MCP SERVER | NO CODE | CLOUD HOSTED

Checkmarx MCP for AI Agents

Programmatic Application Security Analysis and Code Flaw Detection

Checkmarx lets you manage your application security posture directly through natural language commands. Trigger scans on codebases, analyze complex infrastructure flaws (KICS), pinpoint exact lines of vulnerable code, and calculate the optimal fix location—all without leaving your current chat window.

F Quality Score 3.6/100

appsec | sast | sca | code-scanning | cybersecurity | devsecops

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

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.

Checkmarx MCP

10 tools available
Cloud-hosted on Vinkius

Security scanning used to be a dashboard nightmare. You'd spend hours toggling between reports, manually cross-referencing vulnerability severity with specific files, just to figure out where to patch things. This MCP changes that. Instead of navigating complex cyber dashboards, you talk to your agent and it handles the heavy lifting for Checkmarx One.

Need to check if a new deployment breaks security standards? You can ask it to trigger scans across specific projects or even list all containers in an application group. It'll give you status updates and results so you know exactly where you stand. If you're worried about misconfigurations in your IaC, the agent pulls specialized metrics from Terraform, Dockerfiles, and Kubernetes YAML. The best part is that when it finds a flaw, it doesn't just tell you *that* there's a bug; it calculates the precise spot in your code where the patch needs to go. If this sounds too powerful for one tool, remember that Vinkius hosts thousands of MCPs, giving your agent access to every system you use.

Core Capabilities

01 — Scan Codebases and Projects

Get metadata listing all available Checkmarx projects or trigger a new SAST scan on your current codebase.

02 — Analyze Infrastructure-as-Code (KICS)

Focus solely on identifying misconfigurations within specific IaC files like Terraform, Kubernetes YAML, and Dockerfiles.

03 — Retrieve Vulnerability Details

Fetch detailed reports containing vulnerability severity, status, and the exact line of code where a flaw was detected.

04 — Pinpoint Fix Locations (BFL)

Calculate the mathematically optimal spot in your application's execution path to apply a patch that fully resolves a specific security vulnerability.

05 — Manage Scan Status

Check the current status, configuration, and timing of any running or historical Checkmarx scan.

One Click on Vinkius — From Prompt to Execution

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

01 First, connect your AI client using a JWT token to authenticate with your enterprise Checkmarx One environment.

02 Next, ask the agent to perform a specific security action, like running a scan on a project or listing applications. The agent executes the necessary API call and retrieves raw data.

03 Finally, you prompt the agent again, telling it what insight you need—for instance, 'What's the best fix location for this XSS vulnerability?' The MCP processes the data and delivers the actionable answer.

The bottom line is that your AI client becomes a natural interface to complex security infrastructure, turning technical dashboards into simple conversation prompts.

Built For

This MCP is for Security Engineers and DevOps teams who are tired of context switching between IDEs, ticketing systems, and multiple web dashboards. It lets you manage AppSec from one chat window, letting developers focus on code, not clicks.

Security Engineer (AppSec)

You use this MCP to orchestrate vulnerability triage by asking the agent to pull core datasets of severe flaws and then calculating the best patch location without ever leaving your primary workstation.

DevOps/Platform Team

You manage deployment risks by using the MCP to run scans on staging branches, checking for misconfigured KICS results before a merge, or canceling redundant running jobs.

Software Developer

You grab an exact Best Fix Location (BFL) from the agent and ask it to rewrite sanitized logic instantly, speeding up zero-day remediation without deep manual research.

What Changes When You Connect

01 Stop manual vulnerability triage. Instead of opening dozens of reports, you simply ask the agent to analyze core datasets of severe flaws and pinpoint them automatically.

02 Eliminate context switching. You manage everything—from listing applications with `list_applications` to checking specific project details with `get_project`—without ever leaving your chat interface.

03 Get surgical remediation advice. The Best Fix Location (BFL) tool calculates the exact optimal spot in your code for a patch, saving hours of guesswork for developers.

04 Master IaC security checks. Use `get_kics_results` to focus only on misconfigurations inside Terraform or Kubernetes YAML files, ignoring standard source code flaws when necessary.

05 Control your pipeline flow. You can trigger new scans with `run_scan`, check the status with `list_scans`, and even cancel redundant jobs using `cancel_scan`—all via natural language.

Real-World Applications

Reviewing a Merged Pull Request

A developer asks their agent to run a scan on the current project branch and, upon completion, immediately list all critical vulnerabilities. The agent uses `run_scan` followed by `get_scan_results`, summarizing the top 5 issues right in the chat for rapid sign-off.

Auditing Cloud Infrastructure Setup

A platform engineer needs to verify a new Kubernetes deployment. They ask the agent to check the specialized IaC metrics, and the MCP uses `get_kics_results` to isolate misconfigurations in the YAML before they hit production.

Finding the Quickest Code Patch

A security engineer identifies an old XSS vulnerability. Instead of manually tracing the flaw, they ask the agent for the Best Fix Location (BFL). The MCP uses `list_bfl` and returns the exact line number and function call to fix it.

Checking Application Coverage

A manager needs a full view of all microservices under one product umbrella. They ask for an overview, and the agent uses `list_applications` to provide a risk summary across the entire logical product line.

Patterns to Avoid

Treating Security as a Manual Audit

✗ AVOID

Manually logging into Checkmarx, clicking through 'Projects', then running scans one by one, and finally downloading CSV reports to analyze in Excel.

✓ INSTEAD

Start with `list_projects` to see your codebase inventory. Then use the agent to trigger a scan via `run_scan`. Finally, ask it to download results using `get_scan_results`—all without leaving your chat.

Ignoring IaC Flaws

✗ AVOID

Assuming that because the core code passed SAST checks, the surrounding infrastructure (like Dockerfiles) is safe.

✓ INSTEAD

Always check for cloud-level misconfigurations. Use `get_kics_results` to run specialized scans against your Infrastructure as Code before deployment.

Forgetting Scan Status

✗ AVOID

Running a scan and then forgetting if it finished, or not knowing which specific engine (SAST vs SCA) generated the latest results.

✓ INSTEAD

Use `list_scans` to track all historical jobs. If you need details on *how* the job ran, check the configuration using `get_scan_details`.

The Right Fit

You should use this MCP if your security process requires programmatic analysis of code and infrastructure flaws. Use it when you need to calculate precise patch locations (BFL) or when you are managing multiple, interconnected projects under a single application umbrella. Don't use it if all you need is simple compliance reporting; for that, a dedicated dashboard tool might be faster. If your problem is simply listing user accounts or basic

network inventory, this MCP won't help—you'll need an endpoint focused on identity management.

Checkmarx: Automating AppSec Vulnerability Triage

Today, finding deep code flaws involves a painful cycle of clicking through dashboards. You copy vulnerability IDs from one report, paste them into another tool to check the status, and then manually cross-reference the affected lines of code. It's slow, it’s error-prone, and it breaks your focus.

With this MCP, you describe the problem to your agent. Instead of manual copy-pasting, you simply ask for the optimal fix location (BFL). The tool analyzes the complex data flow and gives you a single, precise answer—the exact line number where the patch must go.

Checkmarx: Managing Infrastructure as Code Security

The biggest manual gap is checking cloud infrastructure. You have to switch tools just to verify if your Terraform or Dockerfile has an exposed port or a misconfigured secret, which are completely separate from the application code itself.

Now, you can ask for specialized IaC metrics directly through this MCP. It pulls findings specifically from Kubernetes YAML and CloudFormation, allowing you to audit your entire deployment stack without switching out of your workflow.

Checkmarx: 10 Tools for AppSec Scanning and Code Flaw Management

These tools let your agent manage the entire security lifecycle—from listing all projects to running scans and calculating precise fix locations.

#	TOOL	DESCRIPTION
01	`cancel_scan`	Stops an active Checkmarx scan job immediately, preventing unnecessary resource usage if the code context changes.
02	`get_project`	Retrieves specific metadata for a designated Checkmarx project to ensure you are working with the correct codebase context.
03	`get_kics_results`	Pulls specialized findings that focus only on misconfigurations within Infrastructure as Code (IaC) files like Dockerfiles and Kubernetes YAML.
04	`list_applications`	Lists all defined Checkmarx One Applications, providing visibility into aggregated security risk across multiple microservices or projects.
05	`list_bfl`	Calculates and returns the Best Fix Location (BFL) by referencing a specific vulnerability rule ID and scan ID.
06	`list_projects`	Provides an inventory of all available Checkmarx One Projects, along with their metadata and linked applications.
07	`list_scans`	Lists all historical or active scans for a project, showing the status, targeted branch, and timestamps to help you track job history.
08	`run_scan`	Triggers a new Checkmarx One code scan, commonly used in CI/CD pipelines to enforce security quality on pull requests.
09	`get_scan_details`	Checks the precise status and configuration of a specific scan, detailing which engines (SAST, SCA, KICS) were run and their timing.
10	`get_scan_results`	Downloads SAST findings for a completed scan, providing vulnerability severity, state, description, and the exact lines of code affected.

See It in Action

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

U What are the most critical vulnerabilities found in my API service?

Security Findings Summary

SEVERITY	COUNT	CWE CODE
Critical	2	CWE-89 (SQL Injection)
High	3	CWE-79 (XSS)

Details: The two critical flaws are in `/api/users.js` at line 142 and `/auth/login.ts` at line 87. Use the BFL tool to find the perfect patch for each.

U Check my new deployment files for security issues.

KICS Report: Infrastructure Flaws

Found 1 misconfiguration in `docker-compose.yml` :

- **Issue:** Exposed internal database port (Port 5432).
- **Line:** L7.
- **Recommendation:** Change the exposed network ports to private and use a service mesh for communication. This prevents external access.

U I'm worried about old vulnerabilities in the project.

Project: WebPortal v2

Scan Status: Completed (ID #4521)
- **Total Vulnerabilities:** 12
- **Breakdown:** Critical: 2, High: 3, Medium: 7.

I recommend starting with the two critical SQL Injections. I can run a BFL analysis to give you the exact fix path for those immediately.

Frequently Asked Questions

01 **How does Checkmarx MCP help me manage my application security findings?**

It lets your AI agent analyze complex vulnerability reports through natural conversation. Instead of manually navigating dashboards, you ask it to pull core datasets of flaws and get immediate summaries.

02 **Can I use Checkmarx MCP for cloud infrastructure checks?**

Yes, the MCP includes tools that focus specifically on Infrastructure as Code (IaC). It reads specialized metrics from Terraform, Kubernetes YAML, and Dockerfiles to find misconfigurations.

03 **What if I need a specific patch location for code flaw?**

You can ask the agent to calculate the Best Fix Location (BFL) for any vulnerability. It tells you the exact optimal spot in your application's code where the patch needs to be applied.

04 **Does Checkmarx MCP help me with continuous integration?**

Absolutely. You can use it to trigger new scans automatically when a pull request is opened, ensuring that security quality is checked continuously throughout your CI/CD pipeline.

05 **What kind of projects can I list and analyze with this MCP?**

The tool lets you inventory all available Checkmarx Projects and Applications. This gives you a complete overview, allowing you to check security metrics across multiple related microservices or products.

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 `"checkmarx": { "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

Checkmarx 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	Checkmarx MCP
Server ID	`019d756e-34c4-7303-b2e4-d79b36281968`
Platform	Vinkius Cloud for AI Agents
Endpoint	`https://edge.vinkius.com/{token}/mcp`

LICENSE & USAGE