Back to blog

How Real Teams Are Powering AI Security with the Contrast MCP Server

By Jacob Mages-Haskins, Principal Software Engineer
October 30, 2025

When we introduced the Contrast Model-Context Protocol (MCP) Server a few months ago (read Supercharge your vulnerability remediation with Contrast MCP), the promise was clear: to give AI agents access to the rich security context within the Contrast Graph, which is a real-time application security data model that continuously maps, updates and correlates security insights across applications, APIs and infrastructure.

assisted secure development.

As a quick refresher, the MCP Server is the crucial bridge that gives your Large Language Model (LLM) agent the tools it needs to securely interact with the Contrast Security API and access the data in your Contrast Graph.

In this post, we'll move past the introduction to highlight real customer use cases, and then provide the full quick-start guide to setting up the Contrast MCP Server in your own environment.

The Contrast MCP Server in Action: Driving Real-World Outcomes

Here are a few high-value use cases our early adopters are implementing:

Use Case 1: AI-Driven Vulnerability Remediation

This is the most impactful use case. When a vulnerability is found, the AI agent doesn't just see a line of insecure code. Because of the MCP Server, the agent can access the full security context from Contrast, including:

The environment and application details.

The AI agent uses this complete picture, not just the code snippet, to generate a precise, context-aware fix or a secure code refactor, drastically reducing the time required for a human to step in and improving the accuracy of the AI coding assistant.

Here are example prompts that you can use with your AI coding assistant:

> What are the most critical vulnerabilities in this application that Contrast knows about?
>
> In Contrast, which routes for this application have critical CVEs?
>
> Fix the most important vulnerability identified by Contrast.

Use Case 2: Automating Security Posture Reports

Instead of manually pulling data from different dashboards, an AI agent, connected via the MCP Server, can quickly query the

Identifying the most critical, reachable vulnerabilities.

Aggregating security metrics across multiple applications.

Translating technical findings into business risk (e.g., building an HTML report with graphs and high-level summaries for a CISO presentation).

This capability automates security intelligence and reporting, freeing up security teams to focus on mitigation.

Example report-building prompt:

As a CISO, I want to present the Contrast data for this project to business people. Build me an HTML report that uses nice graphs and pictures to display a simple, but comprehensive picture of the security position of this application, what would happen if this were running on the public internet, and what are the chances and timeframe of exposure, and what could be lost.

Generate custom reports with the Contrast MCP Server from data in the Contrast Graph.

Contrast MCP Server IDE Setup: The Quick-Start Guide

Inspired by these advanced use cases? Here's how you can set up the Contrast MCP Server in your own environment to start

It is imperative to note here, though, that exposing your Contrast vulnerability data via an MCP Server to an LLM that trains on your data is dangerous. You must use an enterprise-grade, "sandboxed" LLM that guarantees your data is not used for further training. Examples include models targeted to the enterprise market, like those available through AWS Bedrock or those used for GitHub Copilot and Claude Code.

It is your responsibility to thoroughly research and validate your specific LLM provider's data usage and security policy before deploying the Contrast MCP Server.

Setting up Contrast MCP Locally

The first step is setting up the Contrast MCP Server on your local machine. Refer to the documentation at https://github.com/Contrast-Security-OSS/mcp-contrast to find the most appropriate setup method for your IDE or project.

Example: GitHub Copilot in Visual Studio Code

Here is a concrete example for a popular setup. For all others, check the official documentation first!

Install the Contrast MCP Server in Visual Studio Code with Docker from this installation link.

Download the latest JAR file (requires Java 17+): https://github.com/Contrast-Security-OSS/mcp-contrast/releases/latest

2. Configure VS Code

Create a new config file at the root of your project: .vscode/mcp.json

This file tells your setup how to run the Java-based MCP Server and provides the necessary details to connect to the Contrast Security API (like your API keys and host name). Most of the connection details can be found on the User Settings page of the Contrast Security UI.

Last, Enable the Tool in Copilot:

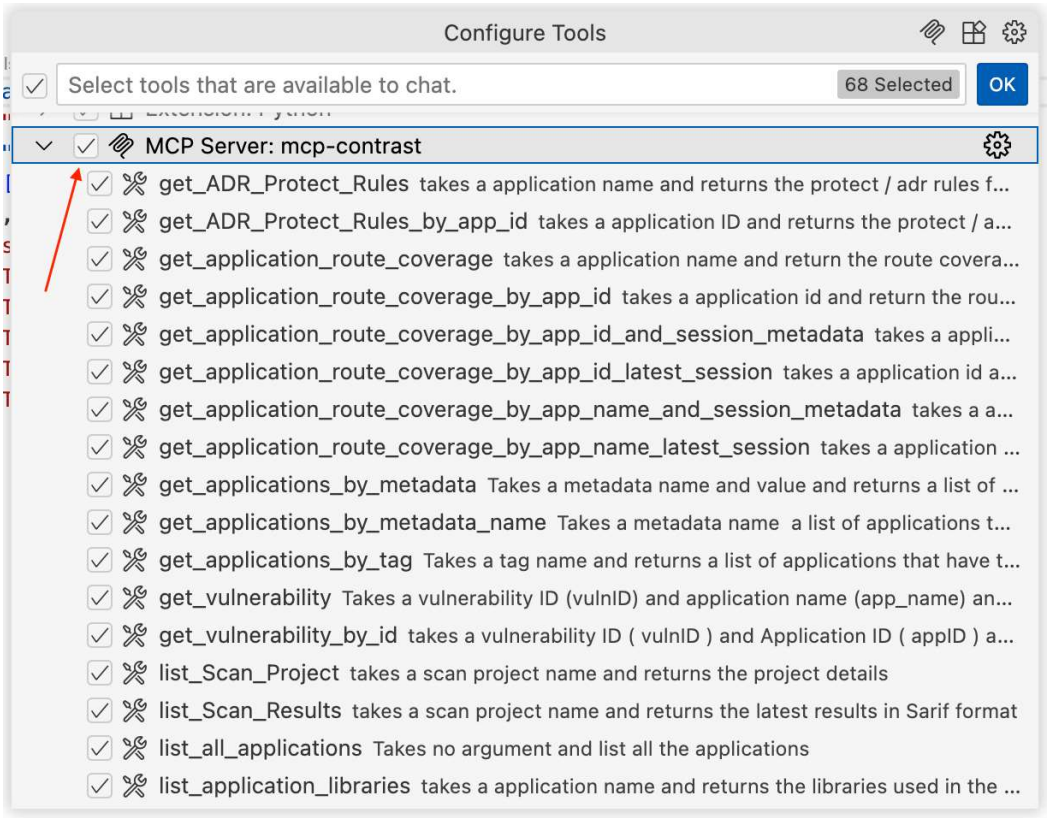
Open the Copilot chat pane.

Put Copilot into "Agent" mode.

Click the tools icon.

In the tool dialog window, enable the Contrast MCP Server tools.

Put Copilot in Agent mode and click the tools icon



Enable the Contrast MCP Server in the tools dialog

Power of the Contrast Graph + AI

Now that the Contrast MCP Server is installed and enabled in your environment, your AI agents can finally use the data in the

and begin turning your security insights into immediate, powerful intelligence for your AI development workflow.

Contrast MCP Server | Model Context Protocol (MCP) | LLM | Large Language Model

Jacob Mages-Haskins, Principal Software Engineer

Jacob Mages-Haskins is a Principal Software Engineer at Contrast Security, where he leads innovation in AI-driven cybersecurity. With decades of experience building software across higher education, HVAC manufacturing, telemedicine, live events, and fintech, Jacob brings a uniquely broad perspective to the next generation of application-layer protection. He originally conceived the Contrast Security Model-Context Protocol (MCP) Server, a flexible platform that

security. Driven by curiosity more than credentials, Jacob is passionate about the intersection of AI, software engineering, and cybersecurity innovation. When he's not pushing the boundaries of secure development, you'll find him reading, gardening, or exploring the rugged coastline and trails of Maine.

NEXT

Contrast and Microsoft Sentinel: Closing the Application-Layer Blind Spot

PREVIOUS

Dynamic Application Security Testing (DAST) Can't Keep Pace with AI-Generated Code: The Runtime Security Imperative

Loving our content?

First Nar Last Nar

Company Email*

Get the latest content from Contrast directly to your mailbox. By subscribing, you will stay up to date with all the latest and greatest from Contrast.

HQ Country*

We take your privacy seriously at Contrast; security is what we're all about in the first place! We use the information you provide to us on the basis of legitimate interest to make sure you get more information about the topics that may be of interest to you. Contrast also partners with third parties from time to time and may share your contact information with them. By submitting this form, you agree to our collection and use of your information in accordance with our Privacy Policy. You may opt out at any time here.

Submit

Contrast Application and API Security Testing (AST)

Contrast One™

Contrast Assess (IAST)

Contrast Software Composition Analysis (SCA)

Contrast Scan (SAST)

By team

SecOps

AppSec

Developer

CISO

By industry

Technology

Financial services

Healthcare

Insurance

Become a partner

Find a partner

Visit partner portal

Glossary

©Contrast Security 2025

Privacy Matters

Terms of Service

SECURITY