The Next Security Frontier: Understanding and Mitigating Risks in the MCP Ecosystem

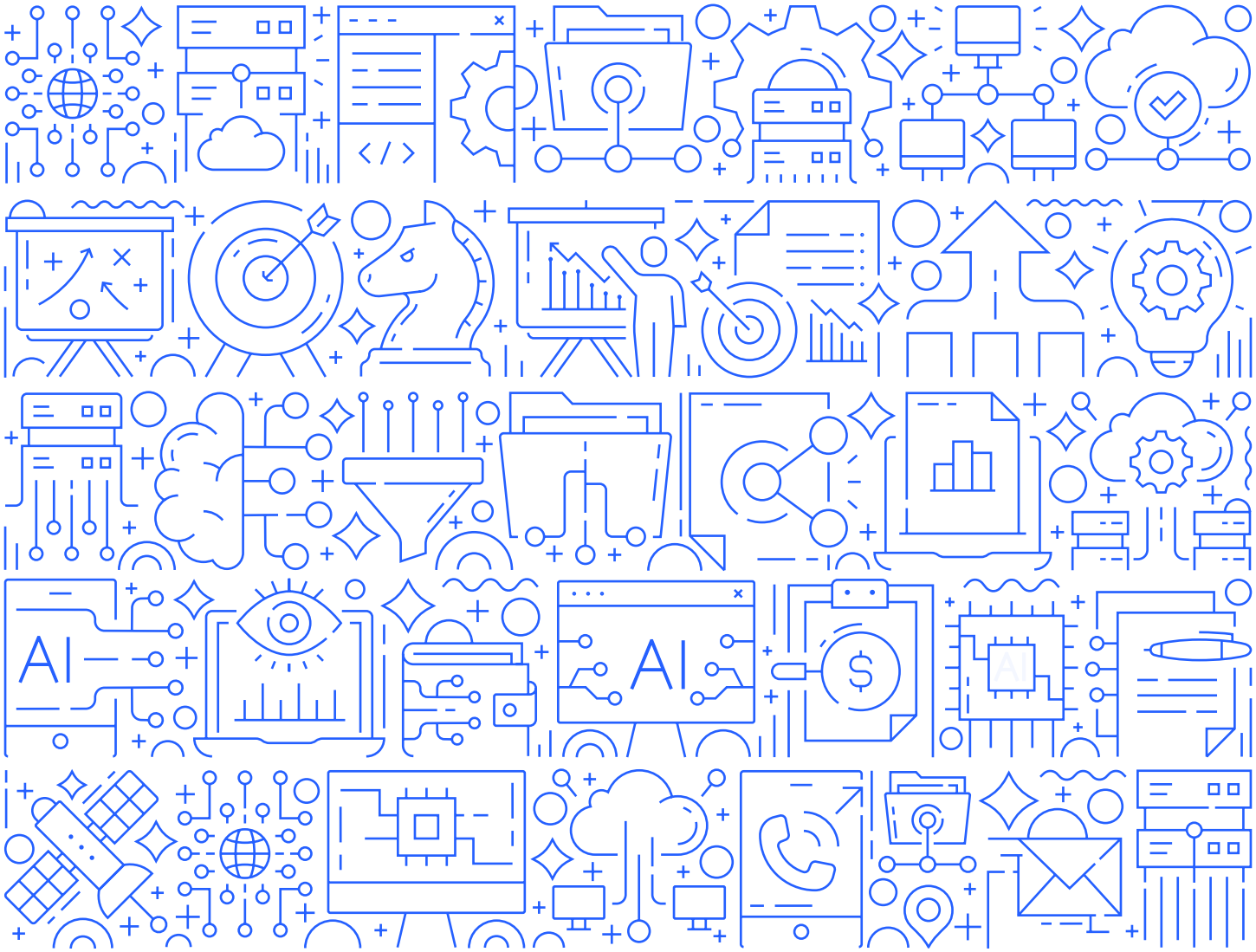
By : Yiwen Xu

Executive Summary

The **Model Context Protocol (MCP)** is quickly becoming a core standard in the AI ecosystem. Originally built to simplify the interaction between agents and external tools, MCP is growing rapidly, offering teams a plug-and-play model that reduces custom integration and accelerates development.

But this rapid adoption comes with trade-offs. Developers are often forced to choose between convenience and security, especially when running MCP servers or scaling beyond local experiments.

This paper highlights key security risks in today's MCP tooling, presents data on their real-world prevalence, and offers practical guidance for building secure, production-ready agent infrastructure.



A Promising Protocol at a Critical Juncture

What started as a niche experiment in tool invocation is quickly becoming foundational to the modern AI stack. MCP is now embedded in tools like VS Code, integrated into chat interfaces like Claude Desktop, and powering agents like Gordon. At Docker, we've seen this momentum firsthand: our MCP products saw over 7 million pulls just months after launch.

But despite the rapid adoption, the developer experience around MCP is still maturing. Most teams run into the same roadblocks:

Discovery is fragmented

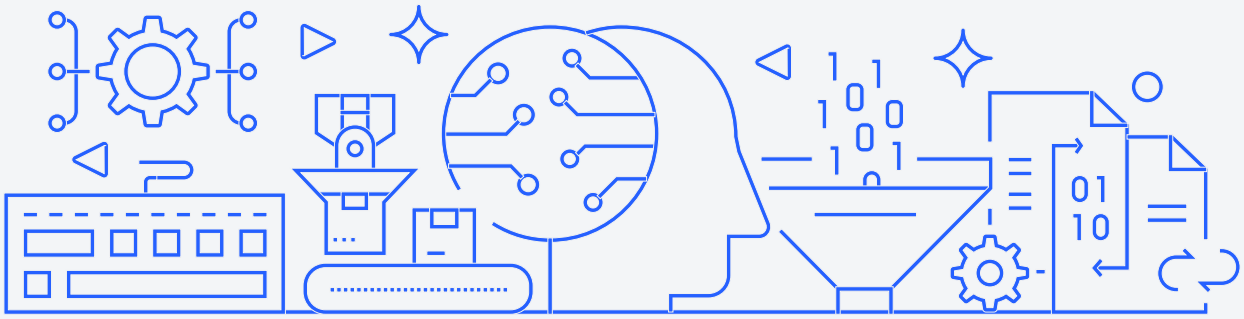
There's no centralized registry for official or trusted MCP servers. Many developers rely on unvetted servers from the open internet, putting their agents, applications, and connected systems at risk.

Security is lacking

MCP servers are often run via npx or uvx commands, exposing systems to unverified code. Credentials are typically passed as plaintext environment variables, increasing the risk of leaks and credential theft.

It's not enterprise-ready

Current MCP tools lack the controls and features enterprises need, like access policies, audit logging, and standardized security practices. Scaling beyond experimentation remains a challenge.



Understanding the Emerging Risk Surface

The most critical vulnerability in MCP-based systems isn't a single exploit; it's the design of the protocol itself.

MCP's client-server architecture allows agents to query available tools, interpret their capabilities through natural language descriptions, and invoke them autonomously. This flexibility is powerful, but every step in that flow introduces new attack vectors. And attackers are starting to take notice.

We're now seeing a new class of threats specifically engineered to exploit how MCP works:

MCP Rug Pull

A malicious server alters a tool's description after it's been approved, changing agent behavior without the user's knowledge.

MCP Shadowing

A malicious server injects a tool description that alters the agent's behavior toward a trusted service or tool.

Tool Poisoning

Hidden instructions are embedded in tool descriptions, invisible to users but interpreted by AI models, manipulating agent behavior from within.

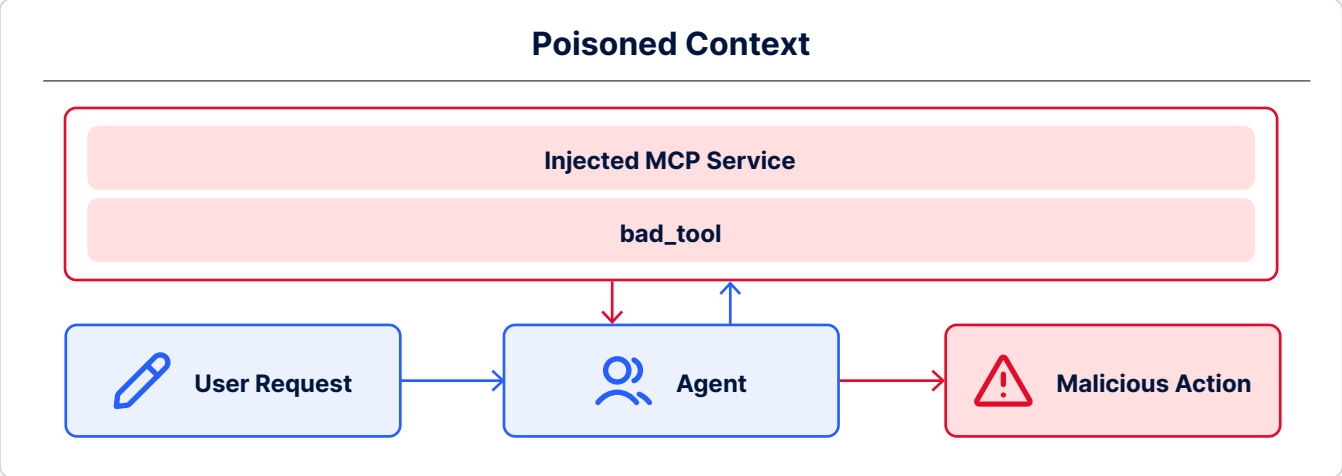


Figure 1: This illustration shows tool poisoning, which is an MCP vulnerability that allows attackers to hijack agents and manipulate them into performing unauthorized actions without the user's knowledge.
(Adapted from Invariant Labs)

These aren't theoretical risks. They're already being observed in the wild. Research from Queen's University in Canada found that 7.2% of MCP servers contained general vulnerabilities, with credential exposure accounting for over half of all detections. Even more concerning: 5.5% showed signs of MCP-specific tool poisoning attacks.

The broader picture is even more urgent. According to Accenture's State of Cybersecurity Resilience 2025 report, a shocking 90% of companies lack the maturity to defend against AI-enabled threats. When combined with the rapid proliferation of MCP tools, this leaves many organizations significantly more exposed to evolving attack vectors.

90% of companies lack the maturity to counter today's AI-enabled threats.

Source: Accenture State of Cybersecurity Resillience 2025

For developers and businesses, these gaps aren't just theoretical; they have real, measurable consequences. When agents interact with unverified tools or execute poisoned instructions, the result is often compromised credentials, leaked data, and cascading failures across connected systems. Every incident pulls engineering teams off roadmap work, slows product velocity, and drains resources responding to breaches that could have been prevented with stronger security controls. As MCP adoption accelerates, so do the stakes. And the cost of ignoring these risks will only grow.

As MCP Usage Grows, So Does the Need for Security and Control

Discovery, Security, and Standards Are Evolving

The MCP ecosystem is evolving fast, and we're seeing progress on several key fronts. Tool discovery is getting a major upgrade. An official MCP Registry is now in development. Think of it as the "yellow pages" for MCP servers. Docker will provide the secure runtime and distribution backing for these listings, ensuring developers have a trusted way to discover and run tools. Meanwhile, the MCP spec is actively improving OAuth, which will help bring credential management more in line with modern security practices.

Shifting Usage Patterns

We're also seeing clear trends in how MCP is being used. Remote MCP servers are becoming more common, moving beyond the traditional local, stdio-based setups. In parallel, MCP clients are evolving to support these remote workflows. Developers are no longer just connecting to a single server they host themselves; they're running multiple MCP servers in tandem. We see this firsthand across usage patterns on Docker.

At the same time, enterprises are ramping up their investment in agentic AI. A recent Capgemini study found that the number of companies piloting or scaling agent systems has jumped from 10% to 37% in just one year, a 3.5x increase.

The takeaway: we've moved past the era of hand-curated, locally hosted MCP setups. As agents begin connecting to dynamic mixes of local and remote servers, the complexity and the risk grow. We need better ways to manage it all: discovery, security, credentials, policies, and scale.

Requirements for Scaling MCP Tools Securely and Reliably

As MCP adoption moves from experimentation to production, organizations need a clear path for managing and securing this growing surface area. Whether you're running a few agents locally or scaling across environments, here are the core requirements to run MCP tools safely and effectively:

- **Access trusted MCP servers**
- **Keep secrets and credentials safe**
- **Run MCP servers securely**
- **Block known vulnerabilities**
- **Manage multiple MCP servers**
- **Centralize logging and monitoring**
- **Support dynamic, multi-agent environments**
- **Enforce consistent security policies**
- **Handle authentication and access control**
- **Deploy seamlessly across dev, test, and production**

What Secure MCP Infrastructure Looks Like

Securing MCP systems will require a different kind of architecture, one that assumes tools may be untrusted, metadata may be misleading, and agents may misinterpret instructions.

The following design principles should guide all production deployments of MCP:

Govern Discovery

Organizations must move from GitHub scraping to curated registries. Approved tools should be packaged as containers, vetted, versioned, and signed with verification of both provenance and integrity.

Isolate All Execution

Every MCP server should run inside a locked-down container. This includes minimal privileges, memory and CPU limits, read-only file systems, and blocked outbound network access unless explicitly required.

Remove Environment-Based Secrets

Credentials should never be passed via environment variables. Use secure stores and inject secrets only at runtime, scoped to a specific workload.

Monitor and Enforce Behavior

Every tool invocation should be logged, with policies in place to alert or block suspicious behavior. This includes monitoring for unexpected outputs, malformed responses, or prompt manipulation indicators.

Use an MCP Gateway to Scale Securely

Connect multiple MCP servers through a single, centralized gateway. This enables unified configuration, authentication, credential management, and threat detection, allowing organizations to enforce consistent security policies across all AI tools from one place.

How Docker Helps

Docker provides the foundational infrastructure to scale MCP securely and reliably. It does this by leveraging its unique strengths as both a provider of trusted content and secure container runtimes.

With the Docker MCP Catalog, developers and teams have access to over 100 curated, containerized MCP servers, each with a verifiable record of provenance and integrity. This gives teams a trusted starting point for building agentic AI applications without the risk of pulling unvetted tools from the open internet.

The Docker MCP Toolkit makes it easy to connect these servers to MCP clients and run them securely. Each server runs in an isolated container with strict runtime controls and minimal privileges, aligned with best practices for secure execution. Docker's secrets management eliminates the need to pass credentials via environment variables. Instead, secrets are injected at runtime, scoped to the workload, reducing the surface area for credential exposure.

To scale securely, the Docker MCP Gateway centralizes configuration, authentication, access control, credential handling, and threat detection. It allows enterprises to connect multiple MCP servers through a single entry point while applying consistent security and compliance policies. Paired with logging and monitoring integrations, Docker gives teams visibility into tool behavior and the ability to enforce policies, all without sacrificing developer speed or flexibility. Finally, the Docker MCP Gateway is designed for both development and production use. It runs as a containerized service and integrates seamlessly with Docker Compose, making it easy to orchestrate agentic applications across environments.

Docker helps organizations move MCPs from experimentation to production by taking a different approach, one that brings simplicity, structure, and security to the growing complexity of managing these systems.

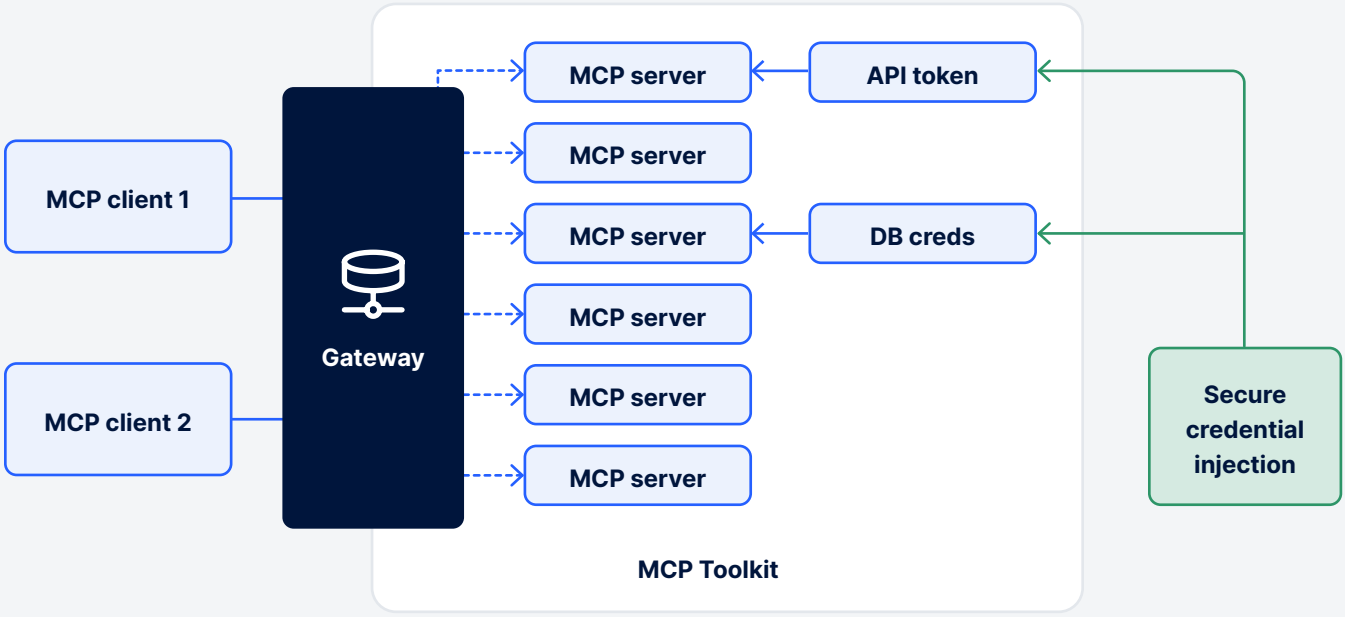


Figure 2: Architecture diagram of Docker's MCP Toolkit and Gateway, enabling scalable MCP server management and centralized control.

Conclusion: Preparing for the Next Phase of MCP

MCP is no longer a niche protocol; it's quickly becoming a core infrastructure for AI. To support this shift, organizations need scalable, secure systems for tool discovery, authentication, and policy enforcement.

Docker offers exactly that: a full-stack foundation with curated tools, secure runtimes, authentication and secrets management, and centralized control.

With the right guardrails in place, teams can harness MCP's full potential securely, reliably, and at scale.

Next Steps

- Talk to Docker about securing your agent infrastructure
- Discover hundreds of curated MCP servers on the Docker MCP Catalog
- Learn more about Docker MCP Toolkit
- Explore Docker MCP Gateway on GitHub

References

- **MCP Security Notification: Tool Poisoning Attacks, Invariant Labs**
  https://invariantlabs.ai/blog/mcp-security-notification-tool-poisoning-attacks
- **Accenture State of Cybersecurity Resilience 2025, Accenture**
  https://www.accenture.com/us-en/insights/security/state-cybersecurity-2025
- **Rise of agentic AI: How trust is the key to human-AI collaboration, Capgemini**
  https://www.capgemini.com/wp-content/uploads/2025/07/Final-Web-Version-Report-AI-Agents.pdf