MCP SERVER | NO CODE | CLOUD HOSTED

Security Hacker MCP

Audit open-source code and hunt supply chain threats instantly.

Google Deps.dev Security Hacker turns your AI client into a specialized DevSecOps auditor. It instantly scans open-source packages and full GitHub repositories for deep supply-chain vulnerabilities, known CVEs, and governance gaps across npm, PyPI, Maven, and more. You get to hunt down hidden threats that basic scanners miss.

F Quality Score 3.6/100

supply-chain-security | dependency-analysis | vulnerability-scanning | open-source-security | devsecops

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/google-depsdev-security-hacker

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 Deps.dev Security Hacker MCP

4 tools available
Cloud-hosted on Vinkius

This MCP connects your agent directly to Google's Open Source Insights (deps.dev). It lets you perform serious security audits on any open-source code, making your AI client a true DevSecOps auditor. Instead of running multiple command-line tools or cross-referencing documentation pages for vulnerabilities, your agent handles the entire process in chat.

Need to check if an old version of Express is safe? Just ask. Need to know every single dependency that package relies on, including the ones you never knew existed? The agent maps the whole tree. You can even paste a GitHub URL and get a full governance score based on industry best practices. Vinkius hosts this MCP so your AI client can access all of these checks from one place. It's what developers actually need when they're worried about supply chain attacks.

Core Capabilities

01 — Audit single dependencies

Check any open-source package across major ecosystems (npm, PyPI, etc.) for known security flaws and adherence to governance standards.

02 — Map hidden supply chain risks

Trace the full dependency tree of a package to find indirect or deep-level vulnerabilities that aren't immediately obvious.

03 — Scan repository health

Run comprehensive security and governance audits on an entire GitHub repository, checking for best practices like code reviews and fuzzing.

04 — Look up vulnerability details

Retrieve specific technical information about a given CVE or GitHub Security Advisory ID so you know exactly what's compromised.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/google-depsdev-security-hacker** — connect your AI agent in three steps.

01 Tell your agent which package, repository, or vulnerability ID you need to check. For example: 'Audit the PyPI package requests.'

02 The MCP connects to Google's deps.dev API and executes the necessary security scan (e.g., mapping dependencies or checking governance).

03 You get a plain-language report detailing known CVEs, high-risk components, OSSF scores, and immediate upgrade recommendations.

The bottom line is you get professional-grade supply chain security analysis without leaving your chat window.

Built For

The DevSecOps Engineer who gets burned checking dependency chains manually. The Security Architect worried about third-party risk. The Senior Developer who needs to quickly vet a new open-source library before committing code.

DevSecOps Engineer

Uses the MCP daily to audit dependencies and repositories, ensuring that every new feature passes security checks without needing dedicated scanning tools.

Security Architect

Runs full governance audits on client-provided GitHub URLs using OSSF Scorecards to vet the overall health of a project's development process.

Software Developer

Needs to quickly check if an older, common package version like Log4j is still safe or if they need to upgrade immediately before running tests.

What Changes When You Connect

01 Never trust a dependency blindly. Use the `get_transitive_dependencies` tool to map every single indirect component, finding hidden malware or unpatched flaws deep in the stack.

02 Cut down on manual research time. Instead of checking multiple documentation sites for CVEs, let your agent use `analyze_dependency` to instantly flag known security advisories across npm, PyPI, and more.

03 Gauge project maturity automatically. Paste a GitHub URL and run `analyze_github_repository`. You get an OSSF Scorecard that tells you if the project even follows basic security best practices.

04 Understand exactly what's broken. If you find a weird CVE ID, use `get_vulnerability_details` to pull down the exact exploit mechanism, severity, and affected versions for immediate patching.

05 Support every major language stack. This MCP works natively across npm (Node.js), PyPI (Python), Maven (Java), Cargo (Rust), and more, so you never have to switch tools.

Real-World Applications

Vetting a new third-party library

A developer finds a promising open-source library but isn't sure if it's safe. They ask their agent to run `analyze_dependency` on the package name, immediately getting an OSSF score and a list of known CVEs without installing anything locally.

Investigating supply chain risks

A security engineer suspects a core service has been compromised by a hidden dependency. They use `get_transitive_dependencies` to map the full tree, spotting an obscure, unpatched component that needs overriding.

Assessing team code quality

A manager wants to know if their internal teams are following security best practices. They input a GitHub URL and run `analyze_github_repository`, getting actionable feedback on branch protection or code review enforcement.

Responding to a critical zero-day alert

A team gets an alert about an old vulnerability ID (like Log4Shell). They use `get_vulnerability_details` with the specific CVE ID, getting immediate confirmation of severity and affected package versions.

Patterns to Avoid

Checking only top-level dependencies

✗ AVOID

A developer runs a basic audit and sees that their main dependency is clean. They assume the project is safe because they didn't look deeper.

✓ INSTEAD

Always run `get_transitive_dependencies`. This ensures your agent maps out the entire tree, catching hidden risks from indirect components you never even knew existed.

Relying on manual documentation searches

✗ AVOID

When a vulnerability pops up, the developer spends hours cross-referencing multiple CVE databases and GitHub advisories to understand its impact.

✓ INSTEAD

Use `get_vulnerability_details`. Give your agent the GHSA or CVE ID. It pulls down all necessary details—impact, affected versions, severity—in one shot.

Ignoring project governance

✗ AVOID

A team adopts a library because it's popular, but the repository has no enforced code reviews or version signing.

✓ INSTEAD

Run `analyze_github_repository`. The OSSF Scorecard immediately flags poor governance practices like missing fuzzing or lack of branch protection.

The Right Fit

Use this MCP if your primary concern is third-party risk, supply chain integrity, or knowing the security posture of open source codebases. You need to audit *what* a project uses (dependencies) and *how* it's built (governance). If you only care about general functionality—like figuring out how to call an external API endpoint—this tool is overkill. For simple task automation, use a dedicated messaging or data management MCP instead. But if the code quality or dependency risk is your blocker, this Hacker toolkit is mandatory.

The headache of auditing open source libraries.

Today, checking one library means opening GitHub, finding its release history, cross-referencing a dozen vulnerability databases, and then running local dependency audits just to find the transitive dependencies. It's a massive copy/paste job that takes hours and relies on you remembering which tool checked what.

With this MCP, you simply ask your agent to audit the package or repository. It handles the complexity of tracking every single layer—from the top-level dependency all the way down into its deepest dependencies—and delivers a clear report in seconds.

Get full visibility with Google Deps.dev Security Hacker

You stop manually gathering scores and reports from disparate sources. You never have to worry about forgetting which specific package version caused a vulnerability; the MCP checks that context for you.

The difference is moving from reactive, painful investigation to proactive, instant assurance. Your AI agent gives you expert-level security analysis on demand.

Google Deps.dev Security Hacker MCP with 4 Tools

These tools let your AI client perform specialized security checks on dependencies, entire repositories, and specific vulnerabilities across multiple programming languages.

#	TOOL	DESCRIPTION
01	`analyze_dependency`	Checks a specific open-source package across multiple systems (npm, pypi, etc.) for security flaws and governance scores.
02	`analyze_github_repository`	Runs a full audit of a GitHub repository to assess its overall development security practices using OSSF Scorecards.
03	`get_transitive_dependencies`	Maps the complete dependency tree for a package, finding hidden or indirect components that pose security risks.
04	`get_vulnerability_details`	Retrieves specific technical details about any known vulnerability using its CVE or GHSA identifier.

See It in Action

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

U Audit the npm package 'express' version 4.16.0 for any security vulnerabilities.

I checked `express` v4.16.0 on deps.dev. It has **4 known security advisories**, including GHSA-8X4W-MXXF-XXXX (High Severity - CVSS 7.5). The project has an OSSF Scorecard of 6.8/10. I strongly recommend upgrading.

U I found this GHSA-xvch-5gv4-984h advisory. What does it do and what packages are affected?

That ID corresponds to a **CRITICAL (CVSS: 9.8)** vulnerability: "Prototype Pollution in Lodash". It allows an attacker to inject properties into Object.prototype. It affects the npm package `lodash` on versions `< 4.17.15`. You must upgrade immediately.

U Check the dependency tree for the pypi package 'requests' v2.28.1. Are there any hidden threats?

I mapped the entire transitive tree (total 8 dependencies). ⚠️ **WARNING: Found 1 hidden threat:** The indirect dependency `certifi` (v2022.6.15) has a known security advisory. You should override this dependency in your requirements.

U Generate a security governance audit for https://github.com/facebook/react

Here is the OSSF Scorecard audit for `facebook/react`. It scores a **7.2/10** overall. They excel in Code-Review (10/10) and Branch-Protection (10/10), but lost points on Pinned-Dependencies (3/10) and lack of Fuzzing (0/10).

U Is the 'log4j' maven package safe to use right now?

Let me check the latest versions on deps.dev. The older 2.14.x versions are famously compromised (Log4Shell - CVE-2021-44228, CVSS 10.0). However, the latest version `2.20.0` has ✅ **No known security advisories** and is safe to use. Always ensure you are pinning the latest version.

Frequently Asked Questions

01 **How does Google Deps.dev Security Hacker check dependencies?**

It connects directly to the deps.dev API and supports major package managers like npm, PyPI, Cargo, Maven, and NuGet for comprehensive coverage.

02 **Can I use analyze_github_repository with private repos?**

The MCP requires a publicly accessible GitHub URL to run the OSSF Scorecard audit. It analyzes public governance practices only.

03 **Does get_transitive_dependencies find everything?**

It maps and scans the entire dependency graph, finding indirect components that could contain hidden security threats or unpatched vulnerabilities.

04 **What is an OSSF Scorecard?**

The OSSF Scorecard is a metric used to rate how well a repository enforces development best practices like code review and branch protection.

05 **Is this better than running local vulnerability scanners?**

Yes. While local tools are good, the Security Hacker MCP provides immediate, centralized analysis across multiple ecosystems without needing to install anything or manage complex environments yourself.

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-depsdev-security-hacker": { "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 Deps.dev Security Hacker 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

DOCUMENT INFORMATION

Generated	June 2026
MCP Server	Google Deps.dev Security Hacker MCP
Server ID	`019eb8c8-e09f-70bc-aaad-612d4d684aad`
Platform	Vinkius Cloud for AI Agents
Endpoint	`https://edge.vinkius.com/{token}/mcp`

LICENSE & USAGE