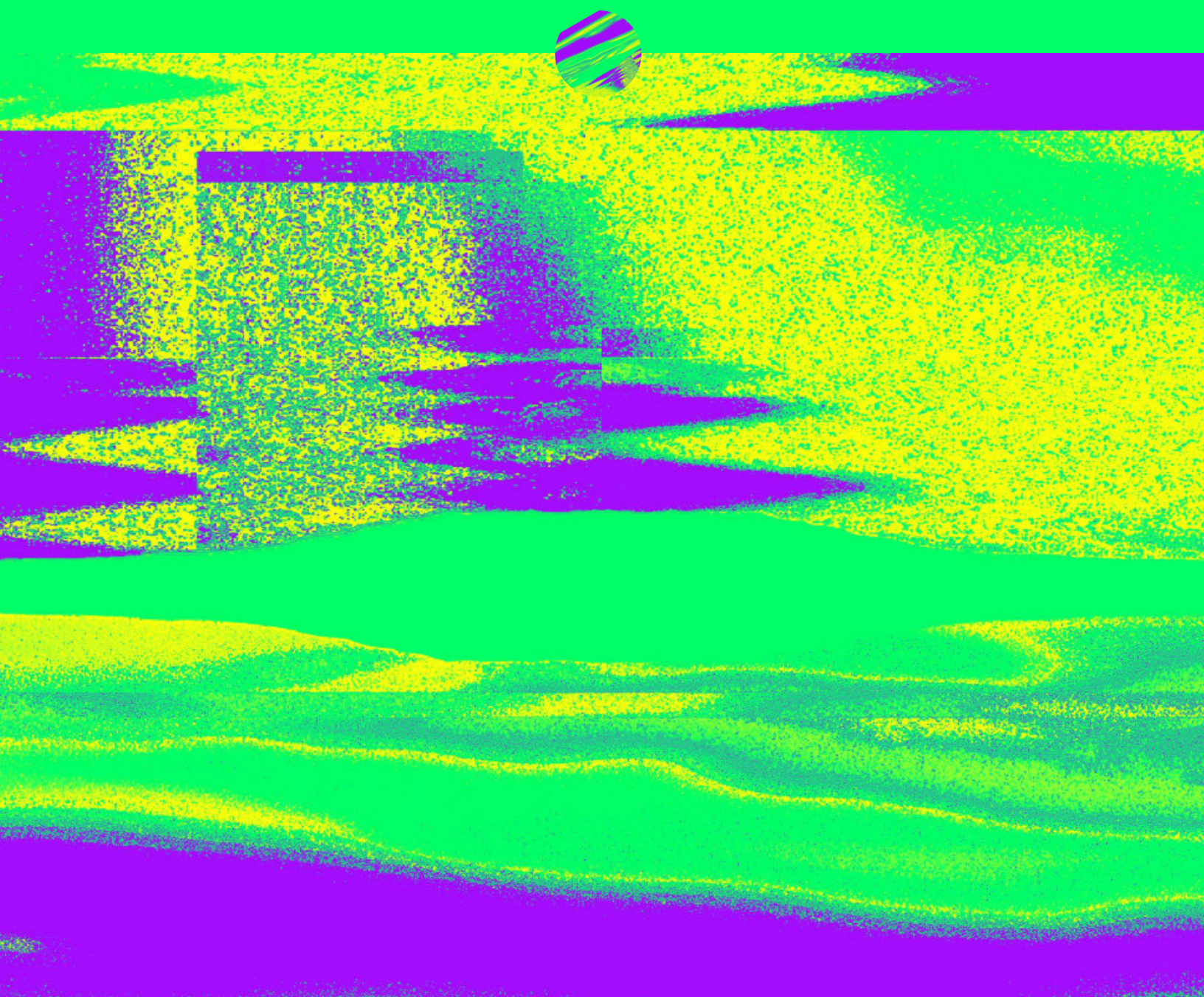
The State of MCP Tools

What 40,000+ MCP Servers Reveal About the Agent-Ready Enterprise



MCP has become the backbone of enterprise AI agents, but the ecosystem has grown faster than the quality of tools it depends on. Here's what the data shows, and what it costs enterprise teams when that quality falls short.

GAME STARTING

Insert coin to begin with top takeaways

Arcade's ToolBench has indexed 43,400+ MCP servers and analyzed 219,069, providing the first large-scale, objective quality assessment of the MCP ecosystem, which has become the de facto standard for connecting AI agents to enterprise software workflows. The verdict is unambiguous: only 0.5% of tools earned an A grade or above. 167,333 tools received an F.

43,400+
MCP servers indexed

219,069
tools analyzed

0.5%
earned an A grade or above

167,333
tools received an F

The tool documentation gap is the single biggest driver of agent failure in production today. Missing tool descriptions appear 6,568 times, the number one issue in the ecosystem. When tools don't tell agents what they do, agents guess. Hallucinations, retry loops, and wasted tokens cost an estimated $5-8 per task for unconstrained agents.

The lion's share of the MCP ecosystem is structurally incompatible with enterprise deployment. ToolBench caps STDIO-only servers at a maximum score of 50; they cannot be accessed by hosted MCP clients. Both STDIO and HTTP servers are counted in headline MCP growth statistics. Only one is deployable at enterprise scale.

Meanwhile, SaaS vendors are making deliberate choices about whether to let enterprise agents in, and those choices are altering procurement decisions. Arcade's data uncovers a widening divide between vendors investing in agent-ready integrations and those locking agents out (whether deliberately or by neglect).

Security in the MCP layer is largely an afterthought. Eighty-eight percent of MCP servers require credentials, but only 8.5% use modern OAuth authentication. The majority rely on insecure, long-lived static secrets, handling identity for production enterprise systems in the least secure way possible.

THE ERA OF AI AGENTS HAS ARRIVED, BUT SO HAS A HIDDEN TAX

Entering 2026, AI agents crossed from enterprise experiments to expectation. The past 18 months have seen a fundamental shift in how organizations think about automation from isolated copilots to agents that take action across entire business systems to make decisions, execute workflows, and operate software the way humans do. This shift has moved AI from a productivity experiment to a strategic priority, and the pressure to move quickly is intense. PwC found that 79% of senior executives report agents are already being adopted in their organizations, and 46% say they are concerned their company may be falling behind rivals.

That FOMO (fear of missing out) is rooted in compressing timelines, loosening budgets, and pushing organizations into production before the infrastructure beneath them is ready.

The Model Context Protocol (MCP) has emerged as the connective tissue that enables AI agents to integrate with enterprise software at remarkable speed. What started as an open protocol for connecting AI agents to enterprise software has become the de facto standard that major AI providers, from OpenAI to Anthropic, Hugging Face, and LangChain, have coalesced around. MCP Server downloads grew from roughly 100,000 in November 2024 to 97 million monthly by March 2026, a 970x increase. At the same time, Gartner projects that this year, 75% of API gateway vendors and 50% of IPaaS vendors will have MCP features.

Monthly downloads

100,000

in November 2024 to

97m

by March 2026

970x

increase

78% of enterprise AI teams report at least one MCP-backed agent in production. Over two-thirds of CTOs have named MCP their default agent-integration standard. MCP has won. It is the protocol enterprises are building on.

What hasn't followed MCP's meteoric rise as the de facto agent-integration standard is quality and consistency across its tools. Arcade built ToolBench as the first large-scale, objective benchmark for MCP server quality to assess which ones are actually production-ready. The answer is sobering: Arcade found that of 43,400 servers and 219,069 tools analyzed, only 0.5% earned an A grade or above. In fact, most tools (167,333) received an F. That ratio is not a rounding error. It is a structural characteristic of an ecosystem that was scaled for breadth before reliability. Organizations building on this ecosystem are absorbing a hidden tax in wasted tokens, agent failures, security risk, and engineering work that compounds at scale.

43,400+
MCP servers indexed

219,069
tools analyzed

0.5%
earned an A grade or above

As MCP has taken over the agent-integration ecosystem, the gaps in its implementation have become the defining challenge for enterprises moving from pilot to production. The findings in this report detail the roadmap for closing these quality gaps that are costing enterprises in production today.

Chapter 1 establishes why the tool documentation gap is the reliability gap and why the most common failure mode in AI agent production has nothing to do with the model.

Chapter 2 tells the story of the protocol divide that makes most of the MCP ecosystem invisible to hosted enterprise agents. Here, we explore why headline server counts overstate what is actually deployable.

Chapter 3 tracks the widening divide between SaaS vendors investing in agent-ready infrastructure and those actively resisting it. This is why vendor openness is becoming a make-or-break factor in procurement.

Chapter 4 explores the security exposure accumulating across the MCP ecosystem, and why it is a quiet disqualifier for enterprise-scale agent deployment.

Chapter 5 concludes with what this means for enterprise agentic strategy and the decisions that will separate organizations that lead in the agent economy from those that play catch-up.

CHAPTER 1: THE TOOL DEFINITION GAP IS THE RELIABILITY GAP

Agents aren't failing because models are bad. They're failing because tools don't tell them what to do.

When an AI agent fails in production, the instinct is to blame the model. If the agent hallucinated, misunderstood the task, or looped endlessly without completing a workflow, then surely that's a model problem. A better one would have figured it out. Most of the time, that instinct is wrong.

Arcade's ToolBench data signals a more mundane culprit: the tools themselves. Specifically, the way tools describe or fail to describe what they do, what they return, and what occurs when something goes wrong. When an agent reaches for a tool and finds no description, it doesn't stop and ask for clarification. The agent infers or guesses, and that guessing is where production AI systems quietly fall apart.

Missing descriptions are the single most common issue across the entire MCP ecosystem. This problem appears more than 6,568 times across the servers that ToolBench analyzed. The number alone is striking, but the downstream consequences make it a real business problem rather than a developer inconvenience. An agent that can't reliably identify the right tool for a task will call the wrong one. It will retry and then consume even more tokens on dead ends. When agentic models require an estimated 5-30x more tokens per task than a standard chatbot, those one-way trips are expensive.

When agentic models require an estimated

5-30x

more tokens per task than a standard chatbot, those one-way trips are expensive.

The problem doesn't stop at missing descriptions. ToolBench surfaces a cascade of related failures that compound in multi-step workflows:

Issue	Occurrences	Agent impact
Missing Description	6,568	Agents guess what a tool does, leading to hallucinations and wrong tool selection
No Error Handling	1,899	400/500 errors create unrecoverable retry loops
No Output Schema	1,407	Agents can't plan multi-step workflows without knowing what a tool returns
No Pagination Guidance	694	Unbounded list returns exceed context windows and degrade performance

The hidden cost of the tool definitions gap shows up in production, at scale, and when the bills arrive. Researchers studying agentic workflows have given it a name: "The Unreliability Tax." This tax represents the additional cost in compute, latency, and engineering time required to compensate for tools that don't behave predictably. For unconstrained agents on sophisticated workflows, that tax can reach $5-8 per task.

For unconstrained agents on sophisticated workflows, that tax can reach

$5-8

The Cost Difference in Practice

Arcade ran eight identical CRM queries through two Attio MCP toolkits (its own and Composio's) to measure the real-world token cost of tool quality differences. Arcade consumed 7,426 tokens total. Composio consumed 747,083. That's a 100x difference for the same tasks against the same data. At an enterprise scale of 100 agents running 200 queries per day, that gap amounts to a difference of $1.97 million in annual token costs at Claude Sonnet pricing. At Opus pricing, it's 5x that.

Full methodology and raw data: arcade.dev/blog/attio-mcp-toolkit-benchmark

The wider pattern is consistent with what enterprises are experiencing in AI. Forty-seven percent of AI users have admitted to making at least one major business decision based on hallucinated content, and knowledge workers now spend an average of 4.3 hours per week fact-checking AI outputs. People assume that as better models come online, this gap will close. However, ToolBench's data reveals a different remedy: better tools. The model works only with what it's given. When that is incomplete, ambiguous, or silent on failure modes, even the best or most advanced model in the world won't deliver the desired results.

47%

of AI users have admitted to making at least one major business decision ...now spend an average of

4.3hrs

per week fact-checking AI outputs.

The good news is that none of this is technically hard to fix. Arcade's 54 Agentic Tool Patterns serve as the standard against which ToolBench scores, and as the remediation path for every issue in the table above. This is what agents should look like to be secure, reliable, and enterprise-ready in production.

CHAPTER 2: MOST MCP SERVERS AREN'T ENTERPRISE-READY

The growth statistics for MCP over a short period of time are genuinely impressive. MCP has notched 97 million monthly SDK downloads and has been adopted across every major AI provider over the past 18 months. Those figures suggest that the MCP ecosystem is vast, mature, and ready for prime time. But the enterprise-ready MCP ecosystem looks larger than it actually is.

MCP has notched

97 mil.

monthly SDK downloads and has been adopted across every major AI provider over the past 18 months.

The eye-popping numbers count every MCP server equally, but not all MCP servers are created equal. The most important dividing line is what's underneath the hood. A significant portion of these servers is architecturally locked out of the environments where enterprise agents actually need to run. These are the STDIO-only servers: tools built to run locally on a developer's machine and communicate through standard input and output. By design, they can't be accessed by hosted MCP clients, where the power enterprise-grade agent deployments happen.

ToolBench makes this structural reality explicit in its scoring system. STDIO-only servers are capped at a maximum score of 50, regardless of how well-documented or secure they might otherwise be. This is for a good reason: HTTP transport is a prerequisite for production-grade, multi-tenant, hosted agent deployment. STDIO-only servers simply don't meet the qualification for deployment across a business at scale.

The ecosystem looks bigger than it is. STDIO-only servers are invisible to hosted agents, which amounts to a significant share of the "available" MCP infrastructure.

Enterprises have spoken about the model they need. Among the 20 most-searched MCP servers, 80% provide remote access. The reality is that the ecosystem's headline numbers are misleading, and much of what's being counted doesn't actually serve the market's needs. For example, Postgres has some 1,200 servers in the market. Yet of those, only one received an A grade on ToolBench's scorecard.

Postgres has

~1,200

servers in the market. Yet of those, only one received an A grade on Tool-Bench's scorecard.

This gap has a second-order consequence that compounds as organizations evolve from evaluation to deployment. Teams that move quickly into MCP adoption without a quality filter in place are prone to accumulating server sprawl: a growing catalog of tools with no central governance, inconsistent security postures, and no reliable way for end users to distinguish production-ready servers from dev experiments.

Teams that move quickly into MCP adoption without a quality filter in place are prone to accumulating

SERVER SPRAWL

Hundreds of disparate servers arise with no central catalog, inconsistent security, and end users who are unable to distinguish safe from unsafe options.

Protocol compliance is the first filter that stops sprawl before it starts, and most servers don't pass it. Organizations that skip that diligence now are setting themselves up for a painful audit later, when IT and security teams come looking for answers that nobody has.

CHAPTER 3: VENDOR OPENNESS IS THE NEW PROCUREMENT VARIABLE

One version of the enterprise AI agent story is entirely about the technology. This side emphasizes which model, framework, and infrastructure is the most advanced or "right" for today's enterprises. Organizations focused on this version are missing the critical other side of the AI agents' story: what happens when the software vendor on the other side of the integration doesn't want to cooperate.

This is the version that doesn't show up in demos. What those don't show is what happens when that software application's API throttles requests from third-party agents, or requires a separate enterprise contract for agentic access. The model works, but the agents fail, and the failure has nothing to do with AI.

Arcade's Toolbench data, cited by The Information in March 2026, puts this dynamic into sharp relief. Across the SaaS landscape, vendors are making deliberate decisions about how much access they will grant to enterprise agents that aren't their own, and those decisions vary dramatically from platform to platform. The result is a clear and widening divide between the vendors that are letting the agents in to get to work, and those that are, deliberately or by neglect, becoming dead ends.

The Open Tier: Betting on Interoperability

On the open side is a consistent pattern. Vendors who have made interoperability a product decision instead of an afterthought. GitHub ships a first-party MCP server with MCP support directly into Copilot, making it one of the earliest major platforms to treat agent connectivity as a core product capability. Stripe went further in March 2026 with the Machine Payments Protocol, an open standard that lets agents make and receive payments autonomously. In GTM, Attio stands out for building API-first in a way that most CRMs aren't. Its API spec is granular enough to support precise, well-optimized tool definitions without guesswork. These vendors are betting that making it easy for agents to work with their platforms will deepen customer relationships and benefit their platforms.

The Closed Tier: Betting on Control

On the more closed side, the motivations are more varied and, in some cases, more candid. Slack recently tightened API restrictions, severely limiting how quickly external agents can access message history. This dynamic first surfaced nine months earlier, when Slack made it difficult for customers to use AI search tools to access their own messaging history. In a similar vein, Gong's API is restricted to enterprise contracts with a separate commercial agreement.

Most Open	Least Open
Github	**Slack**
Figma	**Meta**
Linear	**Workday**
Hugging face	**LinkedIn**
Stripe	**Discord**
Attio	**Gong**

There is no self-serve and no MCP server. For a platform built around sales intelligence, that's a deliberate choice to keep users inside their product rather than feeding context to an agent. A final example, Meta's advertising tools and WhatsApp don't include MCP support at all. Teams that need to manage ad spend through agents report resorting to browser automation with full account access: a workaround that trades a convenience problem for a security one.

> **The vendors who embraced open APIs in the cloud era became the platforms enterprises standardized on. The ones that locked down became the systems that enterprises migrated away from. History is likely to repeat itself.**

The business tea leaves at work here are not difficult to read. The companies on the open list are betting that interoperability drives retention: if agents can work with their platforms more easily, customers will build deeper dependencies and churn less. The companies on the other side of the coin are betting on control, and that owning the access layer is a moat worth defending. History suggests that the open side is the smarter bet.

The vendors that embraced open APIs in the cloud era became the platforms that enabled enterprises to unify workflows. The ones that locked down became the systems that customers migrated away from.

CAN OUR AGENTS WORK WITH THIS?

Is now a core evaluation crtierion

> **History suggests that the open side is the smarter bet.**

For procurement and IT leaders, the implication is practical. MCP openness deserves a mark on vendor scorecards alongside other key decision signals, such as price, security certifications, and support SLAs. “Can our agents work with this?” is now a core evaluation criterion. For the vendors on the closed list, it’s a growing reason for businesses to look elsewhere.

CHAPTER 4: SECURITY IN MCP IS AN AFTERTHOUGHT

As enterprises scale agent deployments, MCP security practices are becoming the biggest liability they aren't watching yet.

Enterprise security teams have spent years building controls around how humans access sensitive systems. The playbook is a carefully established collection of strong authentication and authorization requirements, credential rotation policies, and a zero-trust architecture. What most security guidelines don't yet account for is a new category of actor that needs access to those same systems: the friendly neighborhood AI agent.

Agents can perform the same tasks as humans, but they don't authenticate the way we do. They don't have passwords stored in a password manager or a YuBiKey in their pocket. Agents authenticate through the tools they call. In the vast majority of cases, those tools are handling the credential handoff in ways that would fail even the most basic security audit.

A large-scale analysis of 5,200 open-source MCP servers found that 88% require credentials to operate. Surprisingly, over half of these servers rely on insecure, long-lived static secrets such as API keys and Personal Access Tokens. These credentials never expire, can't be easily revoked, and provide broad access to the systems they unlock. That's a huge problem. Modern auth methods like OAuth, which provide scoped, revocable, and time-limited access, are used by only 8.5% of servers.

To put this in perspective, consider what happens when a static API key is compromised in an agentic AI workflow. Unlike a human user, whose behavior raises flags when it deviates from a pattern, an agent with a compromised credential may look entirely unsuspicious. Static secrets create a security risk at the point of compromise and a detection problem that compounds the longer they remain in place.

A large-scale analysis of

5,200

open-source MCP servers found that

88%

require credentials to operate.

Security is the quiet disqualifier. Enterprises can pilot agents past IT. They cannot scale them past IT. The credential posture of the tools will be scrutinized, and most of the ecosystem will fail that audit.

ToolBench's security scoring systematically captures this risk. For remote servers, the benchmark evaluates the correctness of the OAuth 2.0 flow, PKCE support, authorization server discovery, and 401 challenge handling. These are the basic components of a secure, modern authentication implementation. Yet, few remote servers score well on these dimensions. The technical bar is not especially high: Gartner recommends implementing mandatory security requirements for all MCP servers, including the exclusive use of HTTPS endpoints and a robust OAuth implementation. However, the data shows that the gap between where the ecosystem is and where it should be is largely and significantly unacknowledged.

Part of the problem is structural. Recent reporting from The Register points to a design-level issue in the MCP specification itself. The protocol currently lacks strong, standardized security boundaries that would prevent a compromised tool from escalating its access to adjacent systems. It remains unclear whether this is a bug or a feature, but it's a signal that the security architecture of MCP needs more attention from the people who are building the standards themselves.

For enterprises, the practical consequence is that MCP security practices are becoming a major risk surface area at exactly the moment when organizations are granting more access than ever before. Security teams that haven't yet turned their attention to MCP server credential practices will soon do so. When they do, most of what is currently running in production will not survive the review.

Fixing the tools is the necessary first step. Building the runtime layer is what makes enterprise-scale agent deployments scalable and sustainable

There is a limit to how much can be solved in the tools alone. Even with well-designed tools and proper OAuth implementation, individual MCP servers operating in isolation can't enforce consistent credential policies, manage token lifecycles across integrations, or provide the audit trail that enterprise security teams demand. The runtime layer is where these problems get solved. The credential, token, and API management challenges that enterprises will face at scale require a separate runtime layer: one that enforces access policies consistently and provides centralized governance across all agents, users, and systems.

There is a
LIMIT
to how much can be solved in the tools layer alone

CHAPTER 5: WHAT THIS ALL MEANS FOR ENTERPRISE AGENTIC STRATEGY

The four findings in this report describe different failure modes of AI agents today. They also share a common root cause. While MCP has emerged as the standard protocol for agents, this ecosystem was built at lightning speed by developers addressing immediate gaps, without a stable, shared definition of what "production-ready" means. The result is an infrastructure layer that can look deceptively extensive but reveals its limitations under the sustained demands of enterprise deployment.

This is to be expected as every foundational technology goes through this phase. The early cloud was insecure and unreliable. Early mobile SDKs were inconsistent and poorly documented. The emergence of quality standards gave developers a clear lens to build toward and businesses a clear lens to evaluate against. Arcade's ToolBench is the first attempt to establish that standard for the MCP tools. The benchmark makes the gap between the ecosystem and that standard visible at scale.

The window to close the quality gaps is narrowing. Organizations that define rigorous, well-defined standards for tool quality, protocol compliance, vendor evaluation, and credential security will now have an infrastructure advantage that scales cleanly into the future. The organizations that don't will spend the next several years remediating problems that were entirely predictable and avoidable.

LEVEL UP



Press continue to see conclusions from the data:

Adopting tools based on quality is an infrastructure decision, not a developer detail. The choice of MCP server affects token costs, agent reliability, and security posture in ways that compound at scale. Treating tool selection as a developer convenience rather than an infrastructure commitment is how the Unreliability Tax quietly becomes a budget line. This belongs in the same infrastructure conversation as cloud provider selection, API gateway strategy, and observability tooling.

The SaaS stack audit has a new dimension. Every software vendor in an organization's stack now has a de facto answer to the question: “Can our agents work with this?” For vendors on the closed list, that answer is increasingly a reason to reexamine the relationship. MCP openness (first-party server availability, transport protocol, authentication model) should be part of annual vendor reviews going forward, alongside price, security certifications, and support SLAs. The vendors making it hard for agents to work with their platforms today are betting that customers won't notice or act. Enterprises that add this to their scorecards are the reason that bets won't pay off.

The quality window is closing. Standards are being set right now, at a moment when most of the ecosystem still scores an F. The organizations and vendors who invest in quality practices today will have a structural advantage that compounds as agentic adoption accelerates. The ones who wait will be catching up to a bar that keeps rising.

Recommendations for Enterprise Teams

Team	Priority Actions
Enterprise Buyers	Require ToolBench grade A or above as a baseline filter for MCP server procurement. Add MCP openness to vendor scorecards alongside price, security, and SLAs. Prioritize vendors with first-party remote MCP servers and OAuth-based auth.
Tool Builders	Address the top ToolBench issues first: descriptions, error handling, and output schemas. Move from STDIO to HTTP, as STDIO is not enterprise-viable. Arcade's 54 Agentic Tool Patterns provide the design standard for these fixes.
Platform & IT Teams	Build internal MCP governance before sprawl sets in. Establish a central runtime for policy enforcement, auditability, and trusted MCP discoverability. These are non-negotiables for sustainable MCP deployment at scale.

Ready to build secure, enterprise-ready AI agents?

Get in touch at Arcade.dev.

APPENDIX

About ToolBench
ToolBench is Arcade's open benchmark for evaluating the quality of MCP servers at ecosystem scale. Every server in the index is evaluated across multiple dimensions using two scoring models, one for local servers and one for remote, with a weighted average determining the final grade.

Local servers are scored on Definition Quality (50%), Protocol Compliance (20%), and Supportability (30%). Definition Quality measures how well tools are designed for AI agent consumption: naming clarity, description quality, and parameter schema completeness, informed by Arcade's 54 Agentic Tool Patterns. Remote servers are scored on Protocol Compliance (40%), Security Checks (30%), and Supportability (30%), covering OAuth 2.0 correctness, PKCE support, transport security, and enterprise supportability signals, including SLA tier, compliance certifications, and multi-region availability.

Grade thresholds run from A+ (90–100) through F (below 50). The full scoring rubric, improvement guidance, and scoring API are publicly accessible at toolbench.arcade.dev/methodology.

About Arcade.dev
Arcade.dev is the industry's first MCP runtime enabling AI to take secure, real-world actions. As the MCP runtime, Arcade is uniquely able to deliver secure agent authorization, highly reliable tools, and centralized governance. Arcade helps teams at some of the largest organizations deploy multi-user AI agents that take actions across any system with granular permissions and complete visibility—no complex infrastructure required. Learn more and try it for free at www.arcade.dev.