MAJOR LABS · REPORT · JUNE 2026

The State of MCP

More than a third of the most-used Model Context Protocol servers ship a risky code pattern. The ecosystem is an order of magnitude smaller than it advertises, its usage is concentrated in a handful of packages, and almost no one is measuring any of it. This is a firsthand, open, read-only read of where MCP actually stands.

38,157 advertised across four registries	**~1,200** genuinely evaluable (not 38,000)	**48.5M** monthly npm downloads (tracked)
36% of the 500 most-used ship a risky pattern	**32%** expose an SSRF-shaped pattern in source	**78%** carry an OSI license

Snapshot, June 2026. Every figure is firsthand and checkable at majorlabs.co/data, updated weekly.

Finding 1 — The population is inflated by an order of magnitude

Across four registries we find **38,157** distinct servers advertised. The official MCP registry lists **10,997**, plus 1,909 remote-only hosted servers with no public repo. We deep-scanned **2,468** firsthand and found roughly **1,200** genuinely evaluable: maintained, documented, and not abandoned. That is a 10 to 15x gap between advertised and real. One catalogue auto-indexes ~31,700 repos and reports tens of thousands; the maintained core is a fraction of that.

> *The gap between advertised and evaluable is not a footnote. It is the finding.*

Finding 2 — Adoption is brutally concentrated

Listings count what exists; downloads count what people run. Across 3,706 tracked npm packages we see **48.5M** monthly downloads, but the distribution is top-heavy to the point of distortion.

`@playwright/mcp`	**14.5M / mo**
`chrome-devtools-mcp`	**9.3M / mo**
`firebase-tools`	**7.7M / mo**
`@upstash/context7-mcp`	**5.0M / mo**
`@storybook/addon-mcp`	**4.2M / mo**

A handful of vendor-backed servers carry most of the volume; thousands of community servers see almost none. Counts include CI, mirrors, and bots, so read them as an upper bound; the npm pass is complete, the PyPI pass is pending.

Finding 3 — More than a third of the most-used servers ship a risky pattern

We ran a static, read-only read of the source for the 500 most-used active servers. **We never connect to, run, install, or probe a server.** Of 499 scanned, **36%** ship at least one risky pattern. SSRF surface dominates at **32%** — an outbound request built from a model-supplied argument with no allow-list, the canonical way an agent gets steered at an internal endpoint. Command injection (3.6%) and arbitrary code execution (3.2%) are rarer but more direct. By surface tier: 15 High, 164 Elevated, 320 Low.

What this is *not*: confirmed vulnerabilities, exploit proof, or anything run against a live server. The heuristics favour precision over recall, so every figure is a lower bound. Per-repo findings are held for coordinated disclosure to maintainers, not named here.

> *This is the honest version of the SSRF percentages that circulate unsourced: measured firsthand, from the code, with the method published.*

What it means

For builders: the registry count is noise. Filter to the maintained core, read the source, and do not wire a payment credential or filesystem access into a server you have not looked at. **For the ecosystem:** registries rank by recency and stars, not quality or safety. **For the agentic web:** the protocol shipped fast; the safety layer did not. Mandates, budgets, provenance, and identity are absent by default. That gap is why Major Labs builds them.

Method: four sources deduped (official registry, Glama, Smithery, our GitHub deep-scan), read-only throughout, security sweep static and heuristic over the 500 most-used active servers. Full methodology, raw data, and the open-source scanner: majorlabs.co/data and github.com/major-matters/mcp-scanner.

Major Labs builds open-source primitives for the agentic web: IdentityKit (who), MandateKit (may), BudgetGuard (spends), WitnessKit (did), plus the scanner that produced this report. Cite: Major Labs (2026). The State of MCP. majorlabs.co/reports/state-of-mcp.