What is an MCP gateway?

An MCP gateway is a reverse proxy that sits between AI agents (MCP clients) and the MCP servers they use. It becomes the single policy-enforced entry point: it authenticates agents and users, brokers OAuth tokens scoped to each backend, allows or denies individual tools, applies rate limits and approval gates, filters inputs and outputs, and writes one audit trail across every server. A gateway reduces sprawl and gives you a kill switch, but it does not make a model immune to prompt injection, and it becomes a high-value target itself.

HOW IT WORKS

01 What a gateway controls

- Identity and token brokering. Authenticates the agent and the user it acts for, then obtains a separate token for each backend, bound to that server. This avoids the token passthrough anti-pattern the MCP specification forbids.
- Catalog and allowlist. Only approved servers and versions are reachable; individual tools can be enabled per agent, team, tenant or environment. Destructive tools can be hidden from production entirely.
- Policy as code. Rules such as "this agent may read tickets but never close them" or "writes to finance systems need human approval" are enforced centrally.
- Tool definition pinning. The gateway can record each server's tool descriptions and block or flag changes, which blunts MCP tool poisoning and MCP tool shadowing.
- Content filtering. Pre- and post-call hooks can redact secrets and personal data, and flag injection patterns in tool output.
- Rate limits and quotas per agent and per tool, which limit both abuse and denial of wallet.
- Audit and kill switch. One record of who called what, with which arguments, and the ability to disable a tool or server everywhere at once.

02 What a gateway cannot do

A gateway sees tool calls, not the model's reasoning. If an agent has legitimate permission to send email and a poisoned document convinces it to email your customer list to an attacker, every

SOURCES

[1] Model Context Protocol specification

[2] MCP specification: Authorization

[3] MCP Security Best Practices

[4] OWASP Top 10 for Agentic Applications (2026)

call the gateway sees is allowed. That is agent goal hijacking, and a gateway can only limit it through scoping, approvals and anomaly rules, not prevent it.

Two other limits matter. Local stdio servers on a developer laptop usually never pass through a gateway at all. And the gateway itself now holds tokens for every backend, so a flaw in it (an authorization bypass, an SSRF, a policy that fails open) exposes everything behind it at once.

03 How to test an MCP gateway

Treat it like the high-value API it is:

- Authorization: can one agent or tenant reach another's servers, tools or audit logs? Try IDOR-style changes to server and tool identifiers, and replay tokens across backends.
- Policy bypass: do alternate tool names, case changes, encoded arguments or batched JSON-RPC requests slip past allow and deny rules? Does the policy fail open when the policy engine is slow or down?
- Token handling: does the gateway ever forward the caller's token instead of minting a backend-scoped one? Are backend tokens exposed in errors or logs?
- SSRF: can a registered server URL or OAuth metadata URL point the gateway at internal addresses?
- Agent-level abuse: with the gateway in place, can prompt injection still drive a permitted tool to a harmful outcome? That result tells you whether scoping and approvals are tight enough.

Test your AI application before an attacker does.
securelayer7.net/learn/ai-security/mcp-gateway

Open online