MCP SERVER | NO CODE | CLOUD HOSTED

Google Chat Webhook Notifier MCP

Send structured alerts into your team chat.

The Google Chat Webhook Notifier MCP sends precise, structured messages and rich alerts directly into your designated Google Chat space. It acts as a zero-trust bridge for your AI agent to post status updates, deployment reports, or incident notifications without needing complex OAuth permissions or compromising your corporate data. This is the safest way to give your automation a voice in team chat.

F Quality Score 3.6/100

webhooks | notifications | alerts | team-collaboration | messaging | automation

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/google-chat-webhook-notifier

SCAN TO VIEW

The infrastructure that powers AI agents in the real world.

Vinkius connects AI to the world's software through secure, enterprise-grade infrastructure — enabling real-world execution at scale, built on the Model Context Protocol (MCP).

Your AI Connections Run Through Vinkius Cloud

The world's largest managed MCP catalog

Vinkius is the cloud infrastructure where AI agents connect 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.

Google Chat Webhook Notifier MCP

1 tools available
Cloud-hosted on Vinkius

When you need your automated workflows to talk to your team, this MCP handles the delivery. It bypasses the headache of full Google Workspace integrations that demand massive security scopes across your entire company directory. Instead, it gives your AI agent a surgical tool: a simple webhook URL. This means your agent can drop critical alerts, deployment statuses, and structured engineering reports straight into a specific chat channel without reading any emails or snooping on other conversations. The process is straightforward and secure. Because this MCP is hosted on Vinkius, you connect your preferred AI client once and gain access to this specialized notification capability alongside thousands of others. Your agent gains the immediate ability not just to send plain text updates, but also to build rich Cards v2 layouts—complete with styled sections and interactive buttons—making alerts much easier for people to consume on their phones or desktops.

Core Capabilities

01 — Post status notifications

Sends simple text messages alerting the team that a task has completed, failed, or is running.

02 — Deliver structured alerts

Creates rich Cards v2 layouts that contain styled data, buttons, and formatted sections for complex reports.

03 — Isolate alert scope

Ensures the agent can only write to the single designated chat space, maintaining strict security boundaries.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/google-chat-webhook-notifier** — connect your AI agent in three steps.

01 Your AI client determines that a specific event occurred (e.g., deployment finished).

02 It invokes this MCP's tool to format the message content and specifies the target Google Chat space.

03 The service posts the final notification, rich cards included, directly into the designated chat channel.

The bottom line is that your agent sends a pre-packaged, secure alert package right where you need it, no complex setup required.

Built For

This MCP is for DevOps engineers and SREs who are tired of manually copying status updates from terminal logs or CI/CD dashboards into Slack. It's also useful for incident response teams needing immediate, structured communication during an outage.

DevOps Engineer

Needs to push real-time deployment success or failure messages directly from the build pipeline into a dedicated team chat channel.

Site Reliability Engineer (SRE)

Uses this MCP to send structured, high-priority incident alerts containing context and remediation steps when an outage is detected.

Backend Developer

Requires a way to notify the entire team instantly when a microservice has been successfully updated or rolled back.

What Changes When You Connect

01 Security is built-in. Because this MCP uses a simple Incoming Webhook, the agent can only write to one space; it cannot read emails or interact with other corporate data. This keeps your system strictly contained and safe.

02 Rich formatting makes alerts readable. The agent doesn't stick to plain text. It uses Cards v2 to programmatically generate structured content—think buttons, styled sections, and clear headings—so critical information isn't missed in a wall of words.

03 Setup takes minutes. You skip the entire pain point of configuring service accounts or complex OAuth flows with Google Cloud Projects. If you have a webhook URL, your AI can speak.

04 Instant visibility across teams. Instead of having to check three different dashboards (CI/CD, monitoring, logs) for status updates, one single message appears in Chat that consolidates all the necessary information.

05 Flexible reporting. You don't just send 'Success.' You can use this MCP to deliver detailed reports, like listing which services were updated or what version was deployed, keeping context right where people read it.

Real-World Applications

Post-Deployment Status Update

A developer finishes a feature branch merge. Their agent uses this MCP to immediately send a rich Card alert into the #devops channel, detailing the commit hash and confirming that all automated tests passed. The team knows instantly what was deployed without checking Jira or Slack manually.

Handling System Failures

The monitoring system detects an elevated error rate. The agent automatically calls this MCP to push a highly visible, structured incident alert into the #oncall channel. This message includes links to runbooks and the current severity level, notifying the team faster than any email could.

Automated Audit Logging

A backend process completes a large data migration task. The agent uses this MCP to send a simple confirmation message to management, stating that 10 million records were successfully processed and logged, creating an undeniable audit trail in the chat history.

Workflow Completion Confirmation

A complex multi-step workflow finishes. The agent calls this MCP to send a final notification summarizing all completed tasks (e.g., 'Database migration complete,' 'Cache flushed,' 'Service restarted'), providing one single source of truth for the team.

Patterns to Avoid

Trying to read history

✗ AVOID

Thinking your agent can check if someone mentioned a specific feature two weeks ago in an old chat thread.

✓ INSTEAD

This MCP only sends messages; it cannot retrieve past conversations or search message history. You must use other tools designed for reading and indexing data.

Using generic API calls

✗ AVOID

Writing custom Python code to handle the complex formatting and webhook URL management.

✓ INSTEAD

Use the `send_google_chat_message` tool. It handles the entire message payload structure, including rich Cards v2 JSON, so you only worry about the content.

Over-relying on plain text

✗ AVOID

Sending a long block of raw log output that is hard to read and scroll through.

✓ INSTEAD

Always include rich data in the `cardJson` parameter when calling the tool. This forces structured, scannable information into the message.

The Right Fit

Use this MCP if your primary need is *outgoing* communication—you need to send a notification or status update from an external process (like a CI/CD pipeline or monitoring system) directly into Google Chat. This tool excels at delivering structured, one-way alerts with minimal security overhead.

However, don't use this if you need two-way conversation. If your goal is for the agent to read specific conversations, retrieve history, or respond dynamically based on chat context (like asking 'What did Bob say about X last week?'), this tool won't work—it only publishes messages. Similarly, if you need the agent to manage permissions or see who can access the space, it lacks those controls. For reading data, look for MCPs built around knowledge bases or message retrieval tools instead.

The Pain of Status Copy-Pasting

Today, when a critical job finishes, you're faced with the tedious process: check the Jenkins dashboard for green lights. Then, copy the success message into Slack. If it's an incident, you run to the monitoring tool, grab the error code, and paste that into your on-call chat, all while making sure formatting looks right so people don't ignore it.

With this MCP, the status update flows automatically. Your agent delivers a single, rich alert directly into the designated channel. The team gets everything they need—success/failure, context, and next steps—in one clean message without you lifting a finger.

Send Structured Alerts with send_google_chat_message

You eliminate the need to manually construct HTML or worry about text wrapping. By providing the `cardJson` in the tool, you enforce a clean structure for your alerts; it automatically renders interactive buttons and styled sections that look professional regardless of device.

This means operational status updates are no longer just blocks of text; they are actionable pieces of information that people can digest instantly. It moves communication from simple logging to genuine team coordination.

Google Chat Webhook Notifier with 1 Tool

Use the available tool to programmatically send simple messages or rich, structured alerts into any designated Google Chat space.

#	TOOL	DESCRIPTION
01	`send_google_chat_message`	Sends a notification or message, optionally with rich card formatting, to a specific Google Chat Space webhook.

See It in Action

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

U Notify Google Chat that the database backup is finished.

I've sent the message 'The database backup is finished.' to the Google Chat space.

U Send a rich alert to Google Chat using a Cards v2 layout for a security incident.

The rich Cards v2 alert detailing the security incident has been successfully posted to Google Chat.

Frequently Asked Questions

01 **How does the Google Chat Webhook Notifier MCP work with security?**

It uses a secure Incoming Webhook URL, which is designed only for writing data. This means your agent cannot read private messages or access any part of your corporate email system—it's purely outgoing.

02 **Can the Google Chat Webhook Notifier MCP send plain text and rich cards?**

Yes, it supports both. You can provide fallback plain text in the message body, but you should use the `cardJson` parameter to create structured, visually appealing alerts.

03 **Does this MCP need complex Google Cloud authentication?**

No. The core value of this MCP is its simplicity. It bypasses the complicated OAuth scope management by relying only on a basic webhook URL.

04 **Is the message sent to all users in the chat space?**

The message goes into the designated Google Chat Space, ensuring that every member of that team sees the critical alert instantly. It's ideal for dedicated #alerts channels.

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 `"google-chat-webhook-notifier": { "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

Google Chat Webhook Notifier 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 Google Chat Webhook Notifier. 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	Google Chat Webhook Notifier MCP
Server ID	`019e38a1-2ef8-72cc-984d-3213ce5b0c4b`
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/google-chat-webhook-notifier.