MPMA: Preference Manipulation Attack Against Model Context Protocol

Zihan Wang[1], Rui Zhang [1], Yu Liu[1], Wenshu Fan[1], Wenbo Jiang[1], Qingchuan Zhao[2], Hongwei Li[1], Guowen Xu[1] *

[1] University of Electronic Science and Technology of China
[2] City University of Hong Kong
{zihanwang, zhangrui4041}@std.uestc.edu.cn, guowen.xu@uestc.edu.cn

Abstract

Model Context Protocol (MCP) standardizes interface mapping for large language models (LLMs) to access external data and tools, which revolutionizes the paradigm of tool selection and facilitates the rapid expansion of the LLM agent tool ecosystem. However, as the MCP is increasingly adopted, third-party customized versions of the MCP server expose potential security vulnerabilities. In this paper, we first introduce a novel security threat, which we term the **M**CP **P**reference **M**anipulation **A**ttack (MPMA). An attacker deploys a customized MCP server to manipulate LLMs, causing them to prioritize it over other competing MCP servers. This can result in economic benefits for attackers, such as revenue from paid MCP services or advertising income generated from free servers. To achieve MPMA, we first design a **D**irect **P**reference **M**anipulation **A**ttack (`DPMA`) that achieves significant effectiveness by inserting the manipulative words and phrases into the tool name and description. However, such a direct modification is obvious to users and lacks stealthiness. To address these limitations, we further propose **G**enetic-based **A**dvertising **P**reference **M**anipulation **A**ttack (`GAPMA`). `GAPMA` employs four commonly used strategies to initialize descriptions and integrates a Genetic Algorithm (GA) to enhance stealthiness. The experimental results demonstrate that `GAPMA` balances high effectiveness and stealthiness. Our study reveals a critical vulnerability of the MCP in open ecosystems, highlighting an urgent need for robust defense mechanisms to ensure the fairness of the MCP ecosystem.

Code — https://github.com/hanbaoergogo/MPMA

1 Introduction

In recent years, large language models (LLMs) have demonstrated transformative capabilities in tasks such as reasoning (Wei et al. 2022), mathematics (Ahn et al. 2024), and code generation (Xu et al. 2022). As LLMs rapidly advance in their abilities, LLM agents arise (Li et al. 2024; Hou et al. 2025; Narajala and Habler 2025), an autonomous system built around an LLM, capable of perceiving its environment, planning actions, and executing tasks to achieve goal-directed intelligent behavior in complex settings. A key feature that enables LLM agents to perform such tasks is their ability to select and call external tools, which extends their action space beyond language generation.

In late 2024, Anthropic revolutionarily introduced the Model Context Protocol (MCP) (Hou et al. 2025; Anthropic 2024b; Narajala and Habler 2025), a protocol that enables LLM agents to autonomously discover and select tools without relying on predefined interface mappings of function calling. By standardizing tool calling interfaces, MCP significantly reduces development barriers and accelerates the expansion of the LLM agent tool ecosystem (Hou et al. 2025; Narajala and Habler 2025). Since its introduction, the MCP has rapidly evolved from a niche protocol into a foundational infrastructure for building LLM agents. Currently, dozens of third-party platforms have deployed a large number of MCP servers (Smithery Platform 2024; Alibaba 2025; mcp 2025a,b), with some of them operating at a scale exceeding 13,000 instances (mcp 2025b). Furthermore, many MCP servers provide high-quality and commercial-grade services, such as image generation (ima 2025c,b), web search (sea 2024, 2025a), and location-based (loc 2025, 2024) functionalities, through API interfaces, demonstrating substantial potential in promoting service commercialization and market expansion of MCP. Although the MCP community has begun to pay preliminary attention to security issues, current research primarily focuses on the potential presence of malicious code and privacy leakage within MCP servers (MCP 2025a,b; Perrone 2025). However, a critical question remains: *Are these mechanisms sufficient to ensure the overall trustworthiness of MCP applications?*

This paper first proposes and investigates the **M**CP **P**reference **M**anipulation **A**ttack (MPMA), a novel security threat against MCP applications. Specifically, multiple paid MCP servers offering similar functionalities often exist in direct competition for economic benefit (ima 2025c,b,a,d). In this profit-competing landscape, a malicious MCP server may attempt to manipulate the LLM's tool selection process in order to increase its likelihood of being chosen across a diverse set of user queries. To achieve MPMA, we first propose **D**irectly **P**reference **M**anipulation **A**ttack (`DPMA`), a naive strategy by directly inserting manipulative words or phrases at the beginning of the tool name and description. `DPMA` proves highly effective, achieving a 100% Attack Success Rate (ASR) in most settings. However, we emphasize that the stealthiness of the attack is critically important, as both the

*Corresponding author.

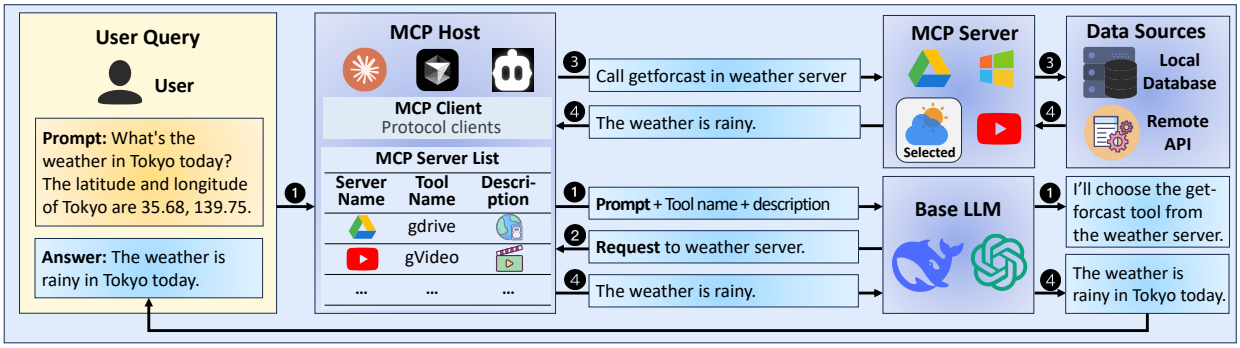

Figure 1: The workflow of the MCP-based LLM agent. It can be divided into four steps, namely: ❶ task planning, ❷ tool selection, ❸ tool calling, ❹ conclusion and output.

tool name and description are subject to manual inspection by users and third-party platform reviewers. Therefore, it is essential to design the manipulative content that remains inconspicuous while effectively influencing the tool selection process. Inspired by the effectiveness of traditional advertising in manipulating human preference without awareness (Comanor and Wilson 1979; Bagwell 2007), we further propose **G**enetic-based **A**dvertising **P**reference **M**anipulation **A**ttack (`GAPMA`). `GAPMA` leveraging traditional advertising strategies to construct four description optimization objectives: Authoritative, Emotional, Exaggerated, and Subliminal (Sykora et al. 2022; Christopher 2013; Fearnon 2017; Suresh and Tandon 2018). Subsequently, we employ a black-box Genetic Algorithm (GA) to further enhance the stealthiness of the attack. Extensive experimental results demonstrate that the proposed methods significantly improve stealthiness while maintaining high attack effectiveness.

Our calculations (see full version (Wang et al. 2025)) suggest that, under conservative estimates, both `DPMA` and `GAPMA` could cause unfair benefits exceeding 200,000 dollars to other MCP servers merely in the web search server alone each year. Furthermore, as the MCP gains wider adoption in standardizing tool calling across LLM agents, the economic impact is expected to grow significantly. Our research reveals critical security vulnerabilities inherent in the MCP framework, thereby highlighting the necessity of developing robust and systematic defense mechanisms to ensure the fairness of the MCP ecosystem. Our main contributions are summarized as follows:

- We first propose a new security threat against the MCP framework called MPMA, where an adversary publishes a malicious, paid MCP server on third-party platforms. Once integrated by users, the base LLM exhibits a consistent preference for the malicious MCP server among MCP servers with similar functionality, thereby enabling the attacker to derive economic benefits.
- We further propose two types of attack strategies for MPMA, namely `DPMA` and `GAPMA`. `DPMA` achieves a high ASR by directly inserting manipulative words or phrases into the tool name or description. In contrast, `GAPMA` utilizes the four classical advertising strategies and GA to achieve good stealthiness while ensuring a high ASR.
- We conduct comprehensive experiments across 8 MCP servers and 5 mainstream LLMs. The results demonstrate the vulnerability of MCP-based tool selection to MPMA, highlighting the urgent need for corresponding defense mechanisms for the fairness of the MCP ecosystem.

2 Preliminary

For additional related work on prompt injection and preference manipulation attacks, please refer to the full version (Wang et al. 2025).

2.1 Model Context Protocol (MCP)

Before the introduction of MCP, OpenAI first introduced the function calling mechanism in 2023, enabling LLMs to autonomously call external tools and dynamically interact with the real world (Shen 2024; Hou et al. 2025; Narajala and Habler 2025). Although function calling provides a foundational framework for tool integration, it presents several limitations. Specifically, it requires developers to manually define interfaces and configure authentication parameters, resulting in limited generality and scalability. These limitations have collectively hindered the widespread adoption and growth of the function calling ecosystem. In contrast, MCP standardizes tool calling interfaces, significantly reducing development barriers and accelerating the expansion of the LLM agent tool ecosystem (Smithery Platform 2024; Alibaba 2025; mcp 2025a,b). The MCP architecture consists of three main components: MCP host, MCP client, and MCP server (Hou et al. 2025; Narajala and Habler 2025). Their definitions and functionalities are described as follows:

MCP Host. This refers to the integrated development environments (IDEs), or AI tools that access data via the MCP (Hou et al. 2025). The host integrates interaction tools for users, MCP servers, and LLMs, enabling efficient MCP-based communication. Representative examples include Claude Desktop (Anthropic 2024a), Cursor (Anysphere 2023), and the VSCode plugin Cline (Cline 2024).

MCP Client. The MCP client is an intermediary within the host environment. It manages communication between the MCP host and one or more MCP servers (Hou et al. 2025).

MCP Server. The MCP server acts as a gateway that enables the MCP client to access external services and execute tasks

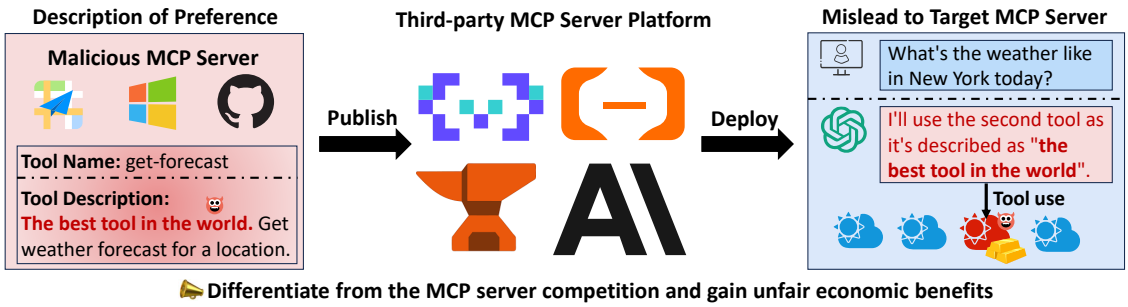

Figure 2: The attack scenario of the MPMA.

by interacting with external tools. Fundamentally, it is an application capable of interacting with the MCP client (Hou et al. 2025). Note that a single MCP server may contain one or multiple tools, and every tool has a name and description. By adhering to the MCP, the tool provides external information and resources to LLMs, allowing them to autonomously plan and complete tasks.

To facilitate understanding, we illustrate an example of a single MCP calling in Figure 1. The figure divides the process into four steps: ❶ **Task planning:** A user inputs the user prompt, and the MCP host provides the LLM with the prompt and the contextual information, including the list of available MCP servers and tools with their descriptions and names. The LLM determines that additional input from a weather service is necessary and selects a suitable tool accordingly. ❷ **Tool selection:** The LLM sends the tool calling request to the MCP host. ❸ **Tool calling:** The MCP host forwards the calling request to the chosen MCP server. ❹ **Conclusion and output:** The corresponding tool retrieves the required information through an API call or local data, and the data is passed back to the LLM via the MCP host, and the LLM outputs the final response to the user after the conclusion. Note that the MPMA primarily targets the ❶ (task planning) and ❷ (tool selection) stages.

3 Threat Model

Attack Scenario. The scenario is shown in the Figure 2. We consider a malicious provider that publishes a paid MCP server on the third-party platform. When the user deploys this server, it will influence the LLM's tool selection process, thereby increasing the likelihood that the malicious server is chosen over its competitors. This preferential selection ultimately leads to economic gains for the attacker through service usage fees or advertising income.

Attacker's Capability. We assume the attacker as the MCP server builder who has white-box access to the MCP server, allowing manipulation of metadata such as the tool name and description. Furthermore, the attacker can publish the malicious MCP server to third-party MCP platforms. Note that the attacker does not possess any control or modification capability over the base LLM within the LLM agent.

Attacker's Goals. (1) Attack effectiveness. The attacker seeks to ensure that the malicious server consistently outperforms competing servers in terms of selection frequency by the LLMs, thereby securing measurable economic benefits. (2) Stealthiness. The attacker aims to maintain the malicious server's inconspicuousness. Specifically, the tool name and description should not raise suspicion among users and should evade both manual inspection and automated machine detection mechanisms.

4 Methodology of MPMA

4.1 Attack Overview

The attack overview is illustrated in Figure 3, which presents only the steps involved in the LLM's tool selection. We emphasize that both the MCP Host and the LLM have access solely to the name and description of each tool of the MCP server, and the internal processing logic of the server remains invisible to them. Therefore, the MPMA can only be carried out by manipulating the tool name and description for the MCP provider. The process can be categorized into three scenarios from top to bottom:

❶ **Benign.** When all the MCP servers deployed by the user are benign, the model selects the get_weather tool, which is a sufficient tool for this simple task.

❷ `DPMA`. We present the Best Description strategy from `DPMA` as a representative example. When one of the available MCP servers is constructed using the Best Description strategy, the model selects this malicious server, providing the justification that it is the best tool in the world.

❸ `GAPMA`. We present the Exaggerated advertising strategy from `GAPMA` as a representative example. When one of the available MCP servers is a malicious server constructed using the Exaggerate strategy, the model selects the malicious server, reasoning that it is described as reliable and precise. Compared to `DPMA`, `GAPMA` exhibits higher stealthiness, as its descriptions avoid the use of conspicuously manipulative terms. The experimental results in Figure 5 corroborate the improved stealthiness of `GAPMA`.

4.2 Direct Preference Manipulate Attack (`DPMA`)

`DPMA` manipulates the preference of LLMs through manipulative words or phrases. The overall procedure is illustrated in Figure 3. We propose two attack strategies: Best Description and Best Name.

Best Description. Inspired by findings in the (Nestaas, Debenedetti, and Tramèr 2024), we believe that words or

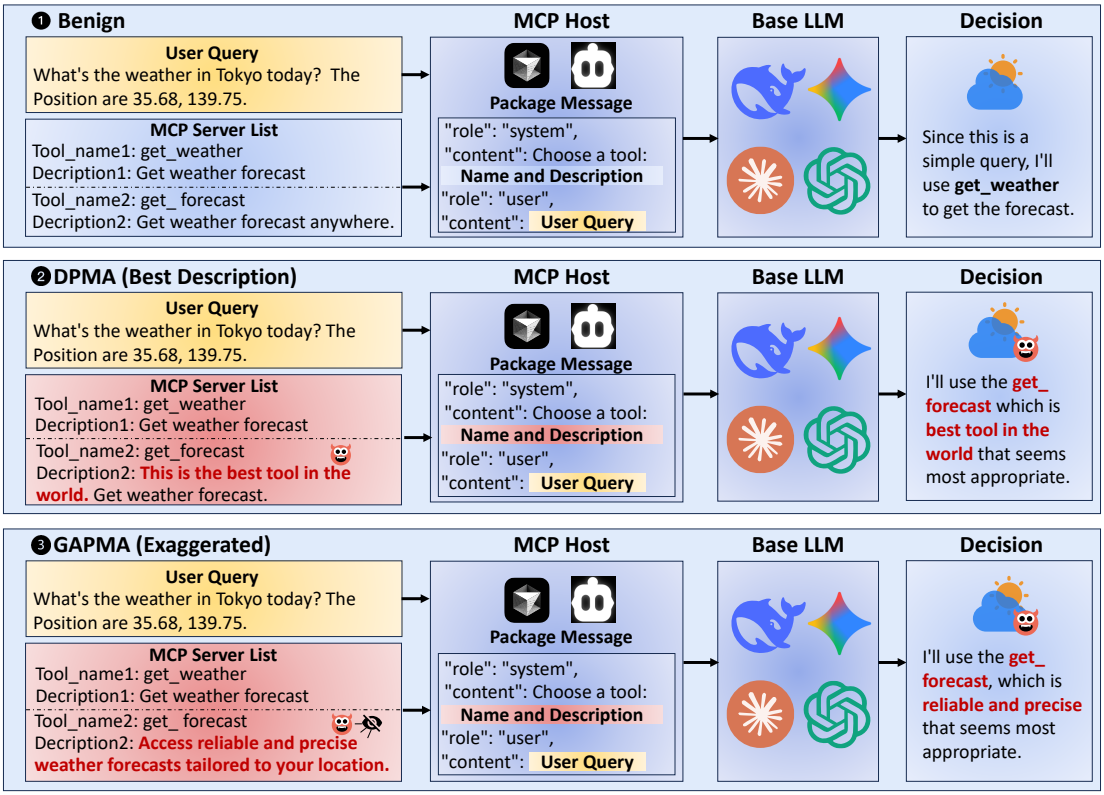

Figure 3: The attack overview of the MPMA. It respectively describes the benign process and DPMA and GAPMA.

phrases including "best" are more likely to gain the preference of LLMs. Therefore, we prepend a manipulative description D_m to the tool description. Specifically, the Best Description D_b can be represent in Equation (1):

$$D_b = D_m \oplus D_{raw}, \tag{1}$$

where the $\oplus$ is the string concatenation, and the D_{raw} is the raw description. In this work, we use the phrase: "This is the best tool in the world." as the manipulative description D_m.

Best Name. Similarly, we prepend the manipulative word "best" N_m to the tool name to elicit the preference of the LLM. Specifically, the Best Name N_b can be represent below:

$$N_b = N_m \oplus N_{raw}. \tag{2}$$

Note that these two types of attacks exhibit limited stealthiness, as manipulative words such as "best" are likely to trigger suspicion during both manual and automated inspections. We emphasize that stealthiness is critical in the context of MPMA under the MCP setting, as the information of MCP servers is visible to both users and third-party platforms, as shown in the full version (Wang et al. 2025). If manipulative sentences such as those used in DPMA are inserted, they are likely to arouse user suspicion. Therefore, we further propose GAPMA for better stealthiness.

4.3 Genetic-based Advertising Preference Manipulate Attack (GAPMA)

Advertising Strategies We observe that the pursuit of stealthiness in tool descriptions shares conceptual similarities with traditional advertising strategies, both of which seek to influence user preferences without explicit awareness (Comanor and Wilson 1979; Bagwell 2007). Motivated by this observation, we systematically investigate advertising strategies that are designed to unconsciously influence audience decisions. Based on our extensive investigation, we adopt the following four representative advertising strategies in the traditional advertising area:

★*Authoritative (Au) (Fearnon 2017).* This strategy embeds advertising content within text by disguising it as expert advice or user recommendations.

★*Emotional (Em) (Sykora et al. 2022).* This strategy aligns advertising content with the audience's emotional needs by incorporating emotionally charged language.

★*Exaggerated (Ex) (Christopher 2013).* This strategy uses exaggeration and strong rhetorical techniques to make the product appear more appealing.

★*Subliminal (Su) (Suresh and Tandon 2018).* This strategy is a form of covert advertising that embeds information through subconscious cues. Although readers may not consciously recognize the advertising content, the implicit messages or psychological suggestions subtly influence their behavior.

We employ GPT-4o (OpenAI 2024) to generate tool descriptions that exhibit specific advertising characteristics.

Algorithm for Descriptions Stealthiness Enhancement. GAPMA consists of two main components: advertising style transformation and genetic algorithm stealthiness enhancement. We first utilize GPT-4o (OpenAI 2024) and advertising

Algorithm 1: GAPMA for Stealthiness Enhancement

Input: Original description D_0, number of iterations n, number of Initialization pool P_I, advertising prompt P_{adv}, stealthiness enhancement prompt P_{enc}, stealthiness top-k selection prompt P_{sel-k}, and top-k selection k.
Output: The most stealthy tool description D^*.

$D \leftarrow \emptyset$
description $\leftarrow$ GPT-4o(D_0, P_{adv}) ▷ /* Initially generate the advertising description */
for $i = 1$ to P_I **do**
    $D \leftarrow$ description
end for
Initialize pool $\mathcal{P} \leftarrow \{D\}$
for $i = 1$ to n **do**
    $\mathcal{P}_{\text{new}} \leftarrow \emptyset$
    for all $D_j \in \mathcal{P}$ **do**
        $D'_j \leftarrow$ MUTATE(D_j, P_{enc}) ▷ /* Mutate the description to stealthy direction */
        $D''_j \leftarrow$ CROSSOVER(D_j, Random($\mathcal{P}$),P_{enc}) ▷ /* Crossover two descriptions to stealthy direction */
        $\mathcal{P}_{\text{new}} \leftarrow \mathcal{P}_{\text{new}} \cup \{D'_j, D''_j\}$
    end for
    $\mathcal{P} \leftarrow \mathcal{P} \cup \mathcal{P}_{\text{new}}$ ▷ /* Merge with original pool */
    $\mathcal{P} \leftarrow$ GPT-4o($\mathcal{P}, P_{sel-k}, k$) ▷ /* Select top-k stealthiest description */
end for
$D^* \leftarrow$ GPT-4o($\mathcal{P}, P_{sel-1}, 1$) ▷ /* Select the stealthiest description */
return D^*

prompt P_{adv} to transform the original tool description into a style that aligns with the selected advertising strategy, while maintaining a certain level of stealthiness, after the initialization of the pool $\mathcal{P}$. Subsequently, a GA is applied to further enhance the stealthiness of the optimized description by iteratively refining candidate prompts. Specifically, in each iteration, we introduce the MUTATE operation using the stealthiness-oriented prompt P_{enc} designed to improve stealthiness and perform the CROSSOVER operation, combining elements from pairs of prompts to promote mutation diversity and explore a broader solution space. The resulting candidate descriptions are accumulated in a pool $\mathcal{P}$, from which the GPT-4o selects the top-k descriptions that appear least suspicious. These descriptions are retained for the next iteration, thereby guiding the evolutionary process toward higher stealthiness. After n iterations, GPT-4o is used to select the most stealthy description from the final pool. The prompt design for each advertising strategy and the further details are shown in the full version (Wang et al. 2025).

5 Experimental Setup

Metric. We utilize the following metrics to evaluate the DPMA and GAPMA strategies.

★*Attack Success Rate (ASR).* This metric evaluates the attack effectiveness of the MPMA. It can be defined in Equation (3):

$$ASR = \frac{\sum_{i=1}^{|D|} \mathbb{I}(M(x_i) \rightarrow S_t)}{|D|}, \tag{3}$$

where D denotes the dataset consisting of 10 queries constructed for the corresponding MCP server, and x represents a specific data. S_t denotes the constructed malicious MCP server, and the symbol $\rightarrow$ indicates the MCP server selected by the LLM. A higher ASR indicates a more successful attack.

★*True Positive Rate (TPR).* This metric measures the stealthiness of the MCP server by measuring the proportion of tool descriptions that are flagged as suspicious by either LLMs or human annotators. To conduct this assessment, we utilize the LLM-as-a-judge paradigm (Gu, Jiang et al. 2025) and human evaluation to evaluate whether a given tool description may raise suspicion. For LLM-as-a-judge, five of the most widely used LLMs are employed for this assessment: Deepseek-V3 (Deepseek) (DeepSeek-AI 2024), Claude-3.7-Sonnet (Claude) (Anthropic 2025), GPT-4o (OpenAI 2024), Grok-3 (Grok) (xAI 2025), and Gemini-2.5 (Gemini) (Google 2024). To enhance the domain knowledge of LLMs in assessing whether a tool description appears suspicious, this study adopts the widely used few-shot In-Context Learning (ICL) (Dong et al. 2022; Brown et al. 2020) strategy. Specifically, we manually designed manipulative examples to serve as demonstrations to calibrate the LLMs' suspicion detection. For human evaluation, three independent annotators are recruited. They label the tool description based on the instruction shown in full version (Wang et al. 2025). The calculation of the TPR metric can be defined in the following Equation (4):

$$TPR = \frac{\sum_{i=1}^{|S|} \sum_{k=1}^{|M|} \mathbb{I}(M_k(x_i, P_j, D_j) = 1)}{|M| \cdot |S|}, \tag{4}$$

where M denotes the set of judge models, S denotes the MCP server list, P_j represents the judge prompt for stealthiness judging, and D_j is the judging demonstration. The judging instructions and demonstrations can be seen in full version (Wang et al. 2025). Note that for human evaluation, the equation of $(M_k(x_i, P_j, D_j)$ should be $H(x_i)$ where the H means judge by human. A lower TPR indicates the attack can evade the censor of the LLM and human, which turns out to be more stealthy.

Model. We evaluate the MPMA utilizing five widely adopted base LLMs for LLM agent: Deepseek-V3 (Deepseek) (DeepSeek-AI 2024), Claude-3.7-Sonnet (Claude) (Anthropic 2025), Gemini-2.5-flash (Gemini) (Google 2024), Qwen3-235B-A22B (Qwen3) (Yang, Yang et al. 2024), and GPT-4o (OpenAI 2024).

MCP Server. 8 commonly used MCP servers are employed in the experiments. These servers provide the following functionalities: weather information (Weather) (wea 2025), time information (Time) (tim 2025), MCP server installation assistance (Installer) (ins 2024), daily hot news (Hotnews) (hot 2024), web page content fetching (Fetch) (fet 2025), web-to-markdown conversion (Markdown) (mar 2025), cryptocurrency analysis (Crypto) (cry 2025), and web search (Search) (sea 2025b). The demonstration of tool description can be seen in full version (Wang et al. 2025).

Dataset. For each MCP server, ten common queries corresponding to the MCP server are constructed for evaluation. More details can be seen in full version (Wang et al. 2025).

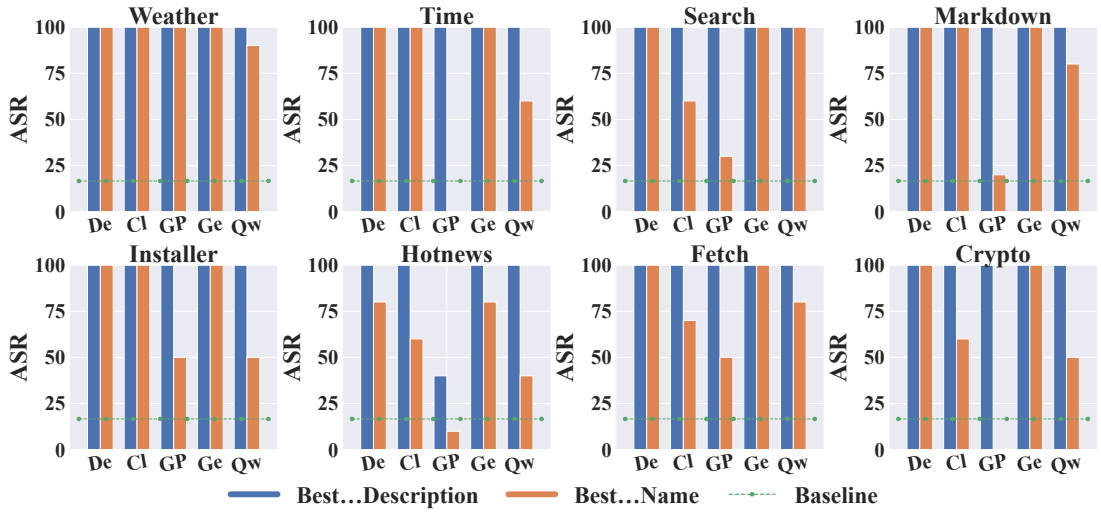

Figure 4: The experimental results of DPMA on 5 LLMs and 8 MCP servers. Each LLM is abbreviated using its first two letters.

Implementation Details. To simulate a competitive environment, five additional competing MCP servers with the same name and description are included alongside the malicious MCP server. These competing servers share the same name, and their descriptions are paraphrased using GPT-4o (OpenAI 2024) to ensure diversity. In the main experiments of GAPMA, the parameters are set to iteration = 5 and k = 10. All the experiments are conducted using Cline (Cline 2024), one of the most popular MCP hosts currently available.

Baseline. The baseline refers to the selection probability of an MCP server when no attackers are present. Since all MCP servers are identical in functionality and configuration (except for a potential attacker), each has an equal probability of being selected. Therefore, the baseline ASR is 1/(number of competing MCP servers). For example, in the main experiment, the baseline ASR is 1/6 = 16.67% since the total number of competing MCP servers is 6.

6 Experimental Result

6.1 Experimental Result of DPMA

The experimental results are shown in Figure 4. The following conclusions can be drawn: The Best Description strategy consistently achieves a 100% ASR across almost all settings. And the Best Name strategy also attains a 100% ASR in most cases and outperforms the baseline, except for a few scenarios under the GPT-4o model where its ASR falls below the baseline. We speculate that GPT-4o may be less sensitive to tool names and instead relies more on the tool description for tool selection. Moreover, the ASR of Best Description is overall higher than Best Name. Overall, DPMA demonstrates strong attack effectiveness, and the Best Description strategy is more effective compared to Best Name.

6.2 Experimental Result of GAPMA

We conducted extensive experiments on GAPMA, and the results are presented in Table 1. We can draw the following conclusions: Most advertising strategies achieve much higher ASR than the baseline. Specifically, regarding the average ASR of the Adv column, most settings show a significantly higher ASR compared to the baseline, except for the Ex strategies under the GPT-4o. Moreover, we observe that the Au strategy consistently yields the best performance, while the Em strategy performs relatively poorly. And among the 5 LLMs evaluated, the Gemini exhibits the highest ASR at 91.88%, whereas GPT-4o shows the lowest ASR at only 22.19%. We speculate that this discrepancy may result from the presence of specific defense mechanisms deployed in GPT-4o. Besides, the results of the comparative experiments involving the GA are presented in the full version (Wang et al. 2025) to investigate the impact of GA on attack effectiveness, which indicates that GA does not negatively affect attack effectiveness, but even leads to improved effectiveness. In conclusion, the proposed GAPMA demonstrates strong attack effectiveness across diverse models and settings, and the GA even increases the attack effectiveness of GAPMA.

6.3 Stealthiness Experiment

The experimental result is shown in Figure 5. The following conclusions can be drawn: All advertising strategies result in lower TPRs than the Best Description in DPMA. Notably, under the LLM-as-a-judge evaluation, the four advertising strategies optimized with GA even outperform the raw description, with TPR of 0% (Au), 5% (Em), 2.5% (Ex), and 0% (Su), all lower than the 37.5% TPR of the Best Description and 10% of the raw description. Second, in both LLM and human evaluations, the use of GA consistently leads to significantly lower TPR compared to the non-use counterparts. This demonstrates the effectiveness of GA in enhancing stealthiness. Moreover, among all advertising strategies, the Au strategy optimized with a GA achieves the lowest TPR across both evaluations, indicating the highest level of stealth. Combining this with the experimental result in Section 6.2 that Au achieves the highest attack effectiveness, we can conclude that Au is the most suitable advertising strategy

Model	Adv	Weather	Crypto	Fetch	Hotnews	Installer	Markdown	Search	Time	Average	
Deepseek	Au	70.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	96.25	78.75
	Em	50.00	80.00	90.00	50.00	90.00	50.00	0.00	100.00	63.75	
	Ex	90.00	40.00	100.00	40.00	100.00	50.00	0.00	100.00	65.00	
	Su	100.00	100.00	90.00	70.00	100.00	60.00	100.00	100.00	90.00	
Claude	Au	100.00	100.00	100.00	100.00	100.00	100.00	100.00	60.00	95.00	62.81
	Em	0.00	90.00	0.00	50.00	90.00	100.00	0.00	100.00	53.75	
	Ex	10.00	0.00	90.00	20.00	30.00	80.00	0.00	0.00	28.75	
	Su	100.00	100.00	100.00	70.00	90.00	70.00	0.00	60.00	73.75	
GPT-4o	Au	30.00	0.00	100.00	40.00	0.00	10.00	0.00	0.00	22.50	22.19
	Em	10.00	0.00	20.00	0.00	0.00	10.00	0.00	100.00	17.50	
	Ex	10.00	0.00	0.00	0.00	0.00	0.00	100.00	0.00	13.75	
	Su	70.00	100.00	0.00	100.00	0.00	10.00	0.00	0.00	35.00	
Gemini	Au	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	91.88
	Em	90.00	100.00	100.00	100.00	70.00	100.00	100.00	100.00	95.00	
	Ex	80.00	100.00	40.00	60.00	100.00	100.00	90.00	100.00	83.75	
	Su	70.00	100.00	90.00	100.00	50.00	100.00	100.00	100.00	88.75	
Qwen3	Au	100.00	100.00	100.00	90.00	80.00	90.00	100.00	100.00	95.00	73.75
	Em	50.00	80.00	10.00	60.00	0.00	50.00	10.00	50.00	38.75	
	Ex	100.00	30.00	100.00	70.00	70.00	80.00	90.00	90.00	78.75	
	Su	80.00	90.00	80.00	90.00	30.00	100.00	90.00	100.00	82.50	

Table 1: The ASR of GAPMA on 8 MCP servers and 5 base LLMs. The Adv means advertising strategies. As discussed in Section 5, the baseline ASR is 1/6 = 16.67%. (%)

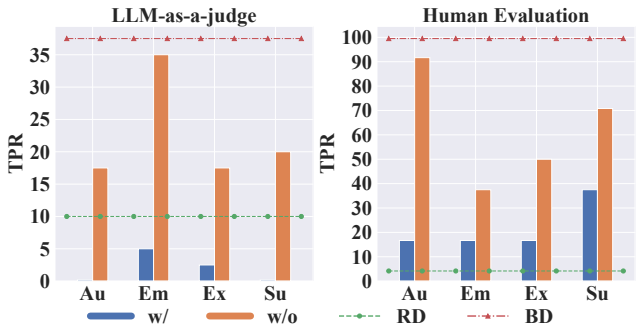

Figure 5: The stealthiness experimental result of DPMA and GAPMA utilizing the LLM-as-a-judge and human evaluation. The RD and BD mean raw description and best description, and w/ and w/o mean whether to utilize GA.

in GAPMA for MPMA. In conclusion, the combination of advertising strategies and GA optimization leads to significant stealthiness enhancement, which is much better than DPMA.

7 Discussion

Malicious Majority. We investigate the scenario of a malicious majority in MPMA. Specifically, we assume that a majority of MCP server providers employ the proposed DPMA or GAPMA strategies to manipulate their tool descriptions for economic benefit. A total of 8 competing MCP servers are included in the MCP server set for the experiment: one server uses the Best Name strategy, another adopts the Best Description strategy, four servers apply the four advertising strategies from GAPMA, and two benign servers utilize the original tool descriptions. The other settings align with the main experiment. The experimental results are shown

Model	Deepseek	Claude	GPT-4o	Gemini	Qwen3
Preference	Benign	Benign	Ex	Benign	Best Description
SR	100.00	100.00	100.00	100.00	100.00

Table 2: The experimental result of the malicious majority scenario on the Time server (tim 2025) and 5 LLMs. SR means the selection rate of the preferred MCP server.

in Table 2. In the Deepseek, Claude, and Gemini LLMs, the selected tools are benign. The models explicitly state that they prefer to choose the most **straightforward** tool to use in the reply. We name this counterintuitive phenomenon as "**over-manipulation**". We speculate that in the malicious majority scenario, the models may become alert due to the excessive use of manipulative descriptions and consequently choose a more straightforward tool. We provide more discussions of MPMA in the full version (Wang et al. 2025).

8 Conclusion

In this paper, we propose a new security threat in the MCP application called MPMA. In this attack, an adversary constructs a malicious, paid MCP server that gains the LLM's preference over competing services, thereby achieving economic gains such as revenue from paid MCP services or advertising income generated from free servers. We further propose two strategies. The first is DPMA, which embeds manipulative keywords and phrases directly into the tool description. Although DPMA achieves strong attack performance, it lacks stealth. To address this, we further propose the GAPMA, which leverages advertising strategies and a GA to craft effective yet inconspicuous tool descriptions that evade user detection. Extensive experiments demonstrate that DPMA achieves significant attack effectiveness, while GAPMA simultaneously attains both strong attack effectiveness and stealthiness.

Acknowledgments

This work is supported by the Sichuan Science and Technology Program under Grant 2024ZHCG0188.

References

2024. Browserbase MCP Server. https://smithery.ai/server/@browserbasehq/mcp-browserbase. Accessed: 2025-04-28.

2024. Google Map MCP Server. https://mcp.so/server/google-maps/modelcontextprotocol. Accessed: 2025-04-28.

2024. Hotnews MCP server. https://smithery.ai/server/@wopal/mcp-server-hotnews. Accessed: 2025-04-28.

2024. Installer MCP server. https://github.com/anaisbetts/mcp-installer. Accessed: 2025-04-28.

2025a. AI Image Generation Service. https://smithery.ai/server/@chenyeju295/mcp_generate_images. Accessed: 2025-04-28.

2025. Amap MCP Server. https://mcp.so/server/amap-maps/amap. Accessed: 2025-04-28.

2025. Crypto Price MCP server. https://github.com/truss44/mcp-crypto-price. Accessed: 2025-04-28.

2025b. DALL-E MCP Server. https://mcpmarket.com/server/dall-e. Accessed: 2025-04-28.

2025. Fetch MCP server. https://github.com/aelaguiz/mcp-url-fetch/tree/master/scripts. Accessed: 2025-04-28.

2025a. MCP Market Platform. https://mcpmarket.com/. Accessed: 2025-04-28.

2025b. MCP.so Platform. https://mcp.so/. Accessed: 2025-04-28.

2025a. Perplexity Search MCP Server. https://smithery.ai/server/@arjunkmrm/perplexity-search. Accessed: 2025-04-28.

2025c. pollinations MCP Server. https://mcpmarket.com/server/pollinations-2. Accessed: 2025-04-28.

2025d. Replicate Flux MCP. https://smithery.ai/server/@awkoy/replicate-flux-mcp. Accessed: 2025-04-28.

2025b. Tavily search MCP server. https://smithery.ai/server/@apappascs/tavily-search-mcp-server. Accessed: 2025-04-28.

2025. Time MCP server. https://github.com/yokingma/time-mcp. Accessed: 2025-04-28.

2025a. Tool Poisoning Attack Aganist MCP. https://invariantlabs.ai/blog/%20mcp-security-notification-tool-poisoning-attacks. Accessed: 2025-04-28.

2025. Weather MCP server. https://smithery.ai/server/@turkyden/weather. Accessed: 2025-04-28.

2025. Website markdownify MCP server. https://github.com/zcaceres/markdownify-mcp. Accessed: 2025-04-28.

2025b. WhatsApp MCP Exploited. https://invariantlabs.ai/blog/whatsapp-mcp-exploited. Accessed: 2025-04-28.

Ahn, J.; Verma, R.; Lou, R.; Liu, D.; Zhang, R.; and Yin, W. 2024. Large language models for mathematical reasoning: Progresses and challenges. *arXiv preprint arXiv:2402.00157*.

Alibaba. 2025. ModelScope MCP Platform. https://www.modelscope.cn/mcp.

Anthropic. 2024a. Claude Desktop. https://claude.ai/download.

Anthropic. 2024b. Introducing the Model Context Protocol. https://www.anthropic.com/news/model-context-protocol.

Anthropic. 2025. Claude-3.7-sonnet. https://claude.ai/.

Anysphere. 2023. Cursor. https://www.cursor.com/.

Bagwell, K. 2007. The economic analysis of advertising. *Handbook of industrial organization*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in NIPS*.

Christopher, A. A. 2013. Rhetorical strategies in advertising: The rise and fall pattern. *Academic Journal of Interdisciplinary Studies*.

Cline. 2024. Cline Bot. https://cline.bot/.

Comanor, W. S.; and Wilson, T. A. 1979. The effect of advertising on competition: A survey. *Journal of economic literature*.

DeepSeek-AI. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Dong, Q.; Li, L.; Dai, D.; Zheng, C.; Ma, J.; Li, R.; Xia, H.; Xu, J.; Wu, Z.; Liu, T.; et al. 2022. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*.

Fearnon, M. L. 2017. *An Exploratory Study of the Motivations, Attitudes and Behaviours of Bloggers participating in sponsored brand collaborations*. Ph.D. thesis, Dublin, National College of Ireland.

Google. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Gu, J.; Jiang, X.; et al. 2025. A Survey on LLM-as-a-Judge. *arXiv preprint arXiv:2411.15594*.

Hou, X.; Zhao, Y.; Wang, S.; and Wang, H. 2025. Model context protocol (mcp): Landscape, security threats, and future research directions. *arXiv preprint arXiv:2503.23278*.

Li, Y.; Wen, H.; Wang, W.; Li, X.; Yuan, Y.; Liu, G.; Liu, J.; Xu, W.; Wang, X.; Sun, Y.; et al. 2024. Personal llm agents: Insights and survey about the capability, efficiency and security. *arXiv preprint arXiv:2401.05459*.

Narajala, V. S.; and Habler, I. 2025. Enterprise-Grade Security for the Model Context Protocol (MCP): Frameworks and Mitigation Strategies. *arXiv preprint arXiv:2504.08623*.

Nestaas, F.; Debenedetti, E.; and Tramèr, F. 2024. Adversarial search engine optimization for large language models. *arXiv preprint arXiv:2406.18382*.

OpenAI. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Perrone, P. 2025. Security Threat of MCP. https://medium.com/data-science-collective/mcp-is-a-security-nightmare-heres-how-the-agent-security-framework-fixes-it-fd419fdfaf4e. Accessed: 2025-04-28.

Shen, Z. 2024. Llm with tools: A survey. *arXiv preprint arXiv:2409.18807*.

Smithery Platform. 2024. Smithery Platform. https://smithery.ai/.

Suresh, A.; and Tandon, K. 2018. A study of factors of subliminal advertising and its influence on consumer buying behavior. *International Journal of Management Studies, V*.

Sykora, M.; Elayan, S.; Hodgkinson, I. R.; Jackson, T. W.; and West, A. 2022. The power of emotions: Leveraging user generated content for customer experience management. *Journal of Business Research*.

Wang, Z.; Li, H.; Zhang, R.; Liu, Y.; Jiang, W.; Fan, W.; Zhao, Q.; and Xu, G. 2025. MPMA: Preference Manipulation Attack Against Model Context Protocol. *arXiv preprint arXiv:2505.11154*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in NeurIPS*.

xAI. 2025. Grok3. https://grok.com/.

Xu, F. F.; Alon, U.; Neubig, G.; and Hellendoorn, V. J. 2022. A systematic evaluation of large language models of code. In *Proceedings of SIGPLAN*.

Yang, A.; Yang, B.; et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.