MCP SERVER | NO CODE | CLOUD HOSTED

Socket.dev (Dependency Security) MCP

Audit your entire software supply chain instantly.

Socket.dev (Dependency Security) immediately scans your open-source packages to hunt down vulnerabilities in your software supply chain. Your agent checks package security scores, analyzes manifest files like `package.json`, and monitors real-time threat feeds for malicious dependencies before you ever run an install command.

A+ Quality Score 100/100

supply-chain-security | dependency-scanning | open-source-security | malware-detection | devsecops | package-analysis

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/socketdev-dependency-security

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.

Socket.dev (Dependency Security) MCP

10 tools available
Cloud-hosted on Vinkius

When developing software, the biggest risk often isn't the code you write; it's the packages you download. This MCP connects your AI agent directly to Socket.dev's security platform, letting you proactively defend against supply chain attacks. Instead of treating dependency checking as a manual, multi-step process that slows down sprints, you pass your manifest files—whether they're for npm, PyPI, or Go—and get an instant audit report. Your agent can check specific packages for known issues or grab the overall security score in seconds. If anything looks suspicious, it flags it immediately and provides details on why it's risky. By connecting through Vinkius, you give your AI client access to this deep layer of security intelligence, allowing you to catch typosquatting and backdoors right inside your chat window or IDE. You stop guessing if a package is safe; you just know.

Core Capabilities

01 — Scan code dependencies

Upload manifest files like `requirements.txt` or `package.json` to create a full security scan of your project.

02 — Check package safety scores

Instantly retrieve the detailed security score and issue alerts for any specific open-source package using its name.

03 — Access real-time threat intel

Pull a live feed listing packages that Socket's engine has recently flagged as malicious or dangerous.

04 — Review and manage reports

List, retrieve, and organize historical security reports for your entire organization.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/socketdev-dependency-security** — connect your AI agent in three steps.

01 Subscribe to this MCP and input your personal Socket.dev API token.

02 Direct your AI agent to use the dependency scanning tools, providing it with the manifest file data (e.g., package names or a full `package.json`).

03 Your agent runs the scan and returns detailed security reports, showing you which packages are vulnerable or if they have high-risk issues.

The bottom line is that your AI client treats dependency security like another searchable function in the conversation, eliminating manual CLI steps entirely.

Built For

Security Engineers and DevOps teams who are tired of waiting for vulnerability reports or manually checking every new package before merging code. If you're worried about a supply chain attack from an obscure dependency, this is your connector.

Security Engineer

Automating the review process for all incoming dependencies and monitoring organizational security posture via list_organizations.

Software Developer

Getting instant feedback on package safety scores or potential issues when they are adding a new library to their local project.

DevOps Engineer

Integrating comprehensive scans into development workflows, allowing them to quickly triage and manage security reports before deployment.

What Changes When You Connect

01 Stop worrying about obscure dependencies. By checking package safety scores, you get a single number that tells you how secure a component is—no guesswork required.

02 Keep your codebase clean by using the `create_scan` tool to upload full manifest files. This provides a comprehensive security audit for every dependency in one go.

03 Stay ahead of bad actors. The dedicated `get_threat_feed` tool gives you real-time alerts on malicious packages, letting you block them before they hit production.

04 Manage compliance effortlessly. Use the report listing tools (`list_reports` , `get_report`) to keep a centralized record of your security posture across multiple projects and organizations.

05 Eliminate manual research time. Instead of searching documentation for known issues, simply ask your agent to run `get_package_issues` on any package name.

Real-World Applications

Auditing a new microservice dependency

A developer needs to add a logging library. Instead of running `npm install` and hoping for the best, they tell their agent to use `get_package_score` on the library's name. The agent instantly returns an A+ score with no critical issues reported, confirming safety before the first line of code is written.

Triage after a major security bulletin

A DevOps team receives a warning about a common vulnerability. They instruct their agent to use `create_scan` on all existing project manifest files, creating multiple scans to identify which internal services are affected and what needs immediate patching.

Checking organizational compliance

A security engineer must ensure that every team meets a minimum dependency safety standard. They use `list_organizations` first, then run targeted scans across all departments to generate a unified report for leadership review.

Reacting to zero-day threats

During an active threat window, the team needs immediate intelligence. The agent runs `get_threat_feed` and immediately flags several packages that have been recently flagged with malware, allowing the team to pull them from deployment lists instantly.

Patterns to Avoid

Checking one package at a time

✗ AVOID

The user manually asks the agent to check 'axios', then asks again for 'lodash', and so on. This is slow, tedious, and fails to capture cross-dependency risks.

✓ INSTEAD

Use `create_scan` by uploading the entire manifest file (like package.json). This single action scans every dependency at once and provides a full risk assessment.

Forgetting historical context

✗ AVOID

The user only checks the current score for a package but has no record of past issues or compliance failures.

✓ INSTEAD

Use `list_reports` first. This retrieves all previous scan records, allowing you to track security improvements and flag if an issue was previously present.

Ignoring the source of risk

✗ AVOID

The user only focuses on known CVEs but ignores newly published malicious packages.

✓ INSTEAD

Periodically run `get_threat_feed`. This ensures you are seeing the most current, real-time alerts about novel malware that haven't been cataloged yet.

The Right Fit

Use this MCP if your primary concern is proactively identifying vulnerabilities hiding deep within third-party open-source dependencies. You need to know what packages you are using and why they might be unsafe. Don't use it if you just need to validate that a package *exists* or check simple API rate limits; those general utility tools will suffice.

If your problem is limited to reviewing the status of an already scheduled scan, then `get_scan` is all you need. But if you need deep analysis—the full picture of scores, issues, and threat intelligence across multiple projects—this MCP handles it. If you are only worried about organizational structure and not package content, stick to pure identity management tools instead.

Dependency auditing used to be a tedious, multi-stage process.

Today, adding a single library means checking its documentation for known vulnerabilities. Then you open a separate dashboard to check the overall security score. You copy package names into one place, and then paste them into another tool just to get a list of historical reports. It’s slow, it requires jumping between three different tabs, and frankly, it's exhausting.

With this MCP, your agent handles all those steps in one conversation. Instead of multiple manual checks, you simply provide the manifest file. The agent runs the full scan, pulls together the score, flags the issues, and hands you a clear report right where you are working.

Get comprehensive dependency security with Socket.dev (Dependency Security).

You no longer have to wait for a dedicated security team to run the full analysis. You can trigger a complete audit, check the real-time threat feed, and get both an immediate score and a historical report—all in one go.

The process moves from 'I hope this package is safe' to 'I know exactly why it's safe.' This MCP puts enterprise-grade supply chain security right into your daily agent workflow.

Socket.dev (Dependency Security): 10 Tools

Use these tools with your agent to run full dependency scans, retrieve security reports, and access real-time threat intelligence for your codebase.

#	TOOL	DESCRIPTION
01	`create_scan`	Uploads manifest files (like package.json) to start a new project-wide dependency scan.
02	`delete_scan`	Removes an existing, unnecessary security scan record.
03	`get_package_issues`	Checks for known vulnerabilities and specific issues associated with a single package name.
04	`get_package_score`	Retrieves the overall security score (e.g., 98/100) for a specified software package.
05	`get_quota`	Shows how many API scan requests are left for the current billing period.
06	`get_report`	Fetches all the detailed results and findings from a specific, completed security report.
07	`get_scan`	Retrieves metadata about a scan to check its current status (running or finished).
08	`get_threat_feed`	Accesses the real-time feed of packages flagged by Socket's analysis engine as malicious.
09	`list_organizations`	Lists all different organizations that the provided API token has permission to access and audit.
10	`list_reports`	Retrieves a list of historical security reports, providing IDs needed for deeper inspection.

See It in Action

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

U Check the security score for the npm package 'axios'.

I've checked the security score for `pkg:npm/axios`. It currently has a high security score of 98/100, with no critical issues detected in the latest version.

U List all security reports for my organization.

I've retrieved the reports. You have 3 recent scans available. Would you like me to get the details for the most recent report (ID: rep_12345)?

U Show me the real-time threat feed from Socket.

Fetching the threat feed... I found several recently flagged malicious packages, including 'discord-selfbot-v14' which was flagged for containing malware. Be cautious if these appear in your dependencies.

Frequently Asked Questions

01 **How do I check the overall safety score using Socket.dev (Dependency Security)?**

You use `get_package_score` and provide the full package identifier, like `pkg:npm/react`. The tool returns a simple numerical score that tells you how healthy the dependency is right now.

02 **Can Socket.dev (Dependency Security) scan multiple manifest files at once?**

Yes. You first use `create_scan` and upload all necessary manifest data, allowing a single job to audit dependencies from various sources like package.json and requirements.txt.

03 **What is the difference between running `get_package_issues` and `get_report`?**

`get_package_issues` gives you specific, immediate alerts for one package. `get_report` provides a comprehensive summary of all findings from an entire scan run.

04 **Do I need to worry about my API usage quota with Socket.dev (Dependency Security)?**

No problem. You can use the `get_quota` tool anytime your agent needs it, which simply tells you how many scan requests are remaining for your account.

05 **Does this MCP help me find brand new malware?**

Yes. The dedicated `get_threat_feed` accesses Socket's real-time intelligence feed, alerting you to packages recently flagged by the community or security experts as malicious.

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 `"socketdev-dependency-security": { "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

Socket.dev (Dependency Security) 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 Socket.dev (Dependency Security). 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	Socket.dev (Dependency Security) MCP
Server ID	`019e38f0-6f7a-708b-b696-97b467e1907e`
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/socketdev-dependency-security.