TRACES OF A PROMPT: UNCOVERING AND MITIGATING PRIVACY RISKS OF EXPOSED STATES IN TEE-PARTITIONED LLM INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Cloud-based LLM services must protect sensitive prompts while maintaining efficient inference. To balance these goals, partitioned inference keeps selected computations inside a trusted execution environment (TEE) while offloading others to an untrusted GPU. However, the intermediate states exposed by this partitioning may reveal private inputs. We introduce **Prompt-State Matching Attack**, which constructs direct, invariant, and joint-state fingerprints for vocabulary-based prompt reconstruction. Given observations of selected states and access to chosen-input probing, the attack achieves near-perfect token recovery through exhaustive vocabulary search in controlled FP32 experiments across four LLMs. We experimentally evaluate token-aligned transformations and analyze how relationships among exposed states support reconstruction. To mitigate this leakage, we propose **StateVeil**, which protects outsourced linear computation using Dual-LPN-based sparse encoding. We provide a conditional reduction for transcript privacy, complemented by concrete attack-cost estimates. Even with the true prefix supplied at each position, the evaluated Euclidean direct matcher achieves 0% token recovery from StateVeil's masked outputs across four models and four protected projections. StateVeil achieves end-to-end latency comparable to the evaluated partitioned-inference baseline, while model quality remains comparable to the original models in most evaluated model–task settings.

1 INTRODUCTION

Cloud-based LLM services process prompts containing personal information, proprietary code, and confidential business data (Huang et al., 2026). Protecting these inputs from an untrusted provider requires balancing prompt privacy, inference efficiency, and model quality.

Trusted execution environments (TEEs) isolate sensitive code and data (Costan & Devadas, 2016). Compared with homomorphic encryption (HE) (Gilad-Bachrach et al., 2016) and secure multiparty computation (MPC) (Wagh et al., 2019), protected plaintext execution can reduce cryptographic computation and communication costs (Tramèr & Boneh, 2019). However, running an entire LLM inside a TEE can limit performance. Partitioned inference retains selected operations inside the TEE and outsources others to untrusted accelerators (Tramèr & Boneh, 2019; Huang et al., 2026). This exposes intermediate values that may reveal prompts without compromising the TEE (Figure 1).

Prior vocabulary-matching attacks recover prompts from exposed KV caches (Luo et al., 2026) and hidden states (Thomas et al., 2025). In TEE-partitioned inference, the attacker's observations depend on which operations are outsourced and which values are jointly exposed outside the TEE. We ask whether these observations retain token-identifying fingerprints, within individual states or across related states, that can be compared with probe fingerprints without access to TEE secrets.

We introduce **Prompt-State Matching Attack**, adapting vocabulary probing to specified exposure and alignment conditions. It constructs direct fingerprints from attention states, invariant fingerprints under token-aligned encodings, and joint fingerprints from simultaneously exposed states. Across four LLMs, unprotected exposure of each of Q, K, V, S, A, O yields 100% token recovery at the first, middle, and last layers, showing that depth alone does not prevent reconstruction in these set-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

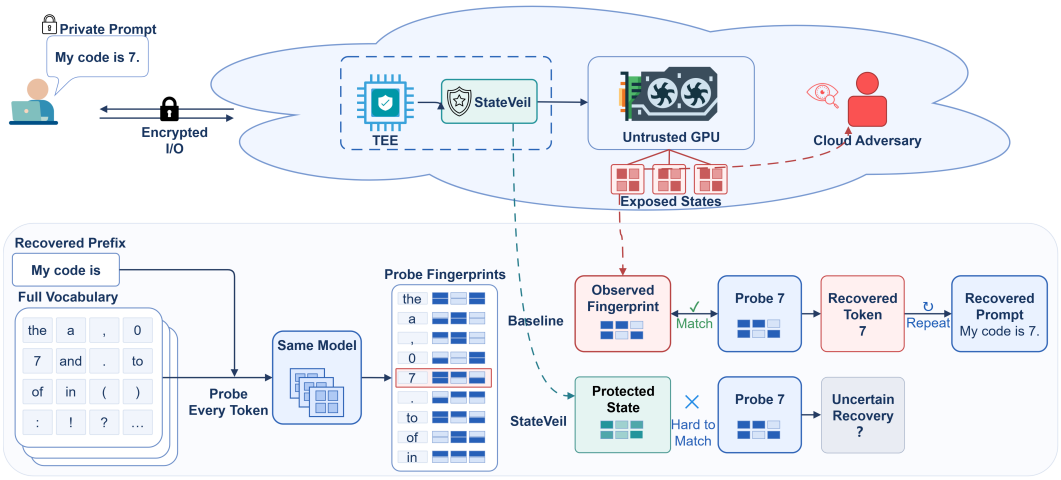


Figure 1: Prompt-state matching in partitioned inference. Victim and probe executions follow the selected protection mechanism; StateVeil aims to prevent reliable fingerprint matching.

tings. Invariant experiments assess recovery from preserved information, while joint-state analysis establishes when paired observations yield comparable fingerprints.

We propose **StateVeil** to protect outsourced linear operations using fresh exact-weight sparse masks over a finite field. A shared public master matrix and projection-specific precomputed products enable sparse mask generation and removal. Exact modular masking and GPU integer computation preserve the reference quantized Linear outputs. Under an exact-weight Dual-LPN assumption (Liu et al., 2024), we derive a joint-transcript privacy bound accounting for cumulative exposure under the shared matrix. The evaluated Euclidean direct matcher achieves 0% token recovery across four models and protected projections, even with the true prefix supplied at each position. Our TDX prototype achieves end-to-end latency comparable to the evaluated partitioned-inference baseline, with task-dependent quality changes from quantization.

Our contributions are:

- **TEE-boundary leakage analysis.** We adapt vocabulary-based reconstruction to specified outsourcing views and identify conditions under which directly exposed states, invariants of token-aligned encodings, or relationships between jointly observable states yield fingerprints comparable across victim and probe executions.
- **StateVeil.** We instantiate fresh exact-weight Dual-LPN masking for outsourced LLM linear operations using a shared public master matrix and projection-specific recovery tables. We provide exact quantized execution and a conditional privacy reduction that accounts for jointly exposed values and repeated observations across projections.
- **Evaluation.** We measure recovery across four LLMs, distinguish measured attacks from analytically implied joint recovery, and evaluate StateVeil's direct-matcher resistance, inference latency, and quantization effects.

2 Background and Related Work

2.1 LLM Inference and Intermediate States

For normalized inputs U, one attention head computes the query, key, and value projections (Vaswani et al., 2017):

$$Q = UW_Q, \qquad K = UW_K, \qquad V = UW_V. \tag{1}$$

Here, W_Q, W_K, W_V are learned projection matrices, with biases omitted. Let $\bar{Q}$ and $\bar{K}$ denote the queries and keys after optional query/key normalization (Yang et al., 2025) and rotary position

embeddings (Su et al., 2021); V undergoes neither operation. Attention computes

$$S = \frac{\bar{Q}\bar{K}^\top}{\sqrt{d_h}}, \qquad A = \mathrm{softmax}(S + \mathcal{M}), \qquad O = AV. \tag{2}$$

Here, d_h is the query/key dimension, $\mathcal{M}$ is the causal mask, and O precedes output projection. Autoregressive inference predicts each next token from the preceding sequence, with causal masking preventing attention to future tokens.

2.2 TEE-Partitioned LLM Inference

Cloud prompts require protection from the service provider. TEEs isolate sensitive code and data from untrusted software (Costan & Devadas, 2016). Compared with HE (Gilad-Bachrach et al., 2016) and MPC (Wagh et al., 2019), they support conventional plaintext computation with potentially lower cryptographic computation and communication costs (Tramèr & Boneh, 2019).

Running an entire LLM inside a TEE can limit performance. Partitioned inference therefore retains selected operations inside the TEE and outsources others to an untrusted GPU. Slalom additively masks outsourced linear layers (Tramèr & Boneh, 2019), while Talaria applies protected outsourcing to LLM inference (Huang et al., 2026). The TEE receives encrypted prompts, protects states before outsourcing, and recovers returned results. This reduces trusted computation but adds protection and communication overhead. Exposed states may still enable prompt reconstruction if their values or relationships retain distinguishing information.

2.3 Prompt Reconstruction Attacks

Hidden-state matching (Thomas et al., 2025) and the KV-cache collision attack (Luo et al., 2026) reconstruct prompts through vocabulary-based state matching. Both append candidate tokens to the recovered prefix and compare their resulting states with the observations to identify the next token. Hidden-state matching also survives certain permutation- and noise-based defenses (Thomas et al., 2025). We focus on attention states and their relationships; KV-cache storage and layer-level hidden states are not separately evaluated.

Two questions remain: *Can other intermediate states support such reconstruction? Can transformed states exposed outside the TEE still enable prompt matching?* Our attack extends the targeted states and comparison methods through direct, invariant, and joint-state matching.

3 Prompt-State Matching Attack

3.1 Threat Model

Adversary's Objective. The adversary aims to recover private user prompts from exposed computation values. We target verbatim reconstruction, which reveals the complete input and constitutes the most severe form of prompt leakage.

Adversary's Capabilities. We consider an honest-but-curious model provider that operates the cloud inference service and its untrusted GPU worker. This provider is the "Cloud Adversary" in Figure 1; it knows the model architecture, weights, tokenizer, inference implementation, protection algorithms, and public parameters, but cannot access TEE memory or secret transformation parameters. The provider runs arbitrary candidate inputs on a separate local copy of the model, never sending those probes to the victim's TEE. Protected local probes use the same protocol and public param- eters but independent secret parameters. Each matching construction must make their fingerprints comparable to the observed victim fingerprints. Victim inference follows the specified model and protection protocol.

3.2 Attack Procedure

Let $x = (x_1, \ldots, x_n)$ be the target input sequence over vocabulary $\mathcal{V}$, where $x_1 = \mathrm{BOS}$ is public and $x_2, \ldots, x_n$ are the user prompt tokens. Victim and probe inputs use the same BOS. The attacker

initializes $\hat{x} = (\text{BOS})$ and recovers positions $i = 2, \ldots, n$, using only the BOS and previously predicted tokens.

Let $\mathcal{O}^\star$ denote the victim's exposed observations and $\mathcal{O}(p)$ the observations obtained by probing input p under the evaluated exposure and protection settings. At position i, an extractor Φ_i constructs a fingerprint from the selected observable entries. For every vocabulary token v, the attacker computes

$$f_i^\star = \Phi_i(\mathcal{O}^\star), \qquad f_i(v) = \Phi_i(\mathcal{O}(\hat{x}_{<i} \| v)), \tag{3}$$

where $\hat{x}_{<i}$ is the reconstructed prefix and $\|$ denotes token concatenation. Given a fingerprint distance D, it selects

$$\hat{x}_i = \arg\min_{v \in \mathcal{V}} D(f_i^\star, f_i(v)). \tag{4}$$

The selected token extends the prefix until the sequence reaches length n; incorrect predictions can affect subsequent matches. For attention-weight matching, recovery begins at the second attention row, since the constant first row corresponds to the known BOS.

In Figure 1, the recovered text prefix is `My code is`. The attacker appends every vocabulary token and compares the resulting probe fingerprints with the victim's fingerprint. In the illustrated matching case, the closest match identifies `7`, which extends the recovered prefix. Appendix A provides pseudocode and execution details.

3.3 Matching Strategies

A fingerprint must be computable from exposed observations and depend only on the corresponding input prefix. For fixed public parameters, an exact construction satisfies

$$\Phi_i(\mathcal{O}_\theta(x)) = g_i(x_{\le i}), \tag{5}$$

where θ denotes secret protection parameters and g_i is independent of θ and the unknown suffix. Fingerprints must also distinguish candidates. Our experiments apply transformations separately at each token position; cross-token mixing requires a separate extraction argument.

Direct State Matching. Let Z_i contain the unencoded state entries selected for position i and its causal prefix. Without transformation, we use $\Phi_i^{\text{dir}}(\mathcal{O}) = \text{vec}(Z_i)$, where vec flattens these entries. Under deterministic exact computation, the correct token with a correct prefix reproduces this fingerprint.

Invariant Matching. For an encoding $\widetilde{Z}_i = T_{\theta,i}(Z_i)$ that acts within position i, choose an extractor I satisfying

$$I(T_{\theta,i}(Z_i)) = I(Z_i). \tag{6}$$

Then $\Phi_i^{\text{inv}}(\mathcal{O}) = I(\widetilde{Z}_i)$ removes the dependence on secret parameters. For example, sorting removes within-token feature permutations, as in hidden-state matching (Thomas et al., 2025). A whole-tensor invariant depending on unrecovered suffix tokens does not satisfy Equation 5.

Joint-State Matching. Joint matching exploits relationships among exposed states. For an observably aligned head, consider encodings of the processed queries and keys $\bar{Q}, \bar{K}$ from Section 2.1:

$$\widetilde{Q} = \bar{Q}R, \qquad \widetilde{K} = \bar{K}R^{-\top}, \tag{7}$$

where R is secret and invertible. If query row $\widetilde{q}_i$ and keys $\widetilde{K}_{\le i}$ through position i are identifiable,

$$\Phi_i^{\text{joint}}(\mathcal{O}) = \frac{\widetilde{q}_i \widetilde{K}_{\le i}^\top}{\sqrt{d_h}} = S_{i,1:i}. \tag{8}$$

The query uses R and every key in the product uses $R^{-\top}$; different executions may use different R. Their cancellation exposes a score fingerprint even when S remains inside the TEE. The construction requires both the paired encoding and observable position alignment.

4 ATTACK EVALUATION

We organize the evaluation around the two questions raised in Section 2.3: **RQ1:** whether other intermediate states support prompt reconstruction and **RQ2:** whether transformed states exposed outside the TEE still enable matching.

Table 1: Token recovery accuracy (%) of direct state matching. Each attack uses all heads of one state from one layer. $\bar{Q}$ and $\bar{K}$ denote states after query/key normalization when present and RoPE. Recovery starts from a known BOS token.

Model	Layer	$\bar{Q}$	$\bar{K}$	V	S	A	O
Llama-3.2-1B	First (0)	100.0	100.0	100.0	100.0	100.0	100.0
	Middle (8)	100.0	100.0	100.0	100.0	100.0	100.0
	Last (15)	100.0	100.0	100.0	100.0	100.0	100.0
Llama-3.1-8B	First (0)	100.0	100.0	100.0	100.0	100.0	100.0
	Middle (16)	100.0	100.0	100.0	100.0	100.0	100.0
	Last (31)	100.0	100.0	100.0	100.0	100.0	100.0
Qwen2.5-7B	First (0)	100.0	100.0	100.0	100.0	100.0	100.0
	Middle (14)	100.0	100.0	100.0	100.0	100.0	100.0
	Last (27)	100.0	100.0	100.0	100.0	100.0	100.0
Qwen3-8B	First (0)	100.0	100.0	100.0	100.0	100.0	100.0
	Middle (18)	100.0	100.0	100.0	100.0	100.0	100.0
	Last (35)	100.0	100.0	100.0	100.0	100.0	100.0

4.1 EXPERIMENTAL SETUP

Models. We evaluate Llama-3.2-1B (Meta, 2024), Llama-3.1-8B (Grattafiori et al., 2024), Qwen2.5-7B (Qwen Team, 2024), and Qwen3-8B (Yang et al., 2025), covering two model families and parameter scales from 1B to 8B. All models run in FP32 evaluation mode.

Datasets. We extract the first user message from each conversation in LMSYS-Chat-1M (Zheng et al., 2024). We sort the messages by the SHA-1 hash of their full text, select a fixed slice of 200 entries, and then deduplicate exact texts, leaving 192 unique prompts. All recovery settings use these prompt texts. Each model applies its own tokenizer without a chat template or a decode–retokenize step.

Evaluation Metrics. Token recovery accuracy is the fraction of non-special target tokens whose predicted IDs match at the same positions, aggregated over all prompts.

4.2 DIRECT STATE MATCHING

To answer RQ1, we evaluate $\bar{Q}, \bar{K}, V, S, A, O$ separately, without transformation, at the first, middle, and last layers on the 192 prompts per model. Each fingerprint concatenates all heads of one state and layer: token vectors for $\bar{Q}, \bar{K}, V, O$, or the valid causal row for S, A. Queries and keys are recorded after optional normalization and RoPE. Starting from the public BOS, we search the full vocabulary using the predicted prefix and Euclidean distance.

Table 1 shows 100% token accuracy in all 72 configurations, implying exact recovery of all 192 prompts per configuration on evaluated tokens. This answers RQ1: when these states are exposed without protection, moving the observation to a deeper layer alone does not prevent reconstruction.

4.3 INVARIANT MATCHING

To address RQ2, we examine transformation families motivated by low-rank masking (Huang et al., 2026), permutation (Yuan et al., 2025), and orthogonal encoding (Belikov & Fedotov, 2026), using

layer 0 states from 192 prompts per model. We consider configurations that preserve token positions. For each head, let C contain token states as rows and X denote its encoding.

Low-Rank Masking. For $X = C + MN$, the private matrix M supplies masking coefficients, while public $N \in \mathbb{R}^{r\times d}$ has rank $r < d$. An orthonormal right-null-space basis $P \in \mathbb{R}^{d\times(d-r)}$ satisfies

$$NP = 0, \qquad XP = (C + MN)P = CP. \tag{9}$$

We evaluate the ideal fingerprint $C_{i,:}P$ on $\bar{Q}, \bar{K}, V, O$, using the same N, P for victim and probes. The rank ratio is $r/d = 1/2$. Thus, each head retains $d - r$ invariant coordinates.

Permutation. Let $\mathbf{c}_i$ collect the selected entries across all heads at position i, and let $\mathbf{x}_i$ be their permutation. Independent victim and probe permutations preserve

$$\mathrm{sort}(\mathbf{x}_i) = \mathrm{sort}(\mathbf{c}_i). \tag{10}$$

We evaluate this extracted fingerprint on all six states. For S, A, the selected entries cover the valid causal prefix.

Orthogonal Mixing. We consider right-multiplicative encoding $X = CR$, where $R \in \mathbb{R}^{d\times d}$ is orthogonal. Each execution uses the same R across token rows, while victim and probe matrices may differ. Right multiplication preserves token positions, allowing the adversary to extract $X_{\le i} = X_{1:i,:}$ and compute

$$X_{\le i}X_{\le i}^\top = C_{\le i}RR^\top C_{\le i}^\top = C_{\le i}C_{\le i}^\top. \tag{11}$$

The fingerprint is the vectorized prefix Gram matrix. We consider $\bar{Q}, \bar{K}, V, A, O$. For A, future attention weights are zero, so its prefix Gram matrix contains no contribution from subsequent tokens.

Table 2 reports 100% recovery from ideal null-space fingerprints and 99.81–100% from sorting the permuted observations. These results answer RQ2 for the evaluated configurations: transformations can preserve sufficient information for prompt matching without revealing their secret parameters.

Table 2: Token recovery accuracy (%) of invariant matching at layer 0. $\bar{Q}$ and $\bar{K}$ are post-RoPE states.

Fingerprint	Model	$\bar{Q}$	$\bar{K}$	V	S	A	O
$\mathrm{vec}(X_iP)$	Llama-3.2-1B	100.00	100.00	100.00	—	—	100.00
	Llama-3.1-8B	100.00	100.00	100.00	—	—	100.00
	Qwen2.5-7B	100.00	100.00	100.00	—	—	100.00
	Qwen3-8B	100.00	100.00	100.00	—	—	100.00
$\mathrm{sort}(\mathrm{vec}(X_i))$	Llama-3.2-1B	100.00	100.00	100.00	100.00	100.00	100.00
	Llama-3.1-8B	100.00	100.00	100.00	100.00	100.00	100.00
	Qwen2.5-7B	100.00	100.00	100.00	100.00	99.81	100.00
	Qwen3-8B	100.00	100.00	100.00	100.00	100.00	100.00
$\mathrm{vec}(X_{\le i}X_{\le i}^\top)$	Llama-3.2-1B	100.00	100.00	100.00	—	100.00	100.00
	Llama-3.1-8B	100.00	100.00	100.00	—	100.00	100.00
	Qwen2.5-7B	100.00	100.00	100.00	—	100.00	100.00
	Qwen3-8B	100.00	100.00	100.00	—	100.00	100.00

4.4 Joint-State Matching

To further address RQ2, we examine paired query/key encodings. KV-Cloak's operator fusion produces $q_i^m = q_i M_1^{-\top}$ and $k_j^m = k_j M_1$ before its separate KV-cache encoding (Luo et al., 2026). An adversary observing GPU projection outputs as they are produced can record these aligned values, including keys before the cache's block permutation and additive mask. STIP likewise applies paired feature permutations to queries and keys (Yuan et al., 2026).

Equation 8 shows that the paired encodings yield an attention-score fingerprint when the relevant heads and token positions are identifiable. Section 4.2 reports 100% recovery with layer-0 score

fingerprints across all heads on the 192 prompts; we reuse that result without collecting additional victim traces. This establishes the fingerprint's discriminative power in this setting.

These findings answer RQ2 under the stated conditions: relationships among jointly exposed states can yield token-identifying fingerprints despite protective transformations. When the required states are observable and aligned, these fingerprints support prompt matching without recovering secret transformation parameters. Privacy analysis must therefore account for the combined information exposed at the TEE boundary.

5 StateVeil

5.1 Threat Model

We use the model-provider capabilities in Section 3.1 and assume honest-but-curious execution. The adversary knows the model and protocol, observes exposed values, and may jointly analyze chosen-input probes and victim executions. The TEE hardware and inference code protect private states and randomness (Costan & Devadas, 2016); users exchange prompts and responses through a secure channel. We study computation-value leakage, excluding active deviations (Ghodsi et al., 2017; Checkoway & Shacham, 2013), TEE compromise (Van Bulck et al., 2018), side channels (Xu et al., 2015), and traffic metadata (McDonald & Bar Or, 2025).

5.2 StateVeil

StateVeil outsources the query, key, value, output, and MLP gate, up, and down projections in every Transformer block. Embedding, normalization, RoPE, attention, nonlinear operations, the output head, token selection, and the KV cache remain inside the TEE. Figure 1 contrasts fingerprint matching with StateVeil's protected computation.

Representation. Consider $Y = CW + \text{bias}$, where $C \in \mathbb{R}^{b\times d}$ is private and $W \in \mathbb{R}^{d\times m}$ is public. StateVeil encodes activations and weights as signed 8-bit integers $C_q, W_q \in [-127, 127]$, using activation-row scales α_i and weight-column scales β_j. A prime $p > 2d\,127^2$ permits unique recovery of integer dot products. Appendix B.1 specifies rounding, zero scales, bias handling, and exact arithmetic.

Public tables and secret masks. A uniformly sampled master matrix is shared across the model. For input width d, a fixed submatrix $A_d \in \mathbb{F}_p^{L_d\times d}$ is used, with $L_d > d$. For each projection, we precompute $B = A_d W_q$ on the host and load the public tables into guest memory before inference.

Each invocation independently samples every row of $S \in \mathbb{F}_p^{b\times L_d}$ from $\mathcal{D}_{L_d,s,p}$: choose a uniformly random s-element support and assign independent uniform coefficients from $\mathbb{F}_p \setminus \{0\}$. Secret randomness is generated inside the TEE and is not reused. This is the exact-weight distribution used in our Dual-LPN assumption (Liu et al., 2024). Parameter interpretation and sharing are detailed in Appendix B.3; concrete values are reported in Section 6.1.

Protected execution. StateVeil forms the encoded input X inside the TEE, obtains the GPU product Z, and privately recovers T. Over $\mathbb{F}_p$,

$$X = C_q + SA_d, \qquad Z = XW_q, \qquad T = Z - SB = C_q W_q. \tag{12}$$

It then restores

$$\widehat{Y}_{ij} = \text{cast}_\rho((\alpha_i\beta_j)\,\text{lift}_p(T_{ij})) + \text{bias}_j, \tag{13}$$

where ρ is the original output dtype and lift_p denotes centered integer lifting (Tramèr & Boneh, 2019). Bias is added in dtype ρ after casting. Shared-memory messages carry X, Z and public protocol metadata; masks, activation scales, and restored outputs remain private. Restored outputs feed subsequent model operations.

5.3 Analysis

Correctness. Since $B = A_d W_q$, subtracting SB cancels the mask exactly. The modulus bound makes centered lifting recover $C_q W_q$. The protocol therefore preserves the specified quantized computation; differences from the original model arise from quantization and floating-point restoration. Appendix B.1 gives the separate exactness conditions for CPU masking and GPU multiplication.

Privacy. Let $\delta_d(\tau)$ be the maximum probability-gap advantage, within time τ, for distinguishing $(A_d, \mathbf{e}A_d)$ from $(A_d, \mathbf{u})$, where $\mathbf{e} \leftarrow \mathcal{D}_{L_d,s,p}$ and $\mathbf{u}$ is uniform in $\mathbb{F}_p^d$. The reduction concerns computation values for challenge executions with the same public operation sequence, dimensions, and termination behavior.

Theorem 1 (Privacy with a shared master matrix) *Under the stated sampling rules and leakage restrictions, an adversary's guessing advantage satisfies*

$$\left|\Pr[\hat{\gamma} = \gamma] - \frac{1}{2}\right| \leq \sum_d Q_d \, \delta_d(\tau') + \varepsilon_{\mathrm{rng}},$$

where γ selects the challenge input, Q_d bounds the total exposed rows of width d, τ' includes reduction overhead, and $\varepsilon_{\mathrm{rng}}$ accounts for replacing practical randomness with ideal sampling.

The budget includes victim and protected probe executions sharing the master matrix. Since $Z = XW_q$ and $B = A_dW_q$ are publicly computable, the bound covers all three matching strategies on their joint transcript. Appendix B.4 gives the proof.

Concrete assessment. Across seven widths, four evaluated classical attacks give minimum estimated costs of $2^{161.76}$–$2^{168.10}$. Under an additional time–advantage heuristic, a proposed cap of $M = 2^{24}$ processed token positions per master matrix yields cumulative exponents of 131.15–131.45. Appendix B.3 gives the estimates and rotation rule.

Cost. Online masking and recovery require $O(bs(d + m))$ field operations, excluding quantization, restoration, and communication. Public-table storage and preprocessing are separate costs, detailed in Appendix B.5.

6 Evaluation

We evaluate StateVeil's prompt privacy, model quality, and inference latency.

6.1 Experimental Setup

Models and datasets. We use the four models from Section 4.1 and follow its LMSYS-Chat-1M (Zheng et al., 2024) evaluation protocol. Quality is evaluated on the full MMLU test set (14,042 questions) (Hendrycks et al., 2021) and SQuAD 1.1 validation set (10,570 questions) (Rajpurkar et al., 2016).

Platform and baselines. StateVeil runs in an Intel TDX guest with 16 vCPUs, 128 GiB of memory, and 16 computation threads on an Intel 6530 server. The guest performs trusted computation on the CPU and outsources masked linear operations to an NVIDIA L40S worker on the untrusted host through synchronous RPC over a 64-MiB `ivshmem` region. We compare against Full-TEE (all inference inside the TEE), Talaria (Huang et al., 2026), and No-TEE (unprotected GPU inference).

Implementation and parameters. StateVeil uses $p = 2^{31} - 1$, $L = \max(4096, 2d_{\mathrm{in}})$, and $s = 128$, where d_{in} is the input dimension of each outsourced linear operation. We reimplement Talaria (Huang et al., 2026) on the same platform, using masking rank $r = 128$.

Metrics. We report token recovery accuracy, MMLU accuracy, SQuAD exact match (EM) and F1, and mean time to first token (TTFT) and end-to-end latency (E2E), measured in seconds from pre-tokenized generation until the corresponding output token is available on the CPU. Timing includes online protection, communication, and model execution; it excludes initialization, offline preprocessing, and warm-up (Appendix B.2).

6.2 Privacy Protection

We evaluate Q, K, V, O separately under unmasked quantization and StateVeil. At each position i, the attacker receives $x_{<i}$, probes every $x_{<i} \| c$ for $c \in \mathcal{V}$, and guesses only x_i. Guesses never enter later probes. The attacker observes X, Z and applies Euclidean distance to integer representatives, without modular subtraction (Appendix C.1). Across four models and four projections, quantization allow 100% token recovery; StateVeil yields 0% under this true-prefix oracle. Since the controls

Table 3: Model quality (%). Each entry reports Original / Quantized.

Model	MMLU Acc.	SQuAD EM	SQuAD F1
Llama-3.2-1B	37.83 / 37.58	27.36 / 26.41	42.36 / 41.68
Llama-3.1-8B	63.89 / 63.99	57.25 / 56.56	72.20 / 71.58
Qwen2.5-7B	71.79 / 71.36	78.06 / 74.74	86.53 / 83.91
Qwen3-8B	72.85 / 72.61	23.60 / 20.31	51.18 / 48.36

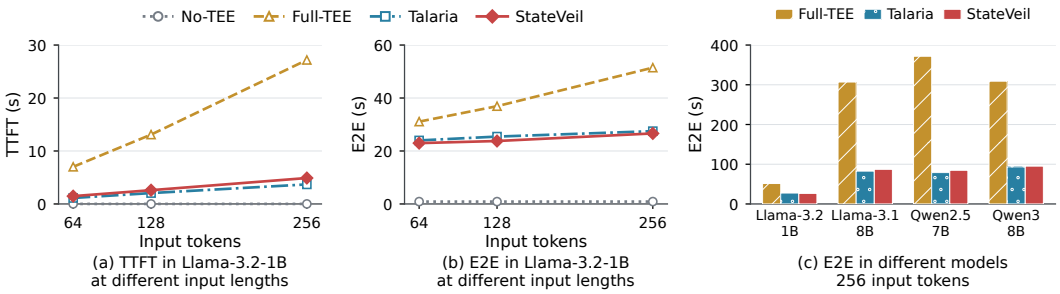


Figure 2: Inference latency of StateVeil and baselines.

recover every token sequentially, their probe prefixes coincide with the ground truth. StateVeil's zero recovery under true-prefix probing rules out error propagation as the cause of failure. These results show that fresh masking disrupts token identification by the evaluated direct matcher, even when the adversary knows the complete preceding context.

6.3 Model Quality

We compare the original models with the quantized computation used by StateVeil on MMLU and SQuAD (Table 3). MMLU accuracy changes by at most 0.43 percentage points, while SQuAD EM and F1 decrease by 0.69–3.32 and 0.62–2.82 points, respectively.

To identify the source of these changes, we compare StateVeil with unprotected quantized computation on actual Linear inputs. Final outputs are bitwise identical in all verified calls, supporting attribution to quantization without additional error from masking or recovery in these executions.

6.4 Inference Performance

Figure 2 reports mean latency for 100-token generations: (a)–(b) vary input length for Llama-3.2-1B, and (c) compares models with 256 input tokens. Without a trusted GPU, all TEE computation runs on the CPU, contributing to the gap from No-TEE. StateVeil reduces TTFT and E2E latency relative to Full-TEE. Compared to Talaria, it has higher TTFT but comparable E2E latency.

7 Discussion and Conclusion

Prompt-State Matching Attack shows that exposed attention states and preserved relationships can support vocabulary-based reconstruction. We evaluate token-aligned transformations and derive joint recovery under specified encoding and alignment conditions. Applicability depends on each deployment's protection configuration and exposed states.

StateVeil combines a conditional transcript-privacy reduction with concrete attack-cost estimates and a proposed exposure budget. True-prefix tests yield 0% token recovery against the evaluated Euclidean direct matcher across four models and four projections. Our TDX prototype achieves end-to-end latency comparable to the evaluated partitioned-inference baseline. Verified Linear outputs exactly match the quantized reference, while quality effects remain task-dependent. Future work will reduce probing costs and improve StateVeil's efficiency.

AI USE STATEMENT

AI USE STATEMENT

We used ChatGPT and Codex to assist with refinement of research ideas and methodology, mathematical analysis and proof drafting, code development and revision, and manuscript writing and LaTeX formatting. The authors directed the research and critically reviewed and revised the AI-assisted material. We take responsibility for the final content and artifacts of this work, including its technical claims, proofs, code, references, and reported experimental results.

ETHICS STATEMENT

This work investigates prompt leakage in TEE-partitioned LLM inference to improve the protection of user privacy. Prompt reconstruction has dual-use implications, as the same techniques used to assess privacy risks could enable unauthorized recovery of sensitive information. Our experiments use publicly available models and benchmark datasets in controlled settings. We report aggregate recovery metrics to limit further exposure of potentially sensitive prompt content. Findings are scoped to the stated observation capabilities and evaluated configurations, with empirical results distinguished from conditional security guarantees. We accompany the attack analysis with StateVeil as a mitigation and intend this research to support authorized security evaluation and the development of more secure inference systems.

REPRODUCIBILITY STATEMENT

Our implementation and experimental instructions are available at `https://anonymous.4open.science/r/stateveil/README.md`. The repository documents dataset selection, software dependencies, attack configurations, StateVeil's numerical implementation, and TDX–GPU deployment, with scripts for prompt recovery, model quality, and inference performance evaluations.

REFERENCES

Anatoly Belikov and Ilya Fedotov. Good-enough LLM obfuscation (GELO). arXiv preprint arXiv:2603.05035, 2026. URL https://arxiv.org/abs/2603.05035.

Stephen Checkoway and Hovav Shacham. Iago attacks: Why the system call API is a bad untrusted RPC interface. In *Proceedings of the Eighteenth International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 253–264. Association for Computing Machinery, 2013. doi: 10.1145/2451116.2451145. URL https://checkoway.net/papers/iago2013/.

Victor Costan and Srinivas Devadas. Intel SGX explained. Cryptology ePrint Archive, Paper 2016/086, 2016. URL https://eprint.iacr.org/2016/086.

Zahra Ghodsi, Tianyu Gu, and Siddharth Garg. SafetyNets: Verifiable execution of deep neural networks on an untrusted cloud. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/hash/6048ff4e8cb07aa60b6777b6f7384d52-Abstract.html.

Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. CryptoNets: Applying neural networks to encrypted data with high throughput and accuracy. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 201–210. PMLR, 2016. URL https://proceedings.mlr.press/v48/gilad-bachrach16.html.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, et al. The Llama 3 herd of models. arXiv preprint arXiv:2407.21783, 2024. URL https://arxiv.org/abs/2407.21783.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2009.03300.

Chung-ju Huang, Huiqiang Zhao, Yuanpeng He, Lijian Li, Wenpin Jiao, Zhi Jin, Peixuan Chen, and Leye Wang. Your inference request will become a black box: Confidential inference for cloud-based large language models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 134–154. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.4. URL https://aclanthology.org/2026.acl-long.4/.

Hanlin Liu, Xiao Wang, Kang Yang, and Yu Yu. The hardness of LPN over any integer ring and field for PCG applications. In *Advances in Cryptology – EUROCRYPT 2024, Part VI*, volume 14656 of *Lecture Notes in Computer Science*, pp. 149–179. Springer, 2024. doi: 10.1007/978-3-031-58751-1_6. URL https://eprint.iacr.org/2022/712.

Zhifan Luo, Shuo Shao, Su Zhang, Lijing Zhou, Yuke Hu, Chenxu Zhao, Zhihao Liu, and Zhan Qin. Shadow in the cache: Unveiling and mitigating privacy risks of KV-cache in LLM inference. In *Network and Distributed System Security Symposium*. Internet Society, 2026. URL https://www.ndss-symposium.org/wp-content/uploads/2026-f258-paper.pdf.

Geoff McDonald and Jonathan Bar Or. Whisper Leak: A side-channel attack on large language models. arXiv preprint arXiv:2511.03675, 2025. URL https://arxiv.org/abs/2511.03675.

Meta. Llama 3.2 model card. Model card, 2024. URL https://huggingface.co/meta-llama/Llama-3.2-1B.

Qwen Team. Qwen2.5 technical report. arXiv preprint arXiv:2412.15115, 2024. URL https://arxiv.org/abs/2412.15115.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392. Association for Computational Linguistics, 2016. doi: 10.18653/v1/D16-1264.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. arXiv preprint arXiv:2104.09864, 2021. URL https://arxiv.org/abs/2104.09864.

Rahul Krishna Thomas, Louai Zahran, Erica Choi, Akilesh Potti, Micah Goldblum, and Arka Pal. Hidden no more: Attacking and defending private third-party LLM inference. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 59434–59469. PMLR, 2025. URL https://proceedings.mlr.press/v267/thomas25b.html.

Florian Tramèr and Dan Boneh. Slalom: Fast, verifiable and private execution of neural networks in trusted hardware. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1806.03287.

Jo Van Bulck, Marina Minkin, Ofir Weisse, Daniel Genkin, Baris Kasikci, Frank Piessens, Mark Silberstein, Thomas F. Wenisch, Yuval Yarom, and Raoul Strackx. Foreshadow: Extracting the keys to the Intel SGX kingdom with transient out-of-order execution. In *27th USENIX Security Symposium (USENIX Security 18)*, pp. 991–1008. USENIX Association, 2018. URL https://www.usenix.org/conference/usenixsecurity18/presentation/bulck.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://arxiv.org/abs/1706.03762.

Sameer Wagh, Divya Gupta, and Nishanth Chandran. SecureNN: 3-party secure computation for neural network training. *Proceedings on Privacy Enhancing Technologies*, 2019(3):26–49, 2019. doi: 10.2478/popets-2019-0035. URL https://petsymposium.org/popets/2019/popets-2019-0035.php.

Yuanzhong Xu, Weidong Cui, and Marcus Peinado. Controlled-channel attacks: Deterministic side channels for untrusted operating systems. In *2015 IEEE Symposium on Security and Privacy*, pp. 640–656. IEEE, 2015. doi: 10.1109/SP.2015.45.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, et al. Qwen3 technical report. arXiv preprint arXiv:2505.09388, 2025. URL https://arxiv.org/abs/2505.09388.

Yu Yu, Kang Yang, Xiao Wang, Anyu Wang, Tianrui Wang, Hanlin Liu, Xinpeng Hao, and Juanru Li. Estimator of LPN problems over any finite fields and power-of-two rings for PCG and MPC applications. Software, 2025. URL https://lpnestimator.com.

Mu Yuan, Lan Zhang, Liekang Zeng, Siyang Jiang, Bufang Yang, Di Duan, and Guoliang Xing. SCX: Stateless KV-cache encoding for cloud-scale confidential transformer serving. In *Proceedings of the ACM SIGCOMM 2025 Conference*, pp. 39–54. Association for Computing Machinery, 2025. doi: 10.1145/3718958.3750509. URL https://doi.org/10.1145/3718958.3750509.

Mu Yuan, Lan Zhang, Yihang Cheng, Miao-Hui Song, Guoliang Xing, and Xiang-Yang Li. STIP: Three-party privacy-preserving and lossless inference for large transformers in production. In *Network and Distributed System Security Symposium*. Internet Society, 2026. doi: 10.14722/ndss.2026.230035. URL https://www.ndss-symposium.org/wp-content/uploads/2026-s35-paper.pdf.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric P. Xing, Joseph E. Gonzalez, Ion Stoica, and Hao Zhang. LMSYS-Chat-1M: A large-scale real-world LLM conversation dataset. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=BOfDKxfwt0.

Algorithm 1 Prompt-State Matching Attack with BOS Initialization

Require: Victim observations $\mathcal{O}^\star$, input length n including BOS, vocabulary $\mathcal{V}$, extractors Φ_i, distance D, and probe procedure Probe
Ensure: Reconstructed user prompt $\hat{x}_{2:n}$

$\hat{x} \leftarrow (\mathrm{BOS})$
for $i = 2, \ldots, n$ **do**
  $f_i^\star \leftarrow \Phi_i(\mathcal{O}^\star)$
  $d_{\mathrm{best}} \leftarrow +\infty$; $v_{\mathrm{best}} \leftarrow +\infty$
  for each $v \in \mathcal{V}$ **do**
    $f_i(v) \leftarrow \Phi_i(\mathrm{Probe}(\hat{x} \| v))$
    $d \leftarrow D(f_i^\star, f_i(v))$
    if $d < d_{\mathrm{best}}$ or $(d = d_{\mathrm{best}}$ and $v < v_{\mathrm{best}})$ **then**
      $d_{\mathrm{best}} \leftarrow d$; $v_{\mathrm{best}} \leftarrow v$
    end if
  end for
  $\hat{x} \leftarrow \hat{x} \| v_{\mathrm{best}}$
end for
return $\hat{x}_{2:n}$

A Prompt-State Matching Algorithm

Algorithm 1 implements Section 3.2. Throughout this appendix, n includes the initial BOS, leaving $n - 1$ user prompt tokens to recover.

A.1 Initialization and Attention-Weight Matching

Both victim and probe inputs begin with the same public BOS token. We set $p_0 = (\mathrm{BOS})$, $k = 1$, and $\hat{x} = p_0$. Recovery starts at position 2 without any known user prompt tokens. The supplied BOS and other target special tokens are excluded from token recovery accuracy.

For causal attention, $A_{1,1} = 1$ in every head, so the first row cannot identify a token. This row corresponds to the known BOS and is not used for token selection. At the first unknown position, attention-weight matching instead uses

$$A_{2,1:2} = \mathrm{softmax}([S_{2,1}, S_{2,2}])\,. \tag{14}$$

At position i, it uses the valid causal entries $A_{i,1:i}$. Unlike the first row, these entries can depend on the candidate token. BOS initialization removes the first-row degeneracy but does not guarantee unique fingerprints; indistinguishable candidates remain unresolved by the selected observations alone.

A.2 Probe Execution and Candidate Selection

$\mathrm{Probe}(p)$ executes the chosen token sequence p using the probe implementation permitted by Section 3.1 and returns the observations admitted by the exposure setting. Protected probes follow the specified parameter policy without access to victim-side secrets. The extractor Φ_i aligns layers, heads, and positions only where the available observations permit this alignment. The matching construction must establish comparability between victim and probe fingerprints.

Each candidate is evaluated using the reconstructed prefix; no ground-truth user tokens or victim states are injected into its forward computation. Search covers the complete vocabulary, including special tokens. All candidates are evaluated, even if a zero distance is found. Exact ties are resolved by the smallest token ID.

A.3 Probe Cost and Error Propagation

Exhaustive search requires $(n - 1)|\mathcal{V}|$ candidate evaluations. Batching reduces execution requests but not the number of candidates. At each position, the victim fingerprint is computed once, and candidate distances are processed incrementally, retaining the best token and distance. Each selected token becomes part of the next probe input, so recovery errors can propagate to later positions.

A.4 Computational Cost

For one model, state, layer, and matching configuration, let n_j denote the length of input j, including BOS. Exhaustive recovery over M inputs evaluates

$$N_{\text{cand}} = |\mathcal{V}| \sum_{j=1}^{M} (n_j - 1) \tag{15}$$

candidates. These are local candidate evaluations, not necessarily external service queries. With a fixed candidate micro-batch size b, the corresponding number of candidate-batch forward calls is

$$N_{\text{batch}} = \left\lceil \frac{|\mathcal{V}|}{b} \right\rceil \sum_{j=1}^{M} (n_j - 1). \tag{16}$$

This count excludes additional calls for prefix initialization and cache updates.

Candidates reuse the KV cache of the reconstructed prefix and execute only up to the selected observation point. Thus, the target layer affects the computation required per candidate. Captured victim states are reused across comparison methods; this reuse does not imply shared candidate computation across layers.

The implementation defaults to candidate micro-batches of 256 within logical batches of 512. Search evaluates the full vocabulary and does not terminate upon finding a zero distance. These optimizations reduce redundant computation and execution overhead without reducing the number of evaluated candidates.

B StateVeil: Implementation and Security Analysis

B.1 Quantization and Exact Arithmetic

Activations are quantized per row and weights per output column. For $C \in \mathbb{R}^{b \times d}$ and $W \in \mathbb{R}^{d \times m}$, define

$$a_i = \max_k |C_{ik}|, \qquad w_j = \max_k |W_{kj}|,$$

and compute the scales in FP64:

$$\alpha_i = \begin{cases} a_i(1/127), & a_i > 0, \\ 1, & a_i = 0, \end{cases} \qquad \beta_j = \begin{cases} w_j(1/127), & w_j > 0, \\ 1, & w_j = 0. \end{cases} \tag{17}$$

The integer codes are

$$\begin{aligned} (C_q)_{ik} &= \text{clip}_{[-127,127]}\big(\text{roundEven}(C_{ik}/\alpha_i)\big), \\ (W_q)_{kj} &= \text{clip}_{[-127,127]}\big(\text{roundEven}(W_{kj}/\beta_j)\big). \end{aligned} \tag{18}$$

Encoding uses FP64 division followed by ties-to-even rounding; scale construction uses multiplication by $1/127$. Weights are encoded once and stored as INT8. Signed codes are interpreted modulo p when forming field values.

For a canonical residue $t \in \{0, \ldots, p-1\}$,

$$\text{lift}_p(t) = \begin{cases} t, & t \le (p-1)/2, \\ t - p, & t > (p-1)/2. \end{cases} \tag{19}$$

Every protected projection is checked against $d\,127^2 \le (p-1)/2$, ensuring $|(C_q W_q)_{ij}| \le (p-1)/2$. Restoration computes $(\alpha_i \beta_j)\,\text{lift}_p(T_{ij})$ in FP64, casts to the original output dtype, and then adds any bias in that dtype.

Distinct arithmetic paths. The CPU forms SA_d and SB using exact integer arithmetic and Mersenne modular reduction. These operations multiply general field elements and require reduction during accumulation. In contrast, the GPU retains signed $W_q \in [-127, 127]$ and checks

$$d(p-1)127 < 2^{53}. \tag{20}$$

With FP64 multiplication and accumulation, every integer product and partial sum in XW_q is then exactly representable. The resulting integer dot products are reduced modulo p and returned as INT32 residues. Equation 20 does not justify FP64 multiplication of two unrestricted field matrices.

For the experimental modulus $p = 2^{31} - 1$, reducing after each integer multiply-add is overflow-safe because $(p-1)^2 + (p-1) < 2^{63}$. Any fused accumulation must enforce an equivalent intermediate bound. The CPU and GPU paths must implement the same field operations; the centered-lifting bound alone does not establish this property.

B.2 Operator Placement and Public Tables

Each Transformer block protects `q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, and `down_proj`. Tokenization, embedding, Q/K normalization when present, RoPE, attention scores and softmax, attention-value multiplication, activations, residual connections, final normalization, `lm_head`, and token selection execute on the guest CPU. The KV cache remains in guest-private memory. Prefill processes the input sequence, while decoding reuses this private cache and processes newly appended tokens.

The host prepares the public master matrix and weight-specific tables before inference. The guest loads and checks these tables during initialization; online execution reads resident memory rather than loading tables from disk. The host GPU stores INT8 weight codes and expands only the active projection into a reusable FP64 workspace.

Secret supports and coefficients are sampled using guest `getrandom()` randomness, independently for each row and projection invocation. The same S generates both the input mask SA_d and the output correction SB within an invocation. Shared memory carries only encoded inputs and encoded results, together with public protocol metadata. Recovery writes restored values exclusively to guest-private memory.

Communication and synchronization. A 64-MiB QEMU `ivshmem-plain` region connects one guest client to one host GPU worker on the same machine. The host maps a shared file backend and the guest maps the device's PCI BAR2. The GPU is not passed through to the guest. Control metadata and two directional slots occupy the region; each slot has usable capacity slightly below 32 MiB. Online requests use a 64-byte binary header and INT32 encoded arrays. SSH is used only for deployment.

The guest writes encoded inputs to the request slot. The host reads a view, transfers inputs to preallocated GPU buffers, and copies computed results through a pinned host buffer into the reply slot. The guest restores directly from that slot into private memory. Shared-slot writes and host–device transfers remain necessary. Release/acquire sequence numbers publish complete messages, and the worker uses a fixed CUDA stream. It publishes a reply only after the device-to-host copy and slot write finish. One request is outstanding per channel; the next begins after the guest consumes and restores the current outputs.

Q/K/V, O, gate/up, and down form four RPC groups with seven GEMMs per block. Each projection retains an independent mask. Without message chunking, an ℓ-block model generating N output tokens uses $4\ell N$ RPCs: one prefill and $N-1$ decoding passes. Slot capacity or row limits can split prefill messages.

Timing and consistency checks. TTFT and E2E start at generation on pretokenized inputs and end when the first and final tokens, respectively, are available on the CPU. Online quantization, mask sampling, communication, transfers, GPU computation, restoration, and remaining model execution are included. VM/model startup, public-table generation/loading, kernel compilation, workspace allocation, warm-up, tokenization, detokenization, and external client networking are excluded. Detailed profiling is separate from latency measurement; guest RPC waiting already includes host service time. In verification runs, unmasked references use actual Linear inputs. Checks compare the fused encoder's C_q, α, the recovered integer product against $C_q W_q$, and final output bytes after restoration, casting, and bias. These checks assess numerical consistency on actual model inputs.

Table 4: Estimated $\log_2$ single-instance classical attack costs for exact-weight masks with $p = 2^{31} - 1$ and $s = 128$.

d	L_d	**Pooled Gauss**	**SD-ISD**	**SD**	**SD2**
2,048	4,096	161.76	161.76	273.10	272.16
3,584	7,168	162.73	162.73	271.24	270.72
4,096	8,192	163.05	163.05	271.00	270.54
8,192	16,384	165.12	165.12	270.49	270.26
12,288	24,576	166.52	166.52	270.57	270.42
14,336	28,672	167.07	167.07	270.65	270.53
18,944	37,888	168.10	168.10	270.85	270.75

B.3 Parameters and Exposure Budget

Let $d_{\max}$ and $L_{\max} = \max_d L_d$ cover all protected input widths. The shared public matrix and its fixed slices are

$$A_{\text{master}} \leftarrow \text{Unif}(\mathbb{F}_p^{L_{\max} \times d_{\max}}), \qquad A_d = A_{\text{master}}[0 : L_d,\, 0 : d]. \tag{21}$$

For the evaluated widths, $L_d = 2d$ and $L_{\max} = 2d_{\max}$. Each slice has the required uniform marginal distribution; overlapping slices are correlated. The proof below explicitly accommodates these correlations.

The modulus and width bounds ensure arithmetic correctness. We require $1 \le s < L_d$. Neither $L_d > d$ nor full column rank establishes Dual-LPN hardness. We assess the experimental values $p = 2^{31}-1$, $L_d = \max(4096, 2d)$, and $s = 128$ using the exact-weight, prime-field distribution (Liu et al., 2024). Here, d is a protected Linear's input width, not a head dimension.

Concrete attack-cost estimates. We use the LPN estimator (Yu et al., 2025), pinned to source revision `969ef60c30cb`, with dual parameters $(N, n, t, q) = (L_d, d, s, p)$ and corresponding primal dimension $k = L_d - d$. We run its exact-noise finite-field routines for Pooled Gauss, SD-ISD, statistical decoding (SD), and statistical decoding 2.0 (SD2). Table 4 reports their single-instance log-costs in the estimator's work-factor convention. Pooled Gauss attains the minimum at every width. The SD-ISD routine includes a fallback to Pooled Gauss, explaining the identical entries in those columns.

The supplementary scripts retain the full commit identifier, source checksum, inputs, and unrounded outputs. They invoke the underlying routines with their declared signatures: the pinned command-line wrapper passes an extra argument to `SDforq`. No cost formula is modified. Estimates cover these four routines; regular-noise routines are not substituted for the exact-weight distribution. They neither exhaust all applicable attacks nor provide memory-cost estimates or upper bounds on the optimal advantage δ_d.

Exposure accounting. For ℓ blocks with hidden width h and MLP width f, let M count token positions processed by prefill and actual decoding forwards during one master-matrix epoch. With independent masks for the seven projections,

$$Q_h = 6\ell M, \qquad Q_f = \ell M. \tag{22}$$

Q_d counts encoded rows, not RPCs: grouped Q/K/V requests still contribute three separately masked rows per token and block. All protected-service executions using this matrix, including chosen-input probes, consume the budget. Locally generated samples using public A_d and attacker-chosen masks instead contribute to adversarial computation time. If models share one master matrix, their per-width counts must be aggregated.

Heuristic budget selection. Let λ_d be the minimum cost exponent in Table 4. To relate these work factors to the transcript bound, we explicitly adopt the additional heuristic $\delta_d(\tau') \lesssim \tau' 2^{-\lambda_d}$. This relation is not established by the estimator or the reduction. Define

$$\omega = \sum_d Q_d 2^{-\lambda_d}, \qquad \lambda_{\text{eff}} = -\log_2 \omega. \tag{23}$$

Table 5: Heuristic exposure budgets for one master matrix serving the indicated model. $M_{\max}(128)$ is an estimated budget, not a proven security limit.

Model	ℓ	h/f	$\lambda_{\text{eff}}(2^{24})$	$\lambda_{\text{eff}}(2^{27})$	$M_{\max}(128)$
Llama-3.2-1B	16	2,048/8,192	131.15	128.15	148,638,905
Llama-3.1-8B	32	4,096/14,336	131.45	128.45	183,904,267
Qwen2.5-7B	28	3,584/18,944	131.33	128.33	168,492,218
Qwen3-8B	36	4,096/12,288	131.28	128.28	162,694,717

Only under this heuristic does substitution give

$$\mathrm{Adv}_{\mathrm{guess}} \lesssim \tau' 2^{-\lambda_{\mathrm{eff}}} + \varepsilon_{\mathrm{rng}}. \tag{24}$$

The runtime τ' includes reduction overhead and preprocessing on the public matrix. A cost exponent is not an advantage bound independent of adversarial runtime; these estimates do not establish a concrete system security level or bound the randomness error.

For a target heuristic exponent b_0, the corresponding token budget is

$$M_{\max}(b_0) = \left\lfloor \frac{2^{-b_0}}{6\ell\, 2^{-\lambda_h} + \ell\, 2^{-\lambda_f}} \right\rfloor. \tag{25}$$

Table 5 uses unrounded costs. The largest common power-of-two budget with estimated $\lambda_{\text{eff}} \geq 128$ is $M = 2^{27}$, but its margin is only 0.15–0.45 exponent bits. We instead propose $M = 2^{24}$, giving $\lambda_{\text{eff}} = 131.15$–$131.45$ and about three additional exponent bits relative to this heuristic target.

For example, $M = 2^{24}$ on Llama-3.2-1B gives $Q_{2048} = 1{,}610{,}612{,}736$ and $Q_{8192} = 268{,}435{,}456$. If several evaluated models share a master matrix, the same cap can be applied to their total processed token positions: the largest per-position coefficient in Equation 25 occurs for Llama-3.2-1B.

Proposed rotation policy. One prefill of I input tokens followed by $N - 1$ decoding forwards consumes $I + N - 1$ token positions. With $I = 256$ and $N = 100$, $M = 2^{24}$ permits $\lfloor 2^{24}/355 \rfloor = 47{,}259$ complete requests; the near-limit alternative 2^{27} permits 378,078. At r such requests per day, these budgets last approximately $47{,}259/r$ and $378{,}078/r$ days. Other execution schedules must count the protected rows they actually process.

Before admitting a request, the proposed policy reserves its exposure budget. If insufficient capacity remains, it completes outstanding work and activates an independently sampled master matrix with new tables $B = A_d W_q$, then starts a new counter epoch. Fresh masks remain mandatory for every invocation; resampling S or resetting counters while retaining A does not begin a new epoch. Table generation and loading must be repeated. Rotation has not been benchmarked and is not included in the reported online latency.

Rotation does not erase prior observations. For a guarantee covering multiple epochs, the hybrid bound accumulates their exposure terms. Under the same heuristic, J equally budgeted epochs reduce the combined exponent by $\log_2 J$. The proposed budget therefore concerns one matrix epoch, rather than an indefinitely long transcript.

B.4 Security Game and Transcript Privacy Proof

Game and permitted leakage. Public parameters comprise the master matrix, slice definitions, quantized weights, and derived tables. The adversary knows the model and may perform preprocessing on these parameters and make adaptive chosen-input probes. It selects two victim prompts $x^{(0)}, x^{(1)}$; a challenger draws $\gamma \leftarrow \{0, 1\}$ and executes $x^{(\gamma)}$. The adversary observes encoded computation values and outputs $\hat{\gamma}$.

Permitted leakage comprises operation identifiers and order, tensor dimensions, call counts, and observable termination. Challenge prompts must induce the same public trace for every realization of private generation randomness. Internal token sampling may change private states, but not the public schedule. Probe choices may depend on earlier observations, subject to public per-width bounds Q_d on total exposed rows.

Fixed-step inference mode. A sufficient service mode fixes input length I, generation budget N, and message-chunking policy as public parameters before execution. For the dense models evaluated here, it performs one prefill and $N-1$ decode forwards, running the same seven protected projections per block even if EOS is sampled; token selection affects only private state. Prefill chunking depends on public dimensions and buffer limits. The service releases its response after the Nth forward. Thus, equal I, N produce the same operation order, tensor shapes, RPC count, and stopping point for every sampled continuation. Observable response lengths must either be padded or included in the permitted leakage. Prompt and response contents use the secure channel; timing and side channels remain outside the analysis. With EOS-triggered early stopping or other input-dependent scheduling, the theorem applies only to challenges with identical leaked traces.

For a time bound τ, define

$$\delta_d(\tau) = \sup_{\mathcal{D}:\,\text{time}\le\tau} \big|\Pr[\mathcal{D}(A, \mathbf{e}A) = 1] - \Pr[\mathcal{D}(A, \mathbf{u}) = 1]\big|, \tag{26}$$

where A is uniform in $\mathbb{F}_p^{L_d \times d}$, $\mathbf{e} \leftarrow \mathcal{D}_{L_d,s,p}$, and $\mathbf{u}$ is independently uniform in $\mathbb{F}_p^d$. The time budget includes adversarial preprocessing after the public parameters are sampled.

Embedding a challenge in the shared matrix. For a row of width d, place a single-instance challenge matrix A in the region defining A_d and fill all remaining master entries independently and uniformly. The resulting master matrix has exactly the prescribed distribution. All other slices and public tables can be computed from it, and every other secret mask can be sampled independently. This simulates joint observations across overlapping slices without assuming that those slices are independent.

Hybrid argument. Fix γ and index potential rows by width and occurrence number, with at most Q_d slots per width. Order these slots by a fixed public rule; unused slots contribute zero to the bound. Let $\mathsf{G}_{\gamma,k}$ replace the first k row masks by independent uniform vectors. For row j, with private quantized input $\mathbf{c}_j$, the reduction embeds its challenge $\mathbf{v}$ through

$$\mathbf{x}_j = \mathbf{c}_j + \mathbf{v}, \qquad \mathbf{z}_j = \mathbf{x}_j W_{q,j}.$$

It advances private model state by directly evaluating the quantized computation, so it does not need the unknown sparse coefficients underlying $\mathbf{v}$. Adjacent hybrids differ by at most $\delta_{d_j}(\tau')$. Summing over rows gives, for either fixed challenge bit,

$$|\Pr[\mathcal{D}(\mathsf{G}_{\gamma,0}) = 1] - \Pr[\mathcal{D}(\mathsf{G}_{\gamma,\text{all}}) = 1]| \le \sum_d Q_d\, \delta_d(\tau'). \tag{27}$$

In the final hybrid, conditionally on the public parameters and prior history h_{j-1},

$$\Pr[\mathbf{x}_j = \mathbf{v} \mid A_{\text{master}}, \mathsf{h}_{j-1}] = p^{-d_j}.$$

Translation by the private input does not alter this distribution. The encoded output is determined by $\mathbf{x}_j$ and public weights. With the same challenge schedule, the final hybrids for $\gamma = 0$ and $\gamma = 1$ are identically distributed. This remains true for adaptive probes, whose choices depend on identically distributed histories.

The triangle inequality bounds the real-transcript probability gap by $2\sum_d Q_d \delta_d(\tau')$. Guessing advantage is half this gap. Adding ε_{rng}, an upper bound on the change in game success caused by replacing practical randomness with ideal sampling, proves Theorem 1. The theorem concerns value privacy under the stated leakage; it does not hide request lengths or stopping behavior. An asymptotic negligible-advantage statement additionally requires hard parameter families and polynomial exposure bounds.

B.5 Storage and Preprocessing Costs

Let $\mathcal{P}$ index protected projections, with input and output widths d_j, m_j. If slices are views of one resident master matrix, public-table storage is

$$L_{\max} d_{\max} + \sum_{j\in\mathcal{P}} L_{d_j} m_j \quad \text{field elements.} \tag{28}$$

For one invocation, sparse mask storage uses $O(bs)$ coefficient–index pairs, and data buffers use $O(b(d+m))$ field elements. Sending INT32 X and receiving INT32 Z transfers $4b(d+m)$ bytes, excluding protocol metadata. GPU storage additionally includes resident INT8 weights and the reusable FP64 workspace.

Precomputing all weight-specific tables requires $\sum_{j\in\mathcal{P}} O(L_{d_j} d_j m_j)$ field operations. Host preparation and guest loading are separate from online inference. For R requests reusing the tables, their average contribution is

$$\frac{T_{\text{prepare}} + T_{\text{load}}}{R}.$$

Amortization consequently depends on measured initialization costs and the number of requests served before the tables change.

C Evaluation Detail

C.1 Matching Encoded Observations

For each compared array, let $X_t^\star$ and $X_{t,c}$ denote aligned victim and probe observations at position t for candidate token c. Field elements use their canonical integer representatives in $\{0,\dots,p-1\}$. The matcher computes

$$D_{t,c} = \left\| \mathrm{vec}(\mathrm{int}(X_t^\star) - \mathrm{int}(X_{t,c})) \right\|_2 . \tag{29}$$

Subtraction uses INT64 arithmetic without modular reduction; the differences are then converted to FP64 for norm computation.

After evaluating the full vocabulary, the matcher selects

$$\hat{x}_t = \min \left\{ c \in \mathcal{V} : D_{t,c} = \min_{v\in\mathcal{V}} D_{t,v} \right\}. \tag{30}$$

Thus, ties in computed distances are resolved by the smallest token ID. There is no acceptance threshold or early termination upon finding a zero distance. In the StateVeil privacy experiment of Section 6.2, every candidate at position t is probed with the ground-truth prefix $x_{<t}$; the selected token is not used when testing position $t+1$. Victim and candidate masks are independent in the fresh-mask setting. Accuracy is computed over non-special target positions under this true-prefix oracle.