Sciences du Nord Nature Science and Technology
Vol. 2, No. 02, December 2025: pp. 158-169
E-ISSN: 3046-6504
P-ISSN: 3047-4922
DOI: https://doi.org/10.71238/snnst.v1i02.235

Context-Aware GenAI Gateway Architecture for Secure Multi-Tenant Foundation Model Access in Enterprise Clouds: An Evidence-Based Review and Reference Model

Kedar Pradhan [1]
[1] Solutions Architect, Amazon Web Services, Austin, USA
Coresponding Author: kedar_pradhan90@yahoo.com

ARTICLE HISTORY
Received Dec, 2025 **Revised Dec, 2025** **Accepted Dec, 2025**

ABSTRACT

While increasingly being made accessible by enterprises to multiple teams and customers via a shared access layer, the security and governance properties of the access layer are dispersed across the systems, security and machine-learning literatures. In this paper, 24 papers (both peer-reviewed systems and security papers, preprints, standards and vendor reference designs) are brought together to analyze how a context-aware generative-AI (GenAI) gateway can enable multi-tenant FM access. The review covers four questions related to identity and context propagation, threats on input and output, equitable sharing of common capacity, and cross-tenant isolation. According to evidence, vendor gateway designs attempt to manage token quotas, metering and semantic caching, and according to the research literature, there are vulnerabilities that haven't been addressed by these: indirect prompt injection, for which 10 defenses have been benchmarked, have failed to be effective, and timing side channels from shared key-value and semantic cache have been observed. The reported efficiencies are: 2-4x serving throughput and up to 98% cost reduction from model cascades, and cannot be directly compared since the units were measured under different conditions. This synthesis is used to extract a 6-layer reference model for associating a per-request context with policy decisions, as well as formal representations of token-weighted admission and policy constrained routing and tenant-scoped caching. Open gaps include end-to-end evaluation of combined controls, the hit-rate cost of cache partitioning and gateway mediation of agentic tool calls. The work is a systematic synthesis and not a report of new experiments.

Keywords: *Cloud Security; Foundation Models; Generative AI Gateway; Large Language Models; Multi-Tenancy; Policy Enforcement; Prompt Onjection; Zero Trust*

INTRODUCTION

It is uncommon for enterprises to expose model endpoints to each application they use with a large language model (LLM) or other foundation model. Instead, a shared layer authenticates callers, applies limits and records consumption, before forwarding calls on. The pattern is exhibited in the public reference designs. Each team gets an API key in an AWS reference solution and each team's usage is logged and API Gateway usage plans throttle requests [1]. Microsoft announced new features for its Azure API Management service for generative-AI, such as a token-limit policy and a token-metric policy and semantic caching of completions [2]. The two designs are extensions of the basic API-gateway pattern for authenticating, throttling, metering and forwarding.

There are three parts of LLM traffic, which differs from the typical API traffic, that undermine inheritance. The data is natural language data that the model interprets, so it can be used to give instructions: prompt injection is the first threat in the OWASP Top 10 for LLM Applications [3] and the NIST Generative AI Profile highlights information security and data privacy as risks generative AI can introduce or amplify. Second, the cost and the natural unit of contention is the

token, not the request, and input and output tokens cost the server differently [4]. Third, time channels can be opened between users by optimizing their performance, sharing some state between requests, for example key-value (KV) caching [5] and semantic caching [6]. Therefore, the context-aware gateway needs to consider the following: its requesters, the data it contains, the models it allows, and common states it might participate.

This review addresses four research questions (RQs):

RQ1: How should tenant identity and request context be established so that policy is decided per request?

RQ2: Which input and output threats can a gateway mitigate, and how strong is the evidence for available defenses?

RQ3: How can shared model capacity and cost be allocated fairly across tenants?

RQ4: Which isolation obligations arise from shared serving state, and where does gateway control end?

The contributions are (i) a structured synthesis of 24 sources, organized around RQ1–RQ4; (ii) a six-layer reference model that maps zero-trust policy enforcement onto GenAI traffic; (iii) formal expressions for token-weighted admission, policy-constrained routing and tenant-scoped caching; and (iv) a gap analysis. The scope is inference-time access. Training-data governance and model supply-chain security are excluded.

LITERATURE REVIEW

NIST SP 800-207 defines Zero-Trust architecture as a policy decision point (PDP) that decides access requests and a policy enforcement point (PEP) that grants, monitors and denies connections (per session) based on contextual signals [7]. All prompts going to a model will pass through the gateway, where context can be added to the prompt, which can then be sent to the model. The gateway is therefore suitable to become a PEP for FM traffic and this is an analysis and not a requirement of the standard.

There are three types of isolation in Saas Architecture [8]. The silo model allocates resources to each tenant, eliminating compliance and cost attribution issues, and blast radius containment, at the cost of efficiency. Resources are shared in the pool and isolation is strictly observed by policy in the pool model. The bridge model is two combined together per layer. The area of FM serving is strongly pushing for pooling. vLLM utilizes the KV cache in pages and supports sharing across and among requests [9]; Punica stores thousands of low-rank adaptation (LoRA) adapters over one base model [10]; Punica stores one copy of the base model and batches requests for different adapters. It's very efficient, therefore, to provide for pooling since it's shared by the state that strict isolation would have split into two.

One way to do this is by using a retrieval-augmented generation (RAG) approach, which involves retrieving documents from an external store during the time of inference [11]. In an enterprise deployment, there is generally a store containing private material for the tenant and a gateway that processes queries from users, and the proprietary context.

METHODS

The study is not systematic review, it is an evidence based synthesis of the literature. Targeted searches were conducted using a general Web search engine on publisher and proceedings

pages (USENIX, ACM, ACL Anthology, MLSys), arXiv, standards bodies (NIST, OWASP) and documentation provided by cloud vendors. The terms used to query were: multi-tenant, LLM serving, LoRA serving, prompt injection, guardrail, LLM routing, side channel, GenAI gateway. The searches and source verification were done after the cutoff, and therefore only material, which was first published on or before the cutoff has been admitted, and further developments are purposefully not included. Because it wasn't possible to search bibliographic databases from which the number of results could then be exported, no statistics of the PRISMA flow is reported.

A source was identified if it was public at the cutoff, answered one or more research questions at inference time, and gave (1) a mechanism, (2) an empirical result, (3) a threat model or (4) a normative control. Technical material wasn't featured in the marketing material. In the same way as in RouteLLM [6] when a preprint is subsequently published in a venue after the cutoff, the preprint is cited. The mechanism, evaluation setting, results (including units), and limitations of each source were tabulated into an evidence matrix that was summarized in Table 1 and Figure 1. All claims were detected as direct, cross-study synthesis/analytical inferences and inferences are identified in the text. Documentation of vendors is not used to determine effectiveness of capabilities, but rather that capabilities were offered.

Table 1. Composition of the Reviewed Corpus

Category	Sources	Role in this synthesis
Standards and guidance	[3], [7], [8], [12]	Normative controls, isolation models, risk taxonomy
Vendor gateway designs	[1], [2]	Capabilities offered (not effectiveness)
Serving and multi-tenancy	[4], [9], [10]	Shared-state efficiency, token-level fairness
Attacks and leakage	[5], [13], [14]	Threat models and empirical attack evidence
Guardrails and defenses	[15], [16]	Input and output controls
Routing and cost	[6], [17]	Cost-aware model selection
Retrieval background	[11]	RAG mechanism and data flow

Note: Peer-reviewed: 11; preprints at cutoff: 7; standards/guidance: 3; vendor documentation: 3 (the AWS isolation whitepaper is counted with standards in Figure 1).

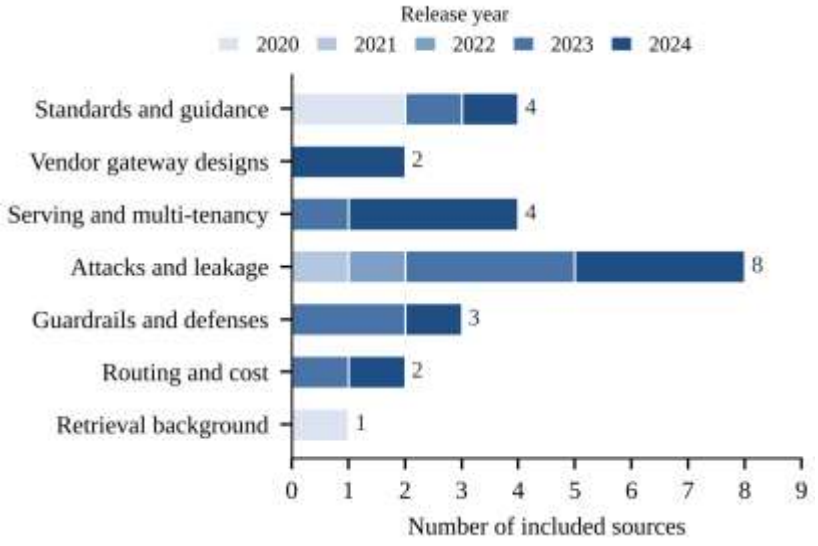


Figure 1. Composition of the 24 included sources by category and release year. Counts are derived from the reference list.

RESULTS AND DISCUSSION

4.1 Identity, Context and Policy (RQ1)

Tenants are mainly differentiated by vendor designs that depend on credentials based on a subscription. The AWS solution includes an API key for every team, and only a few models can be used with an identity and access management (IAM) policy; and for simplicity, the AWS solution excludes organizational authentication from scope.For each team, the AWS solution includes an API key, allows an limited number of models to be used with an identity and access management (IAM) policy, and explicitly excludes organizational authentication from scope for simplicity. Azure is using the API Management subscription to bind its token counter. Both of them assume the name of the tenant, but they don't have more context, like a user's role in the tenant, whether the data provided in the prompt is sensitive or what is being asked for. NIST profile structure that breaks down the function of risk management in three categories: govern, map, measure and manage, while providing guidelines for inventorying GenAI systems, but not providing any mechanism for run-time. Once the normative expectations and enforced context is gone, so is the context. The context of the request r is represented in this paper by the vector $\kappa = (t, u, d, p, m, \varrho)$ composed of the tenant, the user role, the data-sensitivity class, the declared purpose, the requested model and a risk score, which is a modelling device introduced in this paper.

4.2 Input Threats and Defenses (RQ2)

Early on, direct attacks were demonstrated: hand-crafted inputs show that goal hijacking and leaking is possible with GPT-3 [13]. Wei et al. explained that a successful jailbreak was due to conflicting training goals and mismatched generalization [18] and automatically created adversarial suffixes by Zou et al. that generalized across models [19]. In enterprises, where the attacker does not touch the system, indirect injection is more suitable. Greshake et al. have shown that instructions can be seamlessly integrated into data, which can subsequently impact LLM-integrated applications remotely [20]. Liu et al. formalized these attacks, and evaluated five attacks against ten LLMs and seven tasks, where they found that no existing defenses proved successful [21].

Hines et al. however have more positive results, by delimiting, datamarking or encoding the text that is to be attacked, the attack success in GPT-family models is reduced to less than 2%. The two findings are not mutually exclusive. A large number of documents containing a single keyword payload was used in the synthetic corpus and has been shown to be a simple attack that is easy to evaluate in terms of success [15] while Liu et al. employed a larger attack set that contains combined attacks [21]. The cross study reading is that the use of provenance marking is a good baseline but not a guarantee. A gateway is the typical place where it can be applied uniformly as it gathers or sends out the last prompt for all tenants.

The issue of leakage does not just apply to the prompt. Verbatim training sequences can be mined from GPT-2 [22] and leakage of the private retrieval database can be increased by RAG systems [14] while leakage of the training data can be decreased. The retrieval store contains tenant data and the cross tenant exposure is the retrieval scoping that the gateway has to enforce or verify.

4.3 Output Governance (RQ2)

The two sets of complementary output controls of Llama Guard are an instruction-tuned version of the Llama 2-7B

in service received by two backlogged clients [5]. The design of the vendor practice is somewhere in the middle of these extremes: Azure requires that requests limit the number of tokens they contain, and it pre-computes elements of the prompt for the tokens model in which the taxonomy classifies both prompts and responses, and the taxonomy can be tailored through zero- or few-shot prompting, with the authors having the model achieve at least the same accuracy on the OpenAI Moderation dataset and ToxicChat [23]. NeMo Guardrails, however, provides model independent rails which can be programmed and executed by a dialogue-management runtime [16]. One important property for multi-tenancy is the ability to be configured - in this kind of application, different policies are implemented through a shared gateway using per-tenant taxonomies or rails. None of these two, however, provide any indication of the latency or false positive cost to classifiers cascaded in a high throughput gateway. At the time of this writing, output governance is not included in the published gateways' blueprints, nor is it listed in the AWS reference design list of topics, suggesting that it was still not part of published AWS reference designs.

4.4 Fairness, Quotas and Cost (RQ3)

The most popular way to control is per-request rate limits which reduces capacity and affects the client experience if there's any capacity remaining, according to Sheng et al. [4]. They have a virtual token counter (VTC) scheduler that simply charges for the number of input/output tokens, and are work-conserving and have a tight 2× upper bound on the difference that exist in the request, and then processes an inappropriate request; AWS is bandwidth throttle, it adds the cost of the token used in the request at the end of the day. Both the accounting types reduce to the token-weighted cost in (1) which is similar to the cost function in [5] and [1]:

$$c(r) = w_{\text{in}} \cdot n_{\text{in}}(r) + w_{\text{out}} \cdot n_{\text{out}}(r) \tag{1}$$

The n(r) and nout(r) are the number of tokens received by and sent from the server by the request r, and the win and wout are the prices (or coefficients) of the server cost of a single token, respectively. With the admission control function for tenant t over a sliding window Wt with a budget Bt now written as: (2) for the first time,

$$\sum_{r' \in W_t} c(r') + \hat{c}(r) \leq B_t \tag{2}$$

In the case of ĉ(r), the number of elements as input to the request is exactly the number of elements that the gateway will be able to compute before forwarding [2] and is bounded by the maximum-token parameter of the request. There's another lever and that is cost aware routing. FrugalGPT achieved up to 98% cost savings compared to the best individual commercial LLM when using an LLM cascade [17] and RouteLLM optimized between a strong and a weak model for free using preference data and saved more than 50% costs without decreasing the quality of the responses [6]. Both routers do not refer to the tenant policy. A context-aware gateway should first reduce the candidate set in which the models that κ belong to are located, as represented in (3):

$$m*(r) = \arg\min_{m \in \mathcal{M}(\kappa)} \pi_m \hat{c}(r) \;\; \text{s.t.} \;\; q_m(x) \geq \tau_\kappa \tag{3}$$

where $\mathcal{M}(\kappa)$ is the set of models that the PDP permits and πm is the per-token price of model m, qm(x) is a router's predicted quality of prompt x, and τκ is a quality floor for a tenant. In addition

to the binary strong/weak routing of [6] the policy restriction is added to make it generalizable to Equation (3). If $\mathcal{M}(\kappa)$ is empty then the request is denied, not routed.

4.5 Shared State and Cross-Tenant Isolation (RQ4)

Shared state is used at two layers: shared KV blocks are used between requests in the serving engine as in vLLM [9]; semantic caches re-use KV completions for similar prompts in the gateway [2]. Song et al. showed both timing side channels and cache sharing in commercial LLM services [5] and demonstrated timing side channels in both to infer system prompts and other users' prompts. This is in direct contrast to pooled efficiency. The gateway itself fully controls the semantic cache that it runs, and can offer a service that enables limiting the set of entities that can be looked up to just those entries created by the same tenant. A hit is the existence of such an embedding – response pair in the storage set of tenant t, Let St denote the embedding – response pairs stored by tenant t; then a hit is (4):

$$\text{hit}(x,t) \Leftrightarrow \max_{(e',y')\in S_t} \cos(e(x),e') \geq \theta \quad (4)$$

e(·) is an embedding function and θ is a similarity threshold. The trade-off of the lower hit rates when the cross-tenant channel is removed from the gateway cache is not quantified in any of the sources in the corpus by partitioning. There are KV prefix caches that are required on the provider end, out of the gateway control, which require contractual guarantees or which are expected to route sensitive KV tenants to siloed endpoints (per tenant tier bridge model) [8]. Both, multi-adapter serving shares [24] and [10] made a compromise on adapaters weights and batch requests across adapters was out of scope for the works concerning security isolation between adapters.

4.6 Reference Model and Comparative Analysis

4.6.1 Six-Layer Reference Model

All synthesis is compiled in a conceptual model in Figure 2, but would not be implemented and tested in this work. The layer L1 authenticates a caller and collects κ; L2 applies input governance (including redaction of sensitive data, and marking the provenance of untrusted input); L3 admission according to (2) and routing according to (3); L4 adds scope to shared state according to (4) and adds retrieval boundaries; L5 adds output classification according to (2) and writes audit evidence; L6 applies metering according to (1) and writes audit evidence. The separation of policy decisions from the data path is as in [7] realised in a PDP outside the data path. Figure 3 lists eight threats to these layers, which is an analytical judgement and not a measurement.

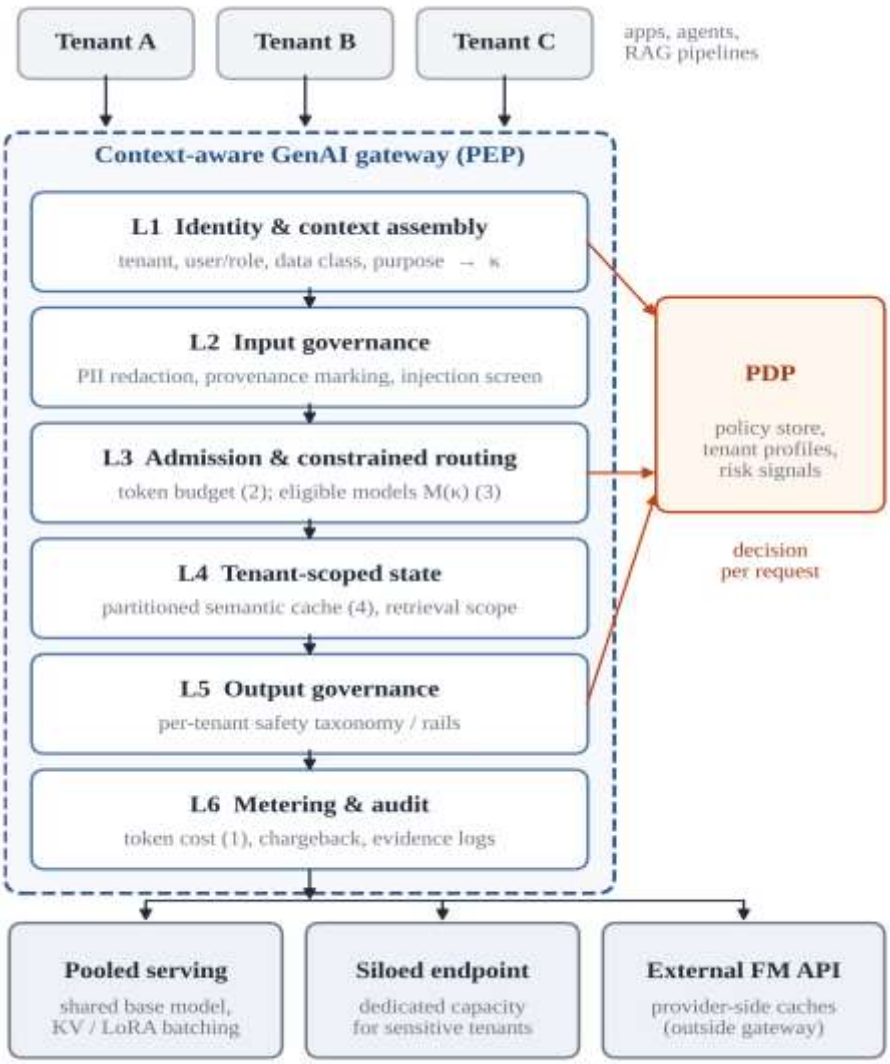


Figure 2. Conceptual six-layer reference model of a context-aware GenAI gateway acting as a policy enforcement point (PEP) with an external policy decision point (PDP). The diagram is a synthesis of the reviewed literature, not a measured system.

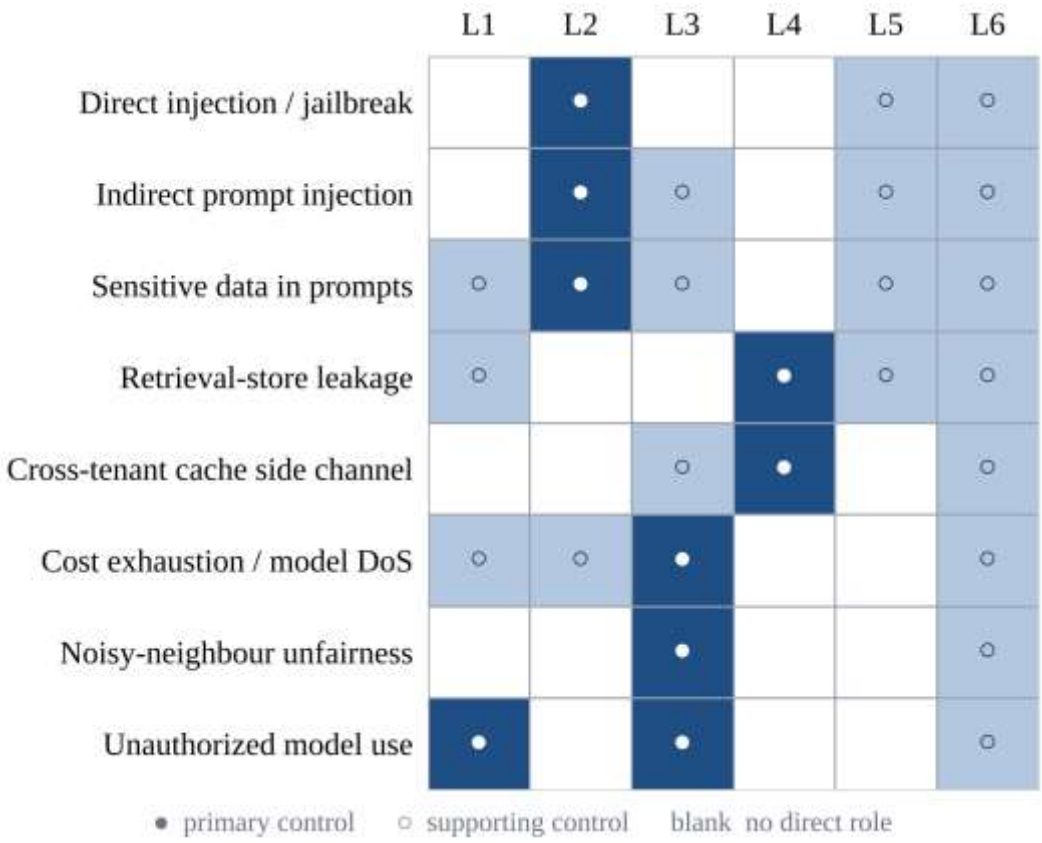


Figure 3. Analytical mapping of threats to gateway layers L1–L6 of Figure 2. Markings reflect the authors' judgment based on Section 4, not empirical measurement.

4.6.2 Quantitative Evidence and Its Comparability

The quantitative results which were verifiable are listed in Table 2. Not to be compared to one another! There are different baselines, models and hardware [9], [10] used to measure throughput gains of vLLM, S-LoRA and Punica; the model price ratio, as well as the benchmark [6] determine the cost reduction of FrugalGPT and RouteLLM; and the attack success rate is heavily dependent on attack corpus [21] and [15]. The overhead of the full model in Figure 2 is not reported for any source, so end to end gateway latency is not known with the full model.

Table 2. Verified Quantitative Results and Their Evaluation Conditions

Work	Reported result	Condition / baseline
vLLM [9]	2–4× serving throughput	Same latency; vs. FasterTransformer and Orca
S-LoRA [24]	Up to 4× throughput; adapters served increased by orders of magnitude	Vs. HuggingFace PEFT and vLLM with naive LoRA; Llama 7B–70B
Punica [10]	12× throughput; +2 ms per token	Fixed-size GPU cluster; vs. state-of-the-art LLM serving systems
VTC [4]	Tight 2× bound on service difference	Analytical guarantee for two backlogged clients; work-conserving

FrugalGPT [17]	Up to 98% cost reduction at best-LLM accuracy, or +4% accuracy at equal cost	LLM cascade over commercial APIs
RouteLLM [6]	More than 2× cost reduction without quality loss	Strong/weak model pair; public benchmarks
Spotlighting [15]	Attack success >50% → <2%	GPT-family models; synthetic 1,000-document keyword-payload corpus
Liu et al. [21]	No defense sufficient	5 attacks, 10 defenses, 10 LLMs, 7 tasks

Note: Values as reported by the original authors; none were recomputed. Rows are not mutually comparable.

4.6.3 Coverage of Vendor Designs

Table 3 presents functions identified in the two sources vs. functions identified in the research literature. The two designs allow tenant identification and token metering [1], [2]. The control mechanisms proposed in the research literature that are said to be required for both source documents are missing: injection screening, output classification, policy-constrained routing and tenant-partitioned caching. If there is no cited document, then there is not a lack of a product; this is made up for in Section 4.7.

Table 3. Functions Documented in Vendor Gateway Sources

Function	AWS [1]	Azure [2]	Research basis
Tenant identification	API key per team	Subscription-keyed counter	PEP/PDP [7]
Model allow-listing	IAM policy	N.D.	Eq. (3)
Token quota	Request throttling; token logging	Token-limit policy	VTC [4]
Metering / chargeback	Daily per-team cost	Token-metric policy	Eq. (1)
Semantic cache	N.D.	Yes (preview)	Risk: [5]
Injection / content filtering	Future work	N.D.	[15], [16]
Cost-aware routing	N.D.	N.D.	[6], [17]
Private connectivity	AWS PrivateLink	N.D.	—

Note: N.D. = not documented in the cited source; this does not establish that the product lacked the function.

Layered controls are said to compound. The probability of it being bypassed is (5) if the probability of attack on layer i is di and attacks are independent.

$$P_{\text{bypass}} = \prod_{i=1}^{L}(1 - d_i) \tag{5}$$

Independence is optimistic. Common weaknesses in adaptive attacks against LLM defenses are found, and the fact that all the defenses tested failed to succeed [21] implies some degree of failure correlation. The assumption of positive correlation between the miss probabilities would result in a higher value of Prob (GOOD) than the value calculated in (5); hence (5) is considered to be a minimum value.

4.7 Research Gaps and Future Directions

Six gaps were summarized in the Table 4. The most critical is end-to-end evaluation without: single controls get tested separately [15] and [23] and gateways connect them in production. The second gap has to do with context integrity. While the NIST profile does not require/prioritize purpose and data-sensitivity labels, it does require them in risk management [12]. Thirdly, issues of adversarial fairness. Clients in VTC and learned routers are not considered to be playing the cost function and attacks on a fair scheduler that exhaust the tokens are not explored. Finally, agentic workloads are more than just the presence of prompts to tool calls, they also involve gateway mediation of tool calls also listed by Greshake et al. as a consequence of indirect injection [20] but not considered in the corpus.

Table 4. Research Gaps Identified in the Corpus

Gap	Evidence	Suggested direction
G1 End-to-end evaluation	Controls tested in isolation [15], [23]	Benchmark latency, false positives and residual risk of chained controls
G2 Cache partitioning cost	Channel shown [5]; hit-rate impact unreported	Measure hit-rate loss under (4) across tenant counts
G3 Context integrity	Profile expects context [12]; no verification mechanism found	Signed context attestations; data-class detection at L2
G4 Adversarial fairness	Schedulers assume honest clients [4]	Token-exhaustion attacks against fair schedulers and routers
G5 Agentic tool calls	Unauthorized API calls listed [20]	Gateway mediation and authorization of tool invocations
G6 Multi-adapter isolation	Isolation outside scope [24], [10]	Security analysis of batched multi-LoRA serving

4.8 Limitations and Threats to Validity

There were some restrictions to these conclusions. The population is small in size, built up by Web search and not from a thorough search of a database, and thus is likely to consist of well cited and publicly available sources. At the time of writing, there were seven sources that were preprints, including side-channel study [5] and spotlighting evaluation [15]. Sources' own summaries were not re-executed with respect to the quantitative values. The vendor documentation changed over time and previous announcements and blog posts were selected, with anything which was not explicitly mentioned in them being considered as undocumented, but not missing. The reference model and (2)–(5) are analytical models which have not been validated by experiments; the threat mapping in Figure 3 is based on the judgment of other analysts.

CONCLUSION

With RQ1, only few pieces of context are identified by published designs to be propagated and a context vector has to be built at the gateway, which is then evaluated by an external PDP to decide per request. While useful and necessary, RQ2 is not enough, as shown by the evidence in the

benchmarks, that current injection defenses do not work on a wider range of attacks. For RQ3, accounting and fair scheduling are much more achievable than rate limits on the requests, and the cost-aware routing for RQ3 should be limited to policy-eligible models. The shared caches are the proven cross tenant channels for RQ4, and the gateway can partition the caches that it owns, but provider-state must be siloed or contractually guaranteed. The 6 layer model and its formal expressions offer a foundation for implementation and for the end-to-end evaluations which were not available in the literature.

REFERENCES

[1] H. Poonawala, B. Pistone, V. Pandey, and A. Tzeveleka, "Build an internal SaaS service with cost and usage tracking for foundation models on Amazon Bedrock," 2024. [Online]. Available: https://aws.amazon.com/blogs/machine-learning/build-an-internal-saas-service-with-cost-and-usage-tracking-for-foundation-models-on-amazon-bedrock/

[2] Microsoft, "Introducing GenAI gateway capabilities in Azure API Management," 2024. [Online]. Available: https://techcommunity.microsoft.com/blog/azureintegrationservicesblog/introducing-genai-gateway-capabilities-in-azure-api-management/4146525

[3] O. Foundation, "OWASP Top 10 for Large Language Model Applications," 2023. [Online]. Available: https://owasp.org/www-project-top-10-for-large-language-model-applications/

[4] Y. Sheng *et al.*, "Fairness in Serving Large Language Models," in *18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24)*, 2024. [Online]. Available: https://www.usenix.org/conference/osdi24/presentation/sheng

[5] L. S. et al., "The Early Bird Catches the Leak: Unveiling Timing Side Channels in LLM Serving Systems," 2024. doi: 10.48550/arXiv.2409.20002.

[6] I. O. et al., "RouteLLM: Learning to Route LLMs with Preference Data," 2024. doi: 10.48550/arXiv.2406.18665.

[7] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero Trust Architecture," 2020. doi: 10.6028/NIST.SP.800-207.

[8] A. W. Services, "SaaS Tenant Isolation Strategies: Isolating Resources in a Multi-Tenant Environment," 2020. [Online]. Available: https://docs.aws.amazon.com/whitepapers/latest/saas-tenant-isolation-strategies/

[9] W. K. et al., "Efficient Memory Management for Large Language Model Serving with PagedAttention," in *ACM Symposium on Operating Systems Principles (SOSP)*, 2023. doi: 10.1145/3600006.3613165.

[10] L. Chen, Z. Ye, Y. Wu, D. Zhuo, L. Ceze, and A. Krishnamurthy, "Punica: Multi-Tenant LoRA Serving," in *Proceedings of Machine Learning and Systems, vol. 6*, 2024. [Online]. Available: https://proceedings.mlsys.org/

[11] P. L. et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Advances in Neural Information Processing Systems, vol. 33*, 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020

[12] N. I. of S. and Technology, "Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile," 2024. doi: 10.6028/NIST.AI.600-1.

[13] F. Perez and I. Ribeiro, "Ignore Previous Prompt: Attack Techniques for Language Models," 2022. doi: 10.48550/arXiv.2211.09527.

[14] S. Zeng *et al.*, "The Good and The Bad: Exploring Privacy Issues in Retrieval-Augmented Generation (RAG)," in *Findings of the Association for Computational Linguistics: ACL 2024*, 2024. doi: 10.18653/v1/2024.findings-acl.267.

[15] K. Hines, G. Lopez, M. Hall, F. Zarfati, Y. Zunger, and E. Kiciman, "Defending Against Indirect Prompt Injection Attacks With Spotlighting," 2024. doi: 10.48550/arXiv.2403.14720.

[16] T. Rebedea, R. Dinu, M. Sreedhar, C. Parisien, and J. Cohen, "NeMo Guardrails: A Toolkit for Controllable and Safe LLM Applications with Programmable Rails," in *EMNLP 2023 System Demonstrations*, 2023. doi: 10.18653/v1/2023.emnlp-demo.40.

[17] L. Chen, M. Zaharia, and J. Zou, "FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance," 2023. doi: 10.48550/arXiv.2305.05176.

[18] A. Wei, N. Haghtalab, and J. Steinhardt, "Jailbroken: How Does LLM Safety Training Fail?," in *Advances in Neural Information Processing Systems, vol. 36*, 2023. doi: 10.52202/075280-3508.

[19] A. Zou, Z. Wang, N. Carlini, M. Nasr, J. Z. Kolter, and M. Fredrikson, "Universal and Transferable Adversarial Attacks on Aligned Language Models," 2023. doi: 10.48550/arXiv.2307.15043.

[20] K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz, and M. Fritz, "Not What You've Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection," in *16th ACM Workshop on Artificial Intelligence and Security (AISec)*, 2023. doi: 10.1145/3605764.3623985.

[21] Y. Liu, Y. Jia, R. Geng, J. Jia, and N. Z. Gong, "Formalizing and Benchmarking Prompt Injection Attacks and Defenses," in *33rd USENIX Security Symposium*, 2024. [Online]. Available: https://www.usenix.org/conference/usenixsecurity24

[22] N. C. et al., "Extracting Training Data from Large Language Models," in *30th USENIX Security Symposium*, 2021. [Online]. Available: https://www.usenix.org/conference/usenixsecurity21

[23] H. I. et al., "Llama Guard: LLM-based Input-Output Safeguard for Human-AI Conversations," 2023. doi: 10.48550/arXiv.2312.06674.

[24] Y. S. et al., "S-LoRA: Serving Thousands of Concurrent LoRA Adapters," in *Proceedings of Machine Learning and Systems, vol. 6*, 2024. [Online]. Available: https://proceedings.mlsys.org/paper_files/paper/2024/hash/906419cd502575b617cc489a1a696a67-Abstract-Conference.html