MCP SERVER | NO CODE | CLOUD HOSTED

OneSignal MCP

Automate Comms, Check Metrics, & Manage Players.

OneSignal MCP lets you manage all customer push notifications directly from your AI agent. Send campaigns to segments, schedule messages, track delivery metrics (like open rates and failure reasons), and list specific player IDs—all without touching the OneSignal dashboard.

A+ Quality Score 100/100

push-notifications | customer-engagement | in-app-messaging | audience-segmentation | message-automation

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

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.

OneSignal MCP

10 tools available
Cloud-hosted on Vinkius

Need to send an urgent announcement or check if a user actually saw that last promotion? This MCP connects your entire customer communication layer through natural conversation. You tell your agent what you need, and it handles the API calls for sending alerts across both mobile and web platforms. Need to target only high-value users who signed up in the last week? Just ask. The agent will manage complex segmentation rules before launching the message. When you're done with the launch, you don't have to leave your workflow just to check metrics; you can fetch detailed reports on delivery status and engagement levels for any campaign. If you prefer a central place to connect all your operational tools, Vinkius brings this MCP alongside thousands of others, giving your agent one command center for everything. It’s about getting immediate answers and executing complex comms strategies without ever opening the OneSignal web interface.

Core Capabilities

01 — Launch targeted campaigns

Send push notifications to specific segments or individual device IDs.

02 — Manage notification schedules

Schedule future messages or instantly cancel notifications that need pulling back.

03 — Analyze delivery performance

Retrieve metrics and detailed reports on how many users received a message and if it failed, and why.

04 — Audit user records

List all registered devices or check specific player details to verify an ID is active.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP and provide your OneSignal REST API Key and App ID.

02 Optionally, supply a User Auth Key if you need full account control for listing all apps or managing configurations.

03 Start chatting with your AI client. You can then ask the agent to send an alert, check a user's status, or pull delivery reports.

The bottom line is: you use natural language prompts to trigger complex communication actions and data retrievals managed by this MCP.

Built For

This connector is built for the Growth Marketing Manager who hates context switching, or the Support Agent who needs instant comms verification. If you spend time jumping between dashboards just to send an alert or check metrics, this MCP saves your afternoon.

Marketing Campaign Manager

Launching A/B tests for push notifications or scheduling a major product announcement across multiple segments.

Customer Support Specialist

Verifying whether a specific user received a critical system alert by checking their player ID metadata and delivery logs.

Product Owner

Testing notification payloads for new features and listing all current applications to maintain an overview of the communication stack.

What Changes When You Connect

01 Launch messages instantly: Use the `create_notification` tool to send campaigns or alerts to any defined segment without leaving your chat environment.

02 Stop wasted effort: If a promotion needs pulling back, simply ask the agent to execute `cancel_notification`. It handles the scheduling logic immediately.

03 Know who saw it: Check delivery status for specific comms using `get_notification` or audit player details with `get_player`, providing immediate accountability.

04 Targeting made simple: The MCP allows you to run complex queries, such as listing all registered devices via `list_players`, ensuring your campaigns hit the right people.

05 Audit everything: Need a full picture of your comms infrastructure? Use `list_apps` and `get_outcomes` to review every configuration point in one chat session.

Real-World Applications

The marketing team needs to send an emergency update.

A manager realizes a critical bug is affecting users. Instead of logging into the web portal, they simply prompt their agent: 'Send a message about the outage now.' The agent uses `create_notification` and targets the 'All Users' segment instantly.

Support needs to prove delivery failure.

A user complains they never got a welcome email. A support specialist asks their agent to check the status of that notification using `get_notification`. The system replies with the exact metrics, confirming if it was sent and where it failed.

Developer needs to clean up old accounts.

A developer identifies a defunct testing account. They ask their agent to delete that player's record using `delete_player` so the test data doesn't clutter the active user list.

Product owner wants an infrastructure overview.

The PO needs to know what notification types exist. They prompt: 'List all connected apps and their details.' The agent runs `list_apps` and `get_app`, giving them a full, clean inventory.

Patterns to Avoid

Checking delivery status manually

✗ AVOID

Having to navigate to the 'Analytics' tab in the OneSignal dashboard, then search for the specific notification ID and wait for reports to load.

✓ INSTEAD

Just ask your agent: 'What was the outcome of notification 550e8400...' The agent executes `get_notification` or `get_outcomes` instantly, giving you metrics without leaving your chat.

Sending a message to an unknown group

✗ AVOID

Trying to guess which segment of users are active. This often leads to sending messages to people who don't care and wasting resources.

✓ INSTEAD

First, ask the agent to run `list_players` to see what devices exist. Then, you can use that list information when calling `create_notification`, ensuring high-value targeting.

Trying to delete a player in multiple steps

✗ AVOID

Finding a device ID via the main dashboard, copying it, and then switching tabs to find the deletion tool.

✓ INSTEAD

Tell your agent: 'Delete this specific player.' The agent uses `delete_player` directly with the provided ID, keeping the entire process conversational.

The Right Fit

Use this MCP if your core requirement involves managing user communication—sending alerts, tracking delivery status, or segmenting audiences. If you need to perform basic CRUD operations (like creating a new record in a CRM) or run complex financial calculations, this isn't the right tool; look for a database-type MCP instead. However, if you only need to read general user profile data without linking it to comms history, a dedicated user management tool is better suited. If your goal is simply to list what apps are available but never actually send anything, `list_apps` works, but remember that this MCP's strength lies in the action: launching communication and pulling metrics on those launches.

Managing customer communications used to be a dashboard nightmare.

Today, if you want to send an alert or check performance, you have to stop what you're doing. You click into the OneSignal portal, find the 'Send' button, select your audience segment from a dropdown menu, write the copy, hit schedule, and then maybe open a *second* tab just to see how many people opened it.

With this MCP, that whole sequence vanishes. You simply tell your agent what you want—a targeted message for new users today—and you get an immediate confirmation of its launch status. Everything happens in one conversation, meaning zero context switching.

The OneSignal MCP gives you command over comms.

You no longer have to manually retrieve lists of users or check if a specific message was delivered. You can ask the agent to pull all registered devices using `list_players`, and then immediately use that data to launch a campaign via `create_notification`.

It changes your work from an operational chore requiring multiple clicks into a direct conversation with measurable outcomes.

OneSignal MCP with 10 Tools

Use these tools to manage everything from sending alerts to checking player records, all through simple conversation with your AI client.

#	TOOL	DESCRIPTION
01	`cancel_notification`	Stops a scheduled alert from being delivered to users.
02	`create_notification`	Sends an immediate or targeted OneSignal message.
03	`delete_player`	Removes a specific device ID from the user's registered devices.
04	`get_app`	Fetches detailed configuration information for one of your OneSignal apps.
05	`get_notification`	Retrieves the current status and details of a specific notification ID.
06	`get_outcomes`	Pulls overall outcome data for your entire OneSignal application.
07	`get_player`	Retrieves comprehensive details about a specific player device ID.
08	`list_apps`	Generates a list of every configured OneSignal application you have access to.
09	`list_notifications`	Lists all historical notifications that were sent for your app.
10	`list_players`	Provides a list of all registered device IDs currently tracked by OneSignal.

See It in Action

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

U Send a push notification to all subscribed users saying 'Flash Sale: 50% off today only!'

Notification sent successfully! It is being delivered to the 'Subscribed Users' segment. Notification ID: 493021-abcd-1234.

U Check the delivery status of notification 550e8400-e29b-41d4-a716-446655440000.

I've retrieved the metrics. The notification reached 1,250 devices with a 98% success rate. 25 deliveries failed due to unsubscribed users.

U List the last 5 registered devices for our app.

I've fetched the latest device registrations. Here are the 5 most recent players, including their device types (iOS/Android) and last active timestamps.

Frequently Asked Questions

01 **Can the OneSignal MCP send messages globally?**

Yes, it supports sending alerts to both mobile and web platforms. You only need to specify the target segment or user ID when calling `create_notification`.

02 **What is the difference between list_players and get_player in OneSignal MCP?**

`list_players` returns a broad inventory of all registered device IDs. If you know the specific ID, use `get_player` to pull detailed metadata for that single player.

03 **Does this MCP support scheduling future notifications?**

Yes. You can schedule messages using the capabilities exposed through `create_notification`, allowing you to plan campaigns days or weeks ahead of time.

04 **How do I check if a user was successfully reached?**

You use the agent to fetch specific data by requesting the notification ID with `get_notification`. This gives you real-time status and delivery metrics for accountability.

05 **Is there a way to mass delete users via OneSignal MCP?**

The toolset provides `delete_player`, which deletes one registered device at a time. You must list the players first using `list_players` and then pass them individually for deletion.

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 `"onesignal": { "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

OneSignal 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 OneSignal. 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	OneSignal MCP
Server ID	`019d75e5-c99b-707d-b205-dc372a188057`
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/onesignal.