# Shadow in the Cache: Unveiling and Mitigating Privacy Risks of KV-cache in LLM Inference

Zhifan Luo[1,2], Shuo Shao[1,3], Su Zhang[2], Lijing Zhou[2], Yuke Hu[1,3], Chenxu Zhao[1,3], Zhihao Liu[1,3], Zhan Qin[1,3]
[1]State Key Laboratory of Blockchain and Data Security, Zhejiang University [2]Huawei Technology
[3]Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security
{luozhifan, shaoshuo_ss, yukehu, zhaocx_7, zhihao_liu, qinzhan}@zju.edu.cn; {zhangsu14, zhoulijing}@huawei.com

***Abstract*—The Key-Value (KV) cache, which stores intermediate attention computations (Key and Value pairs) to avoid redundant calculations, is a fundamental mechanism for accelerating Large Language Model (LLM) inference. However, this efficiency optimization introduces significant yet underexplored privacy risks. This paper provides the first comprehensive analysis of these vulnerabilities, demonstrating that an adversary can reconstruct sensitive user inputs directly from the KV-cache. We design and implement three distinct attack vectors: a direct Inversion Attack, a more broadly applicable and potent Collision Attack, and a semantic-based Injection Attack. These methods demonstrate the practicality and severity of KV-cache privacy leakage issues. To mitigate this, we propose KV-Cloak, a novel, lightweight, and efficient defense mechanism. KV-Cloak uses a reversible matrix-based obfuscation scheme, combined with operator fusion, to secure the KV-cache. Our extensive experiments show that KV-Cloak effectively thwarts all proposed attacks, reducing reconstruction quality to random noise. Crucially, it achieves this robust security with virtually no degradation in model accuracy and minimal performance overhead, offering a practical solution for trustworthy LLM deployment.**

## I. Introduction

Large Language Models (LLMs) have ignited a paradigm revolution in artificial intelligence [24], [36], profoundly impacting various domains and applications, such as machine translation [44], chatbots [40], code generation [14], and content creation [47]. However, the immense scale of these models, characterized by billions or even trillions of parameters, coupled with the need to process extensive input sequences and engage in multi-turn dialogues, presents a substantial challenge to their efficient deployment and inference. This computational demand often translates into high latency and resource consumption, hindering broader accessibility and real-time applicability [22].

To address the efficiency bottlenecks in LLM inference, researchers have proposed several optimization techniques [38], [49]. Among these, the **Key-Value cache (KV-cache)** mechanism has emerged as a crucial and widely adopted solution [28], [47]. During the autoregressive generation process typical of LLMs, the attention mechanism computes key (K) and value (V) matrices for each token based on its preceding tokens. The KV-cache stores these intermediate attention computations (the K and V pairs) for tokens that have already been processed within the input sequence. By reusing these cached K and V pairs for the generation of subsequent tokens, the KV-cache significantly reduces redundant computations. This dramatically accelerates inference speed and improves throughput, especially for tasks involving long contexts or interactive sessions, making LLMs more practical for real-world deployment.

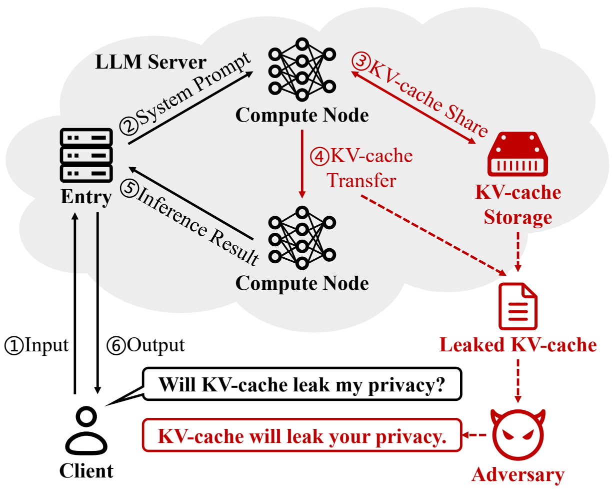


Fig. 1: Overview of the privacy-preserving LLM inference workflow and the associated KV-cache leakage threat model. While user-server communication is encrypted (black lines), the KV-cache is often transmitted and stored in plaintext (red dashed lines), creating a surface for privacy attacks.

However, the storage and potential sharing of the KV-cache introduce significant yet underexplored privacy concerns [39], [41], as illustrated in Figure 1. This vulnerability stems from a critical trade-off made in production systems: while end-user communication with the LLM service is typically encrypted (represented by the black lines), the KV-cache itself is almost always processed, transmitted between compute nodes, and persisted in **plaintext**. This design choice is a concession to performance, as the unacceptable latency overhead from cryptographically securing the often gigabyte-scale KV-cache would violate the stringent demands of real-time inference. Notably, this risk is amplified in the emerging paradigm of confidential Model-as-a-Service (MaaS). Here, TEE adoption for inference privacy [5], [10] introduces a critical architectural trade-off: high-throughput architectures require *deliberately externalizing* the massive KV-cache from the TEE's protection boundary. This architectural design—not a vulnerability—directly exposes the KV-cache to the Cloud Service Provider (CSP), who inherently possesses it without requiring any additional exploit [43].

Crucially, the direct correlation between the KV-cache and user inputs [7], [21], combined with the architectural realities that expose it in plaintext, creates a severe and practical attack vector. A critical research gap therefore exists in mitigating these risks, necessitating a practical and secure defense mechanism.

In this paper, to bridge the research gap, we present the first comprehensive study on the privacy risks of KV-cache in LLM inference. Specifically, we primarily study and answer the following two research questions:

- **RQ1: Is an adversary able to reconstruct user inputs from the KV-caches?**

To address this question, we conduct a systematic investigation, demonstrating that attacks against the KV-cache are not only feasible but also diverse and broadly applicable. We design and implement three distinct classes of targeted privacy-stealing attacks. Each attack exposes inherent privacy risks of KV-caches from a different perspective.

We first explore two relatively direct attack paths. The first is the **KV-cache Inversion Attack**, a method that attempts to leverage known model weight matrices to directly reverse-calculate the input from the KV-cache. The second, more general-purpose approach is the **KV-cache Collision Attack**. This attack reframes input reconstruction as a matching task based on forward computation: an adversary iteratively uses a local model instance to generate KV-caches for candidate inputs and compares them against the intercepted target KV-cache to find a match. Because this method does not rely on any reverse computation, it has broader applicability. Finally, we introduce a novel, semantic-based **KV-cache Injection Attack**. The core of this attack is to leverage the LLM's powerful capability to understand and execute instructions. By appending a specific instruction, such as "Repeat the previous content," to the end of an intercepted KV-cache context, an adversary can induce the model to "echo" or generalize the core information held within the KV-cache.

Collectively, these attacks reveal that privacy leakage from the KV-cache is not merely a theoretical concern. Their diversity and feasibility constitute a significant threat to real-world LLM-based applications, underscoring the urgent necessity of designing specialized and efficient privacy-preserving mechanisms for the KV-cache.

- **RQ2: Can defenders effectively and efficiently mitigate or prevent user privacy leakage from the KV-cache?**

In response to this challenge, this paper provides an affirmative answer by first analyzing the shortcomings of existing privacy-preserving techniques. Conventional methods like full cryptographic encryption [2] or Homomorphic Encryption (HE) [26] introduce prohibitive computational overhead and latency, rendering them impractical for the high-throughput demands of LLM inference. Meanwhile, applying Differential Privacy (DP) [11] requires adding a level of noise that often severely degrades the model's inference accuracy to an unacceptable degree. Even specialized solutions like KV-Shield [41], while lightweight, suffer from critical security flaws. Their fixed shuffling mechanism, as we analyzed in Section V-A, is vulnerable to statistical analysis and is incompatible with modern LLM architectures that use features like Rotary Positional Embedding (RoPE) [32]. These limitations highlight the urgent need for a novel defense mechanism.

Therefore, we propose KV-Cloak, a lightweight and secure mechanism for KV-cache obfuscation designed to overcome these challenges. At its core, KV-Cloak employs a reversible matrix-based obfuscation scheme that guarantees lossless model accuracy. Its security is multi-layered: it applies secret invertible linear transformations to obscure statistical properties, and critically, introduces a one-time random permutation matrix for each data block. This dynamic permutation prevents adversaries from building stable algebraic relations across multiple queries. To further enhance performance, KV-Cloak utilizes operator fusion, an optimization where a portion of the secret obfuscation matrices is algebraically fused into the LLM's attention layer weights offline. This shifts the primary computational cost away from the latency-sensitive online inference phase, striking an effective balance between robust security, lossless accuracy, and high performance.

Our contributions can be summarized as follows.

- **Revealing the privacy risks of KV-cache in LLM inference:** We systematically investigate the privacy risks of the KV-cache by designing and implementing three distinct attacks, Inversion, Collision, and Injection attacks.
- **Proposing a lightweight and effective method to mitigate privacy leakage**: We propose KV-Cloak, a novel and practical defense mechanism that uses a lightweight, reversible matrix-based obfuscation scheme combined with operator fusion to protect the KV-cache without degrading model accuracy and with minimal performance overhead.
- **Conducting extensive experiments to evaluate attacks and defenses:** We conduct a systematic evaluation to empirically demonstrate and quantify the feasibility of attacks that reconstruct user input from the KV-cache of state-of-the-art LLMs, establishing it as a practical and severe threat. We further prove that our proposed KV-Cloak is a highly practical solution that achieves robust security, near-lossless model fidelity, and high efficiency simultaneously. Our experiments show that KV-Cloak, with its negligible impact on accuracy and minimal latency overhead (mostly $< 10\%$), successfully resolves the stark trade-off between security and utility that plagues alternative approaches such as DP.

## II. Background and Related Work

### A. Transformer-based LLM Inference

Prevailing LLMs, such as LLaMA [35], DeepSeek [24] and Qwen [40], are predominantly based on the Transformer decoder architecture [47].

**Self-Attention Mechanism.** For an input token $x_i$, the self-attention layer generates Query ($q_i$), Key ($k_i$), and Value

$(v_i)$ vectors via linear transformations. Crucially, modern architectures incorporate RoPE, denoted as $R^d_{\Theta,i}$:

$$q_i = x_i W_q^\top R^d_{\Theta,i}, \quad k_i = x_i W_k^\top R^d_{\Theta,i}, \quad v_i = x_i W_v^\top. \tag{1}$$

where $W_{(\cdot)}$ are learnable weight matrices. The attention score $a_{ij}$ is computed as the scaled dot product between $q_i$ and preceding keys $k_{j\leq i}$. The final output $o_i$ is the weighted sum of values projected by $W_o$:

$$a_{ij} = \frac{\exp(q_i {k_j}^\top/\sqrt{d})}{\sum_{t=1}^{i}\exp(q_i {k_t}^\top/\sqrt{d})}, \quad o_i = \left(\sum_{j=1}^{i} a_{ij} v_j\right) W_o^\top. \tag{2}$$

**Autoregressive LLM Inference.** LLM inference models the sequence probability $P(x_1,\ldots,x_n)$ through autoregressive decomposition [6]. The generation process predicts tokens sequentially based on the conditional probability:

$$P(x_1,\ldots,x_n) = \prod_{i=1}^{n} P(x_i|x_{<i}). \tag{3}$$

While effective for capturing dependencies, this sequential nature implies that generating $x_i$ depends on the full history $x_{<i}$. Without caching, the model must re-compute the Key and Value vectors for all preceding tokens at each step. This computational redundancy creates a substantial bottleneck for real-time inference.

**KV-cache in LLM Inference.** To mitigate the computational redundancy of autoregressive generation, LLMs utilize the *KV-cache* mechanism [28]. Instead of recalculating the Key $(K)$ and Value $(V)$ matrices for the entire context window at each step, the system persists these intermediate states in memory. When generating a new token $x_i$, the model only computes the current $q_i, k_i, v_i$, performs attention between $q_i$ and the cached history, and subsequently appends the new $(k_i, v_i)$ pair to the cache. This strategy significantly reduces latency but necessitates managing a persistent state proportional to the sequence length.

### B. Privacy Attacks against LLMs

*1) Inference-Phase Inversion Attacks:* Inference-time privacy attacks are categorized into: (1) **Output-based**, reconstructing inputs from final probabilities or embeddings [19], [25]; and (2) **Intermediate-state-based**, exploiting internal representations [37]. While early works like Vec2Text [25] focused on sentence embeddings, Embed Parrot [37] demonstrated that deep hidden states allow for more accurate reconstruction.

However, current research often overlooks the *KV-cache*. Unlike fused hidden states, the KV-cache retains the primitive Key $(K)$ and Value $(V)$ vectors—the raw inputs to the attention mechanism. Crucially, to support autoregressive generation, the KV-cache is architecturally designed for *persistence*, making it a richer and more exploitable surface than transient computational states.

*2) Limitations of Side-Channel Attacks:* Existing threats to the KV-cache are primarily side-channel attacks, such as PromptPeek [39], which infers prompts by measuring latency variations caused by cache sharing. However, the practicality of such timing attacks is severely mitigated by modern memory management.

Systems like PagedAttention [17] manage the KV-cache in non-contiguous blocks (defaulting to 16 tokens). To trigger a cache hit (and thus a timing signal), an adversary must correctly guess an entire block simultaneously. With a vocabulary size $|V| \approx 10^5$, the search space explodes to $|V|^{16}$, rendering brute-force side-channels computationally infeasible. In contrast, we address the *direct access* threat model. Given that KV-caches are often processed in plaintext for performance, a compromised system allows adversaries to bypass probabilistic guessing and perform precise, per-token collision attacks.

## III. Attack Landscape: Input Reconstruction from KV-cache

While the KV-cache serves as a cornerstone for efficient LLM inference, it concurrently introduces a critical and underexplored privacy surface. Unlike deep hidden states that represent fused semantic information, entries in the KV-cache maintain a direct, element-wise correspondence to the tokens in the user's input sequence. This structural characteristic implies that an adversary gaining access to the KV-cache does not merely obtain abstract embeddings, but potentially holds the raw materials to reverse-engineer the original user prompt.

In this section, we investigate the central research question: *Is an adversary able to reconstruct user inputs from the KV-cache?* We first define a realistic threat model grounded in modern cloud architectures, and subsequently detail three distinct attack vectors designed to exploit this vulnerabilities.

### A. Threat Model

This work focuses on the threat landscape of LLM inference services within a confidential computing paradigm. We consider the inference service provider as the adversary.

**Adversary's Objective.** The primary objective of the adversary is to recover the user input prompt from the accessed KV-cache. We target verbatim reconstruction because it represents the most severe breach of privacy, allowing the extraction of exact credentials, PII, or proprietary logic contained in the input.

**Adversary's Capabilities.** We assume the adversary can obtain both the *KV-cache* and the *model weights*, but does not observe the ephemeral runtime activations within the GPU registers. These assumptions are well-grounded in the realities of current confidential computing inference paradigms:

- **KV-cache Access.** The adversary is capable of obtaining the KV-cache stored by the LLM server. This is a realistic assumption, as high-throughput and scalable cloud-native services *deliberately externalize* the large-scale KV-cache to non-secure memory or persistent storage pools to meet performance demands [43]. We thus consider this access a result of an intentional performance-security trade-off, not a traditional vulnerability.
- **Foundation Model Access.** We assume a gray-box setting where the adversary can access the foundation model weights.

This access is realistic through two primary paths: (1) For closed-source services, the CSP inherently owns the model and possesses the weights. (2) For services built on open-source models, the provider may be required to disclose the base model for licensing compliance, or the model can be identified using fingerprinting techniques [8], [27].

## B. Input Reconstruction Attacks from KV-cache

This section investigates the risk of user input leakage from KV-cache data during LLM inference. We systematically present three attack vectors, namely *Inversion Attack*, *Collision Attack*, and *Injection Attack*, for reconstructing the original user inputs from the KV-cache. These attacks differ in complexity, applicability, and exploited vulnerabilities, collectively illustrating the threat landscape.

*1) KV-cache Inversion Attack:* As illustrated in Figure 2a, the KV-cache Inversion Attack functions as a baseline algebraic adversary. It attempts to mathematically invert the attention mechanism by exploiting the linear projection of Key ($k$) and Value ($v$) vectors.

**Attack Formulation.** Based on the forward pass definition in Eq. (1), an adversary possessing the plaintext KV-cache and the model weights ($W_k, W_v$) can theoretically recover the input state $x_i$ of the attention layer via matrix inversion:

$$x_i = k_i(R^d_{\Theta,i})^{-1}(W_k^\top)^{-1}, \ \ x_i = v_i(W_v^\top)^{-1}. \tag{4}$$

**Architectural and Semantic Constraints.** While algebraically sound, the feasibility of this attack in production environments is severely limited by two fundamental constraints: **(1)** The attack strictly requires the projection matrices $W_k$ and $W_v$ to be square and full-rank **invertible**. This condition holds for legacy Multi-Head Attention (MHA) [36] architectures (e.g., LLaMA-7B). However, modern State-of-the-Art (SOTA) models, including LLaMA-3 [12], Qwen [40], and DeepSeek [24], adopt efficiency optimizations such as Grouped-Query Attention (GQA) [3] or Multi-Head Latent Attention (MLA) [30]. These mechanisms typically project inputs into lower-dimensional subspaces, resulting in non-square matrices where unique inversion is mathematically impossible (i.e., the problem becomes ill-posed). **(2)** the attack's effectiveness is largely confined to the **first decoder layer's** KV-cache, as its inversion directly yields the input sequence's embeddings, allowing for precise mapping back to the vocabulary. In contrast, applying Eq. (4) to deeper layers recovers intermediate hidden states. These states represent semantically fused contextual information rather than discrete tokens. Reversing these deep representations into text is a non-trivial problem that necessitates training auxiliary inversion models [37], thereby increasing the adversary's computational cost and reducing fidelity.

Consequently, while the Inversion Attack serves as a theoretical proof-of-concept for KV-cache leakage, its reliance on specific architectural properties (MHA) and layer positions (Layer 0) limits its utility against modern, deep LLMs. This motivates the need for the more robust attacks.

*2) KV-cache Collision Attack:* Unlike the Inversion Attack which suffers from architectural constraints, the **KV-cache Collision Attack** is a universal, forward-matching adversary applicable to any layer. Instead of algebraically reversing the projection, this attack reframes input reconstruction as a search problem: identifying the token in the vocabulary that, when processed by a local model, yields a KV-cache entry most similar to the intercepted target.

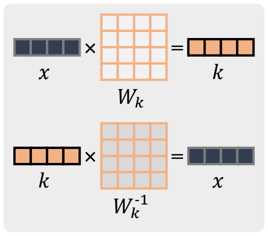
**Fundamental Principle: Reconstruction as Optimization.** The core premise is that the KV-cache exhibits *distance-preserving* properties for identical tokens under the same context. We formalize the attack as finding the optimal token $t^*$ at position $i$ that minimizes the distance metric $\mathcal{D}$ (e.g., Frobenius norm) between the leaked cache $K_{\text{leaked}}$ and the local generation $K_{\text{local}}(t)$:

$$t_i^* = \arg\min_{t\in\mathcal{V}} \mathcal{D}\left(K^{(i)}_{\text{leaked}}, K^{(i)}_{\text{local}}(t|x_{<i})\right). \tag{5}$$

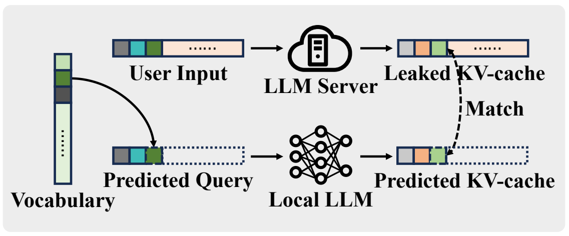
The attack's success hinges on *statistical separability*: the distance for the ground-truth token, $d_{\text{target}}$, must be statistically distinguishable from the distribution of distances for incorrect tokens, $d_{\text{other}}$.

**Attack Procedure.** As illustrated in Figure 2b, the attack proceeds iteratively token-by-token. For each unknown position $i$: **(1)** *Candidate Generation:* The adversary selects candidate tokens from the model's vocabulary $\mathcal{V}$ to test. **(2)** *Local Simulation:* For each candidate token $t$, the adversary performs a forward pass using their local model instance to generate the corresponding KV-cache entry $K^{(i)}_{\text{local}}(t)$. **(3)** *Collision Matching:* The adversary computes the distance $\mathcal{D}$ between the local entry and the intercepted target $K^{(i)}_{\text{leaked}}$. The candidate that minimizes this distance (i.e., creates a "collision" with the leaked state) is identified as the correct token $x_i$, appended to the sequence, and the process advances to $i+1$.

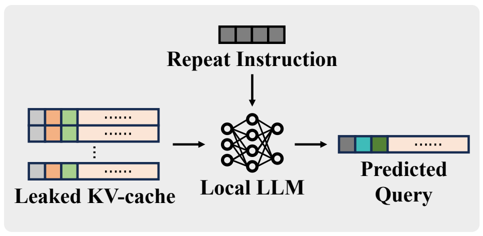
**Implementation Optimizations.** A naive exhaustive search over the entire vocabulary ($\mathcal{V} \approx 10^5$) entails prohibitive latency and VRAM consumption, as verifying the *global minimum* distance requires inferring every candidate token. To render the attack practical, we implement three synergistic optimizations that transform the problem from a global search into an efficient sequential decision process: **(1) Batched Outlier Detection (Enabling Early Exit).** Instead of calculating distances for the full vocabulary to find the minimum, we process candidates in batches. We fundamentally shift the decision logic: rather than comparing a candidate against all other tokens in $\mathcal{V}$, we determine if a candidate constitutes a *statistical outlier* within its current batch (i.e., $d_t < \mu_{\text{batch}} - 3\sigma_{\text{batch}}$). Crucially, this allows for an *early exit*: once a collision is statistically confirmed within a batch, the search terminates immediately without evaluating the remaining vocabulary. **(2) Probability-Guided Prioritization.** To maximize the efficacy of the early exit mechanism, we employ a probability-guided search. We sort the candidate tokens based on the model's predicted probabilities $P(x_i|x_{<i})$. This reordering ensures that the correct token—which typically carries a high probability—is placed in



(a) KV-cache Inversion Attack.



(b) KV-cache Collision Attack.



(c) KV-cache Injection Attack.

Fig. 2: Workflow of the three proposed KV-cache input reconstruction attacks.

the *first few batches*. Consequently, the collision is triggered near the start of the search process, drastically reducing the average number of inferences required. **(3) Search Space Pruning.** Distinct from prioritization, we explicitly truncate the search space by discarding the "long tail" of tokens with negligible probabilities (e.g., keeping only the top-$k\%$ candidates). While prioritization accelerates the average case, pruning bounds the *worst-case complexity*. It prevents the adversary from wasting resources iterating through tens of thousands of implausible tokens in the rare event that the target is not found in the high-probability regions.

**Enhancement via Chosen-Plaintext Attack (CPA).** A critical challenge is determining the optimal decision threshold $\tau$. A static heuristic (e.g., $3\sigma$-rule) is often suboptimal due to the varying entropy of different tokens. However, an adversary with CPA capabilities—who can feed known inputs to the model and observe the resulting KV-cache—can profile the precise statistical behavior of $d_{\text{target}}$ and $d_{\text{other}}$.

This prior knowledge allows for an *adaptive thresholding strategy*. We formulate the success probability for a candidate at rank $r$ as:

$$P(\text{success}|r) = P(d_{\text{other}} > \tau)^{r-1} \times P(d_{\text{target}} < \tau). \quad (6)$$

This formulation explicitly captures the trade-off: a stricter threshold minimizes False Positives (accepting an incorrect predecessor) but risks False Negatives (rejecting the target). By dynamically adjusting $\tau$ based on the expected rank $r$ and the profiled distributions from CPA, the adversary maximizes $P(\text{success})$, significantly boosting reconstruction fidelity compared to static baselines.

*3) KV-cache Injection Attack:* Unlike Inversion and Collision attacks which rely on precise algebraic state matching, the *KV-cache Injection Attack* exploits the semantic instruction-following capabilities of LLMs to exfiltrate private contexts.

**Architectural Constraint: The Missing Query.** Architecturally, a stand-alone KV-cache is computationally dormant. The self-attention mechanism requires a current Query vector ($Q$) to drive the attention scoring (Softmax($QK^\top$)) and state transitions. Consequently, an adversary cannot simply "resume" inference from a stolen cache; a new input stimulus is required to generate the necessary activation state.

**Attack Mechanism: Contextual Hijacking.** To overcome this, we weaponize the LLM's instruction-following nature. As shown in Figure 2c, the adversary appends a crafted directive (e.g., "*Repeat the previous content.*") to the intercepted KV-cache. This injection serves two purposes: **(1) Stimulus Generation.** The directive creates the requisite $Q$ vectors to activate the attention mechanism. **(2) Forced Exfiltration.** The LLM attends to the stolen $K/V$ pairs as historical context. Bound by its alignment training, the model executes the instruction, effectively "echoing" or summarizing the latent private information stored in the cache.

**Strategic Advantages: Robustness and Efficiency.** This attack offers distinct advantages in robustness compared to the Collision Attack. Specifically, it remains effective against KV-cache eviction strategies like H2O [46]. While such lossy compression breaks the strict mathematical correspondence required for algebraic attacks, it typically preserves the *semantic gist* of the context. The Injection Attack successfully exploits these residual semantic vectors to hallucinate or reconstruct private data. Furthermore, the attack is highly efficient, requiring only a single generation pass. This vector underscores that a comprehensive defense must render the KV-cache semantically unintelligible, not just algebraically obfuscated.

## IV. Evaluation of Attacks

### A. Experimental Setup

**Models.** To demonstrate attack universality, we select seven state-of-the-art LLMs spanning diverse parameter scales (1B–8B) and attention mechanisms. Our evaluation prioritizes modern *GQA* architectures, including the LLaMA-3.2 series (1B & 3B-Instruct), LLaMA-3.1-8B [12], and Qwen2.5-Math-7B [40]. Crucially, we incorporate DeepSeek-R1-Distill-LLaMA-8B [13] (LLaMA-3.1-8B-Distilled) to assess robustness against *fine-tuned* models where weight parameters diverge from their base counterparts. Finally, to verify generalizability across architectural evolutions, we evaluate legacy *MHA* models, specifically LLaMA-7B and LLaMA-2-7B.

**Datasets.** To simulate realistic privacy leakage scenarios, we utilize the LMSYS-Chat-1M dataset [48], which comprises real-world user interactions collected from inference services. We construct a test set by randomly sampling 1,000 instances, ensuring a rich variety of dialogue and instruction-following contexts. Analysis of attack generalization across domain-specific datasets is provided in Appendix C3.

**Evaluation Metrics.** BERTScore [45] and ROUGE-L [23] are used to measure the similarity between the original input and the text reconstructed from the KV-cache. BERTScore, based on the `all-mpnet-base-v2` model, is better at capturing semantic similarity, while ROUGE-L primarily reflects lexical-level precision and recall.

### B. *Attack Effectiveness*

This section evaluates the feasibility of reconstructing user inputs via the proposed Inversion, Collision, and Injection attacks. We systematically test these vectors against the models defined in Section IV-A. Detailed ablation studies are provided in Appendix C.

*1) Experimental Settings:* Beyond standard reconstruction, we introduce two specific scenarios to evaluate attack robustness under restricted adversarial knowledge: **(1)** *Fine-tuning Mismatch:* We attack the fine-tuned *LLaMA-3.1-8B-Distilled* assuming the adversary only possesses weights from its base model (*LLaMA-3.1-8B*). This validates effectiveness when exact model weights are unavailable. **(2)** *Cross-Architecture Mismatch:* We employ *LLaMA-7B* parameters to attack *LLaMA-2-7B* to assess generalizability across disjoint architectures.

Regarding data scope, the Inversion and Collision attacks operate on single-layer KV-caches. To characterize layer-wise vulnerability, we evaluate these attacks on the *first*, *middle*, and *last* layers respectively. In contrast, the Injection Attack utilizes the complete, multi-layer KV-cache to leverage the model's full semantic processing capabilities.

*2) Results of KV-cache Inversion Attack:* As detailed in Table I, the efficacy of the Inversion Attack generally hinges on two conditions: **(1)** access to the *first-layer* KV-cache, and **(2)** an algebraically invertible projection matrix, typical of *MHA*. Under these constraints, we achieve near-perfect reconstruction fidelity ($\approx 100\%$), whereas deeper layers yield unintelligible noise ($< 10\%$) due to semantic fusion. Notably, the GQA-based LLaMA-3.2-1B also exhibits high first-layer vulnerability ($\approx 100\%$). We attribute this to the model's high projection rank relative to its hidden state dimension. This distinct property allows the *least squares method* to effectively recover inputs despite the non-square nature of the projection matrix.

**Takeaway 1:** The Inversion Attack is effective against the first-layer KV-cache of MHA models, with exceptions for architectures possessing high-rank projection matrices.

*3) Results of KV-cache Collision Attack:* Unlike the Inversion Attack, the Collision Attack relies on forward-pass matching using the Frobenius norm. As shown in Figure 3, we observe that distances for incorrect tokens ($d_{\text{other}}$) approximate a Gaussian distribution, whereas the target token ($d_{\text{target}}$) manifests as a distinct statistical outlier. By setting an outlier threshold of $3\sigma$ below the batch mean (batch size=256, see Appendix C1), we achieve high reconstruction accuracy across *all* layers of all tested models (Table I). Crucially, this method overcomes the architectural constraints of MHA; it is effective even against the fine-tuned *LLaMA-3.1-8B-Distilled* using only public base model weights, demonstrating robustness against parameter divergence.

**Efficiency via Probability-Guided Pruning.** To enhance practicality, we evaluate truncating the search space based on model-predicted probabilities. Experimental results on LLaMA-3.2-1B (Figure 4) indicate that searching only the top $1/8$ of tokens retains 96.1% of the full-search fidelity. This optimization reduces the average reconstruction time per layer from 5.06s to 2.17s ($< 43\%$ of the original latency), rendering the attack highly efficient for real-time scenarios.

**Takeaway 2:** The Collision Attack is a universal threat, achieving high fidelity across diverse architectures (including fine-tuned models) and layers, with high efficiency via probability pruning.

*4) Collision Attack Enhanced with Prior Knowledge (Collision+):* We further evaluate the potency of integrating adversarial prior knowledge (i.e., adaptive thresholding based on assumed token rank $r$).

**Efficacy on Fine-tuned Models.** As detailed in Table I, assuming a rank $r = 8$ yields near-perfect reconstruction ($\approx 100\%$) across all open-source models. For the fine-tuned *LLaMA-3.1-8B-Distilled*, we analyze the impact of rank assumptions in Figure 5. In the *first layer*, where the actual average target rank is $\approx 4$, using the optimal threshold boosts accuracy to 138.6% of the baseline. Conversely, the *middle layer* shows no improvement, implying an actual rank $> 128$. In the *last layer*, optimal rank alignment ($64 \leq r \leq 128$) increases accuracy to 139.0% relative to the baseline.

**Statistical Justification.** To further illustrate this, we analyze distance distributions using "The Bitter Lesson" excerpt (Figure 3). For the open-source model, the heuristic $3\sigma_{\text{other}}$ threshold yields a 0.13% false-positive rate, capping per-token success at 91.84% (assuming $r = 64$). By switching to a statistically derived threshold, we eliminate false positives, achieving 100% success. Similarly, for the fine-tuned model, the enhanced threshold improves per-token success from 91.79% to 97.82%, dramatically increasing full-sequence reconstruction fidelity.

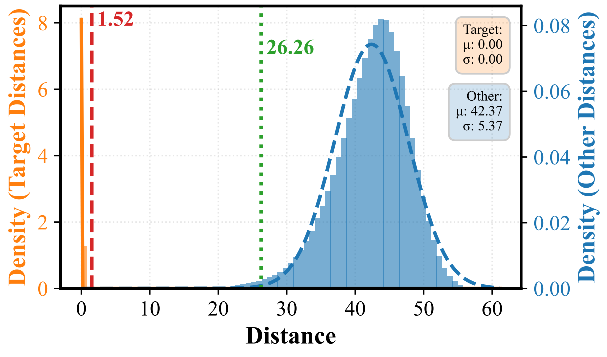
**Takeaway 3:** Augmenting the Collision Attack with model-specific prior knowledge achieves near-perfect ($\approx 100\%$) user-input recovery accuracy.

*5) Results of KV-cache Injection Attack:* We evaluate semantic exfiltration using the optimal directive identified in Appendix C2: "*Repeat the previous content*". As detailed in Table I, this vector achieves an average BERTScore of 0.58 and ROUGE-L of 0.42. While these metrics are lower than those of the exact-match Collision Attack, they confirm significant leakage of the input's core meaning.

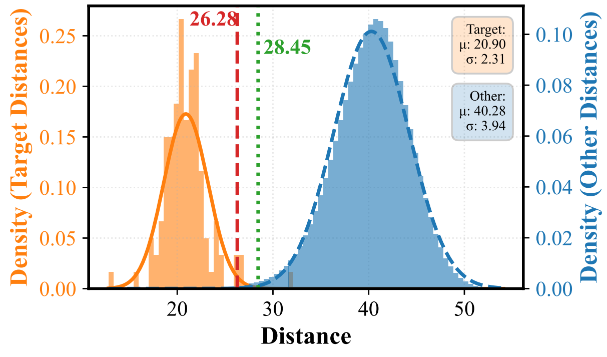
Notably, the attack exhibits peak performance on *LLaMA-7B*. This supports the hypothesis that the *MHA* mechanism retains higher contextual fidelity within the KV-cache compared to compressed variants like *GQA*. Consequently, MHA caches

TABLE I: Reconstruction fidelity (BERTScore and ROUGE-L) of attacks against unprotected KV-cache across different models and layers. "Collision+" denotes the attack enhanced with prior knowledge.

| Type | Model | Metric | Inversion | | | Collision | | | Collision+ | | | Injection |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | First | Mid | Last | First | Mid | Last | First | Mid | Last | All |
| Identical | LLaMA-7B | BERTScore (↑) | 1.000 | 0.065 | 0.092 | 0.449 | 0.769 | 0.611 | 1.000 | 1.000 | 1.000 | 0.765 |
| | | ROUGE-L (↑) | 1.000 | 0.036 | 0.062 | 0.500 | 0.562 | 0.436 | 1.000 | 1.000 | 1.000 | 0.687 |
| | LLaMA-3.2-1B | BERTScore (↑) | 1.000 | 0.084 | 0.057 | 0.877 | 0.791 | 0.894 | 1.000 | 1.000 | 1.000 | 0.544 |
| | | ROUGE-L (↑) | 0.994 | 0.038 | 0.000 | 0.709 | 0.617 | 0.680 | 0.994 | 0.994 | 0.994 | 0.315 |
| | LLaMA-3.2-3B-Instruct | BERTScore (↑) | 0.055 | 0.095 | 0.083 | 0.782 | 0.668 | 0.820 | 1.000 | 1.000 | 1.000 | 0.540 |
| | | ROUGE-L (↑) | 0.000 | 0.000 | 0.000 | 0.732 | 0.456 | 0.621 | 0.994 | 0.994 | 0.994 | 0.324 |
| | LLaMA-3.1-8B | BERTScore (↑) | 0.071 | 0.061 | 0.062 | 0.873 | 0.652 | 0.764 | 1.000 | 1.000 | 1.000 | 0.616 |
| | | ROUGE-L (↑) | 0.000 | 0.000 | 0.001 | 0.825 | 0.443 | 0.564 | 0.994 | 0.994 | 0.994 | 0.447 |
| | Qwen2.5-Math-7B | BERTScore (↑) | 0.229 | 0.105 | 0.105 | 0.918 | 0.552 | 0.783 | 1.000 | 0.983 | 0.996 | 0.422 |
| | | ROUGE-L (↑) | 0.186 | 0.000 | 0.000 | 0.842 | 0.355 | 0.580 | 1.000 | 0.977 | 0.996 | 0.286 |
| Finetune | LLaMA-3.1-8B-Distilled (LLaMA-3.1-8B) | BERTScore (↑) | 0.083 | 0.062 | 0.081 | 0.642 | 0.492 | 0.635 | 0.894 | 0.258 | 0.762 | 0.610 |
| | | ROUGE-L (↑) | 0.000 | 0.000 | 0.001 | 0.633 | 0.227 | 0.413 | 0.868 | 0.122 | 0.479 | 0.421 |
| Cross-Arch | LLaMA-2-7B (LLaMA-7B) | BERTScore (↑) | 0.067 | 0.086 | 0.084 | 0.058 | 0.070 | 0.069 | 0.075 | 0.062 | 0.060 | 0.087 |
| | | ROUGE-L (↑) | 0.038 | 0.057 | 0.031 | 0.013 | 0.000 | 0.002 | 0.052 | 0.017 | 0.024 | 0.019 |



(a) The distributions of LLaMA-3.1-8B model.



(b) The distributions of LLaMA-3.1-8B-Distilled model.

Fig. 3: Distance distributions of target tokens $d_{\text{target}}$ (orange) versus incorrect tokens $d_{\text{other}}$ (blue) in the Collision Attack. The input is an excerpt from "The Bitter Lesson"(see Appendix B). The attack targets the last-layer KV-cache of LLaMA-3.1-8B (left) and LLaMA-3.1-8B-Distilled (right) using weights from the base model. Vertical lines indicate the heuristic threshold ($3\sigma_{\text{other}}$, green dotted) and the prior-knowledge enhanced threshold ($r = 64$, red dashed).

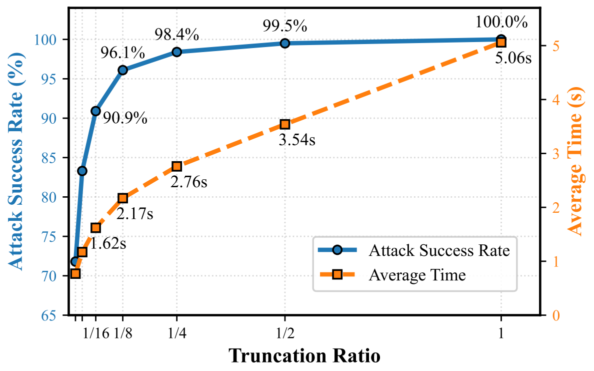


Fig. 4: The effect of truncating the probability-sorted vocabulary on reconstruction fidelity and attack time. Experiments were run with a batch size of 256 and an outlier threshold of $3\sigma_{\text{other}}$.

are more semantically intelligible to the model, increasing vulnerability to instruction-based extraction. This imposes a critical requirement for defenses: a protected KV-cache must be rendered semantically unparsable to the LLM itself, preventing the model from being weaponized to interpret stolen contexts.

**Takeaway 4:** Even without verbatim recovery, Injection Attacks successfully exfiltrate core user intent, necessitating defenses that neutralize semantic intelligibility.

## C. Attack Robustness under Partial Knowledge Scenarios

To assess practical applicability, we evaluate the robustness of our attacks under a relaxed threat model characterized by two realistic constraints: incomplete data interception and model parameter mismatch.

**Incomplete KV-cache Data.** Our analysis reveals distinct data requirements across attack vectors. The *Injection Attack* necessitates the full, cross-layer context and fails when data is fragmented. Conversely, the *Collision Attack* exhibits superior robustness; possessing the KV-cache from *any single layer* is sufficient to perform high-fidelity input reconstruction.

**Model Mismatch.** We further evaluate the scenario where the

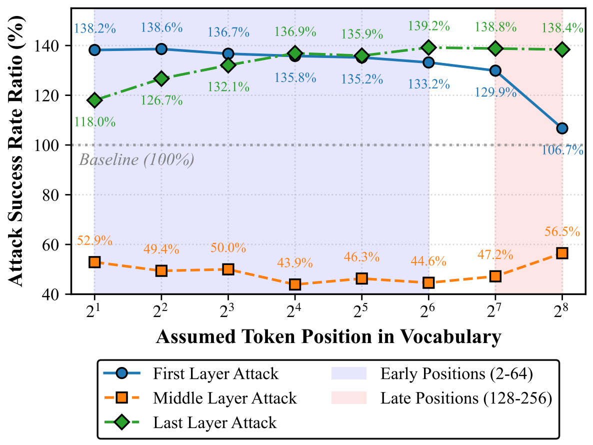


Fig. 5: Collision Attack experiments on LLaMA-3.1-8B-Distilled using optimal thresholds derived for different assumed token ranks ($r$).

adversary's local model differs from the target service model. Experiments confirm that attack effectiveness hinges on the correlation of weight parameters.

- **Fine-tuning Mismatch (Gray-box):** We targeted a fine-tuned model (LLaMA-3.1-8B-Distilled) assuming the adversary only possesses the open-source base model (LLaMA-3.1-8B). Results in Table I show that the attack remains highly effective, suggesting that fine-tuning preserves the fundamental weight correlations exploited by our mechanism.
- **Architectural Mismatch (Black-box):** In contrast, cross-architecture attacks (e.g., using LLaMA-7B to attack LLaMA-2-7B) fail completely. Reconstruction similarity drops to levels statistically indistinguishable from random guessing ($< 0.1$). This confirms that the Collision Attack relies on the algebraic correspondence of weight parameters, which is absent across disjoint architectures.

## V. KV-Cloak: A Lightweight KV-cache Defense

### A. Motivation for KV-Cloak

**Limitations of Existing Privacy-Preserving Techniques.** Current privacy-preserving techniques fundamentally struggle to balance security, efficiency, and utility when applied to the massive, latency-sensitive KV-cache. We analyze their specific limitations below:

- **Cryptographic Methods (Overhead Bottleneck):** Standard cryptographic techniques [4], [34], such as symmetric AES encryption, provide confidentiality by ensuring data remains in ciphertext during storage and transmission, requiring decryption only for legitimate use. A more advanced approach, Homomorphic Encryption (HE) [2], [26], allows for direct computation on encrypted data, theoretically enabling parts of the attention mechanism to operate on the KV-cache without ever decrypting it. However, despite their strong security guarantees, the immense computational overhead and latency introduced by these methods are prohibitive. Given that the KV-cache can be tens or hundreds of gigabytes, applying full encryption or HE is unsuitable for the high-throughput, low-latency requirements of LLM inference.
- **Differential Privacy (Utility Trade-off):** Applying Differential Privacy involves injecting calibrated noise into the Key-Value vectors to mask individual data points [1], [20], [42]. However, the sparse and sensitive nature of attention mechanisms creates a severe utility-privacy trade-off. To achieve a meaningful level of privacy, the required amount of noise would significantly degrade the LLM's inference accuracy to an unacceptable degree.
- **KV-Shield (Security and Compatibility Flaws):** KV-Shield [41] is the only existing lightweight obfuscation scheme for KV-cache. It synchronously permutes attention weight rows ($W_q, W_k, W_v$) to shuffle cache elements, obfuscating them before an attention score is calculated. The obfuscated attention output is then de-obfuscated for subsequent steps. We identify two critical failures in this design: **(1)** *Security Flaws.* First, the simple shuffling preserves the inherent statistical distribution of the data, leaving it highly vulnerable to our proposed *Collision Attacks*. Second, the reliance on a fixed obfuscation key lacks dynamic randomness, rendering the scheme defenseless against CPA. **(2)** *Architectural Incompatibility.* The shuffling disrupts relative positioning, making it incompatible with RoPE used in SOTA models (e.g., LLaMA, Qwen).

**Design Objectives.** The deficiencies of prior works necessitate a specialized defense for KV-cache. We define three mandatory design goals for a practical solution:

- **Robust Security:** The mechanism must resist targeted reconstruction attacks by effectively masking both the algebraic and statistical properties of the cache.
- **Lossless Model Fidelity:** It must preserve the exact mathematical equivalence of the attention mechanism, ensuring zero degradation in generation quality.
- **Negligible Overhead:** The defense must operate with minimal latency, ideally shifting computational costs offline to maintain high inference throughput.

To meet these objectives, we propose **KV-Cloak**, a novel defense mechanism that synergizes reversible matrix obfuscation with operator fusion to deliver robust security, lossless model fidelity, and negligible inference overhead.

### B. Naive Defense: Reversible Linear Obfuscation

To mitigate privacy risks with minimal overhead, we first consider a naive defense that employs reversible linear transformations to obscure the statistical properties of the KV-cache. In the context of modern inference frameworks (e.g., vLLM), we denote the key vectors for a single attention head within a PagedAttention block as a matrix $K \in \mathbb{R}^{b\times d}$, where $b$ is the block size and $d$ is the head dimension. Since the protection mechanism applies analogously to value vectors, we focus our analysis on $K$. The obfuscation transformation is defined as:

$$K' = SKM, \tag{7}$$

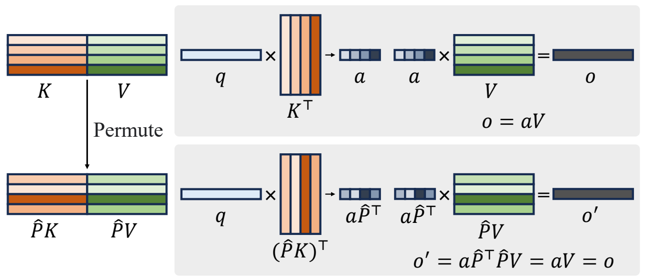


Fig. 6: Illustration of the KV-Cloak block-wise shuffling mechanism. By permuting $K$ and $V$ vectors within a block, the mechanism eliminates positional side channels while maintaining mathematical equivalence in the attention output.

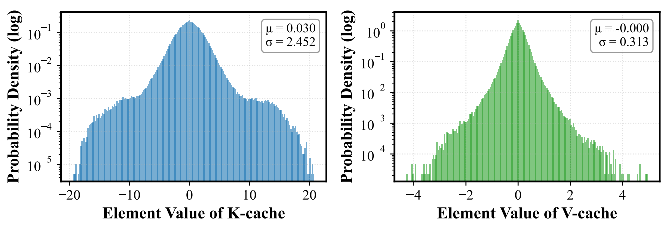


Fig. 7: Value distribution of KV-cache elements ($K$ and $V$) for LLaMA-3.2-1B. The data is collected during inference on "The Bitter Lesson".

where $S \in \mathbb{R}^{b\times b}$ and $M \in \mathbb{R}^{d\times d}$ are secret, randomly generated invertible matrices. This transformation aims to conceal the raw content of $K$ while preserving the matrix dimensions necessary for storage allocation.

**Security Analysis.** Despite its operational simplicity, this scheme is fundamentally insecure under a *CPA* model due to its fixed linear structure. While a real-world adversary cannot generate arbitrary matrix $K$ directly (as $K$ is constrained by the model's embedding projection of valid tokens), we demonstrate that this constraint does not preclude a full compromise. An adversary can circumvent this restriction by mounting a *differential attack* through carefully crafted prompts. Specifically, the adversary chooses two inputs yielding plaintexts $K_1$ and $K_2$, thereby controlling the difference $\Delta K = K_1 - K_2$. Due to the linearity of the scheme, the observed ciphertext difference is $\Delta K' = S(\Delta K)M$. By systematically crafting inputs such that $\Delta K$ approaches a series of standard basis matrices (i.e., matrices with a single non-zero entry), the adversary can isolate and solve for the columns of $S$ and the rows of $M$. This algebraic attack enables full recovery of the secret matrices (up to a scalar ambiguity) with a computational complexity of only $O(b^2d + bd^2)$, rendering the naive defense ineffective against determined adversaries.

## C. One-Time Pad Block-wise Shuffling

*1) Eliminating Redundant Positional Information in KV-cache:* To enhance the security of the obfuscation scheme, we introduce additional randomness by eliminating the architectural redundancy of the KV-cache. We observe that the physical memory ordering of $k, v$ vectors is computationally superfluous during inference, as positional semantics are intrinsic to the vectors via RoPE.

**Mechanism.** We implement a dynamic, block-wise random permutation. By reordering $k, v$ pairs within each block while maintaining their internal correspondence, we decorrelate the physical storage index from the logical token sequence. As illustrated in Figure 6, this operation preserves the mathematical invariance of the attention mechanism while imposing a combinatorial complexity barrier of $b!$ on the adversary. Given typical block sizes $b \in \{16, 32, 64\}$ [17], this renders brute-force matching computationally infeasible.

**Formalization.** The enhanced obfuscation employs a locally generated, one-time pad permutation matrix $\hat{P}$ before applying the linear transformations:

$$K' = S\hat{P}KM. \tag{8}$$

Crucially, for efficiency, $\hat{P}$ is *ephemeral* and does not require storage. The inference engine performs subsequent computations directly on the de-obfuscated permuted state $\hat{P}K$ without reconstructing the original order, thereby incurring zero additional storage overhead for the permutation key.

*2) Rank Preservation and Implicit Key Recovery:* A critical challenge arises when the original matrix $K$ exhibits low rank (e.g., rank 1 in cases of identical repeated tokens). In such scenarios, the permutation entropy collapses, as row shuffling becomes statistically invisible. To guarantee obfuscation robustness, we introduce a secret additive mask matrix $A$ before permutation. Thereby, we can ensure the transformed matrix $(K + A)$ maintains sufficient rank to preserve cryptographic strength of $\hat{P}$. However, explicitly storing the unique one-time pad $\hat{P}$ for every block would reintroduce significant storage overhead. We resolve this by designing $A$ to enable *implicit key recovery* without $\hat{P}$.

**Magnitude-based Positional Embedding.** We exploit the numerical characteristics of attention activations. As shown in Figure 7, elements of $k, v$ vectors are statistically bounded by a threshold $\theta_K$ (e.g., typically $< 100$). Leveraging this, we construct $A$ to function as a "positional beacon" by embedding values significantly larger than $\theta_K$ into specific coordinates. This ensures that the matrix $(K + A)$ maintains enough rank.

**Zero-Storage Recovery.** Since the "beacons" in $A$ dominate the magnitude of $K$, the permuted mask term $\hat{P}A$ remains statistically separable from $\hat{P}K$ within the ciphertext $\hat{P}(K+A)$. During de-obfuscation, the system dynamically identifies the row permutation indices by detecting these high-magnitude outliers, allowing it to reconstruct $\hat{P}A$ on-the-fly. Subtracting this reconstructed mask yields the permuted state $\hat{P}K$ required for inference. This mechanism effectively offloads the storage of $\hat{P}$ into the data itself.

**Formalization.** The complete obfuscation logic integrates this additive masking with the linear and permutation layers:

$$K' = S\hat{P}(K + A)M, \tag{9}$$

where $\hat{P} \in \{0, 1\}^{b\times b}$ is the ephemeral one-time pad, and

$A$ is the structured mask. This composite transformation simultaneously secures cache statistics and enables efficient, stateless recovery without compromising model accuracy.

*3) Security Analysis:* The security of KV-Cloak relies fundamentally on the confidentiality of its secret matrices $(S, M, A)$. Under a standard CPA model, an adversary attempting to recover these keys faces a prohibitive computational barrier. Our analysis indicates that the time complexity to solve for the keys is $O((b^2d + bd^2) \cdot b!)$. Crucially, the factorial term $b!$, introduced by the dynamic block-wise permutation, renders brute-force key recovery computationally infeasible.

This computational hardness is complemented by layered security mechanisms designed to mitigate specific attack vectors: **(1)** The one-time pad permutation severs the explicit token-to-vector positional correspondence at an information-theoretic level, invalidating the premise of Collision Attacks. **(2)** A reversible algebraic transformation via secret matrices $(S, M)$ completely disrupts the cache's statistical distribution properties (e.g., mean, variance), neutralizing profiling and data mining attacks. **(3)** The combined obfuscation renders the cache semantically unintelligible and unparsable by pristine (unmodified) models, thereby thwarting Injection Attacks that attempt to leverage the model's own capabilities.

## D. Implicit Obfuscation via Operator Fusion

To minimize the runtime latency overhead introduced by the obfuscation and de-obfuscation processes, we propose an *operator fusion* strategy. This approach integrates the obfuscation operators directly into the model's weight matrices offline, thereby rendering the online computational impact negligible while maintaining strict mathematical equivalence.

**Transformed Vectors and Invariance.** We introduce two secret, invertible matrices, $M_1, M_2 \in \mathbb{R}^{d\times d}$, to transform the query, key, and value vectors $(q, k, v)$ and the output projection $W_o$. The transformed vectors $(q^m, k^m, v^m)$ and the modified output weight $W_o^m$ are defined as:

$$\begin{cases} q^m = q(M_1^{-1})^\top, \\ k^m = kM_1, \\ v^m = vM_2, \\ W_o^m = W_o(M_2^{-1})^\top. \end{cases} \tag{10}$$

This design ensures that the core attention mechanism remains invariant. Specifically, the attention scores (scaled dot-product of query and key) are preserved:

$$\begin{aligned} q_i^m(k_j^m)^\top &= \left(q_i(M_1^{-1})^\top\right)(k_jM_1)^\top = q_i(M_1^{-1})^\top M_1^\top k_j^\top \\ &= q_i(M_1M_1^{-1})^\top k_j^\top = q_ik_j^\top. \end{aligned} \tag{11}$$

Similarly, the output of the attention head is mathematically identical to the unprotected, guaranteeing lossless accuracy:

$$\begin{aligned} v_j^m(W_o^m)^\top &= (v_jM_2)\left(W_o(M_2^{-1})^\top\right)^\top \\ &= v_jM_2(M_2^{-1})W_o^\top = v_jW_o^\top. \end{aligned} \tag{12}$$

This invariance guarantees that the model's output is mathematically identical to that of the original, unprotected model, ensuring lossless accuracy.

**Fusion under RoPE.** Implementing this fusion requires incorporating the input projection $x$ and the position-dependent RoPE matrix $R^d_{\Theta,i}$. The transformed vectors are expressed as:

$$\begin{cases} q^m = xW_q^\top R^d_{\Theta,i}(M_1^{-1})^\top \\ k^m = xW_k^\top R^d_{\Theta,i}M_1 \\ v^m = xW_v^\top M_2 \\ W_o^m = W_o(M_2^{-1})^\top \end{cases}. \tag{13}$$

From Eq. 13, fusing $M_2$ into the value and output weights is straightforward: we pre-compute $(W_v^m)^\top = W_v^\top M_2$ and $W_o^m = W_o(M_2^{-1})^\top$. However, fusing $M_1$ into $W_k$ and $W_q$ presents a challenge due to the intermediate application of RoPE. The fusion is algebraically feasible *if and only if* the secret matrix $M_1$ and the RoPE matrix $R^d_{\Theta,i}$ **commute**, i.e., $R^d_{\Theta,i}M_1 = M_1R^d_{\Theta,i}$. In this case, Eq. 13 simplifies to:

$$k^m = xW_k^\top(M_1R^d_{\Theta,i}) = x(W_k^\top M_1)R^d_{\Theta,i}. \tag{14}$$

This reordering allows us to absorb $M_1$ into the weight matrix as $(W_k^m)^\top = W_k^\top M_1$. A symmetric logic applies to $W_q$.

To satisfy this commutativity constraint, we structurally design $M_1$ as a invertible block-diagonal matrix composed of $2\times2$ rotation-scaling sub-matrices, analogous to the structure of RoPE (detailed in Appendix A). Additionally, we constrain $M_2$ to be a random invertible rotation-scaling matrix and calibrate the scaling factors of both matrices. This ensures that after the transformation with mask $A$, the padding tokens remain statistically distinguishable as outliers for implicit key recovery.

**Offline Computation and Online Efficiency.** Consequently, the obfuscation operators are fully fused into the attention layer parameters offline. The new pre-computed weights are:

$$\begin{cases} W_q^m = M_1^{-1}W_q \\ W_k^m = M_1^\top W_k \\ W_v^m = M_2^\top W_v \\ W_o^m = W_o(M_2^{-1})^\top \end{cases}. \tag{15}$$

During online inference, the KV-Cloak obfuscation is applied to the cache generated by these fused weights:

$$K' = S\hat{P}(K^m + A), \tag{16}$$

By eliminating explicit online matrix multiplications with $M$, the cost for protecting one KV-cache block is reduced to the multiplications with $S$, $\hat{P}$, and $S^{-1}$. This totals approximately $b^3 + 2b^2d$ floating-point operations. For an LLaMA-3.1-8B instance ($b = 16, d = 128, D = 4096$), this overhead represents merely **0.83%** of the KV-cache re-computation cost (detailed in Appendix F), verifying that KV-Cloak achieves robust security with minimal performance impact.

## VI. Evaluation of KV-Cloak

## A. Experimental Settings

We perform a systematic evaluation across three dimensions: *Security*, *Model Accuracy*, and *Performance Overhead*. To benchmark the efficacy of KV-Cloak, we compare it against three distinct baselines: **(1)** a standard, unprotected system

TABLE II: Comparison of defense mechanisms (KV-Cloak vs. DP) against input reconstruction attacks on LLaMA-7B, LLaMA-3.2-1B, and LLaMA-3.1-8B-Distilled.

| Model | Protect Type | Metric | Inversion | Collision | | | Collision+ | | | Injection |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | First | First | Mid | Last | First | Mid | Last | All |
| LLaMA-7B | Original | BERTScore (↓) | 1.000 | 0.449 | 0.769 | 0.611 | 1.000 | 1.000 | 1.000 | 0.765 |
| | | ROUGE-L (↓) | 1.000 | 0.500 | 0.562 | 0.436 | 1.000 | 1.000 | 1.000 | 0.687 |
| | KV-Cloak | BERTScore (↓) | 0.091 | 0.070 | 0.069 | 0.071 | 0.036 | 0.036 | 0.036 | 0.082 |
| | | ROUGE-L (↓) | 0.068 | 0.000 | 0.000 | 0.000 | 0.044 | 0.044 | 0.044 | 0.000 |
| | $(10^8, 10^{-5})$-DP | BERTScore (↓) | 0.085 | 0.082 | 0.672 | 0.344 | 0.109 | 0.937 | 0.991 | 0.085 |
| | | ROUGE-L (↓) | 0.065 | 0.041 | 0.433 | 0.197 | 0.097 | 0.901 | 0.979 | 0.009 |
| LLaMA-3.2-1B | Original | BERTScore (↓) | 1.000 | 0.877 | 0.791 | 0.894 | 1.000 | 1.000 | 1.000 | 0.544 |
| | | ROUGE-L (↓) | 0.994 | 0.709 | 0.617 | 0.680 | 0.994 | 0.994 | 0.994 | 0.315 |
| | KV-Cloak | BERTScore (↓) | 0.085 | 0.072 | 0.074 | 0.069 | 0.051 | 0.051 | 0.051 | 0.079 |
| | | ROUGE-L (↓) | 0.009 | 0.000 | 0.000 | 0.000 | 0.002 | 0.002 | 0.002 | 0.000 |
| | $(10^8, 10^{-5})$-DP | BERTScore (↓) | 0.633 | 0.849 | 0.763 | 0.849 | 0.973 | 0.995 | 1.000 | 0.393 |
| | | ROUGE-L (↓) | 0.622 | 0.604 | 0.587 | 0.604 | 0.966 | 0.989 | 0.994 | 0.248 |
| LLaMA-3.1-8B-Distilled | Original | BERTScore (↓) | 0.083 | 0.642 | 0.492 | 0.635 | 0.885 | 0.251 | 0.829 | 0.610 |
| | | ROUGE-L (↓) | 0.000 | 0.633 | 0.227 | 0.413 | 0.858 | 0.112 | 0.552 | 0.421 |
| | KV-Cloak | BERTScore (↓) | 0.093 | 0.070 | 0.070 | 0.069 | 0.041 | 0.041 | 0.041 | 0.088 |
| | | ROUGE-L (↓) | 0.002 | 0.000 | 0.000 | 0.000 | 0.003 | 0.003 | 0.003 | 0.000 |
| | $(10^8, 10^{-5})$-DP | BERTScore (↓) | 0.079 | 0.320 | 0.440 | 0.566 | 0.526 | 0.267 | 0.824 | 0.118 |
| | | ROUGE-L (↓) | 0.003 | 0.291 | 0.185 | 0.351 | 0.530 | 0.122 | 0.543 | 0.049 |

(Plaintext), **(2)** a defense based on differential privacy with Gaussian noise (DP), and **(3)** a defense via standard cryptographic encryption (AES).

**KV-Cloak Configuration.** Our implementation involves matrix multiplications with invertible secret matrices $S$ and $M$, and an additive mask $A$. We configure these parameters to balance security and numerical stability:

- **Block Size $b$:** To maintain compatibility with the PagedAttention mechanism (e.g., vLLM), we experimented with standard block sizes of $b \in \{16, 32, 64\}$.
- **Secret Matrices $S, M$:** To minimize precision loss during the matrix inversion required for de-obfuscation, we sample $S$ and $M$ strictly from the orthogonal group.
- **Additive Mask $A$ and Padding:** To preserve numerical precision, the additive mask $A$ and padding values are magnitude-constrained. We sample elements of $A$ uniformly from $[3\theta_K, 4\theta_K]$ and set padding to $1.5\theta_K$. Here, $\theta_K$ denotes the maximum absolute value observed in the K-cache during calibration on an excerpt from "The Bitter Lesson". This methodology is symmetrically applied to the V-cache.

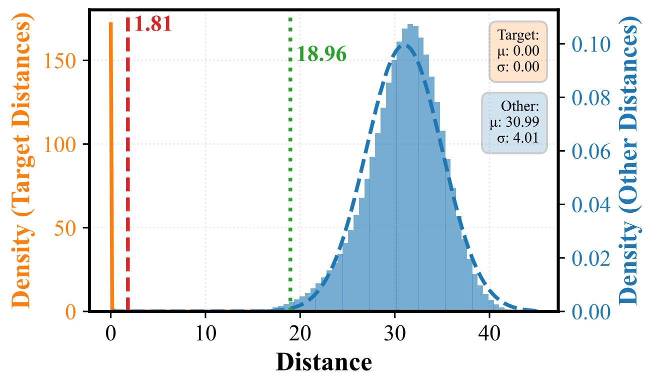
**DP Baseline Configuration.** To establish a strong DP baseline, we conducted an ablation study (detailed in Appendix D) to optimize the trade-off between privacy and utility. We set the clipping threshold to the 50th percentile of the L2 norm distribution observed across the dataset. Based on this, we apply Gaussian noise calibrated for $(\epsilon = 10^8, \delta = 10^{-5})$-DP independently to the K/V caches. We explicitly select $\epsilon = 10^8$ because stricter privacy budgets (e.g., $\epsilon = 10^7$) resulted in near-zero inference accuracy (see Table VIII).

## B. *Evaluation of Security*

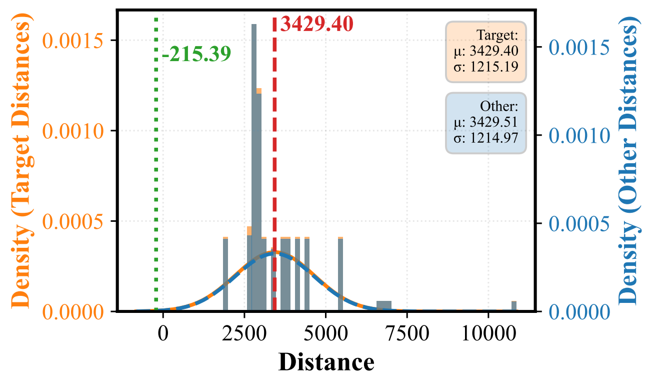
This section evaluates the capability of KV-Cloak against the three proposed reconstruction attacks. We benchmark against the Plaintext baseline and Differential Privacy mechanisms, applying the attacks to the protected KV-cache.

**Defense Efficacy Comparison.** As summarized in Table II (evaluation on the remaining models can be found in Appendix E), the *Plaintext* baseline is highly vulnerable, yielding high attack success rates on all three attacks. In stark contrast, applying KV-Cloak drastically degrades reconstruction quality. The BERTScore of all attack outputs drops to a level consistent with random chance, and the ROUGE-L score falls to nearly 0. These results are statistically indistinguishable from comparing the original input with a random string, demonstrating that semantic reconstruction is entirely disrupted. This proves that KV-Cloak effectively protects the private information within the KV-cache. For the *DP baseline*, security is heavily dependent on the privacy budget $\epsilon$. With a weak budget of $\epsilon = 10^8$, while Inversion and Injection attacks are mitigated, the Collision Attack remains effective, recovering substantial private information.

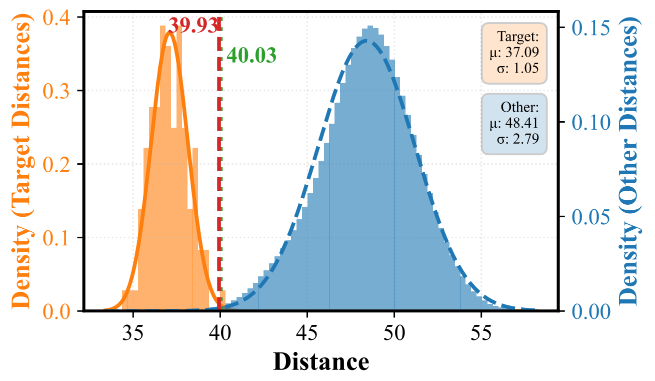
**Statistical Indistinguishability Analysis.** To understand the root cause of defense failure or success, we analyze the distance distributions of target tokens ($d_{\text{target}}$) versus other tokens ($d_{\text{other}}$) for the Collision Attack on LLaMA-3.2-1B (Fig. 8). Under DP protection, the two distributions remain statistically distinguishable. Consequently, utilizing an enhanced threshold with prior knowledge yields a per-token success rate of 94.84% for $(10^7, 10^{-5})$-DP, rising to nearly 100% (equivalent to Plaintext) for $(10^8, 10^{-5})$-DP. Conversely, KV-Cloak achieves *distribution collapse*: the distributions of $d_{\text{target}}$ and $d_{\text{other}}$ become completely indistinguishable. This eliminates



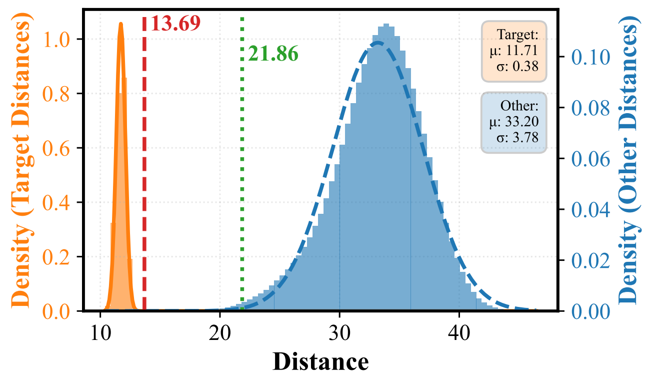
(a) The distributions of LLaMA-3.2-1B model without protection.



(b) The distributions of LLaMA-3.2-1B model with KV-Cloak.



(c) The distributions of LLaMA-3.2-1B model with $(10^7, 10^{-5})$-DP.



(d) The distributions of LLaMA-3.2-1B model with $(10^8, 10^{-5})$-DP.

Fig. 8: Distance distributions of target tokens $d_{\text{target}}$ (orange) versus incorrect tokens $d_{\text{other}}$ (blue) in the Collision Attack. The experiment targets the last-layer KV-cache of LLaMA-3.2-1B using an excerpt from "The Bitter Lesson". Subplots display distributions under four conditions: (a) Plaintext, (b) KV-Cloak protection, (c) $(10^7, 10^{-5})$-DP protection, and (d) $(10^8, 10^{-5})$-DP protection. Vertical lines indicate the heuristic threshold ($3\sigma_{\text{other}}$, green dotted) and the prior-knowledge enhanced threshold ($r = 64$, red dashed).

the statistical separability required for the attack, resulting in a 0% success rate.

**Robustness against Enhanced Attacks.** To assess the robustness of our defense, We further evaluate by subjecting KV-Cloak to the enhanced Collision Attack (utilizing adversarial prior knowledge) across multiple LLMs. As shown in Table II, KV-Cloak demonstrates consistent resilience. Even against this advanced vector, reconstruction accuracy remains near-zero, and outputs are qualitatively equivalent to random noise. This confirms that KV-Cloak effectively neutralizes both algebraic and statistical attack vectors.

**Takeaway 5:** KV-Cloak completely thwarts all proposed attacks, reducing the quality of reconstructed text to a level statistically indistinguishable from random noise.

### C. Inference Accuracy

To rigorously assess model fidelity, we simulate a disaggregated inference service employing a prefill-decode architecture. In this pipeline, the KV-cache generated during the prefill phase is secured using either DP or KV-Cloak, transferred, and subsequently utilized by the decode node for token generation. We employ two standard benchmarks to measure utility: *MMLU* [15], [16] for massive multitask knowledge and *SQuAD* [29] for reading comprehension.

TABLE III: Impact of KV-Cloak on inference accuracy (higher is better) across various models, using a block size of 16.

| Model | Plaintext | | KV-Cloak | | $(10^8, 10^{-5})$-DP | |
|---|---|---|---|---|---|---|
| | MMLU | SQuAD | MMLU | SQuAD | MMLU | SQuAD |
| LLaMA-7B | 0.304 | 0.646 | 0.304 | 0.652 | 0.016 | 0.000 |
| LLaMA-3.2-1B | 0.335 | 0.457 | 0.335 | 0.458 | 0.262 | 0.258 |
| LLaMA-3.2-3B-Instruct | 0.619 | 0.652 | 0.619 | 0.652 | 0.379 | 0.012 |
| LLaMA-3.1-8B | 0.668 | 0.708 | 0.668 | 0.709 | 0.283 | 0.026 |
| LLaMA-3.1-8B-Distilled | 0.584 | 0.568 | 0.584 | 0.570 | 0.108 | 0.001 |
| Qwen2.5-Math-7B | 0.620 | 0.630 | 0.620 | 0.632 | 0.042 | 0.000 |

We evaluated KV-Cloak across all experimental models (results detailed in Table III). The empirical data corroborates our theoretical design: KV-Cloak exhibits *zero degradation* in model performance. Unlike Differential Privacy, which forces a trade-off between utility and privacy via noise injection, KV-

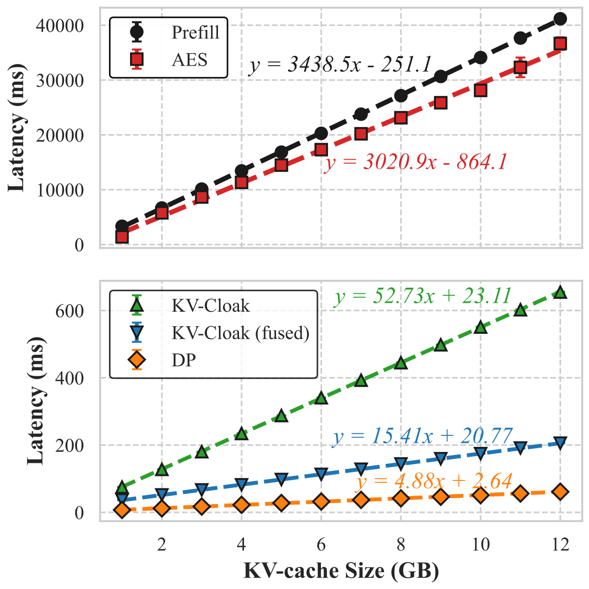


Fig. 9: Micro-benchmark of latency overhead (ms/GB) versus KV-cache size on LLaMA-3.1-8B.

Cloak leverages reversible linear transformations. This ensures that the de-obfuscated attention states are mathematically identical to the plaintext baseline. Consequently, the impact on inference accuracy is negligible across both benchmarks, confirming that KV-Cloak is a practically lossless solution.

**Takeaway 6:** KV-Cloak is virtually lossless, preserving the model's fidelity and core utility without any degradation.

### D. Evaluation of Performance Overhead

**Computational Overhead.** To accurately assess the upper bound of performance impact, we employ a conservative, worst-case micro-benchmark simulating a prefill-decode split. Measurements are conducted on a serial PyTorch implementation without custom CUDA kernels, isolating the algorithmic overhead. As illustrated in Figure 9, latency scales linearly. Standard AES encryption incurs a prohibitive cost of 3020.9 ms/GB, nearly rivaling the prefill latency (3438.5 ms/GB) itself. In stark contrast, KV-Cloak with operator fusion introduces a negligible overhead of just **15.41 ms/GB**. This constitutes merely **0.45%** of the prefill latency, ranking second only to the insecure DP scheme (4.88 ms/GB). In practical deployments, unavoidable network latency would further mask this minimal computation cost, confirming the efficiency of our design.

**Storage Overhead.** The storage Overhead for KV-Cloak's secret matrices $(S, M, A)$ is negligible. The total size, $2n_{\text{kv}}l(b^2 + \frac{3}{2}d+1)$, is orders of magnitude smaller than the protected cache. Quantitatively, for LLaMA-3.1-8B (using block size $b = 16$), the overhead is merely 898 KB; even for a 1T-parameter model like Ling-1T [18] (scaling to $b = 64$), it remains only 20.9 MB. This MB-scale footprint allows all cryptographic keys to securely reside within the limited memory of TEEs with negligible management overhead.

**Takeaway 7:** KV-Cloak introduces negligible computational and storage overhead to the inference pipeline.

## VII. Discussion and Future Work

While KV-Cloak provides a solid foundation for protecting the KV-cache, we identify several limitations, which in turn open up exciting avenues for future research.

**Key Management and Hardware Security Integration.** The security of KV-Cloak fundamentally relies on the confidentiality of its (megabyte-scale) key matrices. Currently, we assume these secrets are protected via TEEs. Future work should explore the deep integration of KV-Cloak with hardware-level security mechanisms, such as TEEs or confidential GPUs, to construct a more robust defense-in-depth architecture. Furthermore, to counter long-term cryptanalysis in persistent services, developing efficient, low-latency key rotation protocols is essential.

**Performance Optimization via Co-Design.** Although our evaluation shows minimal overhead, hyper-scale deployment demands further optimization. At the software level, latency can be masked through the asynchronous generation of One-Time Pad (OTP) matrices, decoupling security operations from the critical inference path. At the hardware level, an algorithm-hardware co-design approach—implementing dedicated GPU intrinsics for block-wise permutation and matrix multiplication—could render the obfuscation cost virtually transparent.

**Adaptation to Quantized Models.** Our current prototype targets floating-point models. As the industry pivots toward integer quantization (e.g., INT8 or INT4) for efficiency, extending KV-Cloak is a priority. This necessitates designing new, lossless reversible transformations suitable for discrete data types, potentially based on mathematical structures like modular arithmetic.

## VIII. Conclusion

This research exposes a critical security flaw at the heart of modern LLM inference systems: the privacy risk of data leakage from the KV-cache. We have demonstrated the feasibility of reconstructing sensitive user inputs through three novel attack strategies, with our Collision Attack proving particularly effective across various models. This underscores the urgent need for dedicated protection mechanisms that do not compromise the efficiency gains the KV-cache provides.

In response, we designed KV-Cloak. By employing a lightweight, reversible obfuscation technique, KV-Cloak neutralizes the identified threats without degrading model accuracy or imposing significant latency. It is designed for seamless integration into existing high-performance inference frameworks like vLLM. Our work provides a vital contribution to building secure and trustworthy AI, offering a blueprint for balancing the competing demands of performance and user privacy in the next generation of LLM services. It establishes that strong privacy protection can be achieved without sacrificing the utility and efficiency that have made these models so powerful.

## Ethic Considerations

This research aims to enhance the privacy and trustworthiness of LLM inference, but we acknowledge the dual-use nature of the vulnerabilities and attack methods we have uncovered. To fulfill our ethical responsibilities and mitigate any potential for misuse, we have committed to a policy of responsible disclosure. Prior to the public dissemination of this paper, we will share our findings, including the details of the vulnerabilities and our proposed defense, with the developers of major affected inference frameworks, such as vLLM. Furthermore, all of our attack validation experiments were conducted exclusively on public and non-sensitive academic datasets. No real user data was involved at any stage of our research. We firmly believe that by taking these responsible measures, the positive contributions of our defensive work, KV-Cloak, will far outweigh the risks associated with the disclosure of these attacks. We are confident that this work will encourage the community to build more trustworthy AI services that are secure by default.

## References

[1] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *ACM SIGSAC Conference on Computer and Communications Security*, 2016.

[2] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A survey on homomorphic encryption schemes: Theory and implementation," *ACM Computing Surveys*, vol. 51, no. 4, pp. 1–35, 2018.

[3] J. Ainslie, J. Lee-Thorp, M. de Jong, Y. Zemlyanskiy, F. Lebrón, and S. Sanghai, "Gqa: Training generalized multi-query transformer models from multi-head checkpoints," *arXiv preprint arXiv:2305.13245*, 2023.

[4] M. N. Alenezi, H. Alabdulrazzaq, and N. Q. Mohammad, "Symmetric encryption algorithms: Review and evaluation study," *International Journal of Communication Networks and Information Security*, vol. 12, no. 2, pp. 256–272, 2020.

[5] APPLE. (2024) Private cloud compute: A new frontier for ai privacy in the cloud. [Online]. Available: https://security.apple.com/blog/private-cloud-compute/

[6] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[7] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson *et al.*, "Extracting training data from large language models," in *USENIX Security Symposium*, 2021, pp. 2633–2650.

[8] Y. Chen, C. Shen, C. Wang, and Y. Zhang, "Teacher model fingerprinting attacks against transfer learning," in *USENIX Security Symposium*, 2022.

[9] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman, "Training verifiers to solve math word problems," *arXiv preprint arXiv:2110.14168*, 2021.

[10] G. Dhanuskodi, S. Guha, V. Krishnan, A. Manjunatha, R. Nertney, M. O'Connor, and P. Rogers, "Creating the first confidential gpus," *Communications of the ACM*, vol. 67, no. 1, pp. 60–67, 2023.

[11] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography Conference*. Springer Berlin Heidelberg, 2006, pp. 265–284.

[12] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan *et al.*, "The llama 3 herd of models," *arXiv preprint arXiv:2407.21783*, 2024.

[13] D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang *et al.*, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," *arXiv preprint arXiv:2501.12948*, 2025.

[14] Y. He, H. She, X. Qian, X. Zheng, Z. Chen, Z. Qin, and L. Cavallaro, "On benchmarking code llms for android malware analysis," in *ACM SIGSOFT International Symposium on Software Testing and Analysis Workshop*, 2025.

[15] D. Hendrycks, C. Burns, S. Basart, A. Critch, J. Li, D. Song, and J. Steinhardt, "Aligning ai with shared human values," *International Conference on Learning Representations*, 2021.

[16] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt, "Measuring massive multitask language understanding," in *International Conference on Learning Representations*, 2021.

[17] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. E. Gonzalez, H. Zhang, and I. Stoica, "Efficient memory management for large language model serving with pagedattention," in *Symposium on Operating Systems Principles*, 2023.

[18] A. Li, B. Liu, B. Hu, B. Li, B. Zeng, B. Ye, C. Tang, C. Tian, C. Huang, C. Zhang *et al.*, "Every activation boosted: Scaling general reasoner to 1 trillion open language foundation," *arXiv preprint arXiv:2510.22115*, 2025.

[19] H. Li, M. Xu, and Y. Song, "Sentence embedding leaks more information than you expect: Generative embedding inversion attack to recover the whole sentence," in *Findings of the Association for Computational Linguistics: ACL 2023*, 2023, pp. 14 022–14 040.

[20] X. Li, Z. Qin, K. Ren, C. Gong, S. Feng, Y. Hong, and T. Wang, "Delay-allowed differentially private data stream release." in *Network and Distributed System Security Symposium*, 2025.

[21] Y. Li, S. Shao, Y. He, J. Guo, T. Zhang, Z. Qin, P.-Y. Chen, M. Backes, P. Torr, D. Tao, and K. Ren, "Rethinking data protection in the (generative) artificial intelligence era," *arXiv preprint arXiv:2507.03034*, 2025.

[22] B. Lin, C. Zhang, T. Peng, H. Zhao, W. Xiao, M. Sun, A. Liu, Z. Zhang, L. Li, X. Qiu *et al.*, "Infinite-llm: Efficient llm service for long context with distattention and distributed kvcache," *arXiv preprint arXiv:2401.02669*, 2024.

[23] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*, 2004, pp. 74–81.

[24] A. Liu, B. Feng, B. Xue, B. Wang, B. Wu, C. Lu, C. Zhao, C. Deng, C. Zhang, C. Ruan *et al.*, "Deepseek-v3 technical report," *arXiv preprint arXiv:2412.19437*, 2024.

[25] J. Morris, V. Kuleshov, V. Shmatikov, and A. M. Rush, "Text embeddings reveal (almost) as much as text," in *Conference on Empirical Methods in Natural Language Processing*, 2023, pp. 12 448–12 460.

[26] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 1999, pp. 223–238.

[27] D. Pasquini, E. M. Kornaropoulos, and G. Ateniese, "Llmmap: Fingerprinting for large language models," in *USENIX Security Symposium*, 2025.

[28] R. Pope, S. Douglas, A. Chowdhery, J. Devlin, J. Bradbury, J. Heek, K. Xiao, S. Agrawal, and J. Dean, "Efficiently scaling transformer inference," *Proceedings of Machine Learning and Systems*, vol. 5, pp. 606–624, 2023.

[29] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ questions for machine comprehension of text," in *Conference on Empirical Methods in Natural Language Processing*, 2016.

[30] Z. Shao, D. Dai, D. Guo, B. L. B. Liu), Z. Wang, and H. Xin, "Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model," *ArXiv*, 2024.

[31] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean, "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," in *International Conference on Learning Representations*, 2017.

[32] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu, "Roformer: Enhanced transformer with rotary position embedding," *Neurocomputing*, vol. 568, p. 127063, 2024.

[33] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto, "Stanford alpaca: An instruction-following llama model," 2023.

[34] E. Thambiraja, G. Ramesh, and D. R. Umarani, "A survey on various most common encryption techniques," *International Journal of Advanced Research in Computer Science and Software Engineering*, 2012.

[35] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[36] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Annual Conference on Neural Information Processing Systems*, vol. 30, 2017.

[37] Z. Wan, A. Cheng, Y. Wang, and L. Wang, "Information leakage from embedding in large language models," *arXiv preprint arXiv:2405.11916*, 2024.

[38] Z. Wan, X. Wang, C. Liu, S. Alam, Y. Zheng, J. Liu, Z. Qu, S. Yan, Y. Zhu, Q. Zhang *et al.*, "Efficient large language models: A survey," *Transactions on Machine Learning Research*, 2024.

[39] G. Wu, Z. Zhang, Y. Zhang, W. Wang, J. Niu, Y. Wu, and Y. Zhang, "I know what you asked: Prompt leakage via kv-cache sharing in multi-tenant llm serving," in *Network and Distributed System Security Symposium*, 2025.

[40] A. Yang, B. Zhang, B. Hui, B. Gao, B. Yu, C. Li, D. Liu, J. Tu, J. Zhou, J. Lin *et al.*, "Qwen2.5-math technical report: Toward mathematical expert model via self-improvement," *arXiv preprint arXiv:2409.12122*, 2024.

[41] H. Yang, D. Zhang, Y. Zhao, Y. Li, and Y. Liu, "A first look at efficient and secure on-device llm inference against kv leakage," in *Workshop on Mobility in the Evolving Internet Architecture*, 2024, pp. 13–18.

[42] L. Yu, L. Liu, C. Pu, M. E. Gursoy, and S. Truex, "Differentially private model publishing for deep learning," in *IEEE Symposium on Security and Privacy*. IEEE, 2019, pp. 332–349.

[43] M. Yuan, L. Zhang, L. Zeng, S. Jiang, B. Yang, D. Duan, and G. Xing, "Scx: Stateless kv-cache encoding for cloud-scale confidential transformer serving," in *Proceedings of the ACM SIGCOMM 2025 Conference*, 2025, pp. 39–54.

[44] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin *et al.*, "Opt: Open pre-trained transformer language models," *arXiv preprint arXiv:2205.01068*, 2022.

[45] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "BERTScore: Evaluating text generation with BERT," in *International Conference on Learning Representations*, 2020.

[46] Z. Zhang, Y. Sheng, T. Zhou, T. Chen, L. Zheng, R. Cai, Z. Song, Y. Tian, C. Ré, C. Barrett, Z. Wang, and B. Chen, "H$_2$o: Heavy-hitter oracle for efficient generative inference of large language models," *Advances in neural information processing systems*, vol. 36, pp. 34 661–34 710, 2023.

[47] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong *et al.*, "A survey of large language models," *arXiv preprint arXiv:2303.18223*, vol. 1, no. 2, 2023.

[48] L. Zheng, W.-L. Chiang, Y. Sheng, T. Li, S. Zhuang, Z. Wu, Y. Zhuang, Z. Li, Z. Lin, E. P. Xing, J. E. Gonzalez, I. Stoica, and H. Zhang, "Lmsys-chat-1m: A large-scale real-world llm conversation dataset," in *International Conference on Learning Representations*, 2024.

[49] Z. Zhou, X. Ning, K. Hong, T. Fu, J. Xu, S. Li, Y. Lou, L. Wang, Z. Yuan, X. Li *et al.*, "A survey on efficient inference for large language models," *arXiv preprint arXiv:2404.14294*, 2024.

## Appendix

### A. Proof of Commutativity with Rotary Position Embedding

Denoting both $R^d_{\Theta,i}$ and the random invertible matrix $M_1$ as a $2 \times 2$ block matrix, we obtain:

$$R^d_{\Theta,i} = \begin{bmatrix} C & -S \\ S & C \end{bmatrix}, M_1 = \begin{bmatrix} T & Y \\ U & Z \end{bmatrix}, \tag{17}$$

where $C, S \in \mathbb{R}^{\frac{d}{2}\times\frac{d}{2}}$ are shown in Eq. (18), assuming that $T, U, Y, Z \in \mathbb{R}^{\frac{d}{2}\times\frac{d}{2}}$ are all random matrices.

$$\begin{aligned} C &= \begin{bmatrix} \cos i\theta_0 & 0 & \cdots & 0 \\ 0 & \cos i\theta_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \cos i\theta_{\frac{d}{2}-1} \end{bmatrix}, \\ S &= \begin{bmatrix} \sin i\theta_0 & 0 & \cdots & 0 \\ 0 & \sin i\theta_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sin i\theta_{\frac{d}{2}-1} \end{bmatrix}. \end{aligned} \tag{18}$$

If the secret matrix $M_1$ and the RoPE matrix $R^d_{\Theta,i}$ commute, then:

$$\begin{bmatrix} T & Y \\ U & Z \end{bmatrix}\begin{bmatrix} C & -S \\ S & C \end{bmatrix} = \begin{bmatrix} C & -S \\ S & C \end{bmatrix}\begin{bmatrix} T & Y \\ U & Z \end{bmatrix}. \tag{19}$$

With $j$ and $k$ as subscripts of matrix elements, Eq. (20) is equivalent to the following system of linear equations:

$$\begin{cases} t_{jk}\cos i\theta_j - u_{jk}\sin i\theta_j = t_{jk}\cos i\theta_k + y_{jk}\sin i\theta_k \\ t_{jk}\sin i\theta_j + u_{jk}\cos i\theta_j = u_{jk}\cos i\theta_k + z_{jk}\sin i\theta_k \\ y_{jk}\cos i\theta_j - z_{jk}\sin i\theta_j = -t_{jk}\sin i\theta_k + y_{jk}\cos i\theta_k \\ y_{jk}\sin i\theta_j + z_{jk}\cos i\theta_j = -u_{jk}\sin i\theta_k + z_{jk}\cos i\theta_k \end{cases} \tag{20}$$

Calculating the equation, the $T, U, Y, Z$ need to satisfy the following relationship:

$$\begin{cases} y_{jj} = -u_{jj}, \\ z_{jj} = t_{jj}, \\ t_{jk} = u_{jk} = y_{jk} = z_{jk} = 0, (j \neq k) \end{cases}. \tag{21}$$

Specifically, the structure of $M_1$ is defined as follows in Eq. (22),

$$M_1 = \begin{bmatrix} t_0 & 0 & \cdots & 0 & -u_0 & 0 & \cdots & 0 \\ 0 & t_1 & \cdots & 0 & 0 & -u_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & t_{\frac{d}{2}-1} & 0 & 0 & \cdots & -u_{\frac{d}{2}-1} \\ u_0 & 0 & \cdots & 0 & t_0 & 0 & \cdots & 0 \\ 0 & u_1 & \cdots & 0 & 0 & t_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & u_{\frac{d}{2}-1} & 0 & 0 & \cdots & t_{\frac{d}{2}-1} \end{bmatrix}. \tag{22}$$

### B. Input Text for Parameter Calibration

The following text, an excerpt from "The Bitter Lesson" by Rich Sutton, was used as model input in our experiments to analyze the numerical characteristics of the KV-cache (e.g., for parameter calibration as described in Section IV-B4).

**Input Text for Parameter Calibration**

One thing that should be learned from the bitter lesson is the great power of general-purpose methods, of methods that continue to scale with increased computation even as the available computation becomes very great. The two methods that seem to scale arbitrarily in this way are search and learning. The second general point to be learned from the bitter lesson is that the actual contents of minds are tremendously, irredeemably complex; we should stop trying to find simple ways to think about the contents of minds, such as simple ways to think about space, objects, multiple agents, or symmetries.

### C. Additional Experiments about Attacks

*1) Ablation Study of Collision Attack:* **Outlier Detection Threshold.** We experimented with different outlier detection thresholds on the LLaMA-3.2-1B model. The results are shown in Table IV. We observed that setting the threshold to $3\sigma_{\text{other}}$ (i.e., treating a value as an outlier if it is more than three standard deviations below the mean of the $d_{\text{other}}$ distribution, which corresponds to approximately 0.13% of a standard

TABLE IV: Impact of different outlier detection thresholds on reconstruction accuracy, with a fixed batch size of 256.

| Model | Layer | Metric | Gap | | | | |
|---|---|---|---|---|---|---|---|
| | | | $2\sigma$ | $2.5\sigma$ | $3\sigma$ | $3.5\sigma$ | $4\sigma$ |
| LLaMA-3.2-1B | First | BERTScore (↑) | 0.531 | 0.783 | 0.877 | 0.821 | 0.719 |
| | | ROUGE-L (↑) | 0.358 | 0.579 | 0.709 | 0.706 | 0.663 |
| | Mid | BERTScore (↑) | 0.419 | 0.619 | 0.791 | 0.820 | 0.724 |
| | | ROUGE-L (↑) | 0.310 | 0.482 | 0.617 | 0.661 | 0.592 |
| | Last | BERTScore (↑) | 0.579 | 0.807 | 0.894 | 0.878 | 0.727 |
| | | ROUGE-L (↑) | 0.455 | 0.615 | 0.680 | 0.648 | 0.485 |
| | Average Time(s) | | 1.17 | 2.13 | 5.06 | 12.65 | 21.58 |

TABLE V: Impact of different batch sizes on reconstruction accuracy, using a fixed outlier detection threshold of $3\sigma_{\text{other}}$.

| Model | Layer | Metric | Batch Size | | | | |
|---|---|---|---|---|---|---|---|
| | | | 64 | 128 | 256 | 512 | 1024 |
| LLaMA-3.2-1B | First | BERTScore (↑) | 0.830 | 0.856 | 0.877 | 0.869 | 0.837 |
| | | ROUGE-L (↑) | 0.711 | 0.723 | 0.709 | 0.666 | 0.594 |
| | Mid | BERTScore (↑) | 0.765 | 0.786 | 0.791 | 0.771 | 0.753 |
| | | ROUGE-L (↑) | 0.590 | 0.617 | 0.617 | 0.588 | 0.554 |
| | Last | BERTScore (↑) | 0.837 | 0.865 | 0.894 | 0.902 | 0.887 |
| | | ROUGE-L (↑) | 0.513 | 0.616 | 0.680 | 0.674 | 0.636 |
| | Average Time(s) | | 12.94 | 7.58 | 5.06 | 4.04 | 4.62 |

normal distribution) yields the highest reconstruction accuracy. A threshold that is too low (e.g., $2\sigma_{\text{other}}$) leads to misidentifying incorrect tokens as the target, thus reducing accuracy. Conversely, a threshold that is too high (e.g., $4\sigma_{\text{other}}$) can cause the correct token to be missed, which also decreases accuracy while significantly increasing the attack time. Therefore, we set the outlier detection threshold to $3\sigma_{\text{other}}$ for all subsequent experiments.

**Batch Size.** We evaluated the effect of different batch sizes on the LLaMA-3.2-1B model. As shown in Table V, with the outlier threshold fixed at $3\sigma_{\text{other}}$, a batch size of 256 achieves the highest reconstruction accuracy. Theoretically, a larger batch size may provide a more robust statistical sample of the $d_{\text{other}}$ distances, leading to higher accuracy. However, our experiments show that as the batch size increases beyond 256, the reconstruction accuracy paradoxically decreases. We attribute this to a mismatch between the batch size and the fixed threshold; a larger batch would likely require a higher, more stringent threshold to maintain accuracy. However, larger batches increase GPU memory consumption, and a higher threshold would multiplicatively increase attack time. To balance accuracy, memory usage, and attack speed, we chose a batch size of 256 for our experiments.

*2) Ablation Study of Injection Attack:* We tested various instructions against each model's KV-cache, with the results shown in Table VI. The instruction "Repeat the previous content." achieved the highest overall reconstruction accuracy across all models, with an average BERTScore of 0.58 and ROUGE-L of 0.42.

*3) Attack Generalization Across Datasets:* To validate the generalization capability of proposed attack, we also conducted evaluations on two datasets from different domains: Alpaca [33] (instruction-following) and GSM8K [9] (mathematical reasoning). As shown in Table VII, our attacks achieved high reconstruction accuracy across all tested datasets. This result confirms that the effectiveness is not confined to specific data distributions or task types, but is instead highly generalizable.

TABLE VI: Impact of different adversarial instructions on Injection Attack success rates.

| Model | Metric | Inject Instruction | | | |
|---|---|---|---|---|---|
| | | Ins1 | Ins2 | Ins3 | Ins4 |
| LLaMA-7B | BERTScore (↑) | 0.765 | 0.716 | 0.557 | 0.598 |
| | ROUGE-L (↑) | 0.687 | 0.606 | 0.449 | 0.473 |
| LLaMA-3.2-1B | BERTScore (↑) | 0.544 | 0.533 | 0.423 | 0.353 |
| | ROUGE-L (↑) | 0.315 | 0.358 | 0.232 | 0.217 |
| LLaMA-3.2-3B-Instruct | BERTScore (↑) | 0.540 | 0.360 | 0.506 | 0.271 |
| | ROUGE-L (↑) | 0.324 | 0.157 | 0.358 | 0.124 |
| LLaMA-3.1-8B | BERTScore (↑) | 0.616 | 0.544 | 0.432 | 0.457 |
| | ROUGE-L (↑) | 0.447 | 0.365 | 0.275 | 0.279 |
| LLaMA-3.1-8B-Distilled | BERTScore (↑) | 0.610 | 0.536 | 0.348 | 0.434 |
| | ROUGE-L (↑) | 0.421 | 0.348 | 0.218 | 0.249 |
| Qwen2.5-Math-7B | BERTScore (↑) | 0.422 | 0.381 | 0.413 | 0.329 |
| | ROUGE-L (↑) | 0.286 | 0.222 | 0.281 | 0.194 |

*Note:* The specific instructions are: "Repeat the previous content." (Ins1), "Summarize the previous content." (Ins2), "Repeat what I said." (Ins3), and "Summarize what I said." (Ins4).

## *D. DP Baseline Parameter Selection*

To establish a robust DP baseline, we first defined its parameterization methodology. We then conducted experiments to select a configuration that balances utility and privacy for comparison against KV-Cloak.

- **Noise Application:** As illustrated in Figure 7, the element distributions of the K and V caches differ significantly. Consequently, we apply $(\epsilon, \delta)$-DP Gaussian noise to the K and V tensors independently.
- **Clipping Threshold $C$:** The noise magnitude in DP is determined by the function's sensitivity, which we control by clipping the Frobenius norm of the K and V tensors. To find an appropriate clipping threshold, we generated 1,000 long sequences (approximately 2,000 tokens each) from the MMLU dataset, recorded the distribution of the resulting KV-cache Frobenius norms, and experimented with thresholds corresponding to different percentiles of this distribution.
- **Privacy Budget $\epsilon$:** This parameter governs the fundamental trade-off between privacy and utility. We evaluated a wide range of $\epsilon$ values to map out their impact on model accuracy.
- **Failure Probability $\delta$:** This represents the probability of the privacy guarantee being violated. We adopt the common standard value of $\delta = 10^{-5}$ for all DP experiments.

Our experimental results, presented in Table VIII, reveal a stark trade-off between privacy and model accuracy for the DP baseline. Under conventionally strong privacy settings (e.g., $\epsilon = 1$ or $\epsilon = 10$), model accuracy on both MMLU and SQuAD collapses to the level of random guessing, regardless of the chosen clipping threshold. Accuracy only begins to recover

TABLE VII: Attack generalization across datasets on LLaMA-3.2-1B.

| Model | Datasets | Metric | Inversion | Collision | | | Collision+ | | | Injection |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | First | First | Mid | Last | First | Mid | Last | All |
| LLaMA-3.2-1B | Alpaca | BERTScore (↑) | 1.000 | 0.942 | 0.910 | 0.936 | 1.000 | 1.000 | 1.000 | 0.570 |
| | | ROUGE-L (↑) | 1.000 | 0.897 | 0.700 | 0.923 | 1.000 | 1.000 | 1.000 | 0.374 |
| | GSM8K | BERTScore (↑) | 1.000 | 0.890 | 0.716 | 0.911 | 1.000 | 1.000 | 1.000 | 0.632 |
| | | ROUGE-L (↑) | 1.000 | 0.784 | 0.493 | 0.800 | 1.000 | 0.999 | 1.000 | 0.496 |
| | LMSYS-Chat-1M | BERTScore (↑) | 1.000 | 0.877 | 0.791 | 0.894 | 1.000 | 1.000 | 1.000 | 0.544 |
| | | ROUGE-L (↑) | 0.994 | 0.709 | 0.617 | 0.680 | 0.994 | 0.994 | 0.994 | 0.315 |

TABLE VIII: Privacy-utility trade-off: LLaMA-3.2-1B inference accuracy under DP with varying noise ($\epsilon$) and clipping thresholds. The 50th percentile norm is highlighted as the baseline for subsequent comparisons.

| Model | Norm Ratio | Metric | $\epsilon$ | | | | |
|---|---|---|---|---|---|---|---|
| | | | 1 | 10 | $10^7$ | $10^8$ | $10^9$ |
| LLaMA-3.2-1B | 50% | MMLU (↑) | 0.051 | 0.052 | 0.052 | 0.262 | 0.299 |
| | | SQuAD (↑) | 0.000 | 0.000 | 0.000 | 0.258 | 0.443 |
| | 90% | MMLU (↑) | 0.053 | 0.053 | 0.045 | 0.259 | 0.309 |
| | | SQuAD (↑) | 0.000 | 0.000 | 0.000 | 0.171 | 0.435 |
| | 95% | MMLU (↑) | 0.052 | 0.053 | 0.045 | 0.252 | 0.309 |
| | | SQuAD (↑) | 0.000 | 0.000 | 0.000 | 0.136 | 0.437 |

TABLE IX: Effectiveness of the Inversion, Collision, and Injection attacks against different layers of the KV-cache from the LLaMA-3.2-1B model, under various DP mechanisms.

| Model | Protect Type | Metric | Inversion | Collision | | | Injection |
|---|---|---|---|---|---|---|---|
| | | | First | First | Mid | Last | All |
| LLaMA-3.2-1B | Plaintext | BERTScore (↓) | 1.000 | 0.877 | 0.791 | 0.894 | 0.544 |
| | | ROUGE-L (↓) | 0.994 | 0.709 | 0.617 | 0.680 | 0.315 |
| | $(10^7, 10^{-5})$-DP | BERTScore (↓) | 0.096 | 0.469 | 0.651 | 0.672 | 0.131 |
| | | ROUGE-L (↓) | 0.073 | 0.353 | 0.402 | 0.336 | 0.067 |
| | $(10^8, 10^{-5})$-DP | BERTScore (↓) | 0.633 | 0.849 | 0.763 | 0.849 | 0.393 |
| | | ROUGE-L (↓) | 0.622 | 0.604 | 0.587 | 0.604 | 0.248 |
| | $(10^9, 10^{-5})$-DP | BERTScore (↓) | 0.994 | 0.808 | 0.786 | 0.886 | 0.524 |
| | | ROUGE-L (↓) | 0.980 | 0.635 | 0.610 | 0.667 | 0.304 |

when the privacy budget is substantially relaxed: at $\epsilon = 10^8$, it reaches 59.13% of the unprotected baseline's accuracy; and at $\epsilon = 10^9$, it improves to 93.61%. This extreme sensitivity is due to the highly sparse nature of the KV-cache, where most elements are near zero. Directly adding noise disproportionately perturbs the cache's delicate structure, severely degrading model performance unless the noise is made negligible by an extremely large $\epsilon$. And its defensive efficacy, presented in Table IX, is strongly correlated with the privacy budget $\epsilon$. With a weak budget of $\epsilon = 10^8$, the accuracy of the Inversion and Injection attacks is reduced, but the Collision Attack can still recover some useful information. As $\epsilon$ is strengthened to $10^7$, the overall attack success rate decreases further. However, the Collision Attack can still achieve a reconstruction with over 55% semantic similarity. Importantly, this protection comes at the cost of model accuracy, a trade-off we will discuss in detail in the next section.

To balance security and accuracy for our comparative analysis, we selected $\epsilon = 10^8$ and a clipping norm at the 50th percentile for subsequent experiments, as this offered a reasonable degree of utility for the DP baseline.

## *E. Evaluation of Security on the Remaining Models*

As shown in Table X, KV-Cloak completely thwarts all our proposed attacks, reducing the quality of any reconstructed text to a level statistically indistinguishable from random noise.

## *F. Performance Analysis and the Impact of Operator Fusion*

A critical aspect of any practical defense mechanism is its performance overhead. In this section, we analyze the computational cost of KV-Cloak and demonstrate the significant efficiency gains achieved through our operator fusion technique.

**Overhead of a Naive Implementation.** Without operator fusion, a naive implementation would apply the obfuscation transform $K' = S\hat{P}(K + A)M$ and its inverse as explicit. Neglecting the computationally inexpensive matrix additions involving $A$, the primary overhead stems from matrix multiplications. For a single KV-cache block of size $b \times d$:

- The obfuscation operation requires approximately $b^3$ (for $S\hat{P}$), $b^2d$ (for $(S\hat{P})K$), and $bd^2$ (for $(S\hat{P}K)M$) floating-point multiplications.
- The de-obfuscation requires an additional $b^2d$ (for $S^{-1}K'$) and $bd^2$ (for $(K')M^{-1}$) multiplications.

This results in a total of approximately $b^3 + 2b^2d + 2bd^2$ multiplications per block per decoding step. To put this into perspective, the cost of re-computing the same KV-cache block from the LLM's hidden states (dimension $D$) is $b \cdot D \cdot d$. For a model like LLaMA-3.1-8B (with $b = 16, d = 128, D = 4096$), the naive obfuscation overhead constitutes a substantial 7.1% of the re-computation cost.

**Efficiency Gains from Operator Fusion.** By fusing the matrix $M$ and its inverse into the model's weights offline, as described in Section V-D, we eliminate the two most expensive online multiplications ($bd^2$ terms). The online obfuscation and de-obfuscation, governed by Eq. (16), now only require approximately $b^3 + 2b^2d$ multiplications.

Revisiting the LLaMA-3.1-8B example, this optimization reduces the computational overhead to just 0.83% of the re-computation cost. This represents a nearly 8-fold reduction in latency compared to the naive implementation (specifically, the new cost is 11.72% of the original overhead). This dramatic improvement makes the runtime performance impact of KV-Cloak minimal and highly practical for real-world deployment.

**Auxiliary Costs.** Our analysis primarily focuses on floating-point multiplications, which dominate the computational cost. However, we acknowledge other minor costs, such as the

TABLE X: Efficacy of Defense Mechanisms Against Input Reconstruction Attacks on the Remaining Models.

| Model | Protect Type | Metric | Inversion | Collision | | | Collision+ | | | Injection |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | First | First | Mid | Last | First | Mid | Last | All |
| LLaMA-3.2-3B-Instruct | Original | BERTScore (↓) | 0.055 | 0.782 | 0.668 | 0.820 | 1.000 | 1.000 | 1.000 | 0.540 |
| | | ROUGE-L (↓) | 0.000 | 0.732 | 0.456 | 0.621 | 0.994 | 0.994 | 0.994 | 0.324 |
| | KV-Cloak | BERTScore (↓) | 0.088 | 0.069 | 0.070 | 0.069 | 0.033 | 0.033 | 0.033 | 0.088 |
| | | ROUGE-L (↓) | 0.000 | 0.000 | 0.000 | 0.000 | 0.042 | 0.042 | 0.042 | 0.000 |
| | $(10^8, 10^{-5})$-DP | BERTScore (↓) | 0.061 | 0.223 | 0.592 | 0.760 | 0.967 | 0.938 | 1.000 | 0.129 |
| | | ROUGE-L (↓) | 0.000 | 0.261 | 0.360 | 0.517 | 0.951 | 0.907 | 0.994 | 0.032 |
| LLaMA-3.1-8B | Original | BERTScore (↓) | 0.071 | 0.873 | 0.652 | 0.764 | 1.000 | 1.000 | 1.000 | 0.616 |
| | | ROUGE-L (↓) | 0.000 | 0.825 | 0.443 | 0.564 | 0.994 | 0.994 | 0.994 | 0.447 |
| | KV-Cloak | BERTScore (↓) | 0.076 | 0.069 | 0.069 | 0.069 | 0.041 | 0.041 | 0.041 | 0.084 |
| | | ROUGE-L (↓) | 0.004 | 0.000 | 0.000 | 0.000 | 0.003 | 0.003 | 0.003 | 0.000 |
| | $(10^8, 10^{-5})$-DP | BERTScore (↓) | 0.076 | 0.343 | 0.526 | 0.614 | 0.639 | 0.986 | 0.999 | 0.115 |
| | | ROUGE-L (↓) | 0.003 | 0.328 | 0.284 | 0.419 | 0.639 | 0.947 | 0.994 | 0.057 |
| Qwen2.5-Math-7B | Original | BERTScore (↓) | 0.229 | 0.918 | 0.552 | 0.783 | 1.000 | 0.983 | 0.996 | 0.422 |
| | | ROUGE-L (↓) | 0.186 | 0.842 | 0.355 | 0.580 | 1.000 | 0.977 | 0.996 | 0.286 |
| | KV-Cloak | BERTScore (↓) | 0.099 | 0.069 | 0.069 | 0.070 | 0.112 | 0.112 | 0.113 | 0.075 |
| | | ROUGE-L (↓) | 0.011 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| | $(10^8, 10^{-5})$-DP | BERTScore (↓) | 0.108 | 0.879 | 0.274 | 0.317 | 0.331 | 0.432 | 0.373 | 0.325 |
| | | ROUGE-L (↓) | 0.018 | 0.790 | 0.100 | 0.143 | 0.336 | 0.445 | 0.404 | 0.208 |

TABLE XI: Impact of PagedAttention block size on LLaMA-3.2-1B inference accuracy under KV-Cloak protection.

| Model | Metric | Plaintext | Block Size | | |
|---|---|---|---|---|---|
| | | | 16 | 32 | 64 |
| LLaMA-3.2-1B | MMLU (↑) | 0.335 | 0.335 | 0.335 | 0.335 |
| | SQuAD (↑) | 0.457 | 0.463 | 0.462 | 0.460 |

generation of the one-time permutation matrix $\hat{P}$, the element-wise additions for the mask $A$, and function call overhead. These costs are considered secondary for several reasons: the generation of $\hat{P}$ can be performed asynchronously in parallel with other computations; matrix addition has a much lower complexity than multiplication; and any remaining overhead can be further optimized through techniques like computation graph optimization and hardware acceleration.

## G. Architectural Compatibility with PagedAttention

The compatibility of KV-Cloak with modern inference engines stems from its core "block-oriented" design principle. All cryptographic operations—both obfuscation and de-obfuscation—are self-contained within a single physical memory block. This design intentionally creates no cross-block dependencies, allowing a memory manager like vLLM to schedule, copy, swap, and share physical blocks freely, without any awareness of their obfuscated contents.

To empirically validate this compatibility, we evaluated KV-Cloak with the most common PagedAttention block sizes: 16, 32, and 64. We measured accuracy on LLaMA-3.2-1B (Table XI) and latency on LLaMA-3.1-8B (Table XII). As shown in Table XI, KV-Cloak is virtually lossless. It preserves the baseline MMLU accuracy perfectly and results in negligible, statistically insignificant variations on SQuAD. In terms of performance (Table XII), the overhead of the optimized, fused

TABLE XII: Computational overhead of KV-Cloak (Fused vs. No Fuse) across different PagedAttention block sizes on LLaMA-3.1-8B.

| Model | Prefill | Type | Block Size | | |
|---|---|---|---|---|---|
| | | | 16 | 32 | 64 |
| LLaMA-3.1-8B | 3438.5 | No Fuse | 52.73 +1.53% | 45.45 +1.32% | 28.60 +0.83% |
| | | Fused | 15.41 +0.45% | 10.17 +0.30% | 12.33 +0.36% |

implementation remains consistently low, adding only $< 0.45\%$ latency relative to the prefill computation across all block sizes.

These results confirm that KV-Cloak's design has no fundamental conflicts with the PagedAttention memory management model. Its negligible impact on accuracy and its low, stable overhead across various block sizes demonstrate that a full integration into an inference engine like vLLM is a practical and feasible engineering task.

## H. Broader Impact of KV-Cloak for LLM Inference Security

Although KV-Cloak targets the KV-cache specifically, it underscores a broader issue: the internal states of large language models represent a rich and vulnerable attack surface. As models grow in scale and architectural complexity (e.g., via Mixture-of-Experts [31]), they produce substantial context-dependent intermediate data—such as activations and attention weights—that may leak sensitive information. KV-Cloak introduces a lightweight, structure-aware obfuscation approach as an alternative to costly cryptographic methods. By exploiting mathematical reversibility, it preserves model accuracy while embedding sufficient algebraic complexity to resist cryptanalysis. This algorithm-architecture co-design paradigm offers a promising direction for enhancing LLM inference security.