MCP SERVER | NO CODE | CLOUD HOSTED

Pumble MCP

Automate communication management inside your team's chat.

Pumble MCP connects your AI agent directly into the Pumble communication workspace. It lets you manage team discussions, read conversation history across every channel, create new project channels on demand, and pull detailed user profiles for accurate tagging. Use it to automate status updates, organize community Q&A sessions, or broadcast standardized announcements without ever leaving the chat interface.

A+ Quality Score 100/100

messaging | team-collaboration | channel-management | automation | user-directory

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

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.

Pumble MCP

10 tools available
Cloud-hosted on Vinkius

You can connect your AI agent right into Pumble to automate how teams communicate. Instead of manually checking different channels for project statuses, you ask your agent to list all open and private discussion hubs; it gives you a full map of your workspace. Need to kick off a new initiative? Your agent creates the channel instantly. It doesn't just talk about messages—it handles them. You can retrieve entire conversation histories or post updates directly into specific threads, even correcting typos on old announcements. For tracking team members, your agent lists all users and pulls key details like emails and time zones so you know who to tag. This capability is available through the Vinkius Marketplace, giving your AI client access to robust communication tools without needing custom integrations.

Core Capabilities

01 — Discovering Channels

The agent can list all public and private channels in your workspace or fetch detailed information about a specific channel.

02 — Managing Communications

You can send new messages, update existing ones, add emoji reactions to acknowledge requests, or delete outdated announcements.

03 — User Directory Lookup

The agent lists all workspace users and pulls detailed profiles, including emails and time zones, for accurate team tagging.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP on the Vinkius Marketplace.

02 Authorize access using your Pumble API Key (which you generate in-app).

03 Your AI client can then run commands, letting you manage channels and messages directly through natural conversation.

The bottom line is that once connected, your agent acts as a native command center for all things happening inside Pumble.

Built For

Operations leads who are tired of manually provisioning project channels and broadcasting status updates across multiple platforms. Community managers struggling to keep up with discussion threads, or engineering teams needing immediate communication about bugs.

Community Manager

Uses the agent to scan entire channel histories for common questions, allowing them to generate and post comprehensive answers in one go.

Operations Lead

Triggers the creation of new project channels instantly and broadcasts identical onboarding messages across dozens of newly provisioned groups.

Engineering Manager

Uses the agent to automatically trigger status updates or bug reports directly into dedicated, high-priority development channels.

What Changes When You Connect

01 Instant Channel Provisioning: Need a new project channel? Use the `create_chat_channel` tool to spin up and name a dedicated discussion hub instantly, eliminating manual setup steps.

02 Centralized Communication History: Never hunt through threads again. With `list_all_channels` and `chat_history_messages`, your agent reads every message in scope so you never miss context.

03 Accurate User Tagging: The `list_workspace_users` tool provides full user profiles, including emails and time zones. Your agent can ensure you tag the right person, reducing reply delays.

04 Message Control: Don't just read; act. You can send updates using `chat_post_message`, or use `chat_update_message` to correct announcements without creating confusing follow-up messages.

05 Quick Acknowledgment: Use `chat_add_reaction` to quickly acknowledge a request with an emoji, keeping the main chat flow clean and professional.

Real-World Applications

Onboarding New Department Teams

An Operations Lead asks their agent: 'Create three new channels for Q3 marketing initiatives (A, B, C) and post the standard welcome message in all of them.' The agent uses `create_chat_channel` and then multiple `chat_post_message` calls to provision and populate everything instantly.

Tracking Down a Specific User's Role

A Community Manager needs to know if John Doe is in the EU time zone. They ask their agent, which uses `get_user_info` on John Doe to pull his full profile data and confirm details like email and time zone.

Summarizing a Long Debate

An Engineering Manager wants a quick summary of the last week's discussion. They instruct their agent to run `list_all_channels` to find the right channel, then use `chat_history_messages` to pull the latest content for review.

Managing Outdated Info

A team needs to delete an old announcement that was posted in error. They direct their agent to locate the message using `chat_history_messages`, and then execute `chat_delete_message` to permanently remove it.

Patterns to Avoid

Copying/Pasting Channel Names

✗ AVOID

A user manually copies five channel names from a list, pastes them into an agent prompt, and then has to adjust the format for each one when calling multiple tools.

✓ INSTEAD

Instead of listing channels individually, ask your agent to run `list_all_channels` first. This gives you a complete, clean map of all available communication hubs to reference.

Ignoring User Details

✗ AVOID

A manager sends an email asking a colleague in a different time zone for input without knowing their working hours.

✓ INSTEAD

Run `get_user_info` on that colleague. The tool returns their detailed profile, including their confirmed time zone, so you can schedule the request appropriately.

Assuming Channel Existence

✗ AVOID

A user tries to post an update into a channel they think exists but hasn't been created yet.

✓ INSTEAD

First, run `list_all_channels` to verify the names. If it doesn't exist, use `create_chat_channel` before posting any messages.

The Right Fit

Use this MCP if your primary need is automating actions *within* a single chat platform like Pumble. You want your agent to manage conversations: creating groups, fetching message history, or updating user data all inside the existing communication flow. Don't use it if you need to analyze structured database records (like CRM customer accounts) or process files stored on an external drive; for that, use a dedicated database-querying MCP type tool instead. If your goal is general knowledge retrieval across multiple disconnected systems (e.g., combining Pumble data with Jira tickets and Salesforce leads), you'll need a multi-source orchestration layer, not just this single channel focus.

The pain of managing team comms without an agent

Right now, when your project hits peak activity, you juggle dozens of open tabs. You have to manually check the general channel for status updates, then switch over to a dedicated bug channel for technical details, and finally go into another chat to find out who needs to be notified. This process is all about clicking, copy-pasting usernames, and remembering which thread holds the most recent truth.

With this MCP, your agent handles that entire mess. Instead of jumping between screens, you ask it to read the history from multiple sources—using tools like `list_all_channels` and `chat_history_messages`. You get a single, synthesized answer without lifting a finger. It's direct communication retrieval.

Automate your team’s setup with Pumble MCP

Forget the manual process of creating a new project workspace from scratch. You used to have to remember who needed to be tagged, create the channels one by one, and then copy-paste the same set of onboarding rules into every single group chat.

Now, you just tell your agent what needs doing. It calls `create_chat_channel` for all required hubs and uses `chat_post_message` repeatedly to deploy the standardized message across the board. The setup is instant.

Pumble MCP: 10 Tools for Communication Ops

These tools let you fully control your Pumble workspace from your AI client. You can perform everything from listing users to creating entire channels.

#	TOOL	DESCRIPTION
01	`chat_add_reaction`	Adds a specific emoji reaction to any message in Pumble.
02	`chat_delete_message`	Permanently deletes a specified message from a Pumble channel. This action cannot be undone.
03	`chat_history_messages`	Retrieves and displays recent messages from any given chat channel.
04	`chat_post_message`	Sends a new message to a specified Pumble channel using provided text and channel ID.
05	`chat_update_message`	Modifies the content of an existing, pre-written message within a chat thread.
06	`create_chat_channel`	Creates and provisions a new communication channel; you can specify its name and whether it should be private.
07	`get_channel_info`	Retrieves comprehensive, detailed information about one specific Pumble channel.
08	`get_user_info`	Pulls full profile details for a single user in the workspace, including their email and time zone.
09	`list_all_channels`	Lists every public and private channel that exists across your entire Pumble workspace.
10	`list_workspace_users`	Retrieves a complete directory listing of all users connected to the Pumble workspace.

See It in Action

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

U List all our active channels in Pumble.

I found 5 active channels in your workspace. The main ones are `#general`, `#dev-updates`, and `#marketing-q4`. Which one should I inspect?

U Post a message in the #dev-updates channel stating that 'Deployment 2.1 is completed'.

Done! I've securely posted 'Deployment 2.1 is completed' to the `#dev-updates` channel.

U Read the last 3 messages from #marketing-q4 and react to the last one with a 'thumbsup'.

Here's what was discussed in #marketing-q4:
1. User A: 'Are the assets ready?'
2. User B: 'Yes, loaded in drive.'
3. User A: 'Perfect, deploying campaigns.'
I also successfully placed a 👍 reaction on User A's last message.

Frequently Asked Questions

01 **How do I securely obtain my API Key?**

Installation in Pumble is unique. In your workspace, click **+ Add apps** in the left sidebar, find the **API** addon, and install it. After doing so, go to any chat input box in Pumble and type `/api-keys generate`. A private, ephemeral message will appear containing your secret token. Paste that token here.

02 **Can my AI automatically reply to unread issues?**

Yes. You can instruct your agent to regularly fetch messages from a specific channel (`chat_history_messages`). The agent can evaluate questions, generate answers based on your knowledge base, and then use `chat_post_message` to post the solution back to the team instantly.

03 **Is it possible for the AI to react with emojis?**

Absolutely. It uses the `chat_add_reaction` functionality. A common use case is having the agent process a batch of tickets reported in Pumble, and instruct it to leave a '✅' checkmark reaction on the original message to signify that work is complete.

04 **What happens if a bot message has a typo?**

Unlike standard webhooks, this integration provides full bidirectional control. If your AI posted something incorrect, simply ask it to update its previous message. It will fetch its message ID and push a new payload using the `chat_update_message` tool instantly without leaving double messages.

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 `"pumble": { "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

Pumble 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 Pumble. 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	Pumble MCP
Server ID	`019d75fa-d3f1-7362-85be-18d0392afb64`
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/pumble.