Sends quick photo guides or specific text instructions to a user directly from the chat interface, keeping communication fast and visual.

Operations Lead

Manages bot health by inspecting webhook status or automatically leaving groups that no longer meet established criteria.

What Changes When You Connect

01 Stop manually checking group rosters. Using the `get_chat_administrators` tool lets your agent instantly audit who has admin access in a large community chat.

02 Need to send instructions? Instead of drafting it and hitting 'send' multiple times, you can use `send_photo` or `send_message` to deliver rich media or text updates directly from your workspace.

03 Managing bot infrastructure used to mean writing code. Now, the agent handles complex setups using tools like `set_webhook` or `delete_webhook`, letting you manage webhooks with simple prompts.

04 Group health matters. You can use `get_chat_member_count` and then instruct your agent to execute `leave_chat` if the group falls below a required size threshold.

05 When troubleshooting, instead of logging into the API dashboard, simply ask for recent updates using `get_updates` to see exactly what happened in the chat.

Real-World Applications

Post-Launch Group Cleanup

The Ops Lead notices a community group has gone dormant. They prompt their agent, which first runs `get_chat_member_count` to confirm low activity, and then executes the `leave_chat` tool, removing the bot from dead groups automatically.

Handling Support Escalations

A support team member needs to guide a user. They use their agent to run `send_photo`, sending a step-by-step image instruction directly into the chat thread, keeping the conversation visual and contained.

Developer Bot Health Check

A developer wants to verify if the bot is connected properly. They ask their agent to run `get_webhook_info` and then immediately use `get_me` to confirm the bot's current status.

Onboarding New Admins

A community manager needs to verify who has permissions. They ask their agent to run `get_chat_administrators`, instantly receiving a list of all high-level users in the chat for review.

Patterns to Avoid

Trying to read raw JSON data

✗ AVOID

When troubleshooting, someone manually runs multiple API calls and gets back large blocks of raw JSON containing status codes, member IDs, and file hashes.

✓ INSTEAD

Don't look at the raw output. Just ask your agent to 'List all admins in this group' or 'Send a message saying X.' The MCP handles parsing the data for you.

Forgetting webhook setup

✗ AVOID

The bot sends messages, but they fail because the connection is broken. The user then manually checks external documentation to see if the webhook needs setting.

✓ INSTEAD

Just ask your agent, 'Check the webhook status.' It uses `get_webhook_info` and tells you immediately if you need to run `set_webhook`.

Confusing file types

✗ AVOID

The user sees a shared photo but doesn't know its source or details. They try clicking through different Telegram tabs looking for metadata.

✓ INSTEAD

Ask your agent to use `get_file` on the media object; it pulls all the necessary information into plain text instantly.

The Right Fit

Use this MCP if your primary need is managing complex, real-time messaging interactions within Telegram. If you spend more time writing code or manipulating API endpoints than actually communicating with users, this is for you. You'll use tools like `send_message` and `get_chat_member_count` to make those communications conversational. Don't use this if your goal is deep data analysis on non-Telegram sources; for that, you need a database connector or an external file system tool. If you just want to read basic messages but don't care about group admin audits or webhook management, other messaging tools might suffice, but nothing beats the depth of control here.

Managing Group Communication is Usually a Pain Point

Today, running a large community chat means hopping between tabs: checking group membership in one spot, sending an announcement from another, and if you need to fix the bot's connection, logging into developer tools. It’s clicking through dashboards just to maintain basic communication flow.

With this MCP, those manual steps disappear. Your agent handles everything—from using `get_chat_administrators` for audits to sending formatted photos via `send_photo`. You just talk to your AI client and get the result.

Manage Chat Administration with Telegram Bot MCP

You no longer need separate code blocks or terminal commands for every task. Setting up a webhook used to be tedious; now, you just tell your agent to run `set_webhook`, and it's done.

It’s simple: the MCP translates complex API calls into natural language instructions. You get control over message delivery and group health without ever touching boilerplate code.

Telegram Bot MCP: 12 Tools Available

These tools allow you to perform specific actions in Telegram like sending messages, auditing user lists, or managing the bot's connection settings via your AI client.

#	TOOL	DESCRIPTION
01	`delete_webhook`	Removes a currently set Telegram web address (webhook).
02	`get_chat_administrators`	Retrieves the list of users who have administrator rights in a chat.
03	`get_chat_member_count`	Counts and reports the total number of members in a specific chat or group.
04	`get_chat`	Retrieves detailed information about a specified Telegram chat.
05	`get_file`	Gets metadata and details for any media file shared with the bot.
06	`get_me`	Retrieves basic status information about the connected Telegram bot itself.
07	`get_updates`	Fetches and reports on the most recent events or messages sent to the bot.
08	`get_webhook_info`	Checks and returns the current status of the active Telegram webhook settings.
09	`leave_chat`	Instructs the connected bot to automatically exit a chat or group.
10	`send_message`	Sends plain text messages to specified users or groups instantly.
11	`send_photo`	Transmits one or more photos along with a message caption.
12	`set_webhook`	Establishes and sets the official web address (webhook) for your bot.

See It in Action

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

U Send 'Hello from Vinkius!' to chat ID 12345678.

Message delivered! I've sent your greeting to chat 12345678. The message ID is 42 and it was delivered instantly.

U Get my Telegram bot details.

I've retrieved your bot info. Your bot name is 'VinkiusHelperBot' (@vinkius_bot). It has access to groups and is currently online.

U Who are the admins in chat -100123456789?

I've found 3 administrators in that chat: 'Marcus R.' (Owner), 'Jane Doe', and 'BotAssistant'. Would you like their specific permission sets?

Frequently Asked Questions

01 **How do I use Telegram Bot MCP to send a photo?**

You simply tell your agent, 'Send a guide photo about X.' It uses the `send_photo` tool, ensuring the image and any caption are delivered correctly.

02 **Does Telegram Bot MCP let me check group membership?**

Yes. Your agent runs `get_chat_member_count` to give you an immediate count of how many people are in the chat, and you can also use `get_chat_administrators` to see who has permissions.

03 **What is the difference between using MCP and just calling the API directly?**

Using this MCP means your AI agent handles the whole process. You don't worry about the syntax; you just ask for the outcome, letting the agent run tools like `get_updates` in the background.

04 **Can I make my bot leave a group using Telegram Bot MCP?**

Absolutely. If your bot needs to exit a chat because it's no longer relevant, you prompt the agent to use the `leave_chat` tool, and it handles the departure.

05 **How do I set up webhooks with Telegram Bot MCP?**

You tell your agent what needs doing. It uses the `set_webhook` tool to establish the necessary connection point for your bot's communication flow.

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 `"telegram-bot-alternative": { "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

Telegram Bot 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 Telegram Bot. 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	Telegram Bot MCP
Server ID	`019d8488-fb76-7318-a797-c0dbd7c44f81`
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/telegram-bot-alternative.