INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-07-01 07:07 UTC

MCP Tool Poisoning Enables Silent AI Agent Hijacking and Enterprise Data Exfiltration

SECURITY ANALYSIS | HIGH | CVSS 7.5

SCC Item ID	SCC-STY-2026-0309
Type	Security Analysis
Severity	HIGH
CVSS Base Score	7.5
Affected Products	Microsoft 365 Copilot, Copilot Studio, Azure AI Foundry, MCP-connected AI agents broadly; GitHub MCP server; postmark-mcp npm package
Published	2026-06-30T13:46:07
Discovery Source	Rss

Executive Summary

Microsoft Incident Response and the Defender research team have documented a practical attack path in which adversaries manipulate Model Context Protocol (MCP) tool descriptions to silently redirect AI agents toward enterprise data exfiltration, requiring no code execution. Testing across 45 real MCP servers demonstrated success rates of up to 72.8%, and a confirmed supply chain incident involving a malicious MCP package has already occurred, moving this threat from theoretical to active. This development signals a structural trust gap in agentic AI architecture that affects any organization deploying AI agents connected to external tools, including Microsoft 365 Copilot, Copilot Studio, and Azure AI Foundry.

Technical Analysis

The attack exploits a foundational design assumption in Model Context Protocol: tool descriptions loaded into an AI agent's context window carry the same functional authority as system-level instructions. According to Microsoft Incident Response and the Defender research team, an adversary who can modify a tool's description field, whether through supply chain compromise, a malicious MCP package, or unauthorized server access, can effectively rewrite the agent's operating directives without touching any executable code. The attack maps to multiple MITRE ATT&CK techniques. Tool description manipulation aligns with T1565 (Data Manipulation) and T1036 (Masquerading), as the poisoned description causes the agent to misrepresent its own actions. Subsequent data collection and staging maps to T1552 (Unsecured Credentials) and T1567 (Exfiltration Over Web Service), while outbound communication via MCP-connected services maps to T1071.001 (Application Layer Protocol). The supply chain vector maps to T1195.001 and T1195.002 (Supply Chain Compromise). The GitHub MCP server and the postmark-mcp npm package represent confirmed real-world examples of the supply

chain exposure surface. Detection is materially complicated because each individual action taken by a poisoned agent, a file read, an API call, a data transfer, appears legitimate in isolation. There is no anomalous binary, no exploit payload, and no command-and-control channel that traditional endpoint and network controls are tuned to catch. The attack's success rate of up to 72.8% across 45 tested MCP servers, as reported by Microsoft, underscores that this is not a niche edge case. The affected surface is broad: any organization using MCP-connected AI agents, including those built on Microsoft 365 Copilot, Copilot Studio, or Azure AI Foundry, is potentially exposed. The underlying weaknesses span improper input validation (CWE-20), injection (CWE-74), improper control of dynamically managed code resources (CWE-913), improper access control (CWE-284), and, where homoglyph rendering is exploited in description fields, CWE-1357. The industry implication is significant: as enterprises accelerate agentic AI adoption, the MCP ecosystem is becoming a high-value target with a security model that has not kept pace with deployment scale.

Action Checklist

1. Step 1: Assess exposure, audit all MCP-connected AI agents in your environment, including Microsoft 365 Copilot, Copilot Studio, Azure AI Foundry, and any internally developed agents; inventory every MCP server and package in use, including third-party and open-source packages such as postmark-mcp and GitHub MCP server
2. Step 2: Review controls, validate that MCP tool descriptions are treated as untrusted input and subject to change-control review (NIST AC-3, AC-6); verify that least-privilege principles govern what data and systems each AI agent can access (NIST AC-6, CIS 5.4); confirm that account and session logging for AI agent actions is enabled and reviewed (NIST AU-2, AU-6, CIS 8.2)
3. Step 3: Update threat model, add MCP tool description poisoning as an explicit attack vector in your threat register, mapping to T1565, T1036, T1195.001, T1195.002, T1567, and T1071.001; assess which AI agent workflows have access to sensitive enterprise data and model the blast radius of a silently redirected agent
4. Step 4: Harden supply chain controls, apply software inventory and integrity controls to all MCP packages (CIS 2.1, CIS 2.2); require that MCP server and package updates go through a reviewed change process before deployment; treat MCP tool descriptions as code artifacts subject to version control and integrity verification (CIS 4.6)
5. Step 5: Communicate findings, brief leadership on the confirmed supply chain incident and the 72.8% empirical success rate reported by Microsoft; frame the risk in terms of enterprise data accessible to deployed AI agents, not as a generic AI vulnerability
6. Step 6: Monitor developments, track Microsoft Security Blog and Microsoft Defender Threat Intelligence for follow-up guidance, updated detection rules, and any patches or architectural changes to MCP server implementations

IR / Forensic Enrichment

Triage Priority	URGENT

Escalation Criteria	Escalate immediately to CISO and legal/privacy counsel if the Microsoft Purview Unified Audit Log (CopilotInteraction record type) or MCP server request logs reveal that an AI agent exfiltrated data containing PII, PHI, or regulated financial records to an external endpoint (e.g., via postmark-mcp email delivery or GitHub MCP server repository writes), triggering breach notification obligations under GDPR, HIPAA, or applicable state privacy law.
Recovery Notes	After containment actions are complete (packages pinned or removed, agent permissions restricted, tool descriptions version-controlled), re-enable AI agent workflows only after each agent's MCP tool descriptions have been reviewed by a human analyst and verified against a cryptographic hash baseline. Monitor Microsoft Purview CopilotInteraction audit logs and Azure AI Foundry activity logs daily for a minimum of 30 days post-remediation, specifically watching for anomalous agent actions such as unexpected data queries to sensitive SharePoint sites or external API calls not present in prior baselines. Re-run 'npm audit' against all MCP packages weekly until Microsoft publishes a formal architectural remediation for MCP tool description trust boundaries.
Forensic Artifacts	MCP tool description manifest files (JSON): the 'description' field of each registered tool — the injection point for poisoning instructions; archive verbatim content with timestamps from all MCP servers before any remediation action | Microsoft Purview Unified Audit Log — CopilotInteraction record type: agent prompt inputs, tool invocations, and data access events for Microsoft 365 Copilot and Copilot Studio; query via 'Search-UnifiedAuditLog -RecordType CopilotInteraction' for the maximum available retention window | Azure AI Foundry activity logs and agent run traces: full request/response chains showing which MCP tools were called by Foundry-hosted agents, what data was retrieved, and what outputs were generated — available via Azure Monitor diagnostic settings | npm package integrity evidence: 'npm list --json' output and SHA-256 hashes of postmark-mcp and GitHub MCP server package contents at the time of discovery, compared against published npm registry checksums to confirm whether a tampered supply chain package was installed | External egress logs (email relay logs from postmark-mcp SMTP calls, GitHub API audit logs for MCP server repository write operations): these logs capture the exfiltration endpoint — the destination to which a poisoned agent was silently redirected to deliver stolen enterprise data

Per-Action IR Details

Step 1: Assess exposure — audit all MCP-connected AI agents in your environment, including Microsoft 365 Copilot, Copilot Studio, Azure AI Foundry, and any internally developed agents; inventory every MCP server and package in use, including third-party and open-source packages such as postmark-mcp and GitHub MCP server

NIST Phase: Preparation

Reference: NIST 800-61r3 §2 — Preparation: establishing situational awareness of assets and attack surface before or during an incident

Controls: CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 2.2 (Ensure Authorized Software is Currently Supported)

Compensating: Run 'npm list -g --depth=0' and 'pip list' on all hosts running MCP servers to enumerate installed packages; cross-reference against a manually maintained spreadsheet of approved MCP packages. Use 'Get-ChildItem -Recurse' on Windows or 'find / -name "*.mcp*"' on Linux to locate MCP configuration files. A 2-person team can divide by platform (cloud vs. on-prem) and consolidate results in a shared spreadsheet flagging any postmark-mcp or unapproved GitHub MCP server instances.

Evidence: This is an inventory step and does not alter live state. Before proceeding, snapshot current MCP server configuration files (e.g., mcp.json, tool manifest files, package.json for npm-based servers) and capture the output of 'npm audit' for all MCP packages to preserve the pre-remediation state of the supply chain. Document all AI agent permission scopes (Microsoft 365 Copilot connector permissions, Azure AI Foundry agent role assignments) as they

exist at discovery time.

Step 2: Review controls — validate that MCP tool descriptions are treated as untrusted input and subject to change-control review (NIST AC-3, AC-6); verify that least-privilege principles govern what data and systems each AI agent can access (NIST AC-6, CIS 5.4); confirm that account and session logging for AI agent actions is enabled and reviewed (NIST AU-2, AU-6, CIS 8.2)

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: validating detection coverage and reviewing control efficacy against the identified attack vector

Controls: NIST AC-3 (Access Enforcement), NIST AC-6 (Least Privilege), NIST AU-2 (Event Logging), NIST AU-6 (Audit Record Review, Analysis, And Reporting), CIS 5.4 (Restrict Administrator Privileges to Dedicated Administrator Accounts), CIS 8.2 (Collect Audit Logs)

Compensating: In the Microsoft 365 Admin Center, navigate to Settings > Copilot > Connected Apps and review each connector's granted permissions against the principle of least privilege. For Azure AI Foundry, use 'az role assignment list --all' to enumerate agent service principal permissions. For MCP session logging where no SIEM exists, enable Azure Monitor diagnostic logs for AI Foundry and export to a local Log Analytics workspace (free tier); configure Microsoft Purview Audit (standard) to capture Copilot interaction events and export via PowerShell: 'Search-UnifiedAuditLog -RecordType CopilotInteraction'.

Evidence: Before modifying any agent permissions or logging configurations, capture current Microsoft Purview Unified Audit Log entries for CopilotInteraction record types covering the prior 90 days via 'Search-UnifiedAuditLog -RecordType CopilotInteraction -StartDate [date] -EndDate [date]'. Also export Azure AI Foundry activity logs and any existing MCP server request/response logs, which would contain tool description content delivered to the agent — this is the primary artifact showing whether poisoned tool descriptions were received and acted upon.

Step 3: Update threat model — add MCP tool description poisoning as an explicit attack vector in your threat register, mapping to T1565, T1036, T1195.001, T1195.002, and T1567; assess which AI agent workflows have access to sensitive enterprise data and model the blast radius of a silently redirected agent

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: understanding the scope, impact, and attack surface to inform containment decisions

Compensating: Document the blast radius analysis in a simple matrix: list each deployed AI agent (Copilot, Copilot Studio bot, Foundry agent), the data sources it can query (SharePoint sites, Exchange mailboxes, SQL databases, APIs), and the external endpoints it can reach (email via postmark-mcp, GitHub repositories via GitHub MCP server). For each, estimate the maximum sensitive data volume accessible in a single agent session. This matrix requires no tooling beyond spreadsheet software and serves as the threat model artifact.

Evidence: This step is analytical and does not alter live state. However, before proceeding to containment actions informed by this threat model, preserve MCP tool description content as it currently exists: pull and archive the tool manifest (tool name, description, input schema fields) from each registered MCP server. In a poisoning scenario, the malicious instruction is embedded in the 'description' field of a tool definition — this field content is the forensic artifact distinguishing a poisoned server from a legitimate one.

Step 4: Harden supply chain controls — apply software inventory and integrity controls to all MCP packages (CIS 2.1, CIS 2.2); require that MCP server and package updates go through a reviewed change process before deployment; treat MCP tool descriptions as code artifacts subject to version control and integrity verification (CIS 4.6)

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment: reducing attack surface and preventing further exploitation while the threat is active

Controls: CIS 2.1 (Establish and Maintain a Software Inventory), CIS 2.2 (Ensure Authorized Software is Currently Supported), CIS 4.6 (Securely Manage Enterprise Assets and Software)

Compensating: Pin all MCP npm packages to verified exact versions in package-lock.json and run 'npm audit --audit-level=high' on every MCP package before deployment. Use 'shasum -a 256' (Linux/macOS) or 'Get-FileHash' (PowerShell) to compute and record hashes of MCP server binaries and configuration files at a known-good state; re-verify hashes after any update. Store MCP tool description JSON manifests in a Git repository with branch protection and required code review so any modification to a tool's 'description' field is a reviewable commit — this directly addresses the poisoning vector where malicious instructions are injected into that field.

Evidence: Before enforcing package pinning or removing any MCP package version, capture 'npm list --json' output and the full contents of all tool description manifests (description fields) from currently running MCP servers. If postmark-mcp or GitHub MCP server is present, archive the exact installed version and its tool descriptions before replacement, as these constitute forensic evidence of whether the supply chain incident affected this environment. Do not uninstall or update packages before this capture is complete.

Step 5: Communicate findings — brief leadership on the confirmed supply chain incident and the 72.8% empirical success rate reported by Microsoft; frame the risk in terms of enterprise data accessible to deployed AI agents, not as a generic AI vulnerability

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: communicating findings, sharing intelligence, and improving organizational awareness to drive risk-informed decisions

Controls: NIST AC-1 (Policy And Procedures)

Compensating: Prepare a one-page executive brief using the blast radius matrix from Step 3 as the quantitative anchor: translate 'AI agent access to SharePoint' into 'up to X GB of sensitive documents accessible per compromised agent session.' Reference the Microsoft Incident Response confirmation of the postmark-mcp supply chain incident as the concrete evidence that this threat is active, not theoretical. No tooling required — the deliverable is a decision memo requesting formal policy approval for MCP change-control procedures and agent permission reviews.

Evidence: This step does not alter live state and requires no volatile capture. Retain all documentation produced in Steps 1–4 (asset inventory, blast radius matrix, audit log exports, tool description archives) as the evidentiary package supporting the leadership brief. These records also serve as the pre-remediation baseline for any future audit or regulatory inquiry related to AI agent data handling.

Step 6: Monitor developments — track Microsoft Incident Response and Defender research publications for follow-up guidance, updated detection rules, and any patches or architectural changes to MCP server implementations

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: integrating threat intelligence and vendor guidance to improve future detection and response capability

Controls: NIST AU-6 (Audit Record Review, Analysis, And Reporting), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Subscribe to the Microsoft Security Response Center (MSRC) RSS feed and the Microsoft Defender Threat Intelligence blog for MCP-related advisories. Set a Google Alert for 'MCP tool poisoning' and 'Model Context Protocol security.' Assign one team member to review these sources weekly and flag any new detection logic (Sigma rules, KQL queries for Microsoft Sentinel, or Defender for Cloud alerts) targeting MCP tool description anomalies. When new detection rules are published, test them against the archived tool description manifests captured in Step 4 to validate local applicability.

Evidence: This is an ongoing monitoring step and does not alter live state. Maintain the tool description manifest archive from Step 4 as a rolling baseline — when Microsoft publishes indicators of compromise or detection signatures related to MCP poisoning, compare them against this archive to determine retroactive exposure. Log all monitoring activities and vendor advisory reviews with timestamps to support future audit trails under NIST AU-11 (Audit Record Retention) requirements.

Detection Guidance

Standard behavioral detection is complicated by the nature of this attack: each agent action appears legitimate in isolation. Hunting should focus on anomalies in aggregate agent behavior rather than individual events. Relevant log sources include Microsoft 365 Copilot audit logs, Azure AI Foundry activity logs, and any API gateway or CASB logs sitting in front of MCP-connected services. Hunt for: AI agent sessions that access an unusually broad set of data sources or file paths relative to the agent's defined function; outbound data transfers initiated by an AI agent to external endpoints or web services (T1567, T1071.001), particularly via postmark-mcp or other email/webhook-capable MCP packages; changes to MCP tool description fields in server configurations, especially changes made outside of a documented change window; new or unrecognized MCP packages loaded into agent environments, including npm packages not present in your approved software inventory (CIS 2.1, CIS 2.3); agent sessions that retrieve credentials, secrets, or configuration data not required by the agent's stated purpose (T1552). Enable and retain audit logs for all AI agent actions per NIST AU-2 and AU-11 to support post-incident reconstruction. Apply least-privilege access controls (NIST AC-6, CIS 5.4) to limit what data repositories and APIs each agent identity can reach, reducing blast radius if a tool description is poisoned. Extend account monitoring principles to AI agent service accounts to detect privilege misuse. The cited Microsoft Incident Response and Defender research may include specific indicators; consult the Microsoft Security Blog and Defender Threat Intelligence for any published IOC values associated with the postmark-mcp supply chain incident.

Framework Mappings

MITRE-ATTACK

- **T1190** — Exploit Public-Facing Application
- **T1059** — Command and Scripting Interpreter
- **T1552** — Unsecured Credentials
- **T1565** — Data Manipulation
- **T1567** — Exfiltration Over Web Service
- **T1195.001** — Compromise Software Dependencies and Development Tools
- **T1036** — Masquerading
- **T1071.001** — Web Protocols
- **T1195.002** — Compromise Software Supply Chain

NIST-800-53R5

- **CA-8** — Penetration Testing
- **RA-5** — Vulnerability Monitoring and Scanning
- **SC-7** — Boundary Protection
- **SI-2** — Flaw Remediation
- **SI-7** — Software, Firmware, and Information Integrity
- **CM-7** — Least Functionality
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring

- **SA-9** — External System Services
- **SR-3** — Supply Chain Controls and Processes
- **SI-10** — Information Input Validation
- **AC-3** — Access Enforcement
- **AT-2** — Literacy Training and Awareness
- **SR-2** — Supply Chain Risk Management Plan

OWASP-TOP10-2021

- **A03:2021** — Injection
- **A01:2021** — Broken Access Control

CIS-V8

- **16.10** — Apply Secure Design Principles in Application Architectures
- **6.1** — Establish an Access Granting Process
- **6.2** — Establish an Access Revoking Process
- **14.2** — Train Workforce Members to Recognize Social Engineering Attacks
- **15.1** — Establish and Maintain an Inventory of Service Providers
- **8.2** — Collect Audit Logs

ISO-27001-2022

- **A.8.26** — Application security requirements
- **A.5.34** — Privacy and protection of personal information
- **A.5.21** — Managing information security in the ICT supply chain

SOC2-TSC

- **CC6.1** — The entity implements logical access security software, infrastructure, and architectures over protected information assets
- **CC7.4** — Responds to identified security incidents
- **CC9.2** — Manages risks associated with vendors and business partners

HIPAA-SECURITY

- **164.312(a)(1)** — Access Control
- **164.308(a)(6)(ii)** — Response and Reporting

NIST-CSF-2

- **GV.SC-01** — Cybersecurity supply chain risk management program
- **DE.CM-01** — Networks and network services are monitored
- **DE.AE-08** — Incidents are declared when adverse events meet the defined incident criteria

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1190	Exploit Public-Facing Application	Initial-Access
T1059	Command and Scripting Interpreter	Execution
T1552	Unsecured Credentials	Credential-Access
T1565	Data Manipulation	Impact
T1567	Exfiltration Over Web Service	Exfiltration
T1195.001	Compromise Software Dependencies and Development Tools	Initial-Access
T1036	Masquerading	Defense-Evasion
T1071.001	Web Protocols	Command-And-Control
T1195.002	Compromise Software Supply Chain	Initial-Access

Sources

Source	URL	Tier
Security News	https://thehackernews.com/2026/06/microsoft-warns-poisoned-mcp-tool...	T2
Connect to MCP Server Endpoints for agents	https://learn.microsoft.com/en-us/azure/foundry/agents/how-to/tools...	T1
Introducing Model Context Protocol (MCP) in Copilot Studio	https://www.microsoft.com/en-us/microsoft-copilot/blog/copilot-stud...	T1
Encountering errors while setting up Sentinel MCP Triage ...	https://learn.microsoft.com/en-us/answers/questions/5858636/encount...	T1

Generated 2026-07-01 07:07 UTC by TJS Security Command Center