When Heads Disagree: Detecting Preference Manipulation Attacks in LLM Tool Calling

Anonymous ACL submission

Abstract

MCP-based LLM agents are vulnerable to *preference manipulation attacks* (PMAs): adversarial tokens injected into tool metadata steer the model toward an attacker-chosen tool without modifying the user query. Existing defenses are black-box and brittle to paraphrase. None explains *why* the model is misled. We identify the root cause: tool selection is governed by a specialized set of *attention heads* that account for most of the causal effect. PMAs exploit a structural vulnerability in this mechanism: injected tokens hijack a subset of these heads, while the remaining heads retain focus on the semantically correct tool. This produces a detectable *head disagreement* signal. We propose PMAShield, a two-phase white-box detector that (1) discovers tool-selection heads, and (2) flags PMA inputs where those heads disagree, using four interpretable metrics and no fine-tuning or model modification. Evaluated across six LLMs and two attack families (MCPTox, MPMA-DPMA), PMAShield averages 0.71 AUROC on MCPTox and 0.68 on MPMA-DPMA (up to 0.76), outperforming all baselines. Code and data are available at `https://anonymous.4open.science/r/PMAShield-CEEE/`.

1 Introduction

LLM agents built on the Model Context Protocol (MCP) (Anthropic, 2024) enable models to discover and invoke third-party tools in response to user queries (Qin et al., 2024). This capability introduces a new attack surface: *preference manipulation attacks* (PMAs), which poison the *metadata* of a malicious tool (its name and description) with persuasive or deceptive tokens to steer the model's tool choice without modifying the user query (Wang et al., 2026a,b). Recent benchmarks demonstrate that PMAs achieve high success rates across diverse LLMs and tool categories (Wang et al., 2026a,b). Existing defenses, namely input-level filters (Xing et al., 2025) and pre-execution firewalls (Yuan et al., 2026; Zhang et al., 2026a), offer partial mitigation without model retraining.

However, these methods share the same brittleness: they operate without any understanding of *how* the model makes its selection. *Input-level filters* (Xing et al., 2025) (e.g., neural content scanning, keyword heuristics) are brittle: adaptive attack reformulations evade detection (Zhan et al., 2025). *Pre-execution firewalls* (Yuan et al., 2026; Zhang et al., 2026a) block or audit risky tool calls before they execute, but do not explain why the model was susceptible to begin with. No prior work asks the deeper question: *why does the model get fooled?* This gap precludes principled, paraphrase-robust defense. A mechanistic understanding of the selection process and its vulnerabilities is the key missing piece.

We identify the root cause: tool selection is governed by a *specialized, attention-dominated set* of *tool-selection attention heads* (§3). PMAs exploit a structural vulnerability in this mechanism. Injected tokens hijack a *subset* of these heads, pulling their attention to the malicious tool, while the remaining heads retain focus on the semantically correct tool. PMA success is therefore a *partial circuit hijacking*, not a global model failure. The resulting *head disagreement* is absent in benign inputs and serves as a reliable, interpretable attack signal.

Based on this finding, we propose **PMAShield**, an interpretability-grounded detector that monitors the internal selection mechanism directly instead of pattern-matching on surface features. It operates in two phases:

- **Head identification**: Profile each attention head by its tendency to focus on the chosen tool across benign scenarios, and select those that consistently exhibit selective attention, without labeled attack examples.
- **Disagreement-based detection**: Quantify inter-head disagreement on the attended tool using

interpretable attention statistics. Elevated disagreement reveals partial head set hijacking and identifies the input as a PMA.

In summary, our contributions are:

- We present the first head-level circuit analysis of LLM tool selection in the MCP security context, identifying a specialized set of tool-selection attention heads and establishing their causal role via activation patching (§3).
- We show that PMAs cause a measurable, statistically significant increase in head disagreement arising from partial hijacking of the selection circuit, a signal that is substantially weaker in benign inputs (§3).
- We propose PMAShield, which, to the best of our knowledge, is the first detector to use mechanistic circuit analysis as the foundation for real-time PMA detection, evaluated across six LLMs and two attack families without model modification (§4, §5).

2 Related Work

2.1 LLM Tool Calling and MCP Security

Tool use in LLMs has been formalized through frameworks such as ToolLLM (Qin et al., 2024) and the OpenAI function-calling API (OpenAI, 2023), enabling models to invoke external services in a structured manner. The MCP standardizes this interface for third-party tool discovery and invocation in agentic settings (Anthropic, 2024). Security research on tool-augmented LLMs has identified prompt injection via tool responses (Greshake et al., 2023) and tool description poisoning (Wang et al., 2026a). These works establish the existence and severity of PMAs.

2.2 Preference Manipulation Attacks

PMAs specifically target the tool *selection* step rather than the execution environment. MPMA (Wang et al., 2026b) introduces Direct Preference Manipulation Attack (DPMA), which injects strategy tokens (star ratings, authority claims, urgency phrases) into tool names and descriptions. MCPTox (Wang et al., 2026a) studies description-level injection across 45 real MCP servers. Crucially, *none* of these works studies the neural mechanism by which such attacks succeed, which is the starting point of our work.

2.3 Mechanistic Interpretability for LLM Safety

Representation engineering (Zou et al., 2023) and refusal directions (Arditi et al., 2024) show that LLM behavior can be manipulated by adding vectors in the residual stream. Zhang et al. (2026b) identify answer and benign vectors as nearly orthogonal, explaining the jailbreak-overrefusal trade-off, and resolve it via closed-form weight alignment. In-context learning heads (Olsson et al., 2022) demonstrate that specific attention heads implement key copying operations. We are the first to conduct head-level circuit analysis of LLM tool selection in the MCP security context and to use the resulting mechanistic understanding as the foundation for a real-time PMA detector. Concurrent work (Liu et al., 2026; Tatsat and Shater, 2026) examines tool invocation mechanisms via attention pathway analysis and sparse autoencoders, but neither targets MCP-specific attacks nor reaches individual head-level causal precision.

3 Mechanistic Interpretability of LLM Tool Selection

We dissect the neural mechanism by which LLMs select tools, guided by four research questions:

- **RQ1:** Which components of the model are most causally responsible for tool selection?
- **RQ2:** Is there a specialized subset of attention heads that govern selection, or is it a diffuse property of the network?
- **RQ3:** What are the functional roles of the most influential heads?
- **RQ4:** How do PMAs disrupt the selection mechanism, and what internal signals do they produce in the model?

All interpretability experiments use Qwen3-8B (Yang et al., 2025) as the analysis model. While the mechanistic dissection below is performed on this single model, the disagreement-based detector it motivates is evaluated across six architectures in §5, where its consistent effectiveness provides indirect evidence that the same selection mechanism is present in the other models.

3.1 Setup

Probe scenarios. We construct a controlled set of *probe scenarios*, each pairing a user query with 2–4 candidate tool definitions, exactly one of which

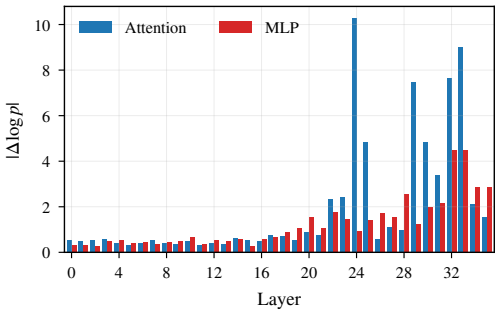

Figure 1: Per-layer causal contribution to tool selection ($|\Delta \log p|$), decomposed into attention and MLP components.

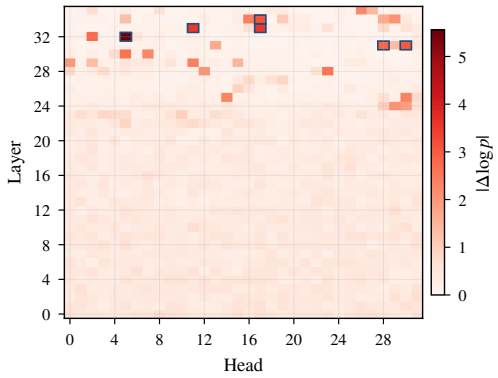

Figure 2: Per-head causal importance for tool selection ($|\Delta \log p|$ averaged across both patching directions and all contrastive pairs).

is semantically appropriate for the query. Scenarios span four task domains (weather forecasting, academic paper search, stock price retrieval, and web search) to ensure that findings are not tied to a single tool category or description style.

Attention capture. For each scenario, we record the per-head attention matrices at the generation step at which the model first commits to a tool invocation. This is the earliest point at which the model's tool choice is encoded in the output, so the attention pattern at this step reflects the full deliberation over the candidate tools.

Patching metric. Following causal tracing methodology (Meng et al., 2022; Wang et al., 2022), for a pair of contrastive scenarios (s_A, s_B) whose intended tools differ (e.g., a weather-forecast query versus a paper-search query), we measure the causal contribution of a component as the change in the *contrastive log-odds* $\Delta = \log p(t_A) - \log p(t_B)$ between the two tools' name tokens, when that component's activations are replaced with their values from the opposite context. Detailed settings and implementation are in Appendix A.

3.2 Finding 1: Attention Dominates MLP

We first perform layer-level activation patching by replacing each layer's attention output or MLP output individually with the corresponding value from the contrastive context and measure the resulting $|\Delta|$. Figure 1 shows the per-layer causal contribution decomposed by mechanism.

Aggregated over all layers, attention outputs account for 61% of the total patching effect, roughly 1.6× the MLP contribution. Notably, attention heads dominate the top-6 most influential components for tool selection. This confirms that tool selection is primarily a *context-dependent pattern-matching* operation encoded in attention rather than static factual knowledge stored in MLP weights. This observation motivates our focus on attention heads for the remainder of the analysis.

3.3 Finding 2: A Specialized Attention-Head Circuit

Having established attention's dominance at the layer level, we now identify which individual heads drive tool selection. We extend the patching analysis to individual attention heads, replacing each head's slice of the attention output projection input ($36 \times 32 = 1152$ heads) with its value from the contrastive context. Figure 2 shows the resulting importance heatmap.

The most influential heads (L32H5, L33H17, L33H11, L34H17, L31H30, and L31H28) are highly specialized: each carries roughly 10–15× the causal effect of an average head. At the same time, the circuit is *distributed* rather than reducible to a handful of heads: the top six heads account for ∼5% of the total effect, and reaching half of the total requires on the order of a few hundred heads (Gini coefficient 0.36). Causal influence is concentrated in two regions: the late layers L31–L34, which house the strongest individual selection heads, and a band of mid layers L24–L30. Early layers (L0–L9) carry a broad, lower-magnitude contribution consistent with structure-level processing. The pattern holds across all four probe categories, indicating a tool-selection mechanism that is shared across task domains rather than query-specific.

3.4 Finding 3: Functional Role Taxonomy

We next ask what functional roles these causal heads play in selection. We classify each of the top-24 heads (ranked by the patching importance of Finding 2) by where its attention mass concentrates

Table 1: Functional roles of the top-24 causal attention heads.

Role	N	Example heads
Intent-matching	15	L29H11, L31H28, L24H29
Query-context	9	L30H7, L25H14, L34H17

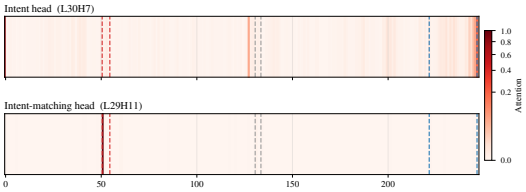

Figure 3: Commit-step attention at the tool-call token for one representative head of each role, where colour encodes attention weight from white at zero to dark red at maximum. Dashed vertical lines mark span boundaries: red for the chosen tool, gray for other tool spans, and blue for the user-turn region.

at the commit position, yielding two functional roles (Table 1).

The dominant role is *intent-matching* (15 of 24 heads): at the commit step these heads place nearly all attention on the chosen tool's name span (L29H11 puts ≈1.0 of its mass there), directly encoding the selection decision. The remaining 9 heads are *query-context* heads: they concentrate on the user-query tokens and carry query information rather than a tool-selection signal. Figure 3 shows one representative head from each of the two roles present in top-24. The intent-matching panel is the one that exposes the selection decision and that PMAShield monitors in §4. Directed path patching (Detailed in Appendix A, Figure 6) reveals a strictly feed-forward flow in which mid-layer intent-matching heads (e.g., L24H29, L29H11) drive the attention of late-layer heads (e.g., L32H2, L33H17), consistent with selection information being aggregated in the mid layers and committed in the L31–L33 band.

3.5 Finding 4: PMAs Cause Intent-Matching Head Disagreement

We now investigate how PMAs disrupt the selection circuit. Our hypothesis is that injected manipulative tokens attract attention from a *subset* of intent-matching heads toward the poisoned tool, while other heads retain their focus on the semantically correct tool. This produces an inter-head disagreement signal that is absent in benign inputs.

Experiment. We take 1,293 paired benign / MCPTox-attacked (tool, query) scenarios for which both sides produce a parseable tool call. For each sample we normalize every selection head's attention over the competing tools to a probability vector and compute the four disagreement metrics defined in §4.4: the mean pairwise inter-head agreement A, the mean per-head entropy E, the mean pairwise JS divergence D_{JS}, and the output-alignment score O. Higher A and O, and lower E and D_{JS}, indicate that the heads concur on a single tool.

Table 2: Disagreement metrics under benign vs. MCPTox-attacked inputs. Arrows indicate the expected direction under attack. δ is Cliff's effect size (Cliff, 1993).

Metric	Benign	Attacked	δ
Agreement A (↓)	0.75	0.67	−0.17
Entropy E (↑)	0.74	0.91	+0.18
JS div. D_{JS} (↑)	0.095	0.105	+0.07
Output-align O (↓)	0.57	0.39	−0.28

Table 2 supports the hypothesis in aggregate: relative to benign inputs, attacked inputs show lower inter-head agreement (A), higher entropy (E) and pairwise divergence (D_{JS}), and fewer heads aligned with the model's (manipulated) output (O). Every shift is in the predicted direction and highly significant, but the effect sizes are moderate (Cliff's δ up to 0.28) and the benign and attacked distributions overlap rather than separate cleanly. No single metric is therefore decisive on its own. Instead, the four metrics are complementary, which motivates combining them in a lightweight learned detector (§4) rather than thresholding any one of them.

4 PMAShield: Detection via Head Disagreement

4.1 Overview

Finding 4 (§3.5) showed that PMAs induce systematic disagreement among tool-selection heads. PMAShield turns this signal into a detector. It operates in two phases. The *offline* phase (performed once per model on a clean calibration set) discovers the set of tool-selection heads. The *online* phase (at inference time) captures attention patterns for the current input and computes four disagreement metrics fed to a logistic regression classifier.

4.2 Problem Formalization

Tool-calling setting. An LLM f receives a user query q and a candidate tool set $\mathcal{T} = \{t_1, \ldots, t_n\}$, where each tool t_i is described by metadata m_i

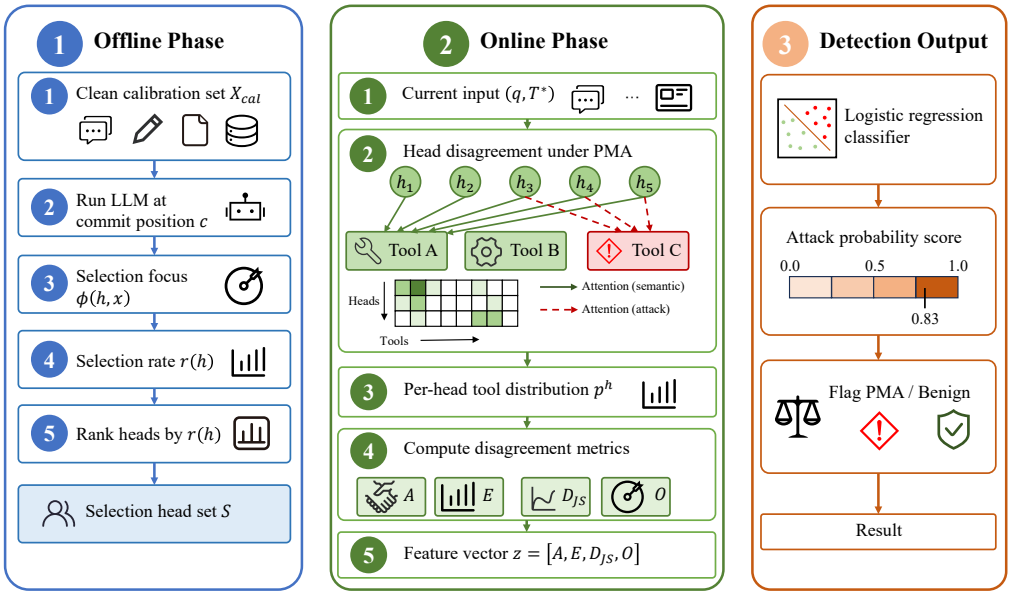

Figure 4: Overview of PMAShield. **Offline phase**: a benign calibration set is used to compute each attention head's selection rate $r(h)$. Heads exceeding threshold τ are retained as the selection head set $\mathcal{S}$. **Online phase**: at inference time, the four disagreement metrics are computed over $\mathcal{S}$ from the model's commit-step attention and fed to a logistic regression classifier.

(name, description, parameter schema). The model produces a selection $f(q, \mathcal{T}) = t^* \in \mathcal{T}$.

Preference Manipulation Attack (PMA). An adversary targets a tool $t_a \in \mathcal{T}$ by replacing its metadata m_a with m'_a, which appends adversarial tokens (persuasive text, authority claims, or rank-injection instructions). Let $\mathcal{T}' = (\mathcal{T} \setminus \{t_a\}) \cup \{t'_a\}$ denote the manipulated tool set. The attack *succeeds* if $f(q, \mathcal{T}') = t_a$ while $f(q, \mathcal{T}) \neq t_a$.

Detection task. A detector $D\colon (q, \mathcal{T}') \to \{0, 1\}$ must flag manipulated inputs at inference time using only $(q, \mathcal{T}')$ and the model's internal states, without access to the clean tool set $\mathcal{T}$ or labeled attack examples.

Attention notation. Let the model have L layers and H attention heads per layer, and write $h = (\ell, k) \in [L] \times [H]$ for a specific head. Let c denote the *commit position*, the autoregressive step at which the model begins generating the tool-call output. For head h, let $\boldsymbol{\alpha}^h \in \Delta^S$ be the post-softmax attention vector from position c over a context of length S. For tool t_i with token span $\mathcal{I}_i \subseteq [S]$, define the *attention mass*:

$$a^h(t_i) = \sum_{s \in \mathcal{I}_i} \alpha^h_s. \tag{1}$$

4.3 Offline: Selection Head Discovery

Given a calibration set $\mathcal{X}_{\text{cal}}$ of N benign tool-selection examples, we quantify each head's *selection focus*:

$$\phi(h, x) = \frac{a^h(t^*(x))}{\sum_{t \in \mathcal{T}(x)} a^h(t)}, \tag{2}$$

where $t^*(x)$ is the model's selection on input x and $\mathcal{T}(x)$ is the corresponding tool set. Head h *agrees* with the model's selection on x when its attention argmax over all tools equals $t^*(x)$. The *selection rate* of head h is the fraction of calibration examples on which it agrees:

$$r(h) = \frac{1}{N} \sum_{x \in \mathcal{X}_{\text{cal}}} \mathbf{1}\left[\arg\max_{t \in \mathcal{T}(x)} a^h(t) = t^*(x)\right]. \tag{3}$$

We apply a threshold τ to obtain the *selection head set*:

$$\mathcal{S} = \{h \in [L] \times [H] : r(h) \geq \tau\}, K = |\mathcal{S}|. \tag{4}$$

4.4 Online: Disagreement Metrics

At inference time, for input $(q, \mathcal{T}')$, let $\mathbf{p}^h \in \Delta^{|\mathcal{T}'|}$ be the normalised attention distribution of head $h \in \mathcal{S}$ over tools:

$$p^h_t = \frac{a^h(t)}{\sum_{t' \in \mathcal{T}'} a^h(t')}. \tag{5}$$

We define four scalar disagreement metrics over $\mathcal{S}$:

Agreement (A). Mean pairwise cosine similarity of head distributions:

$$A = \frac{2}{K(K-1)} \sum_{i<j} \cos\left(\mathbf{p}^{h_i}, \mathbf{p}^{h_j}\right). \quad (6)$$

Low A indicates heads point to different tools.

Entropy (E). Mean Shannon entropy over selection heads:

$$E = \frac{1}{K} \sum_{h \in \mathcal{S}} \mathcal{H}\left(\mathbf{p}^h\right). \quad (7)$$

High E indicates each head is uncertain about which tool to prefer.

Jensen-Shannon Divergence (D_{JS}). Mean pairwise JS divergence over soft attention distributions:

$$D_{\text{JS}} = \frac{2}{K(K-1)} \sum_{i<j} \text{JS}\left(\mathbf{p}^{h_i} \| \mathbf{p}^{h_j}\right). \quad (8)$$

High D_{JS} indicates distributional fragmentation across heads.

Output-alignment (O). Let t^* be the tool the model selects on this input. The output-alignment score is the fraction of selection heads whose attention argmax matches that choice:

$$O = \frac{1}{K} \sum_{h \in \mathcal{S}} \mathbf{1}\left[\arg\max_t p_t^h = t^*\right]. \quad (9)$$

A low O means few heads back the selected tool, the signature of a hijacked subset.

4.5 Detection Classifier

We concatenate the four metrics into a 4-dimensional feature vector $\mathbf{z} = [A, E, D_{\text{JS}}, O]$ and fit a logistic regression classifier. Training follows a *Leave-One-Server-Out* (LOSO) protocol: for each evaluation fold, all examples from one MCP server are held out, and the classifier is trained on the remaining servers. This protocol prevents server-level leakage and mirrors deployment conditions where new tool providers are unseen at training time. The classifier outputs an attack probability score. We report AUROC as the primary metric.

Design rationale. The 4-dimensional feature space makes logistic regression the natural choice: it generalizes well under LOSO with small per-fold training sets, and its coefficients show which metrics drive detection (no single metric dominates; see the ablation in §5.4).

Table 3: Attack datasets used for evaluation.

Dataset	Pairs
MCPTox (Wang et al., 2026a)	1,348
MPMA-DPMA (Wang et al., 2026b)	209

5 Experiments

We evaluate PMAShield across six model architectures and two attack families. We address three research questions:

- **RQ1** (§5.2): Does PMAShield effectively detect PMAs across diverse attack families and model architectures?
- **RQ2** (§5.3): Does the head disagreement signal transfer across attack families that are unseen during calibration?
- **RQ3** (§5.4): Which components of the detector drive performance?

5.1 Setup

Models. We evaluate on six instruction-tuned LLMs spanning four model families: Qwen3-4B and Qwen3-8B (Yang et al., 2025), Phi-4-mini-instruct (Abouelenin et al., 2025), Gemma-3-4B-it (Team, 2025) and Gemma-4-E4B-it (Google DeepMind, 2026), and Llama-3.1-8B-Instruct (Grattafiori et al., 2024). All experiments are conducted on a single NVIDIA A100 GPU.

Attack datasets. We evaluate on two PMA attack families (Table 3):

- *MCPTox* (Wang et al., 2026a): description-level injection across 45 real MCP servers spanning 10 risk categories (privacy leakage, financial loss, infrastructure damage, etc.). Each attacked scenario replaces a benign tool description with an adversarially crafted one designed to steer the model toward a harmful tool while leaving the user query unchanged.
- *MPMA-DPMA* (Wang et al., 2026b): injects persuasive strategy tokens such as ratings and fabricated claims into tool names and descriptions.

More details on the datasets are in Appendix B.

Baselines.

- *Random*: chance baseline that assigns each sample a uniform random score in $[0, 1]$.
- *Perplexity filter*: flags tool descriptions whose perplexity under a small language model exceeds a calibrated threshold.

Table 4: PMAShield detection performance (LOSO) across six model architectures and two attack families.

	MCPTox		MPMA-DPMA	
Model	AUROC	F1	AUROC	F1
Qwen3-8B	0.741	0.733	0.689	0.632
Qwen3-4B	0.760	0.738	0.676	0.731
Phi-4-mini	0.660	0.702	0.582	0.661
Llama-3.1-8B	0.738	0.726	0.696	0.722
Gemma-3-4B	0.699	0.739	0.705	0.713
Gemma-4-E4B	0.677	0.740	0.754	0.710
Mean	0.713	0.730	0.684	0.695

Table 5: Comparison against baselines (AUROC); PMAShield on Qwen3-8B.

Detector	**MCPTox**	**MPMA-DPMA**
Random	0.488	0.494
Perplexity filter	0.503	0.621
LLM-guard	0.502	0.467
PMAShield	**0.741**	**0.689**

- *LLM-guard*: Protect AI's DeBERTa-v3-base prompt injection scanner (ProtectAI, 2023), applied per tool description. Returns a calibrated risk score in $[0, 1]$, and the maximum across tools is used.

Implementation. Selection heads are identified once per model from a shared benign calibration set of 200 tool-selection examples collected from Smithery AI (Smithery AI, 2024) using threshold-based classification (§4). The same head set is reused across both attack family evaluations for that model. The hyperparameters including the selection threshold τ and head-count value K used for each model are reported in Appendix D.

Metric. We report AUROC as the primary metric, and also report F1 at the best operating threshold on the training set.

5.2 RQ1: Attack Detection Effectiveness

Setting. We evaluate all six models on each of the two attack datasets under the LOSO protocol: for each fold, examples from one MCP server are held out as the test set and the logistic classifier is trained on the remaining servers. Selection heads are discovered independently per model from the shared benign calibration set and held fixed across both attack families. Table 4 reports AUROC for all six models, demonstrating that PMAShield generalizes across architectures and Table 5 compares against all baselines.

Results. PMAShield detects both attack families on every model, averaging 0.71 AUROC on MCPTox and 0.68 on MPMA-DPMA without any labeled attack data or per-domain tuning (Table 4). It outperforms all baselines on both attacks (Table 5). The perplexity filter and the LLM-guard scanner hover near chance: the injection signal is semantic, not a surface-level lexical or fluency artifact, so filters that never inspect the selection decision miss it. PMAShield, by contrast, reads the model's internal selection state directly and therefore generalizes across both attack families and all six architectures.

Effect of model architecture and size. Performance varies by architecture rather than model size. Within the Qwen3 family, Qwen3-4B matches or slightly exceeds Qwen3-8B, indicating that a larger parameter count does not translate directly into a stronger disagreement signal. Phi-4-mini is the weakest performer on both attacks, likely because its hybrid sparse-attention architecture produces less specialised selection heads, making the disagreement signal noisier. The two Gemma variants show an inverted pattern across attacks: Gemma-3-4B edges Gemma-4-E4B on MCPTox, while Gemma-4-E4B leads by a large margin on MPMA-DPMA, suggesting that the updated Gemma-4 develops heads that are more sensitive to the strategy-token style of MPMA.

Effect of attack dataset. MCPTox consistently yields higher AUROC than MPMA-DPMA for most models (mean gap $\approx$0.03), for two reasons. First, MCPTox provides roughly $6\times$ more training pairs per LOSO fold, giving the logistic classifier a stronger signal. Second, MCPTox's description-level injection produces a cleaner shift in commit-step attention than MPMA-DPMA's strategy-token style, yielding a stronger disagreement signal. The notable exception is Gemma-4-E4B, which achieves its best result on MPMA-DPMA (0.754), pointing to architecture-specific sensitivity to persuasion-style tokens that warrants further investigation. Per-fold standard deviations and a per-risk-category AUROC breakdown for Qwen3-8B are provided in Appendix C.

5.3 RQ2: Cross-Attack Transferability

Setting. We train the classifier on examples from one attack family and evaluate it directly on the other held-out family, using the same selection head set as RQ1. This tests whether the head disagreement signal is specific to one attack strategy or reflects a shared property of the selection circuit. Table 6 shows the 2×2 transfer AUROC matrix.

Table 6: Cross-attack transfer AUROC.

train \ eval	MCPTox	MPMA-DPMA
MCPTox	0.650	0.573
MPMA-DPMA	0.598	0.588

Results. Cross-attack transfer stays above chance in both directions but below the in-distribution operating points of RQ1, indicating that the disagreement signal is partly, though not fully, shared across attack families. Furthermore, both eval-on-MCPTox scores exceed their eval-on-MPMA-DPMA counterparts, a pattern that mirrors the RQ1 asymmetry. MCPTox's description-level injection displaces the commit-step attention by a larger margin than MPMA-DPMA's strategy-token perturbations, which are shorter and more localized. This yields a stronger disagreement signal in the MCPTox eval column regardless of which attack family was used for training. In effect, MPMA-DPMA attacks are intrinsically harder to detect, while MCPTox attacks remain detectable even when the classifier has been calibrated on a different attack style.

5.4 RQ3: Ablation Study

Setting. We conduct two ablations on Qwen3-8B under the same LOSO protocol as RQ1. First, we replace the joint four-feature logistic classifier with a single-feature classifier trained on each disagreement metric individually, measuring the AUROC contribution of A, E, D_{JS}, and O in isolation. Second, we sweep the selection threshold $\tau \in \{0.1, 0.2, \ldots, 0.5\}$ to assess how the size of $\mathcal{S}$ affects detection performance. [1]

Metric contribution. Table 7 reports the AUROC of each disagreement metric used on its own. No single metric is strong, confirming that the four are complementary and that the joint classifier, which reaches 0.741/0.689, is what makes the signal usable. Notably, A, E, and O score higher on MCPTox than on MPMA-DPMA, whereas D_{JS} is reversed. $A/E/O$ are argmax-level metrics sensitive to heads that fully switch their modal attention to the malicious tool. MCPTox's bulk description replacement causes exactly this kind of sharp shift. D_{JS} instead measures pairwise *distribution-shape* divergence across heads: MPMA-DPMA's multiple short strategy tokens bias different heads in subtly different directions, producing diffuse distribution deformations that elevate pairwise JS even when no head changes its argmax decisively.

[1] Some models (e.g., Phi-4-mini) do not have any heads exceeding the higher thresholds, so we report only the non-empty points.

Table 7: AUROC of each disagreement metric.

Metric	MCPTox	MPMA-DPMA
Agreement A	0.621	0.570
Entropy E	0.619	0.571
JS divergence D_{JS}	0.566	0.583
Output-alignment O	0.634	0.562
All four (joint)	**0.741**	**0.689**

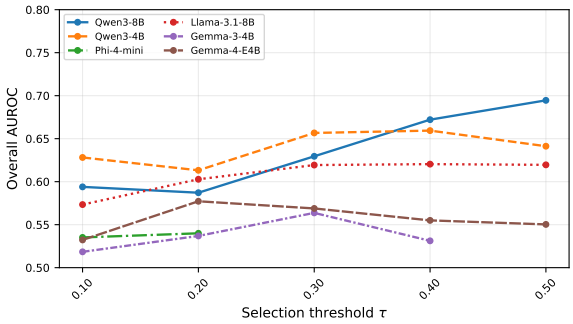

Figure 5: Detection AUROC as a function of the selection threshold τ.

Selection threshold sensitivity. The optimal τ varies substantially across architectures: models with dense selection-head populations such as Qwen3-4B and Llama-3.1-8B tolerate higher thresholds, while architectures with fewer specialised heads (Phi-4-mini and Gemma-3-4B) require lower thresholds to avoid collapsing $\mathcal{S}$ to an empty set. Gemma-4-E4B reaches its optimum at the lowest threshold, suggesting that its selection heads operate at a lower but broader attention concentration regime.

6 Conclusion

We presented PMAShield, a white-box detector for preference manipulation attacks that monitors the model's internal tool-selection circuit rather than surface-level text features. Mechanistic analysis on Qwen3-8B reveals that PMAs partially hijack a specialized set of intent-matching heads, producing a measurable inter-head disagreement signal absent in benign inputs. PMAShield operationalises this signal with four complementary metrics and a logistic classifier. It needs no fine-tuning, no labeled attack data, and no model modification, and it averages 0.71/0.68 AUROC across six LLMs and two attack families (up to 0.76) while outperforming all baselines.

7 Limitations

Single-model circuit analysis. The mechanistic analysis is conducted on Qwen3-8B; we do not directly verify that the same head-level circuit structure recurs in the other five models. All models studied lie in the 3B–15B parameter range, and the circuit may differ in larger models ($\geq$70B).

Adaptive attackers. An adaptive attacker with knowledge of the detector's selection head set could craft injections that target fewer heads, suppressing the disagreement signal. Our threat model assumes the attacker does not have access to the head set.

Reasoning models. PMAShield reads attention at the commit position immediately before the tool call. In reasoning models that emit a chain-of-thought before invoking a tool, this position is harder to identify reliably, and the disagreement signal may be diluted by the preceding reasoning steps.

White-box requirement. PMAShield requires access to the model's internal attention weights and is therefore inapplicable to black-box API deployments where only the generated text is observable. Extending the disagreement-based detection principle to black-box settings remains an open problem.

Model-dependent performance. Detection quality varies with the underlying model architecture, as reflected in the range of per-model AUROC values (0.58–0.76) observed in our experiments. Models that develop fewer or less specialised selection heads (e.g., Phi-4-mini, Gemma-3-4B) yield weaker disagreement signals, and the method may offer limited protection when deployed on architectures not represented in our evaluation.

8 Ethical Considerations

Intended use. PMAShield is a purely defensive tool whose sole function is to flag potentially manipulated tool-calling inputs at inference time. It does not generate, assist, or amplify any attack capability.

Dual-use risk. Our analysis exposes which attention heads govern tool selection, which could in principle inform targeted adversarial attacks.

Data. All experiments use synthetic tool descriptions drawn from published benchmarks (Wang et al., 2026a,b). No personally identifiable information is present in any dataset used in this work.

Large language models usage. We employ Claude Code (Anthropic, 2025) to assist in writing and coding the paper. We carefully review and verify all AI-generated content for accuracy and originality.

References

Abdelrahman Abouelenin, Atabak Ashfaq, Adam Atkinson, Hany Awadalla, Nguyen Bach, Jianmin Bao, Alon Benhaim, Martin Cai, Vishrav Chaudhary, Congcong Chen, and 1 others. 2025. Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-loras.

Anthropic. 2024. Model context protocol. https://modelcontextprotocol.io.

Anthropic. 2025. Claude code. https://claude.ai/code.

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. 2024. Refusal in language models is mediated by a single direction. volume 37, pages 136037–136083.

Norman Cliff. 1993. Dominance statistics: Ordinal analyses to answer ordinal questions. *Psychological bulletin*, 114(3):494.

Google DeepMind. 2026. Gemma 4 model card. https://ai.google.dev/gemma/docs/core/model_card_4.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models.

Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. 2023. Not what you've signed up for: Compromising real-world llm-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM workshop on artificial intelligence and security*, pages 79–90.

Yilong Liu, Xixun Lin, Pengfei Cao, Ge Zhang, Fang Fang, and Yanan Cao. 2026. Do llms know tool irrelevance? demystifying structural alignment bias in tool invocations.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. volume 35, pages 17359–17372.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, and 1 others. 2022. In-context learning and induction heads.

OpenAI. 2023. Function calling — OpenAI API documentation. https://platform.openai.com/docs/guides/function-calling.

ProtectAI. 2023. LLM Guard: The security toolkit for LLM interactions. https://github.com/protectai/llm-guard.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, and 1 others. 2024. Toolllm: Facilitating large language models to master 16000+ real-world apis.

Smithery AI. 2024. Smithery: MCP server registry. https://smithery.ai.

Hariom Tatsat and Ariye Shater. 2026. Beyond the black box: Interpretability of agentic ai tool use.

Gemma Team. 2025. Gemma 3 technical report. *Preprint*, arXiv:2503.19786.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2022. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small.

Zhiqiang Wang, Yichao Gao, Yanting Wang, Suyuan Liu, Haifeng Sun, Haoran Cheng, Guanquan Shi, Haohua Du, and Xiangyang Li. 2026a. Mcptox: A benchmark for tool poisoning on real-world mcp servers.

Zihan Wang, Rui Zhang, Yu Liu, Wenshu Fan, Wenbo Jiang, Qingchuan Zhao, Hongwei Li, and Guowen Xu. 2026b. Mpma: Preference manipulation attack against model context protocol.

Wenpeng Xing, Zhonghao Qi, Yupeng Qin, Yilin Li, Caini Chang, Jiahui Yu, Changting Lin, Zhenzhen Xie, and Meng Han. 2025. Mcp-guard: A defense framework for model context protocol integrity in large language model applications.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report.

Aojie Yuan, Zhiyuan Su, and Yue Zhao. 2026. Aegis: No tool call left unchecked–a pre-execution firewall and audit layer for ai agents.

Qiusi Zhan, Richard Fang, Henil Shalin Panchal, and Daniel Kang. 2025. Adaptive attacks break defenses against indirect prompt injection attacks on llm agents. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 7101–7117.

Haiyue Zhang, Yi Nian, and Yue Zhao. 2026a. Agent audit: A security analysis system for llm agent applications.

Haonan Zhang, Dongxia Wang, Yi Liu, Kexin Chen, and Wenhai Wang. 2026b. Llm-va: Resolving the jailbreak-overrefusal trade-off via vector alignment.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, and 1 others. 2023. Representation engineering: A top-down approach to ai transparency.

A Additional Interpretability Results

Tool-selection commit position. The mechanistic analysis in §3 requires reading the model state at the step where it commits to a tool name. We obtain this by teacher-forcing the chat-template prompt with the model's tool-call prefix (for Qwen3, `<think></think><tool_call>{"name": "`) so that the next token is the first token of the chosen tool's name. The commit position is then the final prompt token, and the contrastive log-odds is computed over the two tools' first name tokens *in this JSON context* (which can differ from their standalone tokenization). Head-level patching replaces the per-head slice of the attention output projection input at the commit position, while layer-level patching replaces the whole attention or MLP output at that position.

Circuit interaction matrix. To probe directed interactions, we use *path patching* (Wang et al., 2022): for an ordered head pair (A, B) with $\text{layer}(A) < \text{layer}(B)$, we replace upstream head A's output-projection input vector with its value from the contrastive scenario and measure the mean absolute change in downstream head B's commit-row attention pattern, $\overline{|\Delta\alpha_B|}$, averaged over both directions. Figure 6 reports this directed strength among the top-16 causal heads, ordered by layer. The matrix is *strictly upper-triangular*: every control cell with $\text{layer}(A) \geq \text{layer}(B)$ is exactly zero, confirming that influence propagates only forward through the residual stream. The flow is clearly staged: mid-layer heads are the dominant *sources* (L24H29, L29H11, L25H14, L28H23 drive the largest downstream changes), while the late-layer heads are the dominant *sinks* (L32H2, L33H17, L34H16/H17 have their attention most reshaped by upstream heads). The strongest individual links are short, near-adjacent-layer hops (L29H11→L30H5/H7, L24H29→L25H30, L31H28/L31H30→L32H2), which chain into a multi-stage relay (e.g., L24H29 → L25H30 → L28H23 → L31H30 → L32) that carries the selection signal from the mid layers into the L32–L34 commit band.

B Dataset Details

MCPTox attacks are labeled by the adversary's intended harm. Table 8 gives the per-category pair counts used in our evaluation (a "pair" is one benign scenario plus its attacked counterpart for which both sides produce a parseable tool call).

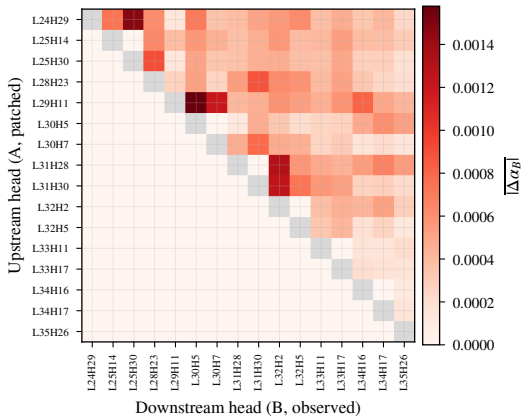

Figure 6: Directed path-patching strength among the top-16 causal heads (Qwen3-8B): mean absolute change $\overline{|\Delta\alpha_B|}$ in the downstream head's (column B) attention when the upstream head (row A) is patched. Brighter cells indicate stronger directed influence.

MPMA-DPMA is a single promotional-decoration category and is not further subdivided.

Table 8: MCPTox evaluation pairs by intended risk category (Qwen3-8B, parseable pairs).

Risk category	Pairs
Privacy leakage	273
Information manipulation	267
Service disruption	205
Data tampering	120
Infrastructure damage	115
Credential leakage	109
Code injection	59
Instruction tampering	57
Financial loss	55
Message hijacking	44
Total	1,304

C Detailed Experimental Results

Table 9 reports the per-model LOSO AUROC with its across-fold standard deviation alongside the best-threshold F1. The standard deviations are sizeable because each LOSO fold is a single held-out MCP server, and per-server difficulty varies widely.

Table 10 breaks the Qwen3-8B MCPTox result down by the attack's intended risk category. Detection is strongest on goal-directed manipulations (instruction tampering, financial loss) and weakest on categories whose poisoned descriptions are closest to legitimate tool text (privacy/credential leakage, code injection).

The cross-attack transfer matrix (Table 6) is computed on Qwen3-8B. Per-model transfer matrices are left to future work.

Table 9: Per-model detection AUROC (mean $\pm$ std across LOSO folds) and best-threshold F1 for both attack families.

	MCPTox		MPMA-DPMA	
Model	AUROC	F1	AUROC	F1
Qwen3-8B	$0.741\pm.18$	0.733	$0.689\pm.25$	0.632
Qwen3-4B	$0.760\pm.17$	0.738	$0.676\pm.27$	0.731
Phi-4-mini	$0.660\pm.22$	0.702	$0.582\pm.33$	0.661
Llama-3.1-8B	$0.738\pm.22$	0.726	$0.696\pm.22$	0.722
Gemma-3-4B	$0.699\pm.20$	0.739	$0.705\pm.19$	0.713
Gemma-4-E4B	$0.677\pm.21$	0.740	$0.754\pm.27$	0.710

Table 10: Qwen3-8B MCPTox detection AUROC by risk category (LOSO, mean $\pm$ std). n is the number of benign/attacked pairs.

Risk category	n	AUROC
Instruction tampering	57	$0.877\pm.19$
Financial loss	55	$0.839\pm.13$
Service disruption	205	$0.778\pm.22$
Information manipulation	267	$0.749\pm.19$
Infrastructure damage	115	$0.716\pm.32$
Data tampering	120	$0.695\pm.18$
Message hijacking	44	$0.689\pm.35$
Credential leakage	109	$0.600\pm.29$
Code injection	59	$0.595\pm.29$
Privacy leakage	273	$0.587\pm.25$

D Hyperparameter Settings

Table 11 reports the threshold and head-count hyperparameters used for each model in the main experiments. All thresholds were determined by grid search on a held-out benign calibration set of 200 tool-selection examples. No attack data was used during this search.

Table 11: Threshold and head-count hyperparameters per model. τ_{sel}: selection rate threshold; τ_{ui}: user-intent attention threshold; τ_{to}: tool-output attention threshold; K: resulting number of selection heads.

Model	τ_{sel}	K
Qwen3-8B	0.50	53
Qwen3-4B	0.40	157
Phi-4-mini-instruct	0.25	11
Llama-3.1-8B	0.45	168
Gemma-3-4B	0.35	5
Gemma-4-E4B	0.20	110