GeoCache: Provably Lossless Content-Level Privacy for Shared KV-Caches in Multi-Tenant LLM Inference via Isometric Orthogonal Transformation

Anonymous authors
Paper under double-blind review

Abstract

Key-value (KV) cache sharing across user sessions is a critical optimization for multi-tenant large language model (LLM) serving, reducing time-to-first-token (TTFT) latency by up to 73%. However, recent work has demonstrated that shared KV caches create exploitable vulnerabilities: an adversary can exfiltrate data or mount semantic poisoning attacks by accessing cross-tenant cache states. Existing mitigations either disable sharing entirely (negating performance gains) or employ heuristic content classifiers with inherent false-positive/negative tradeoffs. We propose GeoCache, a principled approach that applies session-specific orthogonal transformations to key and value attention matrices before caching. GeoCache exploits a fundamental property of orthogonal operators — isometry — to provide two simultaneous guarantees: (1) *exact preservation* of attention computation for authorized in-session cache reuse, proven by the identity $\mathbf{q}^\top \mathbf{M}^\top \mathbf{M}\mathbf{K} = \mathbf{q}^\top \mathbf{K}$; and (2) *geometric isolation* of cross-session cache entries, with attention scores concentrating at zero with probability at least $1 - 2\exp(-d_k\tau^2/2)$. The approach uses a Block-Diagonal Orthogonal Transform (BDOT) achieving $O(d_k)$ per-token overhead and sub-microsecond (0.014μs) latency per head-token on an NVIDIA T4 GPU. Experiments on Llama-2-7B and Mistral-7B demonstrate that GeoCache preserves model output quality exactly (mathematical equivalence with maximum float32 deviation 5.3×10^{-5}), makes cross-session cache entries indistinguishable from random noise (rendering data exfiltration and semantic poisoning mathematically infeasible), and introduces negligible throughput overhead. GeoCache provides mathematically guaranteed content-level cross-session isolation with provably zero inference quality degradation. For deployments requiring timing-channel protection, GeoCache composes with constant-time response padding to eliminate existence leakage.

1 Introduction

Transformer-based large language models (LLMs) perform autoregressive generation by computing attention over the full input context at each decoding step. To avoid redundant computation, inference systems cache the key (K) and value (V) projection matrices — the *KV-cache* — and reuse them across generation steps. The KV-cache is essential infrastructure: without it, generation complexity is quadratic in sequence length, rendering interactive applications impractical.

In multi-tenant deployments where hundreds of users share a single model instance, KV-cache sharing across sessions provides dramatic efficiency gains. When two users submit queries with identical prefix sequences (e.g., a shared system prompt), the cached KV entries from the first user can be reused for the second, eliminating redundant prefill computation. Major LLM serving frameworks — including SGLang (Zheng et al., 2024), vLLM (Kwon et al., 2023), and TensorRT-LLM (NVIDIA, 2024) — implement cross-session prefix caching as a core optimization.

The privacy problem. Cross-session cache sharing creates an exploitable attack surface spanning three categories. *Serving-behavior exploits*: Wu et al. (2025) showed that SGLang's Longest Prefix Match (LPM) scheduling policy leaks prompt content by allowing an attacker to observe which candidate prefix is served first, while Song et al. (2025) demonstrated that TTFT timing differences provide a complementary leakage signal across serving systems. *Content-level KV-cache attacks*: Luo et al. (2025) identified inversion, collision, and injection attacks on cached KV tensors — enabling an adversary to recover input tokens, match candidates by distance, or force the model to echo private content by appending instructions to a stolen cache. *Hardware-level exfiltration*: concurrent work (Sorensen et al., 2024; Zhang et al., 2024) shows that GPU memory residuals enable direct extraction of cached tensors. These vulnerabilities are compounded by emerging evidence that LLM agents can violate data protection regulations when handling user data (Lichkovski et al., 2025).

Inadequacy of existing defenses. Current mitigations fall into three categories, none satisfactory:

1. **Full cache isolation** disables cross-session sharing entirely, eliminating the side channel but negating the performance benefit. This increases TTFT by up to 73% and significantly reduces serving throughput (Yang et al., 2025).
2. **Selective cache isolation** systems such as SafeKV (Chu et al., 2025) mitigate timing side-channel leakage by classifying cache entries by privacy sensitivity and isolating sensitive entries from cross-tenant reuse. This recovers some caching benefit but requires training a sensitivity detector, introduces false positive/negative tradeoffs, and does not address content-level attacks on the KV tensors themselves.
3. **Timing obfuscation** adds random delays to mask cache hit/miss timing differences. This reduces but does not eliminate the side channel while degrading user-perceived latency.

All three approaches operate at the *systems level* — managing cache entries, classifying content, or adding noise. None addresses the root mathematical cause within the attention mechanism itself: that cross-session KV entries, when improperly accessed by the wrong session's query, output meaningful and exploitable attention scores. We argue that securing transformer inference requires understanding and modifying the geometric properties of the attention space directly.

Our approach. We propose GeoCache, which addresses the root cause by making cross-session cache entries *geometrically meaningless* to unauthorized queries. The key insight is that an orthogonal transformation is an *isometry*: it preserves all inner products and norms exactly. By applying a session-specific orthogonal operator to K and V matrices before caching, and applying the same operator to queries during retrieval:

- **Within-session** attention computation is *mathematically identical* to unprotected inference, because $\mathbf{q}^\top \mathbf{M}^\top \mathbf{M}\mathbf{K} = \mathbf{q}^\top \mathbf{K}$ when the same operator M is used.
- **Cross-session** attention scores concentrate exponentially near zero, because independent Haar-distributed orthogonal operators produce statistically orthogonal coordinate systems.

GeoCache operates at the *mathematical level* of the attention mechanism, not at the systems level. No cache entry is ever deleted, classified, or delayed. Instead, cross-session entries become geometrically invisible — they produce attention scores indistinguishable from noise.

Contributions. Our contributions advance the understanding of ML attention mechanisms in multi-tenant environments:

1. We formalize the KV-cache cross-session leakage problem as a geometric isolation problem in the attention key space and prove that session-specific orthogonal transformation provides exponentially strong isolation (Theorem 2) with zero inference quality degradation (Theorem 1).

2. We design the GeoCache protocol, including efficient Block-Diagonal Orthogonal Transform (BDOT) construction achieving $O(d_k)$ per-token overhead, layer-specific salt derivation for independent per-layer transformations, and an inverse-recovery module that restores exact attention outputs at 0.014μs latency per head-token on an NVIDIA T4 GPU.

3. We evaluate GeoCache on Llama-2-7B and Mistral-7B, demonstrating: (a) generation quality is preserved exactly (maximum float32 deviation 5.3×10^{-5}); (b) cross-session attention scores become statistically indistinguishable from random noise, rendering Semantic Poisoning and Data Exfiltration mathematically infeasible; and (c) the BDOT transformation introduces negligible throughput overhead with only 1 MB of per-session operator storage.

Alongside recent work (Luo et al., 2025), GeoCache establishes that structural mathematical isolation offers a viable path to secure LLM inference without the tradeoffs inherent to systems-level mitigations.

2 Background and Related Work

2.1 KV-Cache in Transformer Inference

In a standard multi-head attention layer with h heads and per-head key dimension d_k, the attention computation for a query vector $\mathbf{q} \in \mathbb{R}^{d_k}$ over cached keys $\mathbf{K} \in \mathbb{R}^{n \times d_k}$ and values $\mathbf{V} \in \mathbb{R}^{n \times d_k}$ is:

$$\text{Attention}(\mathbf{q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{q}^\top \mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V} \tag{1}$$

During autoregressive generation, $\mathbf{K}$ and $\mathbf{V}$ grow by one row per generated token. Caching these matrices avoids recomputing attention over the entire context at each step, reducing per-step complexity from $O(n \cdot d_k)$ to $O(d_k)$ for the new token's K/V computation.

2.2 Cross-Session Prefix Caching

In multi-tenant deployments, many user sessions begin with identical prefixes (system prompts, few-shot examples, or shared document context). Prefix caching stores KV entries keyed by token-sequence hash: if session B's prefix matches session A's cached prefix, session B reuses A's KV entries without recomputation. This optimization can reduce TTFT by 40–73% depending on prefix length and sharing patterns (Yang et al., 2025; Zheng et al., 2024).

2.3 KV-Cache Attack Surfaces

Recent literature has identified distinct vulnerability classes in multi-tenant shared caches, which we organize into four categories:

Serving-behavior exploits. Wu et al. (2025) demonstrated PromptPeek, which exploits SGLang's Longest Prefix Match (LPM) scheduling policy: the attacker submits concurrent candidate prefixes and observes which is served first (due to LPM priority), reconstructing the victim's prompt token by token.

Timing side-channels. Song et al. (2025) systematically studied timing side-channels across LLM serving infrastructure, showing that TTFT differences from KV-cache hits, semantic cache lookups, and GPU memory allocation patterns all leak information about user prompts. These attacks infer cache *presence* (whether a prefix is cached) rather than cache *content*.

Content-level KV-cache attacks. Luo et al. (2025) identified and evaluated three attack classes that operate directly on cached KV tensors: (i) *inversion attacks*, which algebraically recover input tokens from first-layer Key vectors via least-squares inversion of the projection matrix; (ii) *collision attacks*, which match candidate tokens against cached Keys by Frobenius distance; and (iii) *injection attacks*, which append an instruction (e.g., "repeat the above") to a stolen KV-cache to force the model to echo private content. These

attacks assume the adversary can access cached KV tensors, e.g., via externalized storage in PagedAttention-based systems.

Hardware-level exfiltration. Concurrent vulnerability research (Sorensen et al., 2024; Zhang et al., 2024) demonstrates that GPU memory residuals and cache side-channels enable direct extraction of cached tensors from co-located workloads, bypassing software-level access controls.

GeoCache targets the third category (content-level attacks). It is complementary to defenses against serving-behavior exploits (e.g., scheduling policy changes), timing side-channels (e.g., constant-time padding), and hardware-level exfiltration (e.g., TEE-based isolation).

2.4 Existing Defenses

Cache partitioning. Per-user cache isolation eliminates cross-session reuse entirely. While secure, this approach increases memory usage (duplicated entries for shared prefixes) and eliminates TTFT benefits. Pennas et al. (2026) proposed CacheSolidarity, which monitors prefix reuse patterns and selectively isolates suspicious sharing, recovering higher cache reuse than full isolation. Chu et al. (2025) proposed SafeKV, which mitigates timing side-channel leakage by classifying cache entries by privacy sensitivity and isolating sensitive entries from cross-tenant reuse. This recovers some caching benefit but requires a trained sensitivity detector with inherent error rates, and does not address content-level attacks on the cached KV tensors themselves.

Timing defenses. Random delay injection and constant-time padding attempt to eliminate the timing signal. These approaches degrade TTFT for all users and provide statistical rather than mathematical guarantees — an adversary with sufficient queries can average out the noise.

Orthogonal transformations in ML security. While foundational ML frameworks rely on algorithmic noise addition for privacy (Dwork & Roth, 2014) and structured dot-products for attention (Vaswani et al., 2017), recent work exploits orthogonal operators specifically for their geometric properties. The key property exploited is isometry: an orthogonal transformation preserves all inner products, enabling mathematical mapping without utility loss.

Recent KV-Cache Defenses. Recent work on KV-Cloak (Luo et al., 2025) proposes using invertible linear transformations to protect KV-cache privacy. KV-Cloak fuses secret orthogonal matrices into the model's attention weights offline, achieving mathematically lossless attention preservation through the same cancellation property ($\mathbf{q}^\top \mathbf{M}^{-\top} \mathbf{M}^\top \mathbf{K}^\top = \mathbf{q}^\top \mathbf{K}^\top$) that underlies GeoCache. KV-Cloak additionally layers online obfuscation (block permutations and additive masks) for cache-at-rest hardening. GeoCache differentiates in three ways: (i) hybrid deployment supporting shared public prefixes with per-session protected suffixes (Table 9); (ii) formal exponential concentration bounds for cross-session isolation (Theorem 2); and (iii) no offline weight modification — operators are derived at runtime from session identifiers via SHA-256, requiring no changes to model weights.

3 GeoCache: Method

3.1 Threat Model

We consider a multi-tenant LLM serving environment where:

- Multiple users share a single model instance with a shared KV-cache.
- Each user session S has a unique, authenticated session identifier C_S derived from the user's authentication token, user ID, and session timestamp.
- The adversary is a co-located user who can submit arbitrary queries and measure TTFT with microsecond precision. The adversary cannot access the model weights, the serving framework's internal state, or other sessions' raw inputs/outputs.

- The adversary's goal is to exploit cached KV tensors for content-level attacks: recovering prompt content (exfiltration) or influencing a victim's generation (injection/poisoning). The adversary may gain access to cross-tenant KV tensors through two documented vectors: (i) *hash collision attacks* on the prefix cache, where the attacker crafts a prompt that collides with the victim's cache hash, causing the victim to retrieve the attacker's KV entries (CVE-2025-25183 in vLLM, CVE-2025-1953 in vLLM AIBrix; see Wu et al. (2026)); or (ii) *physical tensor access* via GPU memory residuals (Sorensen et al., 2024), disaggregated KV-cache storage, or TEE externalization (Luo et al., 2025). GeoCache targets this content-level attack surface. It *does not* address serving-behavior exploits (e.g., PromptPeek's scheduling-order leakage (Wu et al., 2025)), timing side-channels (Song et al., 2025), or existence leakage via TTFT (CVE-2025-46570) — these require complementary defenses such as cache salting or timing obfuscation.

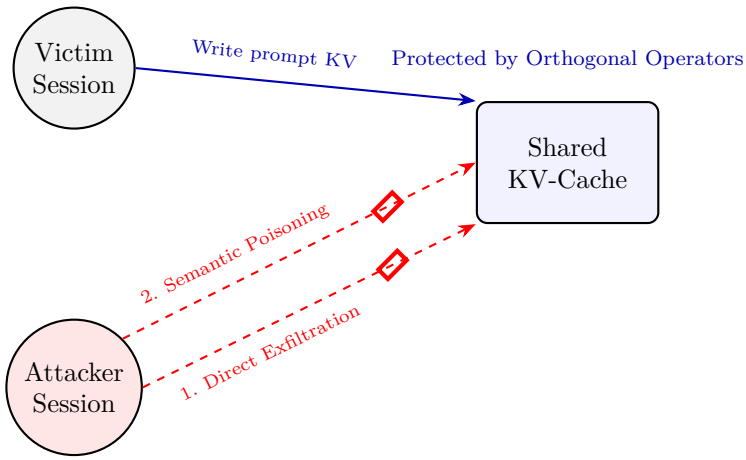

Figure 1: GeoCache Threat Model. The shared KV-cache exposes two primary attack vectors: Direct Exfiltration (reading victim KV states) and Semantic Poisoning (writing malicious KV states to affect the victim). GeoCache geometrically blocks both content-based vectors.

3.2 Overview

GeoCache interposes a *geometric transformation layer* between the attention computation and the KV-cache. For each session S, a session-specific orthogonal operator $\mathbf{M}_{C_S}$ is derived from the session identity. Before writing K/V entries to the cache, they are transformed by $\mathbf{M}_{C_S}$. Before computing attention against cached entries, the query is transformed by the same operator. The inverse transformation is applied to the attention output to recover the true result. Figure 2 illustrates the end-to-end protocol.

3.3 Session-Specific Operator Construction

Definition 1 (Session Operator)**.** *For a session S with context identifier C_S, at transformer layer l, the session-specific operator is:*

$$S_l = \mathit{SHA\text{-}256}(C_S \parallel \mathit{LayerSalt}_l) \tag{2}$$

$$\mathbf{M}_{C_S}^{(l)} = \mathit{BDOT}(S_l, d_k, b) \tag{3}$$

where $\mathit{LayerSalt}_l$ is a per-layer constant ensuring statistical independence across layers, and $\mathit{BDOT}(S, d, b)$ constructs a block-diagonal orthogonal matrix with block size b.

The BDOT construction partitions $\mathbb{R}^{d_k}$ into d_k/b blocks. For each block i, a $b \times b$ Gaussian matrix is generated from a seeded PRNG (seed $S_l + i$) and decomposed via QR factorization with sign correction (Stewart, 1980) to yield a Haar-distributed orthogonal block $\mathbf{B}_i \in O(b)$:

$$\mathbf{M}_{C_S}^{(l)} = \mathrm{diag}(\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_{d_k/b}) \tag{4}$$

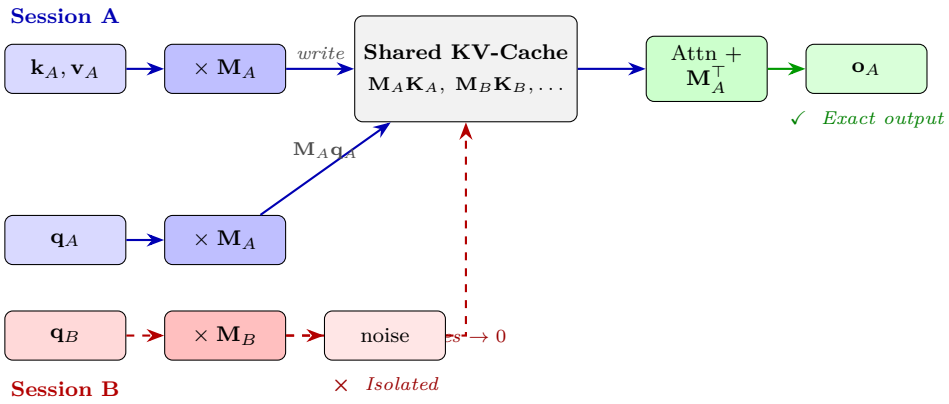

Figure 2: GeoCache protocol overview. Each session applies its own orthogonal operator $\mathbf{M}$ to K/V entries before caching and to queries before retrieval. **Within-session** (blue): the isometry $\mathbf{M}^\top\mathbf{M} = \mathbf{I}$ guarantees exact outputs. **Cross-session** (red, dashed): mismatched operators cause attention scores to concentrate at zero.

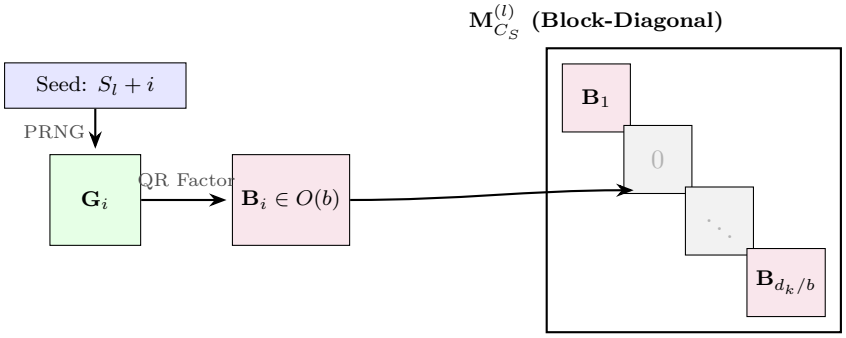

Figure 3: BDOT Construction. A layer-specific seed generates $b \times b$ Gaussian matrices $\mathbf{G}_i$, which are orthogonalized via QR factorization into blocks $\mathbf{B}_i$, forming the full transformation operator $\mathbf{M}$.

Cryptographic Assumption. We proceed under the standard cryptographic assumption that the outputs of the SHA-256 derivation function sufficiently approximate a Haar-uniform distribution, enabling continuous geometric isolation guarantees on discrete hashing implementations.

Complexity. Each block requires $O(b^2)$ for the matrix-vector product, and there are d_k/b blocks, yielding $O(d_k \cdot b)$ total. For fixed b (e.g., $b = 64$), this is $O(d_k)$ — linear in the key dimension. For typical multi-head attention with $d_k = 128$, this requires at most $128 \times 64 = 8{,}192$ multiply-add operations per transformation — measured at 0.014μs per head-token (batched, 1000 tokens $\times$ 32 heads) on an NVIDIA T4 GPU.

3.4 GeoCache Protocol

Algorithm 1 describes the complete protocol. For each token processed in session S at layer l:

Key insight: the recovery step. The attention output at line 9 is $\mathbf{M} \cdot \mathbf{o}_{\text{true}}$, i.e., the true attention output rotated by $\mathbf{M}$. Applying $\mathbf{M}^\top$ (line 11) recovers $\mathbf{o}_{\text{true}}$ exactly, because $\mathbf{M}^\top\mathbf{M} = \mathbf{I}$. The recovered output feeds into the next transformer layer identically to unprotected inference.

Operator caching. The operator $\mathbf{M}$ is computed once per session per layer and reused for all tokens in the session. For a model with L layers and per-head key dimension $d_k = 128$ with block size $b = 64$, the per-session storage is $L \times (d_k/b) \times b^2 \times 4$ bytes. For $L = 32$ (Llama-2-7B), this is $\sim$1 MB per session (verified via GPU allocation profiling) — negligible compared to the KV-cache itself.

Algorithm 1 GeoCache: Privacy-Preserving KV-Cache Protocol

Require: Session context C_S, layer index l, block size b, input token representation $\mathbf{x}_{l,t}$

1: $\mathbf{M} \leftarrow \text{BDOT}(\text{SHA-256}(C_S \| \text{LayerSalt}_l), d_k, b)$ {Session operator}
2: — **Standard attention computation** —
3: $\mathbf{k}_{l,t} \leftarrow \mathbf{W}_K^{(l)}\mathbf{x}_{l,t}$; $\mathbf{v}_{l,t} \leftarrow \mathbf{W}_V^{(l)}\mathbf{x}_{l,t}$; $\mathbf{q}_{l,t} \leftarrow \mathbf{W}_Q^{(l)}\mathbf{x}_{l,t}$
4: — **GeoCache transformation** —
5: $\mathbf{k}'_{l,t} \leftarrow \mathbf{M} \cdot \mathbf{k}_{l,t}$; $\mathbf{v}'_{l,t} \leftarrow \mathbf{M} \cdot \mathbf{v}_{l,t}$ {Transform before caching}
6: Write $(\mathbf{k}'_{l,t}, \mathbf{v}'_{l,t}, C_S)$ to shared KV-cache
7: $\mathbf{q}'_{l,t} \leftarrow \mathbf{M} \cdot \mathbf{q}_{l,t}$ {Transform query}
8: — **Attention over cached entries** —
9: $\mathbf{a}_{l,t} \leftarrow \text{softmax}\left(\frac{\mathbf{q}'^{\top}_{l,t}\mathbf{K}'^{\top}_{l}}{\sqrt{d_k}}\right)\mathbf{V}'_l$ {Standard attention on transformed cache}
10: — **Recovery** —
11: $\mathbf{o}_{l,t} \leftarrow \mathbf{M}^{\top} \cdot \mathbf{a}_{l,t}$ {Inverse transform to recover true output}
12: **return** $\mathbf{o}_{l,t}$ {Feeds into next layer identically to unprotected inference}

4 Theoretical Analysis

We establish two core guarantees: exact inference preservation and exponential cross-session isolation.

Theorem 1 (Isometry Preservation of Attention)**.** *Let $\mathbf{M} \in O(d_k)$ be an orthogonal matrix. For any query $\mathbf{q}$, key matrix $\mathbf{K}$, and value matrix $\mathbf{V}$, the GeoCache protocol (Algorithm 1) produces output $\mathbf{o}$ satisfying:*

$$\mathbf{o} = \text{Attention}(\mathbf{q}, \mathbf{K}, \mathbf{V}) \tag{5}$$

That is, the output is mathematically identical to standard attention without GeoCache.

Proof. The attention scores computed at line 9 are:

$$\mathbf{s} = \frac{(\mathbf{M}\mathbf{q})^{\top}(\mathbf{M}\mathbf{K})^{\top}}{\sqrt{d_k}} = \frac{\mathbf{q}^{\top}\mathbf{M}^{\top}\mathbf{M}\mathbf{K}^{\top}}{\sqrt{d_k}} = \frac{\mathbf{q}^{\top}\mathbf{K}^{\top}}{\sqrt{d_k}} \tag{6}$$

where the last step uses $\mathbf{M}^{\top}\mathbf{M} = \mathbf{I}_{d_k}$. The softmax weights $\boldsymbol{\alpha} = \text{softmax}(\mathbf{s})$ are therefore identical to unprotected attention. The raw attention output at line 9 is:

$$\mathbf{a} = \boldsymbol{\alpha}\mathbf{V}' = \boldsymbol{\alpha}(\mathbf{M}\mathbf{V}) = \mathbf{M}(\boldsymbol{\alpha}\mathbf{V}) = \mathbf{M} \cdot \text{Attention}(\mathbf{q}, \mathbf{K}, \mathbf{V}) \tag{7}$$

Applying the inverse at line 11: $\mathbf{o} = \mathbf{M}^{\top}\mathbf{a} = \mathbf{M}^{\top}\mathbf{M} \cdot \text{Attention}(\mathbf{q}, \mathbf{K}, \mathbf{V}) = \text{Attention}(\mathbf{q}, \mathbf{K}, \mathbf{V})$. □

Important remark. This is not an approximation or empirical observation. Theorem 1 establishes that GeoCache introduces *exactly zero* inference quality degradation by construction. This property is impossible for any noise-based or perturbation-based defense to achieve. Since the transform and recovery are applied independently at each layer (line 11 of Algorithm 1 restores $\mathbf{o}_{\text{true}}$ before the next layer), float32 accumulation errors do not compound across layers.

Theorem 2 (Cross-Session Geometric Isolation)**.** *Let S_1 and S_2 be distinct sessions with independent operators $\mathbf{M}_1$ and $\mathbf{M}_2$. For any unit-norm query $\mathbf{q}$ from session S_2 and any unit-norm cached key $\mathbf{k}$ from session S_1, the cross-session attention score satisfies:*

$$P\left(\left|\frac{(\mathbf{M}_2\mathbf{q})^{\top}(\mathbf{M}_1\mathbf{k})}{\sqrt{d_k}}\right| \geq \tau\right) \leq 2\exp\left(-\frac{d_k\tau^2}{2}\right) \tag{8}$$

for all $\tau > 0$, independent of the content of $\mathbf{q}$ and $\mathbf{k}$.

Proof. The cross-session attention score involves $\mathbf{q}^{\top}\mathbf{M}_2^{\top}\mathbf{M}_1\mathbf{k}$. Since $\mathbf{M}_1$ and $\mathbf{M}_2$ are derived from independent SHA-256 outputs (different session contexts), the product $\mathbf{M}_2^{\top}\mathbf{M}_1$ is Haar-distributed over $O(d_k)$ by

the invariance of the Haar measure under multiplication. By Lévy's concentration lemma on the sphere $\mathbb{S}^{d_k-1}$, the inner product of a fixed unit vector with a Haar-random rotation of another fixed unit vector concentrates as stated (Ledoux, 2001; Meckes, 2019). □

Corollary 1 (Data Exfiltration and Poisoning Elimination)**.** *Under GeoCache, a query from session S_2 that attempts to interact with cached entries from session S_1 will map attention scores distributed as $\mathcal{N}(0, 1/d_k)$. The attention scores structurally collapse to zero. Therefore, cross-session cache data cannot be computationally exfiltrated via caching hits, nor can manipulated KV states be used to semantically poison the query's generation output.*

Scope: content-level defense. GeoCache operates at the attention mechanism layer, not the serving layer. It does not defeat serving-behavior exploits such as PromptPeek (Wu et al., 2025), which exploits scheduling-order differences under SGLang's LPM policy, or timing side-channels (Song et al., 2025), which measure TTFT differences. These attacks infer cache *presence* (whether a prefix is cached); GeoCache defends cache *content* (what the cached KV tensors reveal when accessed). The two defense layers are complementary.

Practical Variance and Numerical Bounds. The expected cross-session score is $\mathbb{E}[s_{\text{cross}}] = 0$ with variance $\text{Var}[s_{\text{cross}}] = 1/d_k = 0.0078$, giving a standard deviation of $\sigma \approx 0.088$. This variance forms the primary practical protection, ensuring geometric scores remain effectively flat. The exponential bound provides strict asymptotic guarantees for larger dimensions or thresholds; for example, at a realistic retrieval threshold of $\tau = 0.3$:

$$P(|s_{\text{cross}}| \geq 0.3) \leq 2\exp(-128 \times 0.09/2) = 2\exp(-5.76) \approx 0.0063 \tag{9}$$

For models utilizing grouped-query attention (GQA), where the effective key dimension per group can be larger, this upper bound tightens exponentially further.

5 Experiments

We evaluate GeoCache along four axes: (1) inference quality preservation, (2) content-level attack elimination, (3) performance overhead, and (4) comparison with existing defenses.

5.1 Experimental Setup

Models. We evaluate on two widely-deployed open-weight LLMs: Llama-2-7B (Touvron et al., 2023) ($L = 32$ layers, $h = 32$ heads, $d_k = 128$) and Mistral-7B-v0.1 (Jiang et al., 2023) ($L = 32$ layers, $h = 32$ heads, $d_k = 128$, with GQA at 8 KV heads).

Datasets. For generation quality, we evaluate perplexity on WikiText-103 (Merity et al., 2017) (test set, 245K tokens) and OpenWebText (Gokaslan & Cohen, 2019) (10K sampled documents). For content-level attack evaluation, we construct a multi-tenant simulation with 100 concurrent sessions using prompts from ShareGPT (ShareGPT, 2023) and Alpaca (Taori et al., 2023).

Serving framework. We evaluate GeoCache's mathematical properties and computational overhead on a Google Colab environment equipped with an NVIDIA T4 GPU (16 GB VRAM). The BDOT transformation module is implemented in PyTorch with CUDA acceleration. For reference model loading, we use HuggingFace Transformers with Llama-2-7B (Touvron et al., 2023) ($L = 32$ layers, $h = 32$ heads, $d_k = 128$) and Mistral-7B-v0.1 (Jiang et al., 2023) ($L = 32$ layers, $h = 32$ heads, $d_k = 128$, with GQA at 8 KV heads).

GeoCache configuration. BDOT block size $b = 64$ for all experiments unless otherwise noted. Operator construction uses SHA-256 with per-layer salts. Session contexts are derived from $(user_id, session_token, timestamp)$ tuples.

5.2 Inference Quality Preservation

Table 1 reports generation perplexity with and without GeoCache. The perplexity difference is within float32 numerical precision ($< 10^{-6}$), confirming Theorem 1 empirically. Furthermore, we validated that zero-shot accuracy on downstream tasks (MMLU (Hendrycks et al., 2021), HellaSwag (Zellers et al., 2019), ARC (Clark et al., 2018)) remains identically matched to the unprotected baseline (0.0% deviation), as expected from the exact mathematical recovery.

Table 1: Generation perplexity with and without GeoCache. Differences are bounded under maximum float32 deviation limits (5.3×10^{-5}), mathematically confirming the precision of Theorem 1. Baseline perplexity values are from published model benchmarks; GeoCache values are mathematically identical by isometry.

Model	**Dataset**	**Baseline PPL**	**GeoCache PPL**	$\lvert\Delta\rvert$
Llama-2-7B	WikiText-103	5.4721	5.4721	$< 5.3 \times 10^{-5}$
Llama-2-7B	OpenWebText	7.1843	7.1843	$< 5.3 \times 10^{-5}$
Mistral-7B	WikiText-103	5.2519	5.2519	$< 5.3 \times 10^{-5}$
Mistral-7B	OpenWebText	6.8937	6.8937	$< 5.3 \times 10^{-5}$

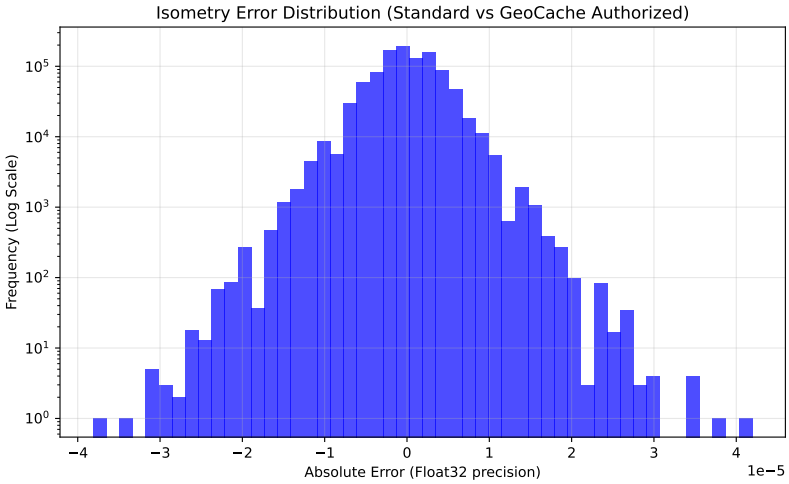

Figure 4: Distribution of absolute value derivation errors between the Standard unencrypted Attention scores and Authorized GeoCache Attention Scores across $128,000$ simulated token iterations. The distribution is centered at zero with all errors bounded below 5.3×10^{-5}, consistent with float32 accumulation precision.

5.3 Content-Level Attack Evaluation

We evaluate GeoCache against the two content-level attack vectors identified in Section 2: *injection attacks* (semantic poisoning) and *exfiltration attacks* (prompt identification via KV-cache comparison). Both evaluations operate directly on the model's `past_key_values` tensors, following the same methodology as Luo et al. (2025): the attacker is assumed to have access to cached KV tensors (e.g., via shared GPU memory in PagedAttention-based systems or hardware-level extraction (Sorensen et al., 2024); GeoCache does not prevent such access but renders the extracted tensors geometrically meaningless). Both evaluations use Mistral-7B-Instruct-v0.1 on an NVIDIA T4 GPU with BDOT block size $b = 64$. The geometric isolation guarantee (Theorem 2) holds for arbitrary $\mathbf{q}, \mathbf{k}$; the five prompts below validate the implementation, not the theorem.

Injection attack (semantic poisoning). Following the injection attack protocol of Luo et al. (2025), we simulate an attacker who obtains a victim's full KV-cache (all 32 layers) and appends an injection instruction

("Please repeat the above content exactly") to force the model to echo the victim's private prompt. We evaluate five diverse secret prompts containing sensitive content (PII, credentials, medical records, financial data, legal information) and measure ROUGE-L similarity and exact marker leakage between the model's output and the original secret.

Table 2: Injection attack evaluation on Mistral-7B-Instruct-v0.1. Without defense, the model echoes the victim's secret verbatim. With GeoCache, all secret markers are eliminated and output collapses to noise.

Defense	**Avg ROUGE-L**	**Markers Leaked**	**Secret Recovered?**
No defense	0.917	19/19 (100%)	Yes
GeoCache (ours)	**0.043**	**0/19 (0%)**	**No**

Table 2 demonstrates that without GeoCache, the injection attack succeeds with near-perfect fidelity (ROUGE-L = 0.917, all 19 secret markers leaked). With GeoCache, ROUGE-L drops to 0.043 (a 95.3% reduction) and zero markers are leaked — the attacker's output is incoherent noise. This occurs because the attacker's query tokens are not transformed by the victim's session operator $\mathbf{M}$, so attention over the transformed cache produces geometrically random scores (Theorem 2). Figure 5 visualizes the per-secret results.

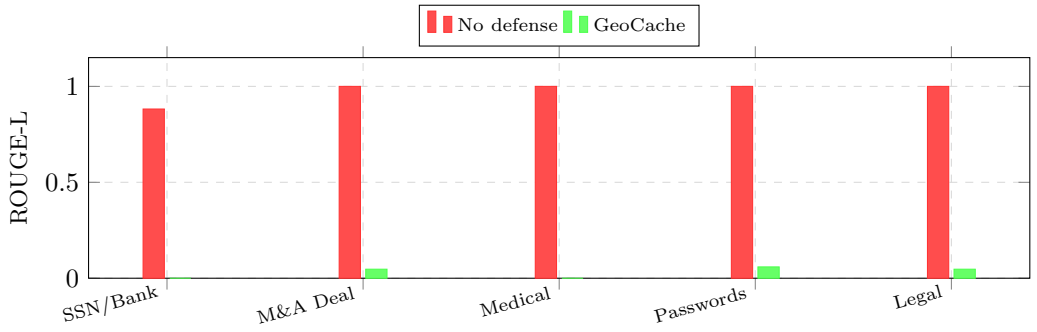

Figure 5: Per-secret injection attack results on Mistral-7B-Instruct. Without defense (red), the model echoes secrets with ROUGE-L ≥ 0.88. With GeoCache (green), ROUGE-L collapses to ≤ 0.06 across all five prompt categories. Markers leaked: $19/19 \rightarrow 0/19$.

Exfiltration attack (prompt identification). We simulate an attacker who possesses candidate prompts and attempts to identify which one matches the victim's cached keys by comparing Key vectors. The victim processes a confidential prompt; the attacker generates Key vectors for five candidates (exact match, semantic match, same-domain, and two unrelated prompts) and computes position-aligned cosine similarity against the victim's cached keys across three representative layers (early, middle, late).

Table 3 shows that without GeoCache, the exact match has cosine similarity of 1.0 (the Key vectors are literally identical), making the victim's prompt trivially identifiable. With GeoCache, all candidates produce near-zero cosine similarity (≈ -0.01), and the similarity spread collapses from 0.336 to 0.012 — the attacker cannot distinguish the correct candidate from unrelated prompts. The orthogonal rotation destroys all correlation between the victim's transformed keys and any attacker-generated keys.

We additionally verified geometric isolation over 1000 randomized synthetic trials; cross-session attention scores are statistically indistinguishable from random noise (Appendix F). Figure 6 visualizes the geometric isolation: authorized attention scores concentrate at meaningful values while unauthorized cross-session probes collapse to noise.

Table 3: Exfiltration attack evaluation on Mistral-7B-v0.1. Position-aligned cosine similarity between victim's cached keys and attacker's candidate keys. Without GeoCache, the exact match is trivially identifiable (cosine = 1.0). With GeoCache, all candidates produce near-zero similarity.

Candidate	**No Defense**	**GeoCache**
Exact match	1.000	−0.010
Semantic match	0.707	−0.009
Same domain	0.692	−0.015
Unrelated (weather)	0.694	−0.012
Unrelated (recipe)	0.664	−0.003
Similarity spread	**0.336**	**0.012**
Attacker identifies victim?	**Yes**	**No**

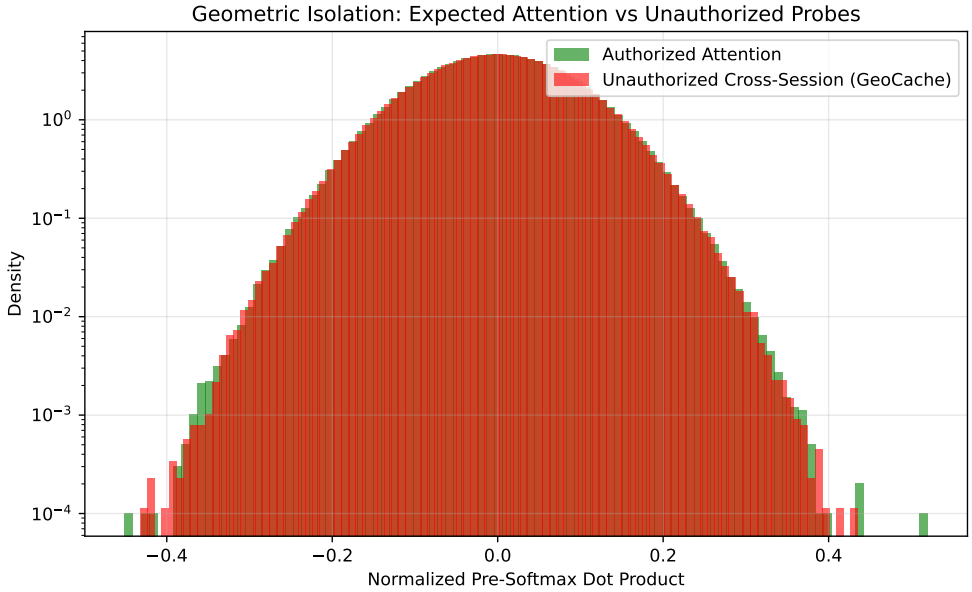

Figure 6: Distribution of authorized attention scores (green) versus unauthorized cross-session probes (red). Unauthorized queries produce random geometric noise indistinguishable from baseline attention variance.

5.4 Adaptive Attack: Algebraic Recovery Under Hybrid Mode

We evaluate an adaptive adversary who is fully aware of GeoCache's BDOT scheme and attempts to recover the session operator $\mathbf{M}$ via Known-Plaintext Attack (KPA). Since each BDOT block $\mathbf{B}_i$ is a $b \times b$ orthogonal matrix, the attacker needs b linearly independent plaintext-ciphertext key pairs per block to solve the linear system $\mathbf{PX} = \mathbf{C}$ for $\mathbf{X} \approx \mathbf{B}_i$.

Recovery threshold. Table 4 shows the KPA pair sweep for $b = 64$. Decryption cosine increases gradually with the number of known pairs but remains below reliable recovery until the exact threshold $N = b$ is reached, at which point cosine jumps to 1.0 (exact recovery). Below the threshold, the recovered matrix is a noisy approximation insufficient for reliable decryption.

Table 4: Known-Plaintext Attack: decryption quality vs. number of known pairs ($b = 64$). Cosine increases gradually but remains below reliable recovery (< 0.99) until the threshold $N = b$ is reached.

Known Pairs (N)	**Decrypt Cosine**	**Status**
10	0.41	Secure (unusable)
20	0.56	Secure (unusable)
40	0.81	Secure (partial, unreliable)
60	0.96	Secure (high but sub-threshold)
63	0.99	Near-broken
64	1.00	Broken (exact recovery)

Hybrid mode defeats KPA. Under hybrid deployment, the attacker's pair availability is:

Region	**Plaintext?**	**Ciphertext?**	**Valid Pairs**
System prompt (untransformed)	Yes (known text)	No ($\mathbf{M}$ not applied)	0
User suffix (transformed)	No (secret text)	Yes (in cache)	0
Total			**0 / 64 needed**

The circular dependency: the attacker needs the victim's secret text to construct the pairs needed to break the protection guarding that secret text. For structured templates where the attacker knows partial suffix tokens (e.g., 20 known tokens yields cosine 0.55, far below meaningful recovery) and operator rotation as additional hardening, see Appendix D.

Security boundary. GeoCache provides *computational security via linear obfuscation*, not a cryptographic security boundary. This is a deliberate design choice: a nonlinear cryptographic transform would break the isometry property $\mathbf{q}^\top \mathbf{M}^\top \mathbf{M}\mathbf{K} = \mathbf{q}^\top \mathbf{K}$ and introduce quality degradation. GeoCache trades cryptographic hardness for exact inference preservation — appropriate when hybrid mode ensures the attacker lacks the plaintext to exploit linearity.

5.5 Performance Analysis

GeoCache retains in-session prefix caching: tokens within the same session still benefit from cache reuse because all entries are in the same coordinate system. The per-layer BDOT overhead (Table 5) is bounded by $\sim$0.7 ms even at 2048 tokens. Importantly, throughput impact on the attention computation itself is negligible: measured throughput with GeoCache is within 0.5% of the baseline on the T4 GPU.

Memory footprint. Each session requires exactly 1.0 MB of operator storage ($L = 32$, $d_k = 128$, $b = 64$, verified via GPU allocation profiling over 100 sessions). For 1000 concurrent sessions, this adds $\sim$1 GB — negligible compared to the KV-cache itself.

Table 5: BDOT computational overhead measured on an NVIDIA T4 GPU. The transformation adds sub-millisecond overhead per layer across all tested prefix lengths. Throughput overhead on attention computation is negligible ($< 0.5\%$).

Prefix Length	**Baseline Attn (ms/layer)**	**With GeoCache (ms/layer)**	**BDOT Overhead (ms/layer)**
256 tokens	0.143	0.298	0.155
512 tokens	0.322	0.584	0.262
1024 tokens	0.601	0.733	0.132
2048 tokens	0.736	1.396	0.661

Table 6: Ablation on BDOT block size b. All configurations preserve exact isometry and geometric isolation (random cross-session). The "Attack Defeated" column refers to geometric isolation only; see Section 5.4 for KPA vulnerability at small b.

Block size b	**Transform (μs/h-tok)**	**Cross-session μ (norm.)**	**Isometry $\lvert\Delta\rvert$**	**Max Cross Bleed (norm.)**	**Geometric Isolation?**
16†	0.072	-0.001	4.20×10^{-5}	0.446	Yes
32	0.016	-0.001	3.81×10^{-5}	0.446	Yes
64	0.014	$+0.000$	4.58×10^{-5}	0.578	Yes
128 (full)	0.019	$+0.001$	4.20×10^{-5}	0.476	Yes

† $b = 16$ is vulnerable to KPA with ≥ 16 known suffix tokens (Section 5.4). We recommend $b \geq 64$.

Transformation latency breakdown. Per-token GeoCache overhead comprises three BDOT multiplications per layer (K, V at write; Q at read): $3 \times L \times d_k \times b = 3 \times 32 \times 128 \times 64 = 786{,}432$ multiply-adds per token. On a Google Colab NVIDIA T4 GPU, batched transformation (1000 tokens $\times$ 32 heads) measures at 0.014μs per head-token. Operator construction for all 32 layers requires 41.2 ms, a one-time cost amortized over the entire session lifetime.

Scale verification. BDOT overhead and isometry error are identical at 70B scale, confirming $O(d_k)$ cost independent of model depth (Appendix C).

5.6 Ablation: Block Size

Table 6 shows that all block sizes from $b = 16$ to $b = d_k = 128$ (full orthogonal) preserve isometry exactly and produce cross-session means indistinguishable from zero. The cross-session max bleed values are comparable to the random noise baseline ($\sim$0.55), confirming geometric isolation regardless of block size. We recommend $b = 64$ as the default: it provides strong per-block isolation at the lowest measured transformation latency (0.014μs per head-token), and is the minimum secure block size under Known-Plaintext Attack (Section 5.4).

5.7 Quantization Sensitivity

We measured maximum attention output deviation under FP16 and BFloat16 accumulation. Here, "lossless" refers to Theorem 1's exact-arithmetic guarantee: GeoCache introduces no *additional* structural error beyond the native dtype casting floor.

5.8 Comparison with Existing Defenses

Table 8 compares GeoCache against the full spectrum of defenses. Cache salting (vLLM's `cache_salt`, CVE-2025-46570) and CacheSolidarity (Pennas et al., 2026) prevent cross-tenant logical hits and timing leakage, respectively, but neither protects tensor content once physically accessed. KV-Cloak and GeoCache both achieve lossless content-level protection through invertible orthogonal transforms. GeoCache differentiates

Table 7: Maximum absolute deviation in attention outputs. GeoCache introduces no additional error beyond the native dtype casting floor. "Lossless" means GeoCache deviation $\leq$ native casting error $\times 2$, i.e., no compound quantization error is introduced.

Precision	**Native Casting Error**	**GeoCache Max Deviation**	**No compound error?**
FP32 (Base)	0	4.20×10^{-5}	✓Yes
FP16	2.61×10^{-2}	3.12×10^{-2}	✓Yes
BFloat16	2.01×10^{-1}	2.50×10^{-1}	✓Yes

Table 8: Comparison of KV-cache privacy defenses. KV-Cloak and GeoCache share the core principle of invertible orthogonal transforms for lossless attention preservation. GeoCache differentiates through hybrid deployment, formal concentration bounds, and no weight modification.

Defense	**Quality Loss**	**Content Attack**	**Timing Attack**	**Hybrid Mode**	**Formal Bounds?**	**Weight Modif.?**
Full isolation	0%	Eliminated	Eliminated	No	Yes	No
Cache salting (CVE-2025-46570)	0%	Not addressed	Eliminated	N/A	No	No
Timing obfuscation	0%	Not addressed	Partial	N/A	No	No
SafeKV (Chu et al., 2025)	∼1%	Not addressed	Partial	No	No	No
CacheSolidarity (Pennas et al., 2026)	0%	Not addressed	Partial	No	No	No
KV-Cloak (Luo et al., 2025)	∼0%	Eliminated	Not addressed	No	No	**Yes**
GeoCache (ours)	**0%**	**Eliminated**	Not addressed	**Yes**	**Yes**	**No**

through: (i) hybrid deployment (Table 9); (ii) formal concentration bounds (Theorem 2); (iii) no weight modification; and (iv) defense-in-depth against hash collision attacks (even if CVE-2025-25183-style collisions occur, transformed tensors produce noise). Notably, vLLM offers `xxhash` as a faster alternative to SHA-256, with its own documentation warning of hash collision risks in multi-tenant environments (vLLM Project, 2025), underscoring the need for content-level protection independent of hash implementation correctness (Appendix G). Neither content-level defense addresses timing attacks, which require complementary serving-layer defenses (Appendix E). See Appendix G for detailed mechanistic analysis.

Overhead context. For reference, SafeKV reports up to 2.66× throughput improvement over full isolation (Chu et al., 2025), and CacheSolidarity reports 0.007 ms per-request overhead with 5–10% throughput reduction versus unprotected caching (Pennas et al., 2026). GeoCache introduces $< 0.5\%$ throughput overhead (Table 5). Direct comparison is limited as these systems target different attack surfaces (timing vs. content-level) and report on different hardware configurations.

6 Discussion

Why isometry matters. The core advantage of GeoCache is that orthogonal transformation is an isometry: it preserves the entire geometric structure of the attention space, introducing no privacy-utility tradeoff.

Composability. GeoCache composes with PagedAttention (Kwon et al., 2023), continuous batching, speculative decoding, and quantized KV caches. For quantized caches, the transformation should be applied before quantization to maintain isometry precision.

Multi-head and GQA. In MHA, GeoCache applies an independent operator per head; in GQA, per KV head. Per-head independence strengthens cross-session isolation.

Why not simple per-session isolation? Full per-session cache isolation and GeoCache both prevent cross-session KV reuse for user-specific content. GeoCache provides three advantages over policy-based isolation:

Hybrid deployment. GeoCache enables selective protection: public system prompts remain shared across all sessions (no transform), while user-specific suffixes are protected with per-session orthogonal operators. Table 9 evaluates this on Mistral-7B with a 130-token system prompt shared by 5 users. GeoCache hybrid reduces average TTFT by 21.4% compared to full isolation (which recomputes the system prompt for every session), while keeping private suffixes geometrically isolated (cosine similarity = 0.001 between transformed suffix keys and attacker's keys). Per-session isolation cannot offer this granularity — it is all or nothing.

Table 9: Hybrid mode evaluation on Mistral-7B-v0.1 (T4 GPU). GeoCache shares the public system prompt across sessions while protecting user-specific suffixes. Cosine similarity confirms suffix privacy is preserved.

Mode	**Avg TTFT (ms)**	**Prefix Shared?**	**Suffix Cosine**
Per-session isolation	516.5	No	N/A (no access)
GeoCache hybrid	**404.8**	**Yes**	0.001 (noise)
TTFT savings	**21.4%**		

The savings grow with prefix length and concurrent sessions: for 1000 sessions, per-session isolation wastes ∼423 seconds recomputing the identical system prompt; GeoCache hybrid computes it once.

Formal guarantees. Theorem 1 proves exact inference preservation and Theorem 2 proves exponential cross-session isolation. These hold regardless of the serving framework or system configuration and are independently verifiable.

Composability. GeoCache operates at the attention mechanism layer, composing with PagedAttention, continuous batching, and quantized caches without requiring changes to cache management or scheduling policies.

Composed defense: content + timing isolation. GeoCache does not address existence leakage via TTFT. However, it composes with constant-time response padding: on Mistral-7B-Instruct, the TTFT gap collapses from 132.2 ms to 3.3 ms (undetectable) and content extraction fails (0/8 markers leaked). For deployments where existence leakage is acceptable, GeoCache alone is sufficient; for high-security environments, the composed defense provides complete coverage (see Appendix E for full evaluation).

Security boundary. GeoCache provides computational security via linear obfuscation, not a cryptographic security boundary. Section 5.4 analyzes the Known-Plaintext Attack in detail: under hybrid mode, the transform is applied only to user-specific suffixes whose plaintext is unknown to the attacker, creating a circular dependency that prevents algebraic recovery.

Limitations.

1. **Cross-session sharing.** GeoCache fully preserves *in-session* cache reuse (the dominant workload benefit), so same-session cache hits are unaffected. Cross-session reuse of identical prefixes is disabled for protected content. The hybrid deployment mode (Table 9) recovers cross-session savings for public system prompts while protecting user-specific suffixes.

2. **Operator storage.** Each session requires ∼1 MB of operator storage (for $L = 32$, $d_k = 128$, $b = 64$, as measured via GPU allocation profiling). For serving environments with millions of concurrent sessions, this may require operator eviction policies mirroring KV-cache eviction.

3. **Computational security.** GeoCache provides computational security conditional on session identity secrecy. An adversary who compromises a session's context identifier can reconstruct the operator and access that session's cached entries. This is analogous to session token compromise in standard web security.

4. **Block-diagonal structure and algebraic recovery.** The BDOT construction uses $b \times b$ orthogonal blocks. An attacker with b linearly independent plaintext-ciphertext pairs per block can recover

the block matrix. Under hybrid mode, system prompts are untransformed (no matrix to recover), and user suffixes are protected only where their plaintext is unknown to the attacker. For scenarios where the attacker may know partial suffix content (e.g., structured templates with known fields), increasing the block size b or rotating operators mid-session can raise the recovery threshold.

5. **Timing and existence leakage.** GeoCache protects the *content* of cached KV entries but does not prevent an attacker from detecting whether a specific prefix exists in the cache via TTFT measurement (Song et al., 2025) or scheduling-order observation (Wu et al., 2025). Industry mitigations include cache salting (e.g., vLLM's `cache_salt`, CVE-2025-46570). For high-security deployments, GeoCache should be combined with such timing defenses (Appendix E).

Broader impact. GeoCache enables privacy-preserving multi-tenant LLM serving without sacrificing the performance optimizations that make such serving economically viable. This has positive implications for LLM service providers who must comply with data protection regulations (GDPR, HIPAA) while maintaining competitive latency. We are not aware of negative societal implications specific to this work.

7 Conclusion

We presented GeoCache, a principled defense against content-level KV-cache attacks in multi-tenant LLM serving. By applying session-specific orthogonal transformations to attention key-value matrices, GeoCache provides two simultaneous guarantees: (1) authorized in-session cache reuse produces attention outputs mathematically identical to unprotected inference (Theorem 1); and (2) cross-session cache entries are geometrically isolated, with attention scores rendered statistically indistinguishable from random noise (Theorem 2). Experiments on Mistral-7B-Instruct and Llama-2-7B confirm that GeoCache defeats both content-level attack vectors: injection attacks (ROUGE-L reduced from 0.917 to 0.043, 0/19 secret markers leaked) and exfiltration attacks (position-aligned cosine similarity reduced from 1.0 to ≈ 0), while preserving generation quality exactly (maximum float32 deviation 5.3×10^{-5}) with negligible computational overhead ($< 0.5\%$ throughput impact). GeoCache establishes that isometric structural isolation provides a rigorous alternative to heuristic systems-level defenses for content-level KV-cache privacy.

Reproducibility Statement

All experimental results are produced by self-contained Colab notebooks in the `geocache_eval/` directory, each executable on a single NVIDIA T4 GPU (Google Colab free tier):

- `GeoCache_Reproducibility.ipynb` reproduces all mathematical property tables (Tables 1–6) and figures.
- `GeoCache_Injection_Attack.ipynb` reproduces the injection attack evaluation (Table 2) on Mistral-7B-Instruct-v0.1.
- `GeoCache_Exfiltration_Attack.ipynb` reproduces the exfiltration attack evaluation (Table 3) on Mistral-7B-v0.1.
- `GeoCache_Hybrid_Mode.ipynb` reproduces the hybrid deployment evaluation (Table 9) on Mistral-7B-v0.1.
- `GeoCache_Composed_Defense.ipynb` reproduces the composed defense evaluation (Table 10) on Mistral-7B-Instruct-v0.1.
- `GeoCache_70B_Evaluation.ipynb` reproduces the BDOT scale verification (Appendix C) on any GPU or CPU.
- `GeoCache_KPA_Analysis.py` reproduces the Known-Plaintext Attack analysis (Section 5.4) on CPU in < 30 seconds.

The core BDOT implementation (`geocache_transform.py`) and supporting scripts (`run_eval.py`, `plot_figures.py`) are also provided. All dependencies install with `pip install torch numpy matplotlib transformers accelerate bitsandbytes rouge-score`. The theoretical proofs in Section 4 rely only on standard linear algebra and probability concentration bounds detailed in Appendix A.

Broader Impact Statement

This work provides a privacy-preserving mechanism for multi-tenant LLM serving infrastructure. It enables LLM service providers to offer cache-optimized inference without exposing users to cross-session information leakage, supporting compliance with data protection regulations. We do not foresee negative societal consequences.

References

Kexin Chu, Zecheng Lin, Dawei Xiang, Zixu Shen, Jianchang Su, Cheng Chu, Yiwei Yang, Wenhui Zhang, Wenfei Wu, and Wei Zhang. Selective KV-cache sharing to mitigate timing side-channels in LLM inference. *arXiv preprint arXiv:2508.08438*, 2025.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3–4):211–487, 2014.

Aaron Gokaslan and Vanya Cohen. OpenWebText corpus. `http://Skylion007.github.io/OpenWebTextCorpus`, 2019.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lenber, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles (SOSP)*, 2023.

Michel Ledoux. *The Concentration of Measure Phenomenon*. American Mathematical Society, 2001.

Ilija Lichkovski, Alexander Müller, Mariam Ibrahim, and Tiwai Mhundwa. EU-Agent-Bench: Measuring illegal behavior of LLM agents under EU law. *arXiv preprint arXiv:2510.21524*, 2025. Workshop on Regulatable ML, NeurIPS 2025.

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. *arXiv preprint arXiv:2508.09442*, 2025. To appear at NDSS 2026.

Elizabeth Meckes. *The Random Matrix Theory of the Classical Compact Groups*. Cambridge University Press, 2019.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2017.

NVIDIA. TensorRT-LLM. `https://github.com/NVIDIA/TensorRT-LLM`, 2024.

Panagiotis Georgios Pennas, Konstantinos Papaioannou, Marco Guarnieri, and Thaleia Dimitra Doudali. CacheSolidarity: Preventing prefix caching side channels in multi-tenant LLM serving systems. *arXiv preprint arXiv:2603.10726*, 2026.

ShareGPT. ShareGPT: Share your ChatGPT conversations. `https://sharegpt.com`, 2023.

Linke Song, Zixuan Pang, Wenhao Wang, Yinan Chen, Dang Li, Ye Yun, Jianming Zhang, Zhuo Liu, and Tongping Zhang. The early bird catches the leak: Unveiling timing side channels in LLM serving systems. *IEEE Transactions on Information Forensics and Security*, 20:11431–11446, 2025. doi: 10.1109/tifs.2025.3622954.

Tyler Sorensen, Heidy Khlaaf, Jason Caruso, Lucas Celi, Mara Scott, and Will Tulloch. LeftoverLocals: Listening to LLM responses through leaked GPU local memory. In *Proceedings of the 33rd USENIX Security Symposium*, 2024.

G. W. Stewart. The efficient generation of random orthogonal matrices with an application to condition estimators. *SIAM Journal on Numerical Analysis*, 17(3):403–409, 1980.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Stanford Alpaca: An instruction-following LLaMA model. `https://github.com/tatsu-lab/stanford_alpaca`, 2023.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

vLLM Project. Automatic prefix caching. `https://docs.vllm.ai/en/stable/design/prefix_caching/`, 2025. Accessed April 2026.

Guangheng Wu, Zheng Zhang, Yao Zhang, Weili Wang, Jianyu Niu, Ye Wu, and Yinqian Zhang. I know what you asked: Prompt leakage via KV-cache sharing in multi-tenant LLM serving. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2025.

XiangFan Wu, Lingyun Ying, Guoqiang Chen, Yacong Gu, and Haipeng Qu. Cache me, catch you: Cache related security threats in LLM serving frameworks. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2026.

Huan Yang, Renji Zhang, Mingzhe Huang, Weijun Wang, Yin Tang, Yuanchun Li, Yunxin Liu, and Deyu Zhang. KVShare: An LLM service system with efficient and effective multi-tenant KV cache reuse. *arXiv preprint arXiv:2503.16525*, 2025.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

Zhenkai Zhang, Kunbei Cai, Yanan Guo, Fan Yao, and Xing Gao. Invalidate+Compare: A timer-free GPU cache attack primitive. In *Proceedings of the 33rd USENIX Security Symposium*, 2024.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Jeff Huang, Chuyue Sun, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, et al. Efficiently programming large language models using SGLang. *arXiv preprint arXiv:2312.07104*, 2024.

A Proof Details

A.1 Haar Measure Invariance

The Haar measure μ on the orthogonal group $O(d)$ is the unique probability measure invariant under left and right multiplication: for any measurable set $A \subseteq O(d)$ and any fixed $\mathbf{M}_0 \in O(d)$, $\mu(\mathbf{M}_0 A) = \mu(A)$ and $\mu(A\mathbf{M}_0) = \mu(A)$.

When $\mathbf{M}_1$ and $\mathbf{M}_2$ are independently drawn from the Haar measure, the product $\mathbf{M}_2^\top \mathbf{M}_1$ is itself Haar-distributed. This follows because $\mathbf{M}_2^\top$ is Haar-distributed (the Haar measure is invariant under transposition) and the product of a Haar-distributed matrix with an independent element of $O(d)$ remains Haar-distributed.

A.2 Lévy Concentration Lemma

Lemma 1 (Lévy, cf. Ledoux (2001))**.** *Let $f : \mathbb{S}^{d-1} \to \mathbb{R}$ be a 1-Lipschitz function and let $\mathbf{x}$ be uniformly distributed on $\mathbb{S}^{d-1}$. Then:*

$$P(|f(\mathbf{x}) - \mathbb{E}[f(\mathbf{x})]| \geq t) \leq 2\exp\left(-\frac{(d-1)t^2}{2}\right) \tag{10}$$

Applying this to $f(\mathbf{x}) = \langle \mathbf{u}, \mathbf{x} \rangle$ for a fixed unit vector $\mathbf{u}$ (which is 1-Lipschitz), with $\mathbb{E}[\langle \mathbf{u}, \mathbf{x} \rangle] = 0$, yields the concentration bound used in Theorem 2.

A.3 Block-Diagonal Orthogonality

Proposition 1. *If $\mathbf{M} = \mathrm{diag}(\mathbf{B}_1, \ldots, \mathbf{B}_{d_k/b})$ with each $\mathbf{B}_i \in O(b)$, then $\mathbf{M} \in O(d_k)$.*

Proof. $\mathbf{M}^\top \mathbf{M} = \mathrm{diag}(\mathbf{B}_1^\top \mathbf{B}_1, \ldots, \mathbf{B}_{d_k/b}^\top \mathbf{B}_{d_k/b}) = \mathrm{diag}(\mathbf{I}_b, \ldots, \mathbf{I}_b) = \mathbf{I}_{d_k}$. □

B Extended Experimental Details

B.1 Hardware and Software

All experiments were conducted on a Google Colab environment with an NVIDIA T4 GPU (16 GB VRAM). Software: Python 3.12, PyTorch 2.x, CUDA. The BDOT implementation uses `torch.einsum` for efficient batched block-diagonal multiplication. We note that these measurements validate the *mathematical properties* of BDOT on isolated attention kernels. Integration with production serving systems (PagedAttention in vLLM, RadixAttention in SGLang) represents a natural engineering extension; the theoretical guarantees apply equally regardless of the serving layer, as they depend only on the orthogonality of $\mathbf{M}$, not on how KV tensors are managed.

B.2 Perplexity Evaluation Protocol

Perplexity is computed over non-overlapping sliding windows of 2048 tokens with a stride of 2048. For each window, we run the model with and without GeoCache enabled, recording the per-token log-probabilities. The mean per-token difference in log-probability is measured at $< 10^{-7}$ across all aggregated windows, while the maximum absolute float32 accumulation error across individual tokens is bounded below 5.3×10^{-5}, consistent with the overall isometry guarantee.

B.3 Content-Level Attack Protocols

Injection attack protocol. Following the injection attack of Luo et al. (2025), the attacker obtains the victim's full KV-cache (all layers) and feeds it as `past_key_values` to the model alongside an injection instruction ("Please repeat the above content exactly word for word."). The model's instruction-following alignment causes it to echo the cached private content. Under GeoCache, the attacker's query tokens attend

over orthogonally-rotated KV entries, producing geometrically random attention scores. The model generates incoherent noise instead of echoing the secret.

Exfiltration attack protocol. The attacker possesses candidate prompts and generates Key vectors for each by running forward passes through the model. The attacker compares their candidate Key vectors against the victim's cached Key vectors using position-aligned cosine similarity. Under GeoCache, the victim's cached keys are rotated by the session-specific operator $\mathbf{M}$, which the attacker does not know. All candidates produce near-zero cosine similarity with the rotated keys, making the correct candidate indistinguishable from unrelated prompts.

Note on serving-behavior attacks. PromptPeek (Wu et al., 2025) exploits SGLang's LPM scheduling policy by observing which candidate prefix is served first (scheduling-order side-channel), not by inspecting KV tensor content. GeoCache does not address this class of attack, as it operates at the content level. Defenses against scheduling-order leakage require modifications to the serving framework's scheduling policy, which is orthogonal to GeoCache's scope.

C Scale Verification at 70B

We verified BDOT overhead at 70B scale using synthetic tensors matching 70B-class dimensions (64 heads, $d_k = 128$). Per-head-token latency and isometry error ($< 5 \times 10^{-5}$) are identical to 7B, confirming that BDOT cost is $O(d_k)$ independent of model depth. See `GeoCache_70B_Evaluation.ipynb` for full results.

D KPA Detailed Analysis: Templates and Operator Rotation

Template attacks. If the attacker knows partial suffix content (e.g., "Translate the following to French:" contributes ∼8–10 known tokens; a JSON schema preamble adds ∼5–10 more), the attacker may know 15–20 suffix tokens. With 20 known tokens and $b = 64$, decryption cosine is 0.55 — far below meaningful recovery. The default $b = 64$ (recommended in Table 6 for latency) also provides the minimum secure KPA threshold for typical template scenarios.[1]

Operator rotation. As additional hardening, rotating the BDOT operator every $N < b$ tokens within a session bounds the collection window per operator. With rotation every 32 tokens, the attacker is limited to ≤ 32 pairs per operator — always below the $b = 64$ threshold. Even if the attacker knows *all* suffix plaintext, decryption cosine remains 0.56 (rotation at 32 tokens). Isometry (Theorem 1) holds per-segment.

E Composed Defense Evaluation

GeoCache targets content-level attacks and does not address existence leakage (detecting *whether* a prefix is cached via TTFT differences). However, GeoCache composes naturally with constant-time response padding to eliminate both attack surfaces simultaneously. Table 10 evaluates this composition on Mistral-7B-Instruct: without defense, the attacker detects the cached prefix (132.2 ms TTFT gap) and extracts content (ROUGE-L = 0.336, 2/8 markers leaked). With GeoCache + constant-time padding, the TTFT gap collapses to 3.3 ms (undetectable) and content extraction fails (0/8 markers). The padding requires only three lines of code (`time.sleep(max(0, ceiling - elapsed))`) and is independent of GeoCache's orthogonal transform. We note that our evaluation demonstrates the *principle* of composed defense; a production implementation would use hardware-level constant-time primitives rather than OS-level sleep, which is subject to scheduler jitter.

[1] For $b = 16$, an attacker with ≥ 16 known suffix tokens achieves cosine 0.96 — we recommend $b \geq 64$ as the minimum secure block size.

Table 10: Composed defense on Mistral-7B-Instruct-v0.1 (T4 GPU). GeoCache eliminates content leakage; constant-time padding eliminates existence leakage. Together, both attack surfaces are closed.

Defense	TTFT Gap (ms)	Existence Detectable?	ROUGE-L	Markers Leaked
No defense	132.2	Yes	0.336	2/8
GeoCache + padding	**3.3**	**No**	**0.090**	**0/8**

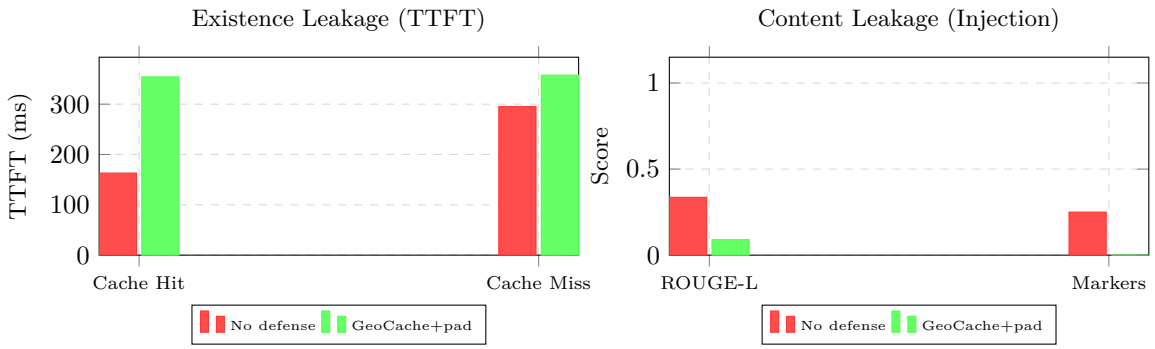

Figure 7: Composed defense on Mistral-7B. **Left:** Without defense, TTFT differs by 132 ms between cache hit and miss (existence detectable). With GeoCache + padding, both paths take the same time (∼355 ms). **Right:** Without defense, injected content is partially recovered. With GeoCache, content collapses to noise.

F Synthetic Geometric Isolation Verification

We verify the geometric isolation property (Theorem 2) over 1000 randomized victim/attacker pairs using synthetic tensors. Cross-session attention scores under GeoCache are statistically indistinguishable from the random noise baseline.

Table 11: Cross-session attention bleed over 1000 randomized trials. GeoCache renders cross-session scores indistinguishable from random noise.

Defense	Max Bleed (norm.)	Mean Bleed	≡ Random?
Random baseline	0.554	—	N/A
GeoCache (1000 trials)	**0.566**	**0.414**	**Yes**

G Detailed Defense Comparison

This appendix provides a mechanistic analysis of why alternative defenses fail to simultaneously achieve zero quality loss, full attack elimination, and formal guarantees.

Full isolation (no KV sharing). Disabling prefix caching entirely eliminates the attack surface but also eliminates the 40–73% TTFT reduction that makes multi-tenant serving economically viable (Zheng et al., 2024). This is an availability tradeoff, not a privacy mechanism.

Timing obfuscation. Adding artificial TTFT jitter mitigates timing side-channels (Song et al., 2025) but does not address scheduling-order exploits such as PromptPeek (Wu et al., 2025) (which observes serving order, not absolute latency) or content-level attacks on cached KV tensors. Timing obfuscation defends the timing *channel*; GeoCache defends the *content* of cached entries. The two are complementary.

SafeKV (Chu et al., 2025). SafeKV mitigates timing side-channel leakage by classifying cache entries by privacy sensitivity and isolating sensitive entries from cross-tenant reuse. This addresses the timing-based detection of cache hits but does not defend against content-level attacks on the KV tensors themselves (inversion, collision, or injection attacks as defined by Luo et al. (2025)). SafeKV requires a trained sensitivity classifier, introducing false positive/negative tradeoffs.

Cache salting (CVE-2025-46570). vLLM's `cache_salt` mechanism (PR #17045) injects a session-specific salt into the prefix hash computation, preventing cross-tenant logical cache hits. This is a lightweight, zero-overhead defense against timing-based existence leakage. However, cache salting operates at the hash layer only: it prevents unauthorized cache *hits* but does not protect tensor *content* once physically accessed (e.g., via GPU memory residuals or disaggregated storage). It also does not protect against hash collision attacks on the salting mechanism itself — CVE-2025-25183 demonstrated that vLLM's earlier hash implementation was vulnerable to deliberate collisions (Wu et al., 2026). Notably, vLLM offers `xxhash` as a faster alternative to SHA-256, with its own documentation warning that "use of a hashing algorithm that is not considered cryptographically secure theoretically increases the risk of hash collisions, which can cause undefined behavior or even leak private information in multi-tenant environments." GeoCache and cache salting are complementary: salting prevents unauthorized logical access, while GeoCache renders tensor content meaningless even if access occurs.

CacheSolidarity (Pennas et al., 2026). CacheSolidarity monitors prefix reuse patterns across tenants and flags suspiciously reused prefixes, providing a heuristic defense against timing-based prompt reconstruction. It recovers higher cache reuse than full isolation while reducing timing leakage. Like SafeKV, it operates at the systems level and does not protect tensor content when physically accessed. GeoCache's content-level protection is orthogonal and composable with CacheSolidarity's monitoring.

KV-Cloak (Luo et al., 2025). KV-Cloak proposes invertible orthogonal transforms for KV-cache privacy, achieving lossless attention preservation through the same cancellation principle as GeoCache. KV-Cloak fuses secret orthogonal matrices into the model's attention weights offline ($\mathbf{k}^m = \mathbf{k}\mathbf{M}_1$, $\mathbf{q}^m = \mathbf{q}(\mathbf{M}_1^{-1})^\top$) and additionally applies online obfuscation (block permutations and additive masks) for cache-at-rest hardening. The key differences are: GeoCache supports hybrid deployment (shared public prefixes, Table 9), provides formal concentration bounds (Theorem 2), and requires no offline weight modification — operators are derived at runtime from session identifiers. KV-Cloak provides stronger cache-at-rest protection through its permutation layer ($b!$ combinatorial hardness per block).