A Survey on Model Context Protocol (MCP) in System Calls

Abhijeet Cholke[1]*, Vinodpuri Gosavi[2], Rohini Patil[3], Mandar Mokashi[4], and Arati Kale[5]

[1,2,4,5] School of Computing, MIT Art, Design and Technology University, India
[3] Department of Computer Science & Engineering, Bharati Vidyapeeth (Deemed to be University) College of Engineering, Pune, India
*abhijeet.cholke@gmail.com
vinodpuri.gosavi@mituniversity.edu.in
patilrohini2402@gmail.com
mandar.mokashi16@gmail.com
arati.kale@mitniversity.edu.in

Abstract. Recently, a new standardized interface called the Model Context Protocol (MCP) has been established which allows reliable interfacing of large language models (LLMs) with the tools and external resources. Due to its structured architecture, it can be analogously termed to system calls in operating systems, since both work as controlled intermediatory between the low-level execution environments and user-level requests. This survey will provide a thorough overview of the literature on MCP with the focus on its system call analogy, security issues, benchmarking and its usage on AI based workflows. We categorize the existing work into four major classifications, namely the security/safety aspect, the MCP server and benchmarking aspect, the investigation on the tool/function calling aspect and the system level observability aspect. In addition to this, we also indicate the open research challenges like the scalability aspect, the interoperability aspect and the need for formal verification. Finally, we remark on the future prospects of the research where MCP should be regarded as an `"syscall layer" for artificial intelligence" serving as the basic building block of the AI-native operating systems and also the multi-agent platforms.

Keywords: Model Context Protocol (MCP), LLM Agents, System Calls Analogy, AI Integration Protocols, Research Maturity Models.

1 Introduction

The rapid development of Large Language Model (LLM) has resulted in agentic systems, such as reasoning, planning, multi-turn interactions with external tools. Such an evolution was partially supported by the Model Context Protocol (MCP), designed to act as a bridge to allow for plug-n-play integration of LLMs with other external services, tools and knowledge-bases. MCP has become known as the “system call

M. Kaushik et al. (eds.), *Proceedings of the International Conference on Responsible, Risk-aware, and Regulated AI (RRRAI 2026)*, Advances in Intelligent Systems Research 210,
https://doi.org/10.2991/978-94-6239-723-1_14

layer" that abstracts modern AI agents [6][11][14] providing portability and modularity, but introducing new challenges in security and governance.

The work being done on the MCP and related tool-use paradigms is multi-front. function calling: improved with concepts such as multilingual pipelines, or decision tokens [2][12], synthetic tool-use data generation (ToolACE) [4] and light-weight implementations on small language models (SLMs) [10]. Related work on agentic architectures examines practical design questions including planning and memory management, as well as infrastructure for serving large-scale agent programs including high-throughput systems such as Autellix [3][5][13][14][15].

Simultaneously, MCP has serious security shortcomings. Studies focused on attacks highlight the security problems caused by function calling [1], insufficiently secure or maliciously configured MCP servers [6][18], and mass poisoning of tools (MCPTox) [20]. To mitigate the damage from these attack vectors, defensive works introduce safety scanners [7], zero-trust enterprise systems [8], observability systems (AgentSight) [17], and taxonomies of agent threats [16]. Collectively, these works demonstrate that MCP is doubly dangerous, as a possible source of capabilities and an attack target.

In this survey, we present an overview of 21 papers giving the synthesis of works on various aspects of MCP methodology, architecture, security, benchmarks and governance. We categorise the papers around 7 themes (function calling, the design of agentic systems, security (attacks and defences), benchmarks, enterprise/govenance, standards/interoperability, adjacent system level work).

2 Literature Survey

The authors analyzed security vulnerabilities in MCP, identifies threats such as prompt injection, malicious servers and protocol abuse, and proposes multi-layered defenses such as request validation and anomaly detection; However, validation is limited to simulations [1]. This was an agentic performance assessment of MCP. environment, emphasizing better interoperability, but more overhead at high. and presents a reusable benchmarking approach, but is only experimented in controlled settings [2]. The work gave a formal definition of MCP in the form of model. verifying that AI workflows are correct and safe, yet only to small scales. examples that cannot be validated in the real world [3]. ToolACE introduced an internally developed data. generation pipeline optimizing LLM function-calling with API of large scale. synthesis and verification, with performance that is comparable to state-of-the-art. models in zero-shot tasks [4]. Multi-turn tool was enhanced by MINT framework. interaction in LLM with dynamic dialog generation, adaptive planning and error. correction, showing superior performance over existing approaches [5].

Toolbench introduced a full benchmark and dataset to test and train LLMs on real-world APIs to use tools in single and multi-turn, which greatly enhanced API selection and accuracy of execution. Nevertheless, there are still difficulties in the generalization of invisible tools and ambiguous intent of the user [6]. This paper has identified significant security vulnerabilities in MCP as risks of arbitrary code execution and

stealing credentials, and demonstrated how LLM can evade security guards; He has suggested McpSafetyScanner as a preventive auditing tool on MCP servers [7]. The paper suggested a zero-trust-based defense-in-depth architecture aimed at securing MCPs, which consists of multi-layer threat modeling and mitigation measures, but it is characterized by the complexity, performance overheads, and a lack of real-world validation [8]. Through MCPBench, the paper assessed the performance and resource efficiency of MCP servers, finds that there is a wide range of variability in accuracy across implementations, and remains uncertain whether MCP in general performs better than simple methods of function-calling [9]. This paper discussed the application of MCP to multi-agent systems and suggested a reference architecture that enhances coordination and integration of tools, however the authors pointed out that MCP has limitation of scalability, security robustness and cannot be compared with other protocols [10]. In the current paper, the authors examined how the Model Context Protocol (MCP) can be used to improve retrieval-augmented generation (RAG) pipelines by offering a consistent way of accessing external knowledge sources. Their work evaluates the performance of MCP-enabled RAG on factual grounding tasks by comparing them to traditional retrieval plugins and APIs. The findings indicate that incorporation of MCP in a pipeline makes it easier to design and more consistent in processing retrieval despite the fact that the enhancement in accuracy may not be very significant. The authors mention that the MCP may coordinate the heterogeneous knowledge tools and also restrictions in latency and partial coverage of domain specific information and lack of real world implementation are also possible [11].

The Paper introduced the Model Context Protocol (MCP) that extended modern Transport Layer Security (TLS) in order to deliver cross-platform AI ecosystem interoperability. In that direction, they are working on enabling the transparent composition of a variety of tools, APIs and services across standardized interfaces that reduce the amount of engineering effort needed to connect proprietary connectors. The paper explains how the MCP can be used in rapid deployment of applications like healthcare data and financial analytics on application areas. Although the research postulates that MCP could be applied to bring about effective diffusion in the multi-tool AI applications, it also presents problems such as ineffective error handling, non-coherent governance frameworks, and the inability to scale to business level workloads [12].

The paper balanced the risks of implementing Model Context Protocol (MCP) to millions of AI systems along the safety and compliance axes with regard to the ethical considerations. The paper also explains why MCP where it manages to facilitate the agentic workflow, will nevertheless leave systems exposed to the risk of leaking by violating privacy, mishandling sensitive information, or non-compliance unless it is appropriately protected. We present a risk assessment model, which categorizes the threats according to their technical, organizational, and legal levels and offers guidelines to the safe implementation of MCP. But the analysis so far is mostly theoretical, and there is very little empirical support or modelling of implementation in reality [13].

This paper investigated how the Model Context Protocol (MCP) can be extended for domain-specific applications, in particular for healthcare and learning. They showcase

dedicated MCP server applications in the sphere of electronic health records (EHRs) and adaptive learning, and reveal the benefits of standardized access as enhanced integration and functionality even in special-purpose domains. Experiments conducted indicate that MCP reduces the integration overhead as compared to traditional tailored APIs, thus enabling the quick roll-out of solution-based deployment with AI. Despite this, the study notes that the work is bounded through obligations to adhere to compliance only at the domain level, potential threats due to the sensitivity of the data involved, and the lack of any scale-based performance metrics [14].

The paper presents a study on the potential of making autonomous AI agents available in scientific discovery workflows via the Model Context Protocol (MCP). They present MCP-enabled pipelines combining simulation tools, databases, and visualization systems and enable the agents to perform automated hypothesis testing and data analysis. In their material science and drug discovery cases they demonstrate more automation and less cost of integration than their previous orchestration-based protocols. However, the study cites limitations such as the absence of sufficient robustness testing, reliance on the quality of the domain-specific tools, and non-scalability to highly complex research environments [15]. Table 1 presents a comparative analysis of ten additional research papers.

Table 1: Comparison of Research Papers

Authors	Year	Focus Area	Key Contribution	Limitations
Gupta et al.[16]	2025	MCP for Explainable AI	Introduced an MCP-based framework for explainability, provenance tracking, and compliance auditing in AI workflows	Limited explanation method coverage, performance overhead, untested in large-scale high-stakes deployments
Zheng et al.[17]	2025	Observability for AI Agents	Developed *AgentSight* using eBPF to bridge semantic gap between LLM intent and system actions; detects prompt injections, reasoning loops,	Dependent on LLM-based semantic analysis; potential latency; lacks validation in large-scale production
Song et al.[18]	2025	Security Risks in MCP	First large-scale empirical study of MCP security, revealing vulnerabilities across 22 servers; proposed security framework for safer MCP deployment	Recommendations not yet tested in production-scale systems; gaps in real-world validation
Patel et al.[19]	2025	Human–AI Collaboration with MCP	Proposed an MCP-based framework for enterprise collaboration, showing improvements in workflow efficiency and integration	User adaptation challenges, possible workflow disruptions, and lack of long-term validation studies

Huang et al.[20]	2025	Defenses for MCP Security	Proposed MCP-Guard, a runtime monitoring and policy framework reducing prompt injection & tool misuse risks	Relies on predefined rules; limited adaptability against novel/adaptive attacks
Zhiqiang et al.[21]	2025	Privacy & Data Protection	Proposes privacy-preserving MCP mechanisms (encryption, differential privacy, access control)	Added overhead; lacks production scale evaluation

A synthesis of 21 research articles on the concept of Model Context Protocol (MCP) is provided in Table 1 and spans the research areas of LLM-enabled tool use, agentic system design, security, benchmarking, and enterprise governance. The comparison identifies essential contributions such as improvements to the functionality of calling, scalable agent structures, and the standardization of evaluation frameworks, along with major gaps of real-world robustness, safety controls and interoperability. The most prevalent theme is security as there has been extensive work done on attack classification and defense mechanisms but there is a lack of practical deployment and verification of these measures. All in all, the analysis indicates that strong benchmarks, better safety frameworks, and massive empirical validation are essential to effective MCP adoption. The research is dominated by security, and a large fraction of studies are centered on attacks, defense and auditing, and research on emerging benchmarks shows that underlying vulnerabilities of current LLMs in MCP-driven tasks are fundamental. Moreover, the infrastructure and design studies make the MCP an underlayer to the LLM agents, and although it can be applied across all domains, there are still issues of integration, governance, and real-life implementation.

3 Model Context Protocol (MCP): An Overview

Core Architecture

MCP follows a client–server model:

MCP Client (e.g., an LLM or agent framework): Initiates requests, manages context, and interprets results. **MCP Server:** Features are made available as tools, resources, and prompts that the client can call upon. **Communication Channel:** It is a JSON-RPC protocol over a communication channel with synchronous communication and streaming error and response management. This isolation allows modularity, you can add a new server, i.e., a file system explorer, a financial data retriever, or a code execution sandbox, to any MCP compliant client, without rewriting the integrations.

Capabilities

Tools are the functions that perform an action, such as executing code, performing a web search, or interacting with APIs. Resources used as persistent external data stores such as documents, databases, or vector indexes. Prompts served as predefined instruction templates that can be reused for common workflows.

Applications

MCP also brings about new security risks like Over-broad tool permissions can expose sensitive resources, malicious or poisoned servers can influence agent reasoning and lack of observability makes debugging and compliance harder. These threats have inspired several defence strategies, such as auditing structures (McpSafety Scanner), observeability layers (AgentSight), zero-trust designs, and attack taxonomies (MCPLIB, MCPTox).

Security & Challenges

MCP introduces new security risks such as Over-broad tool permissions can expose sensitive resources, malicious or poisoned servers can manipulate agent reasoning, and lack of observability complicates debugging and compliance. These risks have motivated multiple defence directions, including auditing frameworks (McpSafetyScanner), observability layers (AgentSight), zero-trust designs, and attack taxonomies (MCPLIB, MCPTox).

Evolution and Outlook

MCP remains immature as system calls have developed over decades with stable interfaces, auditing tools and security models. Current research is on Establishing robust defaults (least-privilege, signed manifests), designing benchmarks to execute based evaluation (MCPBench, MCP-Universe), improving interactions with other protocols (ACP, ANP, A2A), developing enterprise-ready governance frameworks to ensure long term sustainability. In this direction, MCP will probably be the default standard of LLM-to-tool integration, just as system calls are the basis of OS-application communication.

4 MCP and System Call Analogy

The comparison of MCPs and system calls accentuates their similarity as an. interface layer, at which point higher-level entities specify the specific functions of the lower-level. standardized contracts of services. Both are abstract and portable, system. OS-level complexities are concealed by calls, and MCP allows the LLM agents to communicate with each other transparently. with a wide range of external tools without lock-in with vendors. They also act as critical security data, boundaries, which need rigid validation and control processes to avoid misuse. leakage or unsecure execution. System calls have grown to become stable and well-governed. and auditing tools, MCP continues to develop and needs the same in. standardization, observability and robustness testing.

Table 2: Analogy od SysCalls & MCP

OS System Calls	MCP Requests
Kernel mediates access to hardware	MCP server mediates access to tools
Provides abstraction for developers	Provides schema for AI models

Requires secure sandboxing	Requires safe execution & audits
Monitored via tracing tools (eBPF, strace)	Audited via MCP safety scanners

Recent research also makes a case in favor of observability methods that integrate syscall-level monitoring with AI interactions, which supports the similarities between the two. Both system calls and MCP requests, as demonstrated in Table 2, serve as an intermediary and offer an abstraction layer and enforce security boundaries and, therefore, are controlled and secure interactions with underlying resources. Moreover, since the system calls could be monitored with the help of tools like STRESS and eBPF, MCP is based on auditing tools like safety scanners that ensure the integrity and reliability of AI activities.

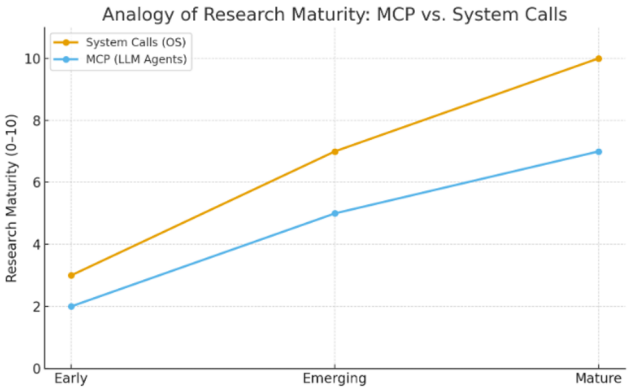

Fig 1. Analogy of SysCalls & MCP

As Figure 1 indicates, the system calls of operating systems (orange line) demonstrate a steady high research maturity in early, emerging, and mature stages, which are indicative of the long-standing nature of their development. On the contrary, the MCP (blue line) of the LLM agents demonstrates a growing yet relatively low level of maturity which means that it is still developing. On balance, the figure indicates that MCP traces the same developmental path as system calls, albeit on an earlier developmental path.

5 TAXONOMY OF RESEARCH ON MCP

Some of the main areas that the literature can be classified into are security attacks, defense and governance mechanisms, tool-exploitation methods, benchmarking, and basic architecture studies of MCP-based systems. The works that are security-oriented prevail, with an emphasis on attack classification, tool poisoning, auditing frameworks, zero-trust architectures, and observability solutions, and methodological works are discussing more sophisticated functions-calling techniques, including Decision Token and ToolACE. Also, benchmarks like MCP-Universe and MCPBench, design-oriented

surveys and protocol studies, are crucial sources of information about the evaluation, interoperability, and changing architecture of agentic LLM systems..

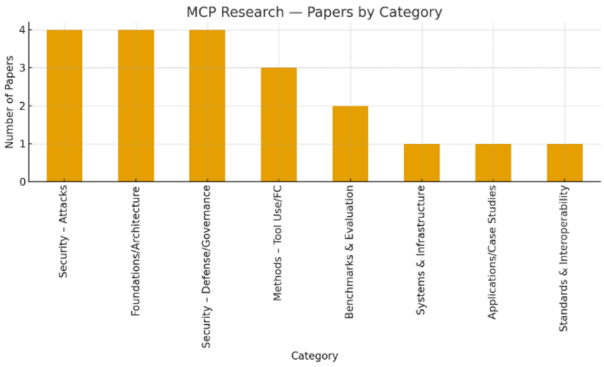

Fig 2. Category Wise Papers

In Figure 2 The chart shows that the research on MCP is concentrated mostly in such areas as security (attack and defense/governance) and basic architecture which means high attention is paid to the issues of system design and security. The methodological research in the use of tools and their use calling is also quite important, albeit somewhat to a lesser degree. Conversely, other areas that have been relatively under-researched include benchmarking, infrastructure, applications and interoperability implying that such areas are either new or not well developed areas of research.

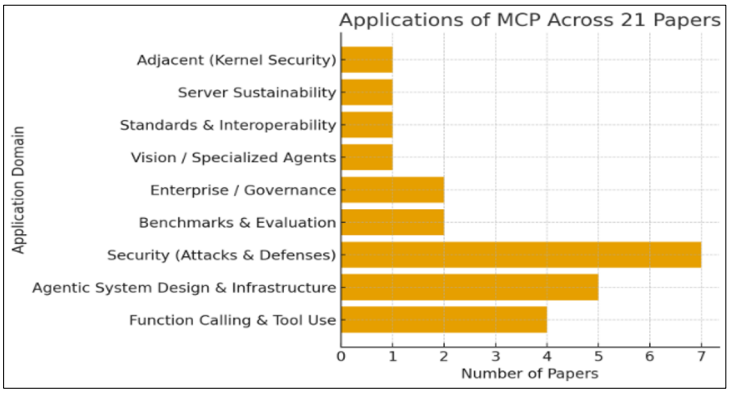

Fig 3. Applications

As Figure 3 indicates, the most significant area of MCP research application is "Security (attack and defense), then there is "Agent system design and infrastructure" and then there are "Function calling and tool use" with a strong emphasis on system security and core architectural development. Comparatively, on the other hand, in other fields, benchmarking, enterprise governance, interoperability, and specialized applications, there are relatively fewer studies, meaning that they are new or under-

researched. In general, the distribution indicates an orientation of current MCP studies towards safety and underlying design, as opposed to practical and operational spheres.

6 FUTURE RESEARCH DIRECTIONS

Future studies of MCP ought to focus on protocol-level security features like least privilege enforcement, fine-grained authorization, and a secure tool manifest to minimize threats like tool poisoning and unauthorized access. More sophisticated runtime observability, provenance tracking, and execution-based benchmarking frameworks, which can assess beyond performance safety, robustness, and adversarial resilience, are also highly demanded. Moreover, work should be done to ensure secure defaults, server health checking, tool description sanitization, and explainable decisions to minimize vulnerabilities and enhance trust in the agent action. Lastly, interoperability, lightweight/edge deployment, human-in-the-loop governance, and OS-level sandboxing will be essential to create a scalable, secure and enterprise-ready MCP ecosystem.

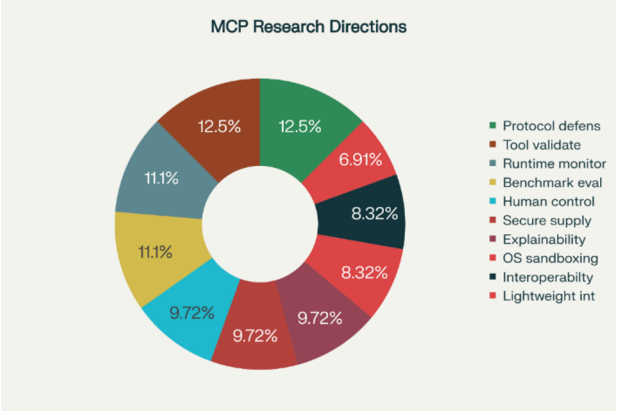

Fig 4. Applications

Figure 4 presents the distribution of research areas in MCP, where protocol-level protection and device validation are the most popular areas, with each having 12.5 percent, and a focus on security and device integrity. Much attention is also paid to runtime monitoring and benchmarking, which proves the relevance of the assessment and management of MCP systems in practice. Middle-weighted regions like human-in-the-loop control, persuasiveness and secure supply chains indicate the emerging concerns about transparency and governance. Lightweight integration, OS sandboxing, and interoperability on the other hand have been less discussed and thus are still in their research phases.

7 CONCLUSION

The Model Context Protocol (MCP) is a significant move towards the ability to bridge the gap between the reasoning ability of large language models and real-world behavior and is moving beyond a simple interoperability layer to basic abstractions like system calls that allow structured interaction with tools, data, and services. The literature review shows that it is broadly applicable in the orchestration of functions, multi-agent systems, security analysis and benchmarking, not to mention its dual nature as an enabler of advanced capabilities and a possible attack surface. Regardless of these developments, MCP is still at an early development stage, and it needs to be standardized, version-stable, with a solid security platform, and well-managed, including human-in-the-loop monitoring. Key research areas include protocol-level security, sound evaluation methodologies, runtime observability, and safe integration. In general, MCP has great opportunities to be turned into the de facto standard of tool-integrated AI systems, though innovation needs to be accompanied by safety, reliability, and transparency.

References

1. Z. Wu, H. Gao, J. He, and P. Wang, “The Dark Side of Function Calling: Pathways to Jailbreaking Large Language Models,” Dec. 2024, [Online]. Available: http://arxiv.org/abs/2407.17915
2. Y.-C. Chen, P.-C. Hsu, C.-J. Hsu, and D. Shiu, “Enhancing Function-Calling Capabilities in LLMs: Strategies for Prompt Formats, Data Integration, and Multilingual Translation,” Dec. 2024, [Online]. Available: http://arxiv.org/abs/2412.01130
3. C. Sypherd and V. Belle, “Practical Considerations for Agentic LLM Systems,” Dec. 2024, [Online]. Available: http://arxiv.org/abs/2412.04093
4. W. Liu et al., “TOOLACE: WINNING THE POINTS OF LLM FUNCTION CALLING.” [Online]. Available: https://huggingface.co/Team-ACE.
5. M. Luo et al., “Autellix: An Efficient Serving Engine for LLM Agents as General Programs,” Feb. 2025, [Online]. Available: http://arxiv.org/abs/2502.13965
6. X. Hou, Y. Zhao, S. Wang, and H. Wang, “Model Context Protocol (MCP): Landscape, Security Threats, and Future Research Directions,” Apr. 2025, [Online]. Available: http://arxiv.org/abs/2503.23278
7. B. Radosevich and J. Halloran, “MCP Safety Audit: LLMs with the Model Context Protocol Allow Major Security Exploits,” Apr. 2025, [Online]. Available: http://arxiv.org/abs/2504.03767
8. V. S. Narajala and I. Habler, “Enterprise-Grade Security for the Model Context Protocol (MCP): Frameworks and Mitigation Strategies,” May 2025, [Online]. Available: http://arxiv.org/abs/2504.08623
9. Z. Luo, X. Shi, X. Lin, and J. Gao, “Evaluation Report on MCP Servers,” Apr. 2025, [Online]. Available: http://arxiv.org/abs/2504.11094
10. I. Kavathekar, R. Donakanti, P. Kumaraguru, and K. Vaidhyanathan, “Small Models, Big Tasks: An Exploratory Empirical Study on Small Language Models for Function Calling,” Apr. 2025, [Online]. Available: http://arxiv.org/abs/2504.19277

11. A. Ehtesham, A. Singh, G. K. Gupta, and S. Kumar, "A survey of agent interoperability protocols: Model Context Protocol (MCP), Agent Communication Protocol (ACP), Agent-to-Agent Protocol (A2A), and Agent Network Protocol (ANP)," May 2025, [Online]. Available: http://arxiv.org/abs/2505.02279
12. H. Song et al., "Beyond the Protocol: Unveiling Attack Vectors in the Model Context Protocol (MCP) Ecosystem," Aug. 2025, [Online]. Available: http://arxiv.org/abs/2506.02040
13. P. Belcak et al., "Small Language Models are the Future of Agentic AI," Jun. 2025, [Online]. Available: http://arxiv.org/abs/2506.02153
14. A. Sarkar and S. Sarkar, "Survey of LLM Agent Communication with MCP: A Software Design Pattern Centric Review."
15. A. Brasoveanu, M. Moodie, and R. Agrawal, "Textual evidence for the perfunctoriness of independent medical reviews," in CEUR Workshop Proceedings, CEUR-WS, 2020, pp. 1–9. doi: 10.1145/nnnnnnn.nnnnnnn.
16. M. A. Ferrag, N. Tihanyi, D. Hamouda, L. Maglaras, and M. Debbah, "From Prompt Injections to Protocol Exploits: Threats in LLM-Powered AI Agents Workflows," Jun. 2025, [Online]. Available: http://arxiv.org/abs/2506.23260
17. Y. Zheng, Y. Hu, T. Yu, and A. Quinn, "AgentSight: System-Level Observability for AI Agents Using eBPF," Aug. 2025, [Online]. Available: http://arxiv.org/abs/2508.02736
18. Y. Guo et al., "Systematic Analysis of MCP Security," Aug. 2025, [Online]. Available: http://arxiv.org/abs/2508.12538
19. Z. Luo et al., "MCP-Universe: Benchmarking Large Language Models with Real-World Model Context Protocol Servers," Aug. 2025, [Online]. Available: http://arxiv.org/abs/2508.14704
20. C. Yang, Z. Zhao, and L. Zhang, "KernelGPT: Enhanced Kernel Fuzzing via Large Language Models," in International Conference on Architectural Support for Programming Languages and Operating Systems - ASPLOS, Association for Computing Machinery, Mar. 2025, pp. 560–573. doi: 10.1145/3676641.3716022.
21. Z. Wang et al., "MCPTox: A Benchmark for Tool Poisoning Attack on Real-World MCP Servers," Aug. 2025, [Online]. Available: http://arxiv.org/abs/2508.14925