Security Risks and Best Practices for Implementing the Model Context Protocol

Bailey Williams

College of Science, Department of Computer Science

CS 463: Cryptography for Cybersecurity

Dr. Jonathan Takeshita

December 5, 2025

The Model Context Protocol (MCP) provides a standardized connection between Large Language Models (LLMs) and external services (Karimova & Dadashova, 2025, p. 50). Prior to the introduction of the MCP in 2024 by Anthropic, developers seeking to integrate external services with LLMs had to develop individual tool-call programs for each LLM - external service combination (*Introducing the Model Context Protocol*, n.d.). In contrast, the MCP allows developers to develop integrations using a single framework, which can be used with any LLM capable of performing tool-calls, making it much easier to connect external services. While the MCP standard is designed to be a secure way to standardize connections, there are still known security risks and vulnerabilities.

The MCP was conceived by United States-based Artificial Intelligence (AI) startup Anthropic in 2024. In their announcement, Anthropic released the specifications and Software Development Kits (SDKs) and open-source MCP server repositories, alongside support for using MCP with their own model family, Claude (*Introducing the Model Context Protocol*, n.d.). As previously described, this release enabled developers to standardize how external services would integrate with LLMs. The peer-reviewed papers published since overwhelmingly focus on explaining the specifications of the protocol, comparing the MCP to other forms of integration services, and discussing the security advantages and pitfalls of the protocol (Karimova & Dadashova, 2025; Yu et al., 2025; Ntousakis et al., 2025).

In *The model context protocol: a standardization analysis for application integration* by Sevinj Karimova and Ulviya Dadashova, the researchers analyzed the current architecture and implementation of the MCP protocol. The research team found that there were numerous development, modularity, and security benefits to implementing the protocol, as it provides better cross-client compatibility compared to direct integration or framework-specific

implementations. While the research team found that the protocol has strong potential, they emphasized the need for wider usage to further validate these findings.

There have been four released versions of the MCP specification. The first one, version 2024-11-05, lays out the fundamentals of the protocol which have been built upon since then. The following paragraph is based on this version, changes between this version and the latest one, version 2025-11-25, will be discussed in the paragraph after.

The MCP standard uses JSON-RPC 2.0 messaging to communicate with the three parts of the protocol: hosts, clients, and servers. The MCP host is the LLM application that initiates the connection, the client is the connector within the LLM application that assists in the connection, and the server provides context to the host (*Specification (Version 2024-11-05)*, n.d.). The context provided could include data for the LLM to then use, prompts for users to use with the LLM, or tools that the LLM can then execute. The host may allow the server to initiate agentic behavior, however this should be explicitly approved by the user prior to initiation. Servers should operate in isolation from the rest of the LLM application and other servers to prevent any cross-contamination or exposure to data they should not have access to. Refer to the Appendix for a flow chart created by Anthropic, which provides a visual representation of the interactions between hosts, clients, and servers. In this initial version, there is no authentication or authorization specification, however there is emphasis on the need for human input and approval of actions taken by the different parts of the MCP protocol.

Since the first release of the MCP protocol, there have been three updates. Major updates have centered around strengthening security measures, including the addition of an authorization framework using OAuth 2.1, replacing HTTP+SSE with streamable HTTP transport for added flexibility, and tool annotations to better understand what type of behavior they do in version

2025-03-26 (*Key Changes (Version 2025-03-26 Changelog)*, n.d.). Version 2025-06-18 helped strengthen the protocol further including safeguards to prevent malicious MCP servers from obtaining real access tokens and developing security best practices, which cover various known attacks and mitigation efforts and will be discussed in more detail later on (*Key Changes (Version 2025-06-18 Changelog)*, n.d.). The latest version has added incremental scope consent to authorization flows and worked to further enhance authorization workflows (*Key Changes (Version 2025-11-25 Changelog)*, n.d.). These updates all work towards strengthening the framework to ensure that it becomes a secure specification guide.

The MCP implementation specification released by Anthropic requires following the OAuth 2.1 security best practices (*Authorization*, n.d.). This includes ensuring authorization occurs over HTTPS and requires redirect URIs to be either localhost or over HTTPS. In addition, MCP clients are required to include the resource parameter, which is then validated by the corresponding MCP server to ensure that the authorization tokens used by the client were issued for that specific use. These measures help ensure that authorization is dynamically verified by MCP servers so that a compromised client is unable to exploit old authorization tokens to make invalid requests appear valid.

In *A Survey on Agent Workflow – Status and Future*, by Chaojia Yu, Zihan Cheng, Hanwen Cui, Yishuo Gao, Zexu Luo, Yijin Wang, Hangbin Zheng, and Yong Zhao, their investigation of MCP implementation was focused on the security risks the protocol poses to AI workflows. They discussed the risks of malicious tool calls and poisoned or fake MCP servers. For malicious tool calls, the attacker changes the description of the tool call to include hidden instructions after the MCP server has been authorized by the end user. The server may then be instructed to steal user data or some other malicious action (Yu et al., 2025). The most common

instance of poisoned or fake MCP servers is the registration of MCP servers by attackers with names similar to the real server to trick end users into installing them instead. There is also the risk of an MCP server becoming compromised by a malicious update. Since these MCP servers are either masquerading as trusted servers or become compromised after installation and authentication by the end user, the attacker is able to circumvent the OAuth 2.1 authentication entirely.

To protect against data leakage from MCP attacks, the paper *Securing MCP-based Agent Workflows* by Grigoris Ntousakis, Julian James Stephen, Michael V. Le, Sai Sree Laya Chukkapalli, Teryl Taylor, Ian M. Molloy, and Frederico Araujo propose the SAMOS system as a solution. SAMOS works at the gateway level, intercepting MCP tool calls and ensuring that security policies are followed. SAMOS requires additional customization during the setup process to ensure that the appropriate metadata is included in each tool call. Once configured, the system then inspects each tool call made to identify potentially malicious ones based on the end-user's policies.

Based on Anthropic's specification and the related works outlined above, the following best practices are recommended when implementing the MCP protocol. First, when connecting premade MCP servers, ensure that the servers come from legitimate sources. If your company is implementing MCP servers within the product or development lifecycle, there should be a policy on evaluating MCP servers to ensure that illegitimate ones are not used. Second, tool calls should be limited to those that are required for your use cases, extraneous tool calls consume the LLM's context window and expand the potential attack surface. Third, MCP servers should be updated with caution and only after validating the update does not introduce malicious code. Fourth, companies utilizing the MCP protocol should provide LLM-specific security awareness training

to end-users so that they understand different attacks to watch out for and know the importance of verifying tool calls the LLM application is requesting to make, rather than blindly allowing all calls to be made. Finally, security teams should audit MCP usage within the organization to monitor for malicious behavior.

The MCP protocol has the potential to span industries with its versatility in MCP server offering. Anthropic has built the capability to connect MCP servers to Claude Code, allowing developers the ability to pull from external sources, including work tickets from platforms like Linear or Jira and databases from platforms like Postgres (*Connect Claude Code to Tools via MCP*, n.d.). This allows developers to stay on the same window for longer, eliminating the need to constantly switch tabs and dig through websites to find relevant information. Anthropic also claims that thousands of MCP servers have been built by members of the open-source community, enabling rapid adoption across industries (*Code Execution with MCP*, n.d.).

Anthropic's competitor, OpenAI, has also developed their own agent builder that allows the use of MCP servers, however these options are fairly limited at this time (*Connectors and MCP Servers*, n.d.). Future developments in the field include working to further secure tool calls to prevent known attacks and the continued expansion throughout industries. To provide added security, the implementation of a trusted MCP server directory may be a good start, however this would be challenging for an open-source project. It is likely that companies will instead develop their own MCP servers for integrating tools to use internally, rather than having to validate external servers.

In conclusion, the MCP protocol allows for the integration of external services with LLM applications. This has enabled the standardization of integrations for LLMs, providing a way for any applicable LLM to connect to any service through its client-server architecture. This is not

without its vulnerabilities, despite the requirement of OAuth 2.1 authentication, there are still attacks able to circumvent this and steal end user data. While there has been rapid adoption of the protocol throughout multiple industries, future work includes strengthening the standard to address known attacks. In the meantime, best practices include ensuring MCP servers are from legitimate sources and carefully updating them to avoid compromising trusted servers.

References

Authorization. (2025, June 18). Model Context Protocol. https://modelcontextprotocol.io/specification/2025-06-18/basic/authorization

Code execution with MCP: Building more efficient AI agents. (n.d.). Retrieved December 5, 2025, from https://www.anthropic.com/engineering/code-execution-with-mcp

Connect claude code to tools via MCP. (n.d.). Claude Code Docs. Retrieved December 5, 2025, from https://code.claude.com/docs/en/mcp

Connectors and MCP servers. (n.d.). OpenAI Platform. Retrieved December 5, 2025, from https://platform.openai.com

Deng, Z., Guo, Y., Han, C., Ma, W., Xiong, J., Wen, S., & Xiang, Y. (2025). Ai agents under threat: A survey of key security challenges and future pathways. *ACM Computing Surveys*, *57*(7), 1–36. https://doi.org/10.1145/3716628

Introducing the model context protocol. (n.d.). Anthropic. Retrieved November 24, 2025, from https://www.anthropic.com/news/model-context-protocol

Karimova, S., & Dadashova, U. (2025). The model context protocol: A standardization analysis for application integration. *UNEC Journal of Computer Science and Digital Technologies*, *1*(1), 50–59. https://doi.org/10.30546/UNECCSDT.2025.01.050

Key changes (version 2025-03-26 changelog). (n.d.). Model Context Protocol. Retrieved December 5, 2025, from https://modelcontextprotocol.io/specification/2025-03-26/changelog

Key changes (version 2025-06-18 changelog). (n.d.). Model Context Protocol. Retrieved December 5, 2025, from https://modelcontextprotocol.io/specification/2025-06-18/changelog

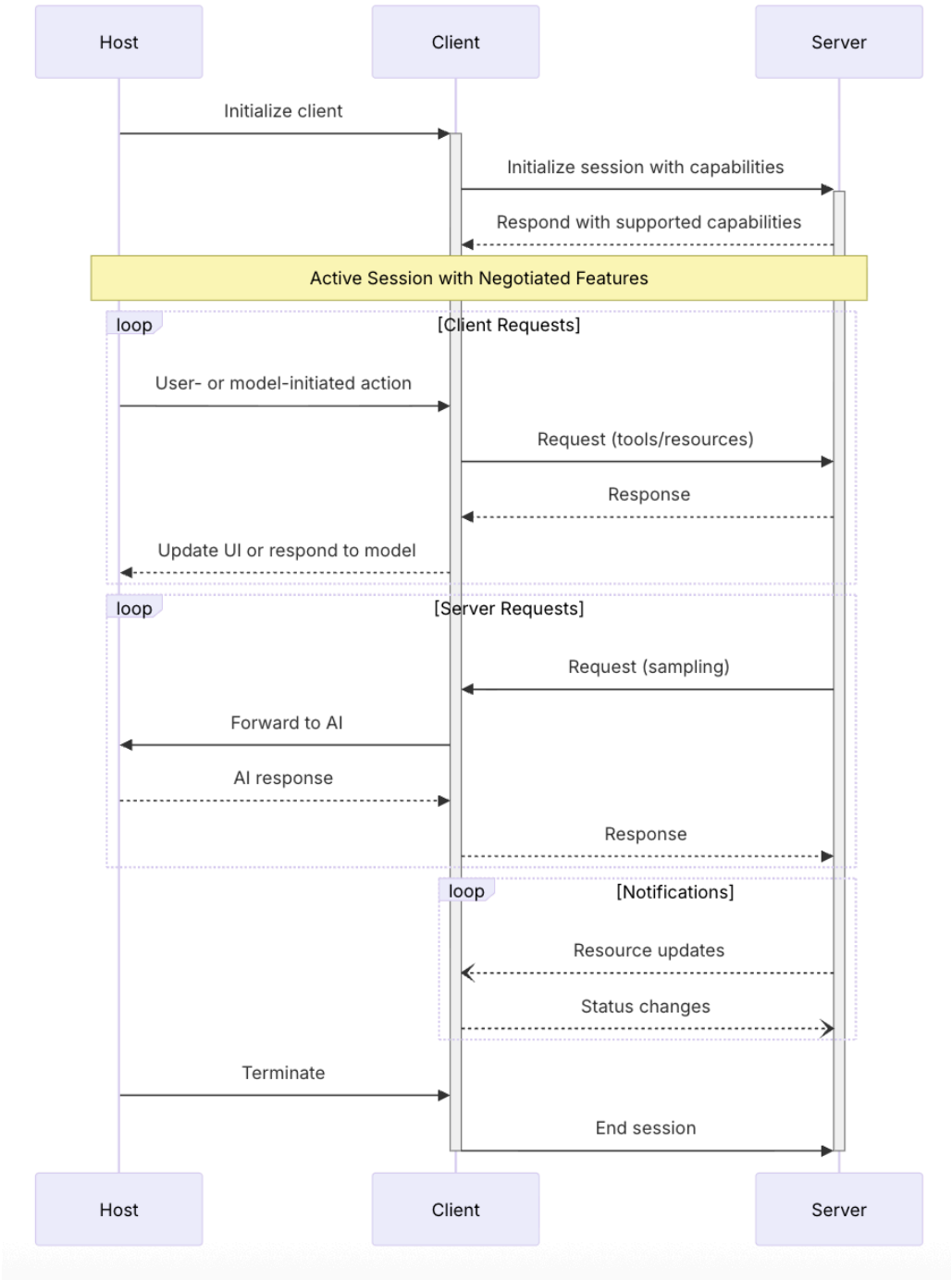
Key changes (version 2025-11-25 changelog). (n.d.). Model Context Protocol. Retrieved December 5, 2025, from https://modelcontextprotocol.io/specification/2025-11-25/changelog

Ntousakis, G., Stephen, J. J., Le, M. V., Chukkapalli, S. S. L., Taylor, T., Molloy, I. M., & Araujo, F. (2025). Securing mcp-based agent workflows. *Proceedings of the 4th Workshop on Practical Adoption Challenges of ML for Systems*, 50–55. https://doi.org/10.1145/3766882.3767177

Specification (version 2024-11-05). (n.d.). Model Context Protocol. Retrieved December 5, 2025, from https://modelcontextprotocol.io/specification/2024-11-05

Yu, C., Cheng, Z., Cui, H., Gao, Y., Luo, Z., Wang, Y., Zheng, H., & Zhao, Y. (2025). A survey on agent workflow – status and future. *2025 8th International Conference on Artificial Intelligence and Big Data (ICAIBD)*, 770–781. https://doi.org/10.1109/ICAIBD64986.2025.11082076

Appendix



Note. A visual depiction of the host-client-server relationship in the MCP protocol. From Anthropic's Model Context Protocol Authorization specification (modelcontextprotocol.io/specification/2024-11-05/architecture).