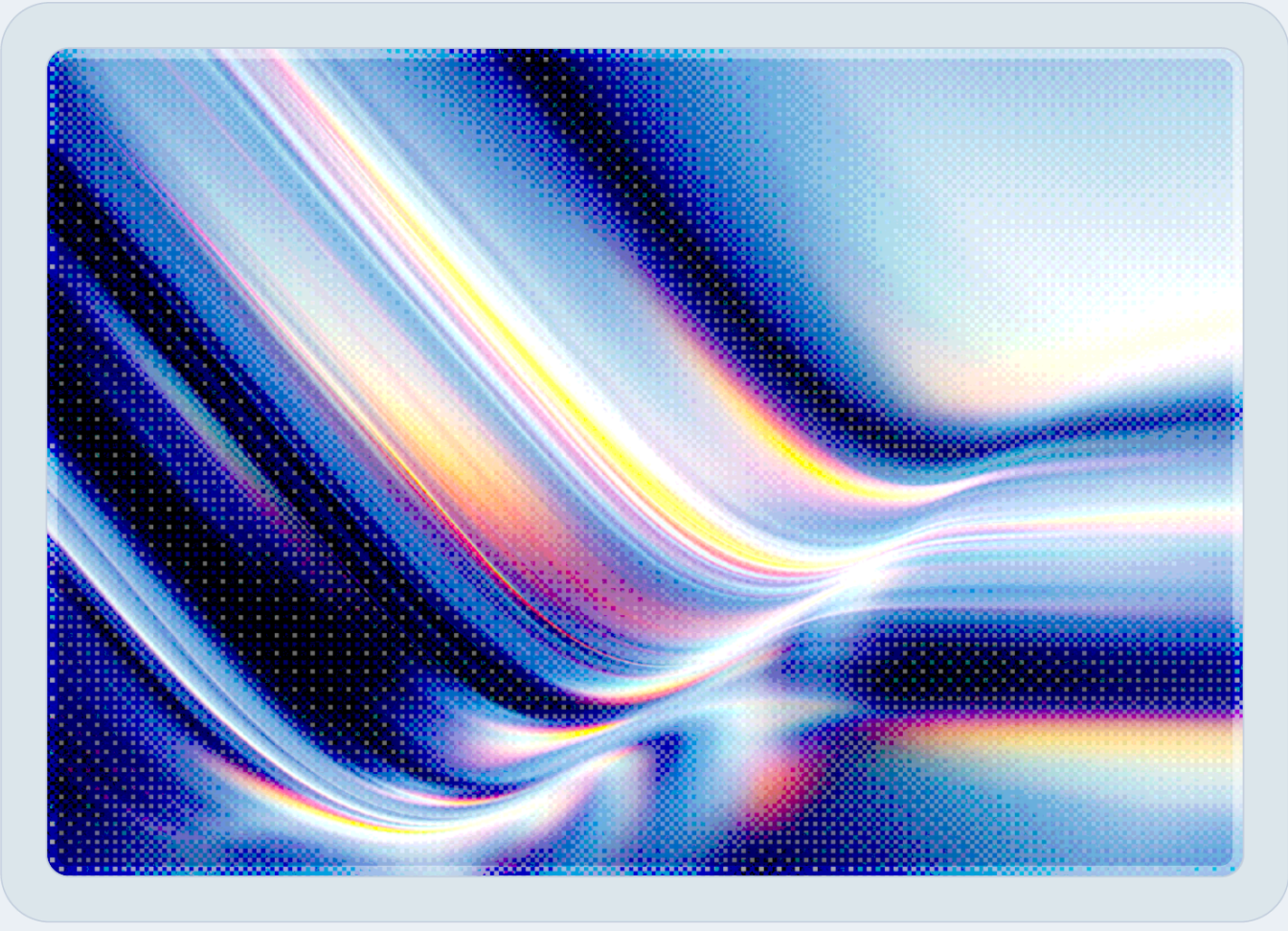
CSAI Foundation | Cloud Security Alliance

When the Inference Engine Leaks: Cross-Tenant KV Cache Exposure

Security Implications and Guidance for Disaggregated LLM Serving

2026-11-02

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

© 2026 Cloud Security Alliance. Some rights reserved.

You may download, store, display, view, print, redistribute, and link to this document in its original, unmodified form, provided that attribution to the Cloud Security Alliance is maintained and all trademark and copyright notices remain intact.

This document may not be modified or altered. You may quote portions of the document as permitted by the Fair Use provisions of the United States Copyright Act, provided that attribution is given to the Cloud Security Alliance.

This document may be shared on professional and social media platforms in its original form with attribution.

This document was generated with AI assistance and has not undergone official CSA review and approval processes.

Key Takeaways

On October 26, 2026, Cloudflare and the maintainers of the open-source SGLang inference engine jointly disclosed a cross-request key-value (KV) cache exposure affecting the GLM-5.3 model on Cloudflare Workers AI [1][2]. Between August 27 and September 9, 2026, 155 responses to very long prompts contained content derived from other customers' recent requests to the same model [1]. The incident did not stem from a malicious tenant or a novel attack. It arose from an engine bug in disaggregated prefill/decode serving, combined with a deployment where the prefill GPU memory pool was smaller than the advertised context window [1][2]. That matters because it shows that the performance machinery of modern LLM serving can, in ordinary operation, cross the boundary between customers.

The central lesson is that the isolation boundary in LLM serving now sits inside GPU memory. The KV cache, the per-request attention state held on accelerators, is pooled, reused, and transferred between machines, and bugs in its lifecycle can expose one tenant's data to another with no error, log line, or failed HTTP status [1][2]. Disaggregated serving adds a further risk in the form of an asynchronous transfer path. SGLang's advisory describes a second, independent mechanism in which late one-sided RDMA writes could land in KV slots that had already been reassigned to another request, affecting disaggregated configurations from the v0.4.x line through v0.5.21 [2].

Cache sharing is also a deliberate, measurable side channel. Peer-reviewed research has shown that cross-tenant prefix caching in vLLM and SGLang lets an attacker reconstruct other users' prompts through timing [3][4], and audits have found global cache sharing at production API providers [5]. Operators should therefore upgrade SGLang to v0.5.22 or later (v0.5.23 for NIXL and MORI transfer backends), verify KV pool size against context length on every node, and scope prefix-cache reuse to a tenant identity [1][2][7]. Consumers of hosted inference, for their part, should treat cache isolation as a contractual and assurance question, asking providers to state whether caches are shared across customers and how cross-request isolation is tested.

Background

Transformer-based language models compute a key and value vector for every token at every attention layer. Inference engines retain these vectors, the KV cache, so that each new output token does not require recomputing attention over the entire preceding context. For long-context models, KV cache memory dominates accelerator usage. Engines such as vLLM and SGLang therefore manage it as a paged pool: fixed-size blocks are allocated to requests, freed when requests finish, and reused for subsequent requests. To save further compute, both engines support prefix caching. When a new request begins with the same token sequence as an earlier one, as with a shared system prompt, the engine reuses the cached blocks instead of recomputing them [3][7].

Disaggregated serving goes a step further. Prefill is the compute-heavy pass that processes the input prompt, while decode is the memory-bandwidth-bound loop that generates output tokens. The two phases run on separate GPU pools so each can be scaled and tuned independently. After prefill finishes, the resulting KV cache is moved to a decode node over a high-speed interconnect, typically through RDMA transfer backends such as Mooncake, NIXL, or MORI [2]. A router coordinates the pairing: decode reserves memory pages for the incoming request, and prefill writes the computed cache into them. The design is intended to improve throughput and latency, but the result is a distributed protocol in which two machines must agree on which memory belongs to which request.

The security relevance of this architecture follows from its multi-tenancy. In a hosted inference service, requests from unrelated customers share the same model replicas, the same KV pools, and, unless deliberately isolated, the same prefix caches. Conventional cloud isolation controls such as virtual machines, network segmentation, and identity-scoped storage do not apply inside a single inference process. Customer separation there depends on the correctness of the engine's memory bookkeeping and on the policy governing cache reuse. The Cloudflare incident is a concrete public example of that boundary failing in production.

Security Analysis

The public record now contains two distinct engine-level failure mechanisms disclosed together by Cloudflare and SGLang, alongside a body of research showing that cache sharing can be exploited deliberately. The subsections below examine each in turn before drawing out the implications for operators and customers.

The Cloudflare GLM-5.3 Incident

According to Cloudflare's post-incident report, GLM-5.3 launched on Workers AI on August 27, 2026 with a 1,048,576-token context window. However, the KV cache pool on its prefill GPUs held only 681,280 tokens [1]. SGLang sizes this pool at startup from remaining GPU memory, and nothing enforced that it match the configured context length. Cloudflare reports that it knew of the mismatch before launch but expected the engine to reject oversized prompts with an error. Cloudflare also reports that its pre-launch tests validated only HTTP response codes and engine performance at long contexts, not output correctness [1].

The engine did not behave as expected in disaggregated mode. When a prompt exceeded the prefill pool, SGLang's rejection path replaced the prompt with a one-token placeholder and marked the request for abort. The disaggregation admission queues, however, checked whether a request had *finished* rather than whether it had been *rejected*, so the request continued [1]. The router had already told the decode node the full prompt length, and decode reserved pages accordingly. Prefill then ran on the placeholder and transferred a single page of KV cache. Decode accepted the transfer and attended over every reserved page, including stale pages that still held other users' attention state. The client received an HTTP 200 and a coherent reply that incorporated content from other customers' requests [1]. The underlying bug had been reported upstream as SGLang issue #30233 on July 6, 2026 [1].

A security researcher, Gadi Evron of Knostic, reported the behavior through Cloudflare's HackerOne program on September 8 [1]. Cloudflare declared an incident the following day and reproduced the leak from two accounts within 25 minutes. It set the model to private at 14:00 UTC and restored it later on September 9 after enlarging the prefill pool and deploying an engine patch [1]. Cloudflare excluded its own and the researcher's test traffic from the count of 155 affected responses. It reports that the remaining requests matched ordinary long-context usage in timing and volume, and it found no evidence of deliberate exploitation [1]. Cloudflare states that GLM-5.3 was the only affected Workers AI model, because its other disaggregated deployments had KV pools larger than their context lengths [1].

A Second Mechanism: Late RDMA Writes

SGLang's coordinated advisory widens the scope beyond Cloudflare's configuration [2]. In disaggregated serving, the transfer backends issue asynchronous one-sided RDMA writes from prefill into decode memory, and the network interface provides no mechanism to cancel a work request once it has been posted. Earlier SGLang code freed a decode request's KV slots immediately on abort. If the allocator reassigned those slots to a new request before in-flight writes from the aborted request landed,

the new request's cache was silently overwritten [2]. The maintainers identify three preconditions: disaggregated serving with a transfer backend, termination mid-transfer, and slot reuse before the writes complete. Long contexts widen the window.

The maintainers state that affected defaults span from the v0.4.x line in early 2025 through v0.5.21, or v0.5.22 for the MORI backend. The advisory also describes a separate variant that requires decode-side hierarchical cache offload, which is off by default; operators who have enabled that feature should consider themselves potentially exposed [2]. Critically, the maintainers note that operators cannot confirm past exposure, because this corruption produces no error or log entry [2].

The table below summarizes the two disclosed engine-level mechanisms and their remediations.

Mechanism	Trigger	Effect	Primary fix	Deferred-release protection on by default
Admission rejection bypass	Prompt exceeds prefill KV pool in disaggregated mode	Decode attends over stale, unpopulated pages containing prior requests' state	#36988 (in v0.5.20); #38935 rejects at intake (v0.5.21)	v0.5.21 and later
Late RDMA write after slot reuse	Request aborted mid-transfer; slots reassigned before writes land	Another request's KV cache silently overwritten	Deferred KV release with ABORT/ABORT_ACK draining (#35049 and related)	v0.5.21 for Mooncake and NIXL; v0.5.22 for MORI*

Source: SGLang security update [2] and Cloudflare incident report [1]. *Follow-on fixes for failed transfers shipped in v0.5.22 for Mooncake (#43719) and in v0.5.23 for NIXL and MORI (#45973) [2].

The deferred-release fix keeps an aborted request's slots reserved until every prefill rank acknowledges that its outstanding writes have drained, or until a timeout expires. The maintainers describe the 30-second default timeout as "a practical backstop, not a proof" [2]. They report throughput differences

within ±2 percent across 21 benchmark runs, which suggests the fix carries little performance cost [2]. SGLang also added an opt-in Adler-32 checksum on prefill-to-decode transfers (`--disaggregation-enable-kv-checksum`), which is off by default [2].

The Deliberate Side Channel: Shared Prefix Caches

The Cloudflare incident was accidental, but the broader research record shows that KV cache sharing can also be exploited on purpose. Wu et al. presented PromptPeek at NDSS 2025. They showed that because vLLM and SGLang share KV cache blocks across requests with matching token prefixes, an adversary can tell whether its own request matched another user's by observing whether sharing occurred, and that signal is enough to reconstruct other tenants' prompts [3]. Song et al. independently demonstrated timing side channels in both KV caches and semantic caches, verified in SGLang among other systems, using a token-by-token search to recover system prompts and peer users' prompts [4]. At the API level, Gu et al. ran statistical timing audits of commercial LLM APIs, published at ICML 2025, and detected global cross-user cache sharing at seven providers, including OpenAI [5].

The engines provide some mitigations, though these depend on correct configuration and have their own failure modes. vLLM supports an optional per-request `cache_salt` that is mixed into the hash of the first cache block, so only requests carrying the same salt can reuse each other's cached blocks [7]. Its documentation also warns that non-cryptographic block hashes raise collision risk that could "leak private information in multi-tenant environments" [7]. Salt propagation can also fail, as one recent CVE shows. CVE-2026-105752 describes vLLM tool-continuation flows that dropped `cache_salt` and placed continuation prefixes in the shared, unsalted namespace, which permits a tenant to confirm, though not directly read, another tenant's prior prompts via cached-token counts. Exploitation requires the `/v1/responses` endpoint and an operator-enabled tool server; Red Hat rates it Low severity, with the upstream fix in vLLM 0.30.0 [8]. A recent preprint argues that the inference engines it examined compute cache keys over token sequences alone, without binding them to the issuing principal, and proposes an HMAC-derived per-principal salt as the necessary isolation control [6]. That work evaluates its defense in simulation, so its efficiency estimates should be treated as preliminary.

Implications

Taken together, these findings suggest that KV cache isolation should be treated as a first-class tenant boundary, comparable to storage or network isolation, rather than as an internal performance detail. Three properties make it particularly difficult to manage. First, failures can be silent: both SGLang mechanisms produced well-formed, successful responses, and the RDMA race leaves no forensic trace [1] [2]. Second, failures are often configuration-dependent. The Cloudflare exposure required a pool smaller

than the context window, a condition that changes whenever weights, quantization, parallelism, or memory settings change [1]. Third, conventional testing may not catch them. Cloudflare's pre-launch checks validated response codes and performance rather than output integrity [1].

For organizations consuming hosted inference, the incident indicates that confidentiality depends on engine internals the customer cannot inspect. Standard assurance questionnaires do not typically address whether KV or prefix caches are shared across customers, how cross-request isolation is tested, or how quickly engine-level security fixes are deployed. The research on API-level cache sharing [5] shows that cross-customer cache reuse has occurred at multiple major providers. Customers should ask these questions directly rather than assume isolation.

Recommendations

The recommendations below are organized by time horizon. Immediate actions close the specific engine defects disclosed in October 2026, short-term mitigations add detection and isolation controls that do not depend on any single fix, and strategic considerations address how organizations govern inference infrastructure as shared-tenancy infrastructure.

Immediate Actions

Operators running SGLang in disaggregated prefill/decode mode should upgrade to v0.5.22 or later, which Cloudflare recommends for all operators [1]. Version 0.5.21 is the first release that protects Mooncake and NIXL by default, while MORI requires a build that includes #37077, which ships in v0.5.22 [2]. Earlier branches were not backported, so patching an older release line is not an option. The advisory also lists follow-on fixes for failed transfers, in v0.5.22 for Mooncake (#43719) and in v0.5.23 for NIXL and MORI (#45973); operators on those two backends should therefore treat v0.5.23 as the current baseline [2]. Operators who cannot upgrade immediately from v0.5.19 or v0.5.20 should set `SGLANG_DISAGGREGATION_DEFERRED_DECODE_KV_RELEASE=1` on both prefill and decode roles. On v0.5.20 they should also bound the Mooncake transfer batch and raise the hold timeout to 60–120 seconds, as the maintainers advise [2].

Independently of version, operators should read the KV pool size from the startup log of every prefill and decode node and compare it against the configured context length, lowering the context length wherever the pool is smaller [1]. This check must be repeated after any change to model weights, quantization, tensor or pipeline parallelism, or memory fraction settings. Operators should also confirm that their router cancels both halves of a prefill/decode pair when either side fails [2].

Short-Term Mitigations

Over the coming weeks, operators should add integrity and isolation checks that do not depend on HTTP status codes. They should enable `--disaggregation-enable-kv-checksum` and Cloudflare's recommended `--kv-canary` corruption detection, then confirm in logs that requests are actually being verified [1][2]. Cloudflare notes that the canary and checksum do not yet cover every model architecture, so operators should confirm coverage for each deployed model [1]. On v0.5.22 and later, where the metric was introduced (#41128), monitoring should alert on `sglang:decode_deferred_kv_release_total{outcome}`; a high ratio of `timeout` outcomes indicates that the acknowledgment path is not draining reliably [2]. Release testing should include canary-marker tests, in which unique strings submitted from one account are checked for in responses to another account, across the full range of supported prompt lengths, including requests at and beyond the context limit [1].

For prefix caching, a safer default in a multi-tenant service is to scope cache reuse to a tenant or principal. In vLLM this means setting a per-tenant `cache_salt` on every request path, upgrading to a release containing the CVE-2026-105752 fix, and using the default cryptographic block hash [7][8]. Where global sharing is retained for cost reasons, operators should limit it to content that is genuinely public, such as a provider-published system prompt, and document the residual timing risk established by PromptPeek and related work [3][4].

Strategic Considerations

Over the longer term, organizations should bring the inference engine into scope for the same threat modeling, change management, and assurance that govern other shared-tenancy infrastructure. Disaggregated serving is a distributed memory protocol, and Cloudflare's ongoing work on page-level integrity tags and formal verification of its RDMA transfer protocol [1] shows the kind of engineering assurance the architecture may ultimately require. Organizations that build or customize serving stacks should consider binding cache identity cryptographically to the requesting principal, as proposed in recent research [6], rather than relying on salts that individual code paths may omit.

Enterprises purchasing hosted inference should add cache isolation to vendor due diligence. Questions should ask whether KV and prefix caches are shared across customers, what engine and version is deployed, how quickly upstream security fixes are applied, and whether output-integrity testing is part of release gating. Data classification policies should account for the possibility that highly sensitive prompts warrant dedicated or isolated inference capacity. Finally, the industry would benefit from coordinated

disclosure norms for inference engines that match those for hypervisors and kernels. The joint Cloudflare–SGLang disclosure [1][2], a coordinated release accompanied by version-specific remediation guidance, offers a useful template.

CSA Resource Alignment

CSA's **AI Controls Matrix (AICM) v1.1** [9] provides a control framework for operationalizing these recommendations, and several of its domains are relevant to the findings. Data security and privacy lifecycle controls address the confidentiality of prompts and attention state held in shared accelerator memory. Infrastructure and virtualization security controls can be extended to tenant segmentation inside the inference process. Change control and configuration management controls can require the pool-versus-context-length verification described above, and should require that mismatches be resolved, not only detected; Cloudflare was aware of the mismatch before launch [1]. Threat and vulnerability management controls govern timely uptake of upstream engine fixes such as SGLang v0.5.22 and v0.5.23 and the vLLM `cache_salt` correction. The **AICM v1.1 Auditing Guidelines for Model Providers** [14] supply control-by-control verification procedures that auditors and providers can use to test these controls in practice. Organizations mapping their inference platforms to AICM should explicitly scope KV and prefix caches as shared resources subject to tenant-isolation controls.

AI Organizational Responsibilities: Core Security Responsibilities [12] addresses the division of security duties for AI systems, including data protection across the model lifecycle. This incident sharpens a gap in that division. In hosted inference, the provider controls the cache isolation boundary entirely, and the customer has no technical means to verify it. CSA's guidance on assigning responsibility can help enterprises make that boundary explicit in contracts. Relatedly, **Securing LLM Backed Systems: Essential Authorization Practices** [10] argues that authorization decisions must be enforced outside the model itself. The cache-salt failures discussed here [6][8] illustrate the same principle at the infrastructure layer: tenant identity must be bound to every cached artifact by the system, not inferred from content.

For threat modeling, CSA's **MAESTRO** framework [11] treats the deployment and infrastructure layer as a distinct source of risk in AI systems, and inference-engine memory management, cache sharing policy, and inter-node KV transfer belong in that layer's threat inventory. Finally, **STAR for AI** [13] offers a mechanism for providers to disclose their controls publicly. Hosted inference providers can use their STAR for AI submissions to state their cache-sharing policy and cross-request isolation testing, giving customers the assurance evidence this incident shows is currently missing.

References

[1] Cloudflare. "Cross-request KV cache exposure in the GLM-5.3 Workers AI model, and our remediation with SGLang." The Cloudflare Blog, October 26, 2026.

[2] LMSYS Org / SGLang Team. "SGLang security update: improving request isolation." LMSYS Org Blog, October 26, 2026.

[3] Wu, G., Zhang, Z., Zhang, Y., Wang, W., Niu, J., Wu, Y., and Zhang, Y. "I Know What You Asked: Prompt Leakage via KV-Cache Sharing in Multi-Tenant LLM Serving." Network and Distributed System Security (NDSS) Symposium, 2025.

[4] Song, L., Pang, Z., Wang, W., Wang, Z., Wang, X., Chen, H., Song, W., Jin, Y., Meng, D., and Hou, R. "The Early Bird Catches the Leak: Unveiling Timing Side Channels in LLM Serving Systems." arXiv:2409.20002 (accepted to IEEE Transactions on Information Forensics and Security), revised October 2025.

[5] Gu, C., Li, X. L., Kuditipudi, R., Liang, P., and Hashimoto, T. "Auditing Prompt Caching in Language Model APIs." Proceedings of the 42nd International Conference on Machine Learning (ICML), PMLR 267, 2025.

[6] Addagada, T. C. "Governing the KV Cache: Preventing Timing Side-Channel Leakage in Multi-Tenant LLM Inference." arXiv:2608.09225, August 2026.

[7] vLLM Project. "Automatic Prefix Caching — Design Documentation." vLLM Documentation, accessed November 2, 2026.

[8] Red Hat. "CVE-2026-105752." Red Hat Customer Portal, 2026.

[9] Cloud Security Alliance. "AI Controls Matrix (AICM) v1.1." Cloud Security Alliance, accessed November 2, 2026.

[10] Cloud Security Alliance. "Securing LLM Backed Systems: Essential Authorization Practices." Cloud Security Alliance, 2024.

[11] Huang, K. "Agentic AI Threat Modeling Framework: MAESTRO." Cloud Security Alliance Blog, February 6, 2025.

[12] Cloud Security Alliance. "AI Organizational Responsibilities: Core Security Responsibilities." Cloud Security Alliance, 2024.

[13] Cloud Security Alliance. "STAR for AI." Cloud Security Alliance, accessed November 2, 2026.

[14] Cloud Security Alliance. "AICM v1.1 Auditing Guidelines for Model Providers (MP)." Cloud Security Alliance, accessed November 2, 2026.