International Journal of Advance Industrial Engineering
©2025, All Rights Reserved

E-ISSN 2320 –5539
Available at http://ijaie.com/

Research Article

Securing enterprise LLM gateways & misconfigured proxy exposure detection + mitigation framework for exposed LLM endpoints, API gateways, model download hooks

Ankita Sharma
TSB Bank, London

ABSTRACT

The quick process of enterprise adoption of large language models (LLMs) via API gateways, reverse proxies, and orchestration layers has enabled a novel category of security threats related to architecture misconfigured as opposed to model-specific behavior. Open LLM endpoints, permissive proxy settings, and unprotected model download handles are facilitating more and more data leakage, unauthorized inference, supply-chain failure and trust boundary breaches. This paper describes a system level detection and mitigation system framework to secure enterprise LLCM gateways based on the co-design principles of cyber-physical systems, embedded security, and safety-critical architectures.

The framework is both based on semantic configuration analysis and runtime monitoring, and secure-by-design architecture patterns to highlight early exposure detection and design-time mitigation of the gateway. The paper shows how the knowledge of cross-domain automotive, IoT, and CPS security can be used to design resilient LLM infrastructure to enable scalable, trustworthy, and governable enterprise AI deployments.

Keywords: *Huge Language Models; API Gateways; Proxy Misconfiguration; AI Infrastructure Security; System-Level Co-Design; Supply Chain security; Enterprise AI Governance*

INTRODUCTION

The blistering pace at which large language models (LLMs) are being embraced by enterprises has fundamentally changed how organizations manage knowledge, automate, support decision-making, and interact with humans. These models are implemented using application programming interfaces (APIs), reverse proxies, and centralized Akaka Luoghorngu gateways, and are being incorporated into business-critical enterprise processes, such as customer service and software development, security operations and planning. Although these gateway-based architectures facilitate scalability, observability and enforcement of policies, they also bring forth novel and uncharted security threats of malconfigured proxies, open endpoints, and implicit trust relationships between multifaceted system tiers.

Systematically, enterprise LLM gateways are similar to large-scale cyber-physical and embedded systems, where employing tightly-coupled interactions among software services, hardware resources, networks, and control logic emerges as functionality. It has been long established by previous studies in system-level design and hardware-software co-design that security and safety vulnerabilities are not always a result of the individual components, but rather of architectural incompatibilities, interface ambiguities, or configuration mistakes across abstraction layers (Sangiovanni-Vincentelli, 2007; Teich, 2012; Abdallah et al., 2010). Such observations are becoming more applicable to infrastructures based on LLM, in which gateways serve as valuable points of coordination between users, models, external services and data sources.

The recent literature on AI-powered and cyber-physical systems indicates that the traditional perimeter-focused security models do not fit the complex, distributed infrastructure, which is founded on the orchestration of systems and relies on shared trust domains (Seshia et al., 2016; Wan, 2018). Misconfigured API gateways and proxies, used in enterprise deployments of the LLM, may accidentally reveal model endpoints to an unauthenticated user, facilitate immediate and data exfiltration, or allow unsanctioned behavioral manipulation of the model using unverified inputs and downstream integrations. These risks are compounded by the fact that the deployment is a fast-paced process, the deployment of generic cloud-native proxy settings is reused, and the company does not have much experience with AI-specific threat models.

The difficulty is also exacerbated by the increased use of third-party model artifacts, plugin ecosystems,

*Author for Correspondence: mail id ankita.sharma@tsb.co.uk
DOI: https://doi.org/10.14741/ijaie/v.12.4.1

and automated model download hooks, which present supply-chain vulnerabilities similar to the ones presented by embedded systems and IoT systems (Chaduvula et al., 2018; Kandah et al., 2019). The study of security and trust in large-scale connected systems proves that the identity binding weak, the lack of integrity verification and the lack of opaque update mechanisms can compromise the resilience of the system, despite the formal security of its individual components (Suhail et al., 2020; Gossner and Duvvury, 2015). The problems in the context of LLM gateways are the unverified updates in models, unstable dependency resolution, and too much privilege of the gateway, which removes the distinction between trust and mistrust.

The new area of work on safety-security co-design, semantically rich architecture patterns offers a promising basis on which to overcome these challenges. Co-design methods also permit systematic identification and alleviation of vulnerabilities introduced through configuration, unlike a system where security is viewed as an afterward control measure (Dantas and Nigam, 2023; Dunnihoo, 2015). The same principles have been effectively implemented in the automotive, aerospace, or embedded industries, with the current best practice being a fail-operational architecture and integrated safety-security reasoning (Chen, 2008; Kohn et al., 2016; Wang et al., 2023).

It is against this background that this article will look at security risks of poorly configured enterprise LLM gateways and proxy infrastructure with an especial focus on exposed LLM endpoints, API gateways, and model download hooks. It promotes a detection and mitigation model based on system-level and hardware-software co-architecture, cross-domain lessons learned within cyber-physical systems, embedded security and scalable AI architecture. The article provides a more resilient and visionary way of securing enterprise AI implementations in increasingly complex and interconnected settings by presenting the framing of LLM gateway security as an architectural and co-design issue, and not a purely operational one.

Architectural Overview of Enterprise LLM Gateways

The use of large language models (LLMs) by an enterprise has led to the accelerated development of specialized architectural layers that mediate the access of organizational applications with foundation models. These layers are more generally known as enterprise LLM gateways, which are control planes, controlling routing, authentication and policy enforcement, observability and integration with broader digital infrastructure. Systemically, LLM gateways are similar to intricate cyber-physical and embedded systems architectures where heterogeneous building blocks have to be simultaneously optimised in performance, safety and security. In turn, the knowledge of their architectural structure is a condition to knowing the risks of misconfiguration and developing effective detection and mitigation measures (Sangiovanni-Vincentelli, 2007; Teich, 2012; Seshia et al., 2016).

Conceptual Role of LLM Gateways in Enterprise Systems

Conceptually, an enterprise LLM gateway is a mediation abstraction in between client-facing applications and backend model providers. Such abstraction separates application logic and model specific interfaces, permitting vendor and deployment mode portability, such as cloud-hosted, on-premise and hybrid configurations. As with the case of hardware-software co-design abstractions, the gateway ensures isolation of concerns and allows coordinated multilayer optimization (Abdallah et al., 2010; Morshed and Morshed, 2021).

Architecturally, the gateway gathers the control over some sensitive functions in the form of credential management, rate limiting, and instant preprocessing. Nevertheless, such centralization also brings the issues

Table 1: Architectural Components, Functions, and Security Implications of Enterprise LLM Gateways

Architectural Layer	*Primary Function*	*Typical Technologies*	*Key Trust Assumptions*	*Common Misconfiguration Risks*	*Security Implications*
API Ingress Layer	Request intake and normalization	API gateways, ingress controllers	Auth enforced at edge	Open routes, weak auth	Unauthorized LLM access
Reverse Proxy Layer	Traffic routing and load balancing	NGINX, Envoy	Correct rule ordering	Overly permissive rules	Endpoint exposure
Policy Engine	Prompt and response enforcement	Rule engines, semantic filters	Policies up to date	Policy drift	Data leakage
Identity & IAM	Authentication and authorization	OAuth, mTLS	Trusted identity source	Token reuse	Privilege escalation
Observability Layer	Logging and monitoring	SIEM, telemetry agents	Logs complete	Blind spots	Undetected abuse
Model Orchestration	Model selection and routing	Orchestrators, SDKs	Trusted model sources	Insecure hooks	Supply-chain compromise

of single points of failures and concentration of trust, which are issues extensively reported in the design of systems at the level of a CPS (Dunnihoo, 2015; Wan, 2018). When organizations expand the use of LLC in various departments, the gateway is an important infrastructural element that can spread systemic risk when it is improperly configured.

Enterprise LLM Gateway Conceptual Architectural Components.

Enterprise LLM gateways are normally made up of a number of closely knitted parts which take care of a specific issue of operation. These are API ingress controllers, reverse proxies, authentication and authorization services, policy engines, logging and telemetry modules and model orchestration interfaces. These components have a reflection of the layered embedded and automotive architectures, in which the safety- and security-critical functions are spread across software and hardware boundaries (Chen, 2008; Kohn et al., 2016).

In practice API gateways and reverse proxies are used as the initial defense, which do request normalization and routing. Prompts and responses are then subjected to policy engines which perform semantic and syntactic validation on them and observability components gather runtime metrics to aid in auditing and anomaly detection. Model orchestration layers handle the access to internal or third parties' models, such as versioning and fallback. These components pose an architectural challenge when they are deployed in a heterogeneous environment, where configuration drift is more likely, and there are inconsistent trust assumptions (Cauwenberghs et al., 2023; Wesling, 2020).

Trust Boundaries and Deployment Models.

Enterprise LLM gateways are implemented with a variety of models such as centralized cloud-native service to distributed, edge-enabled architecture. The centralized deployments are easy to govern, but put the risk all in a single place, the distributed deployments enhance the resilience by exchanging complexity of coordination. These trade-offs are similar to large-scale application of IoT and CPS, where the scope of trust is dynamic and is reconfigured (Kandah et al., 2019; Gogineni et al., 2023).

The problem of trust boundaries in LLM gateway architectures is especially problematic because of the presence of human generated inputs, machine generated prompts and automated downstream actions. Mismatch between the perceived and the actual trust boundaries may reveal internal endpoints or avoid authentication systems. Previous studies in the area of CPS security stress that these boundary breaches tend to be more of an architectural, as opposed to a technical, failure (Chaduvula et al., 2018; Zardini, 2023).

Enterprise Identity, Security, and Compliance Layer Integration.

LLM gateways are hardly used in a vacuum and tend to be heavily integrated with enterprise identity and access management (IAM), security information and event management (SIEM) and frameworks of compliance. Through such an integration, centralized implementation of organizational policies is possible but it also generates long chains of dependencies. Aerospace, automotive, and embedded firewall co-design experience points to the fact that security controls need to be formally modeled on the architectural level to avoid emergent vulnerabilities (Pesé et al., 2017; Wang et al., 2023).

Cryptographic identity binding, mutual authentication and key lifecycle management are getting increasingly used to reduce unauthorized access. These strategies are based on the existing practice of IoT and CPS trust

Table 2: LLM Gateway Security Components, Threats, and Mitigation Principles

Component	*Threat Category*	*Attack Vectors / Threat Actors*	*Potential Impacts*	*Mitigation Principles*
API Gateway	Exposure / Misuse	Open routes, misconfigured auth, bypassed rate limits (external or insider)	Unauthorized access, data leakage, model abuse	Zero-trust, RBAC, rate limiting, API key rotation, traffic inspection
Reverse Proxy	Exposure / Integrity	Default-open routing, request smuggling, route tampering	Direct LLM access, tampered responses, privilege escalation	Hardened config, access control, encrypted channels, route validation
Service Mesh	Exposure / Data Leakage	Misconfigured sidecars, permissive inter-service communication	Unauthorized service access, internal data leaks	Mutual TLS, service identity enforcement, monitoring
Model Endpoints	Misuse / Integrity	Unprotected REST/gRPC, high-volume or malicious queries, partial access	Resource exhaustion, inference manipulation, biased outputs	Endpoint hardening, authentication, rate limiting, anomaly detection
Download Hooks / Repositories	Exposure / Integrity	Open repo endpoints, unverified model artifacts, dependency injection	Malicious model deployment, IP theft, corrupted dependencies	Signed models, access control, automated CI/CD verification

architecture, such as identity supported by hardware and resilience-based architecture (Suhail et al., 2020; Gossner and Duvvury, 2015). The inability to properly implement them at the gateway level may lead to the exposure of APIs and insecure proxy routes.

Interfaces Model-Access, Orchestration and Supply-Chain.

One of the characteristics of enterprise LLM gateways is that they control the access to models and related artifacts. This involves forwarding of requests to other models, application of usage quotas and model download hooks or plug in interfaces. Systemically, these functions are similar to those found in embedded and micro/nano system design where supply-chain interface-defined interfaces are considered the most important (Cauwenberghs et al., 2023; TUD et al., 2022).

External repositories and third-party services are also likely to be exposed to model orchestration layers, creating more attack surfaces. Architectural validation has been found to be crucial in secure design automation and system-level verification research to avoid the compromising nature of such interfaces to the overall trust of the system (Seshia et al., 2016; Dantas and Nigam, 2023).

Risk of Architectural Complicatedness and Misconfiguration.

The risk of misconfiguration is increased as the sophistication of enterprise LLM gateway architectures increases. Functionality is layered between proxies, policies and orchestration services and as a result, unwanted interactions can occur resulting in exposed endpoints or bypassed controls. This effect is consistent with the literature on system-level design, where leakage of complexity and abstraction are some of the major contributors to security problems (Sangiovanni-Vincentelli, 2007; Teich, 2012).

In CPS and IoT systems, empirical research has shown instances where architectural blind spots are the causes of misconfigurations as opposed to implementation issues (Wan, 2018; Kandah et al., 2019). In this regard, the design of the secure LLM gateway requires the use of architectural clarity and explicit modeling of assumptions.

In summary, this section has examined the architectural foundations of enterprise LLM gateways, emphasizing their role as critical control points within complex, multi-layered systems. By situating LLM gateways within established system-level and co-design frameworks, it becomes evident that many security risks originate from architectural complexity and misaligned trust assumptions rather than isolated implementation errors. A rigorous architectural understanding is therefore essential for developing robust detection and mitigation frameworks capable of addressing misconfigured proxies and exposed LLM endpoints in enterprise environments.

Threat Model: Misconfigured Proxies and Exposed LLM Endpoints

Enterprise deployment of large language models (LLMs) increasingly relies on layered gateway architectures, reverse proxies, and API mediation services to manage access, scalability, and observability. While these components are designed to abstract complexity, they also introduce systemic security risks when misconfigured or insufficiently governed. Similar to cyber-physical and embedded systems, LLM infrastructures operate across multiple trust boundaries, where failures at architectural interfaces can cascade into large-scale compromise. This section develops a structured threat model for misconfigured proxies and exposed LLM endpoints, grounding the analysis in system-level co-design, CPS security, and hardware–software integration literature.

Architectural Attack Surfaces in Enterprise LLM Gateways

Enterprise LLM gateways typically sit between client applications and backend model services, performing routing, authentication, rate limiting, and policy enforcement. However, architectural heterogeneity arising from service meshes, container orchestration, hybrid cloud deployments, and third-party model providers creates fragmented control points. Prior system-level design research demonstrates that such fragmentation increases the likelihood of inconsistent policy enforcement and unintended exposure (Sangiovanni-Vincentelli, 2007; Teich, 2012).

Misconfigurations often emerge at proxy layers where default-open routing rules, disabled authentication modules, or overly permissive network access controls allow direct access to LLM inference endpoints. From a CPS perspective, these gateways resemble integration layers in automotive or aerospace systems, where failures at interfaces have historically led to safety and security violations (Abdallah et al., 2010; Chen, 2008). As with embedded architectures, the attack surface is not confined to software logic but spans configuration artifacts, orchestration metadata, and runtime communication channels (Seshia et al., 2016).

Threat Actors and Adversarial Capabilities

Threat actors exploiting exposed LLM gateways range from opportunistic attackers conducting automated scans to sophisticated adversaries targeting enterprise AI infrastructure for data exfiltration or model abuse. In line with IoT and CPS threat models, adversaries may possess partial system knowledge, leveraging misaligned trust assumptions between gateway components (Chaduvula et al., 2018; Kandah et al., 2019).

Capabilities include crafting unauthorized inference requests, bypassing authentication through misrouted proxy paths, and abusing internal service-to-service trust relationships. These capabilities mirror those observed in large-scale distributed CPS environments, where identity and trust mismanagement amplify systemic risk (Suhail et al., 2020). Importantly, attackers need not compromise the LLM itself; control-plane weaknesses alone can enable high-impact misuse.

Attack Vectors Enabled by Proxy Misconfiguration

Misconfigured proxies enable several distinct but interrelated attack vectors. These include unauthenticated endpoint exposure, request smuggling across trust zones, and privilege escalation via improperly scoped API keys. Similar vulnerabilities have been observed in automotive embedded firewalls and CPS gateways, where configuration errors undermined otherwise robust security mechanisms (Pesé et al., 2017; Kohn et al., 2016).

From a co-design standpoint, these attack vectors arise when security is treated as an add-on rather than a design constraint. System-level co-design literature emphasizes that performance optimization, scalability, and security must be jointly considered, as isolating one objective often degrades the others (Dunnihoo, 2015; Morshed & Morshed, 2021). In LLM infrastructures, aggressive latency optimization can inadvertently disable inspection or validation layers, increasing exposure risk.

Cascading and Systemic Impacts of Exposed LLM Endpoints

Exposed LLM endpoints can trigger cascading failures across enterprise systems. Unauthorized access may lead to sensitive prompt leakage, indirect disclosure of proprietary data, or abuse of downstream integrations. CPS research has long shown that local failures can propagate through tightly coupled systems, resulting in disproportionate impact (Wan, 2018; Zardini, 2023).

In AI-enabled enterprises, such cascades may include excessive compute consumption, regulatory non-compliance, and erosion of organizational trust. These risks parallel fail-operational concerns in automotive systems, where maintaining functionality under partial compromise is critical (Kohn et al., 2016). Without architectural containment mechanisms, LLM gateway breaches can undermine both operational resilience and strategic AI adoption.

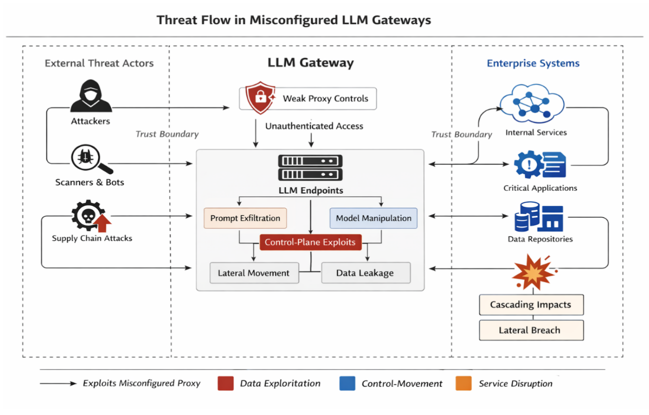

Figure 1: Conceptual Threat Flow Diagram for Misconfigured LLM Gateways

Trust Boundary Violations and Supply-Chain Exposure

Beyond runtime inference, exposed gateways often interact with model download hooks, plugin ecosystems, and external artifact repositories. Trust boundary violations at these interfaces introduce supply-chain risks analogous to those identified in micro/nano system integration and embedded update mechanisms (Cauwenberghs et al., 2023; Gossner & Duvvury, 2015).

If proxies fail to enforce provenance or integrity checks, adversaries may inject malicious models or tampered dependencies. Such risks echo broader concerns in future connectivity systems, where heterogeneous integration complicates end-to-end security assurance (TUD et al., 2022). Consequently, LLM threat modeling must extend beyond inference endpoints to encompass the full lifecycle of model acquisition and deployment.

In sum, this section has demonstrated that misconfigured proxies and exposed LLM endpoints represent a systemic security threat rooted in architectural complexity and insufficient co-design. Drawing from CPS, embedded, and system-level design literature, the threat model highlights how interface-level failures, trust boundary violations, and configuration errors enable high-impact attacks without direct model compromise. Addressing these risks requires treating LLM gateways as critical system components, where security properties are designed, verified, and maintained across the entire enterprise AI architecture rather than enforced in isolation.

Detection Framework for LLM Gateway Exposure

Enterprise deployment of large language models (LLMs) increasingly relies on gateway architectures that mediate access between users, applications, and model backends. While these gateways provide scalability, observability, and policy enforcement, they also introduce new vectors for security exposure when misconfigured or insufficiently monitored. Misconfigured proxies, overly permissive routing rules, unsecured API endpoints, and unverified model download hooks can enable unauthorized access, data leakage, and systemic misuse of LLM capabilities.

This section presents a comprehensive detection framework for identifying and analyzing LLM gateway exposure within enterprise environments. Drawing on system-level design, cyber-physical systems (CPS) security, and hardware–software co-design literature, the framework emphasizes early detection, semantic awareness, and architectural consistency across heterogeneous AI infrastructures (Sangiovanni-Vincentelli, 2007; Teich, 2012; Seshia et al., 2016). The proposed approach integrates static, dynamic, and semantic detection layers to address the complexity and scale of modern enterprise LLM deployments.

Architectural Attack Surfaces in LLM Gateways

LLM gateways form a critical mediation layer that aggregates API traffic, authentication services, rate limiting, logging, and model orchestration. From a system-level perspective, these gateways resemble embedded and CPS control layers, where faults or misconfigurations propagate rapidly across dependent subsystems (Abdallah et al., 2010; Wan, 2018).

Key architectural attack surfaces include exposed REST or gRPC endpoints, reverse proxy misrouting, insufficient isolation between tenants, and implicit trust in downstream model services. Similar to automotive and aerospace systems, the interaction between software control logic and infrastructure configuration creates emergent vulnerabilities that cannot be detected through component-level analysis alone (Chen, 2008; Kohn et al., 2016). Effective detection therefore requires visibility into both logical policy definitions and runtime execution paths.

Static Configuration and Policy Analysis

Static detection constitutes the first layer of the framework and focuses on identifying insecure configurations before deployment. This includes analyzing gateway configuration files, proxy routing rules, authentication policies, and access control lists for violations of security best practices. Techniques adapted from hardware–software co-design and design automation enable systematic verification of architectural constraints and trust boundaries (Teich, 2012; Seshia et al., 2016).

Static analysis is particularly effective at detecting overly permissive endpoint exposure, missing authentication requirements, insecure default settings, and undocumented model access routes. By treating LLM gateways as system-level artifacts rather than isolated software components, static analysis tools can uncover architectural inconsistencies that mirror failures observed in CPS and IoT deployments at scale (Chaduvula et al., 2018; Kandah et al., 2019).

Runtime Monitoring and Behavioral Detection

While static analysis addresses design-time risks, runtime detection is essential for identifying dynamic misconfigurations and active exploitation. This layer monitors live gateway traffic, request patterns, and model invocation behaviors to detect anomalies indicative of exposure or misuse. Drawing from CPS monitoring and resilience engineering, runtime detection emphasizes deviation from expected operational envelopes rather than signature-based threat identification (Dunnihoo, 2015; Zardini, 2023).

Techniques such as traffic profiling, rate anomaly detection, and trust-graph validation enable continuous assessment of gateway behavior under real workloads. In large-scale deployments, federated and distributed monitoring approaches are necessary to maintain visibility without introducing prohibitive overhead (Gogineni et al., 2023). These methods parallel runtime assurance strategies used in automotive safety systems and embedded firewalls (Pesé et al., 2017).

Semantic and Pattern-Based Validation

Beyond syntactic correctness, secure LLM gateway operation depends on semantic alignment between architectural intent and system behavior. This subsection introduces semantic detection mechanisms that validate gateway configurations against predefined security and safety patterns. Inspired by semantically rich architecture patterns, this approach encodes design knowledge into reusable templates that capture correct trust relationships, access flows, and failure handling strategies (Dantas & Nigam, 2023).

Semantic validation enables detection of subtle misconfigurations that pass static checks but violate higher-level design principles, such as unintended privilege escalation paths or inconsistent enforcement of identity policies. Similar approaches have proven effective in system-level design reasoning and CPS verification, where correctness emerges from coordinated component behavior rather than isolated compliance (Sangiovanni-Vincentelli, 2007; Seshia et al., 2016).

Integration with Supply-Chain and Model Integrity Monitoring

LLM gateways often mediate access to external model repositories, plugins, and update mechanisms, making them critical checkpoints for supply-chain security. Detection mechanisms must therefore extend to model download hooks, artifact verification, and dependency management. Lessons from micro/nano systems integration and embedded security highlight the importance of provenance tracking, integrity validation, and controlled update paths (Cauwenberghs et al., 2023; Gossner & Duvvury, 2015).

By integrating supply-chain monitoring into the gateway detection framework, enterprises can identify unauthorized model modifications, tampered artifacts, and insecure dependency updates. This holistic view aligns

Table 3: Common LLM Gateway Exposure Vectors and Static Detection Indicators

Detection Layer	*Key Techniques*	*Primary Risks Addressed*	*Cross-Domain Inspiration*
Static analysis	Config parsing, rule verification	Misconfiguration, open endpoints	HW/SW co-design
Runtime monitoring	Traffic anomaly detection	Active exploitation, misuse	CPS resilience
Semantic validation	Pattern matching, policy reasoning	Privilege escalation	System-level design
Supply-chain monitoring	Integrity checks, provenance	Model tampering	Embedded security

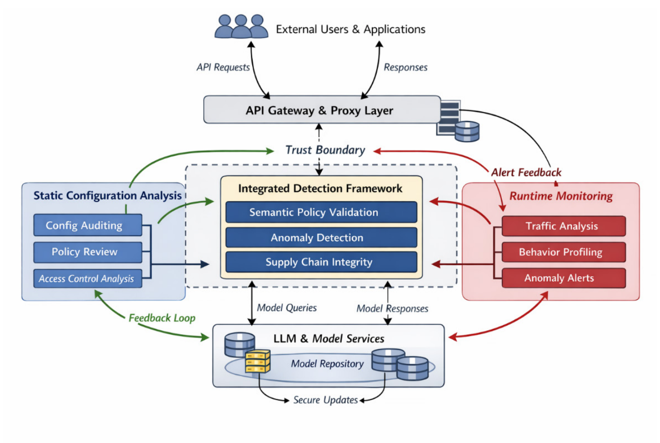

Figure 2: Conceptual Architecture of a Multi-Layer Detection Framework for Enterprise LLM Gateway Exposure

with emerging perspectives on secure heterogeneous integration and future connectivity systems (Wesling, 2020; TUD et al., 2022).

Overall, this section has presented a multi-layer detection framework for identifying LLM gateway exposure in enterprise environments. By combining static configuration analysis, runtime behavioral monitoring, semantic validation, and supply-chain integrity checks, the framework addresses both design-time and operational security risks. Grounded in system-level design and co-design principles, the approach recognizes LLM gateways as critical architectural components whose security depends on holistic reasoning rather than isolated controls. Such a detection framework provides a foundational capability for mitigating misconfigured proxy exposure and strengthening the resilience of enterprise AI infrastructures.

Mitigation Strategies Through Secure Co-Design

The mitigation of security risks associated with enterprise LLM gateways and misconfigured proxies cannot be effectively achieved through isolated, after-the-fact controls. Instead, it requires a secure co-design approach, where security, functionality, performance, and resilience are jointly addressed across architectural layers from the earliest stages of system conception. Drawing from established principles in hardware–software co-design, cyber-physical systems (CPS), and safety-critical embedded domains, secure co-design emphasizes the integration of trust, verification, and fail-safe mechanisms into the core architecture of AI-enabled infrastructures (Teich, 2012; Abdallah et al., 2010; Sangiovanni-Vincentelli, 2007).

In the context of enterprise LLM gateways, secure co-design provides a structured framework for mitigating risks such as exposed endpoints, overly permissive proxies, insecure model access paths, and implicit trust assumptions between components. This section details mitigation strategies grounded in system-level co-design theory and contemporary security engineering practices, with a focus on embedding safeguards directly into LLM gateway architectures rather than relying solely on perimeter defenses.

Zero-Trust Gateway Architecture and Policy Enforcement

A foundational mitigation strategy is the adoption of zero-trust principles within LLM gateway architectures. In a co-design context, zero trust is not merely an access-control policy but an architectural assumption that no component, service, or proxy is inherently trusted (Kandah et al., 2019). Each interaction with an LLM endpoint must therefore be explicitly authenticated, authorized, and continuously validated.

Secure co-design enables zero-trust enforcement by aligning gateway policies with system-level trust models, ensuring that identity management, cryptographic verification, and policy engines are tightly integrated into the gateway fabric (Pesé et al., 2017). This approach mirrors practices in automotive and aerospace embedded systems, where safety and security constraints are jointly enforced across hardware and software boundaries (Chen, 2008; Kohn et al., 2016). For enterprise LLM deployments, this includes mutual authentication between services, fine-grained role-based or attribute-based access control, and strict scoping of API keys to prevent lateral misuse.

Secure Proxy Segmentation and Trust Boundary Definition

Misconfigured proxies often arise from poorly defined trust boundaries within complex enterprise systems. Secure co-design addresses this challenge by enforcing explicit segmentation of proxy layers, ensuring that routing, transformation, and logging functions are isolated according to their security criticality (Dunnihoo, 2015).

From a system-level perspective, proxy segmentation reduces the blast radius of configuration errors by preventing unintended exposure of internal LLM endpoints to external networks. This principle aligns with CPS and IoT architectures, where layered isolation is essential for maintaining resilience in the presence of partial failures or compromises (Chaduvula et al., 2018). Secure co-design further mandates that proxy configurations be treated as verifiable design artifacts, subject to formal review and validation rather than ad hoc operational changes (Seshia et al., 2016).

Cryptographic Identity Binding and Secure Communication Channels

Another critical mitigation strategy is cryptographic identity binding between LLM clients, gateways, and backend model services. Co-design principles emphasize that identity, trust, and communication security must be jointly optimized rather than independently implemented (Gossner & Duvvury, 2015).

In enterprise LLM gateways, this involves binding service identities to cryptographic credentials that are validated at runtime, thereby preventing impersonation

Table 4: Detection Layers, Techniques, and Security Outcomes

Exposure Vector	*Description*	*Static Detection Indicators*	*Related Design Insight*
Unauthenticated endpoints	Public access to LLM APIs	Missing auth directives, open routes	System-level trust violation
Over-permissive proxy rules	Broad routing without validation	Wildcard paths, default allow rules	Architectural abstraction leakage
Insecure model hooks	Unverified model downloads	No checksum or provenance checks	Supply-chain risk
Tenant isolation failure	Cross-tenant data exposure	Shared namespaces or tokens	CPS isolation breakdown

and unauthorized proxy traversal (Suhail et al., 2020). Secure communication channels, including encrypted transport and message-level integrity checks, are co-designed with performance considerations to avoid introducing unacceptable latency, reflecting long-standing trade-offs studied in embedded and high-performance systems (Morshed & Morshed, 2021).

Fail-Operational and Resilient Gateway Design

Secure co-design also incorporates fail-operational and resilience-oriented design patterns, adapted from safety-critical automotive and CPS domains. Rather than defaulting to complete service shutdown upon anomaly detection, LLM gateways can be designed to degrade gracefully, maintaining essential functionality while isolating compromised components (Kohn et al., 2016; Zardini, 2023).

This approach mitigates both security and availability risks, ensuring that enterprises remain operational even during partial misconfigurations or active attacks. By co-designing detection, isolation, and recovery mechanisms, gateway architectures can support automated rollback of unsafe configurations and dynamic re-routing of traffic through verified safe paths (Dantas & Nigam, 2023). Such resilience-focused strategies are particularly important in large-scale AI infrastructures, where manual intervention may be too slow to prevent cascading failures.

Design-Time Verification and Continuous Assurance

A defining characteristic of secure co-design is the emphasis on design-time verification coupled with continuous assurance. Instead of relying solely on runtime monitoring, mitigation strategies incorporate formal and semi-formal verification techniques to validate gateway configurations before deployment (Seshia et al., 2016).

Architectural patterns, policy rules, and proxy configurations are treated as first-class design elements that can be analyzed for security properties such as least privilege, non-bypassability, and fault containment (Dantas & Nigam, 2023). Continuous assurance mechanisms then monitor deviations from verified configurations, enabling early detection of drift that could reintroduce exposure risks. This lifecycle-oriented approach reflects best practices in system-level design automation and CPS engineering (Cauwenberghs et al., 2023).

In sum, mitigation strategies for enterprise LLM gateway vulnerabilities are most effective when grounded in secure co-design principles that integrate security, performance, and resilience across architectural layers. By adopting zero-trust assumptions, enforcing proxy segmentation, binding identities cryptographically, designing for fail-operational behavior, and embedding verification into the system lifecycle, organizations can substantially reduce the risks posed by misconfigured proxies and exposed LLM endpoints. Drawing on decades of research in hardware–software co-design and cyber-physical systems, secure co-design offers a robust and scalable foundation for protecting AI-enabled enterprise infrastructures in increasingly complex and interconnected environments.

Model Download Hooks and Supply-Chain Security

Enterprise deployment of large language models increasingly relies on dynamic model acquisition mechanisms such as model download hooks, plugin-based extensions, containerized artifacts, and external model registries. These mechanisms enable rapid scalability, model updates, and vendor interoperability, but they also introduce critical supply-chain vulnerabilities. Similar to embedded systems and cyber-physical platforms, the integrity of upstream components directly affects downstream system safety, security, and reliability. This section examines model download hooks as a distinct attack surface within enterprise LLM gateways and proposes a structured security perspective grounded in system-level co-design, hardware–software integration, and secure-by-construction principles.

Architecture of Model Download Hooks in Enterprise LLM Systems

Model download hooks refer to automated or semi-automated processes through which LLM gateways retrieve model weights, adapters, fine-tuned layers, or inference plugins from internal or external repositories. These hooks are typically integrated into orchestration layers, CI/CD pipelines, or runtime initialization phases. From an architectural standpoint, they resemble firmware update paths in embedded and automotive systems, where trust assumptions are embedded deep within system design (Abdallah et al., 2010; Chen, 2008).

In heterogeneous enterprise environments, model artifacts may originate from open-source hubs, cloud vendor registries, or third-party research collaborators.

The growing complexity of these integration paths mirrors trends in heterogeneous system integration observed in micro/nano circuits and future connectivity platforms (Cauwenberghs et al., 2023; Wesling, 2020). Without explicit trust boundaries and verification layers, model download hooks can bypass traditional perimeter defenses, creating latent vulnerabilities that are difficult to detect post-deployment.

Threat Landscape and Attack Vectors in LLM Supply Chains

The LLM supply chain is exposed to a range of adversarial actions, including model poisoning, trojan insertion, dependency confusion, and unauthorized model substitution. These threats parallel long-documented risks in embedded system supply chains and cyber-enabled manufacturing ecosystems (Chaduvula et al., 2018; Kandah et al., 2019). In the context of LLMs, a compromised model artifact can leak sensitive prompts, introduce covert behaviors, or degrade decision quality in safety-critical enterprise workflows.

Furthermore, the increasing use of automated download hooks amplifies the risk of large-scale compromise, as malicious artifacts can propagate rapidly across distributed deployments. Similar challenges have been observed in IoT and CPS environments, where scale and automation magnify the impact of trust failures (Suhail et al., 2020; Messaoud et al., 2023). These dynamics necessitate a shift from reactive vulnerability patching to proactive architectural assurance.

Trust, Provenance, and Integrity Verification Mechanisms

Establishing trust in model download hooks requires robust provenance tracking and integrity verification mechanisms. Techniques such as cryptographic signing, hash-based verification, and secure key lifecycle management draw directly from best practices in hardware–software co-design and embedded firewall architectures (Pesé et al., 2017; Gossner & Duvvury, 2015). Provenance metadata, including training origin, fine-tuning history, and dependency lineage, enables semantic validation beyond simple checksum verification.

From a system-level perspective, semantically rich architecture patterns can encode trust assumptions explicitly, allowing automated reasoning about whether a model artifact complies with enterprise security policies (Dantas & Nigam, 2023). This approach aligns with broader design automation efforts in CPS, where formal methods are used to verify safety and security properties before runtime (Seshia et al., 2016).

Secure Co-Design Approaches for Model Supply Chains

Applying co-design principles to LLM supply chains emphasizes the joint optimization of security, performance, and operational flexibility. Lessons from system-level design argue that security controls should be embedded across hardware, software, and organizational processes rather than isolated at runtime boundaries (Sangiovanni-Vincentelli, 2007; Teich, 2012). For LLM gateways, this includes aligning model acquisition logic with gateway authentication, authorization, and monitoring layers.

Fail-operational concepts from automotive systems further suggest that LLM infrastructures should maintain degraded but safe operation when model integrity cannot be verified, rather than defaulting to unrestricted execution (Kohn et al., 2016; Wang et al., 2023). Such strategies reinforce resilience while preserving enterprise continuity.

Governance, Compliance, and Cross-Sector Implications

Beyond technical controls, model download hook security intersects with governance, compliance, and accountability frameworks. As enterprises increasingly deploy LLMs in regulated sectors, supply-chain assurance becomes a prerequisite for auditability and risk management. Analogous challenges have been documented in CPS governance and secure infrastructure development, where compliance gaps often arise from opaque integration practices (Zardini, 2023; Wan, 2018).

Embedding supply-chain security into enterprise AI governance structures ensures alignment between technical safeguards and organizational responsibility. This cross-sector integration reflects broader trends in secure digital infrastructure and intelligent manufacturing ecosystems (Annapareddy et al., 2022).

Overall, model download hooks represent a critical

Table 5: Supply-Chain Risks and Mitigation Strategies for Model Download Hooks

Risk Category	*Description*	*Potential Impact*	*Mitigation Strategy*
Model Poisoning	Injection of malicious behaviors during training or fine-tuning	Data leakage, biased outputs	Cryptographic signing, provenance validation
Unauthorized Substitution	Replacement of trusted models with malicious artifacts	Loss of integrity, covert control	Trusted registries, access control
Dependency Confusion	Exploitation of naming collisions in model repositories	Large-scale compromise	Namespace isolation, semantic verification
Update Rollback Attacks	Downgrade to vulnerable model versions	Reintroduced vulnerabilities	Version pinning, secure update policies
Supply-Chain Opacity	Lack of visibility into model origin	Reduced auditability	Metadata-rich model manifests

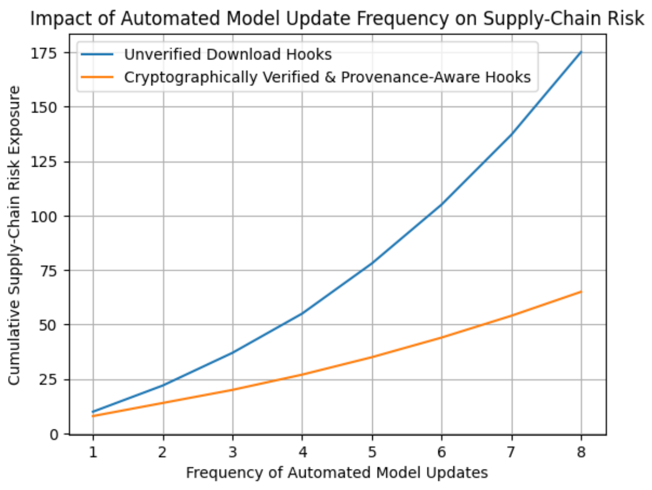

Figure 3: Effect of Automated Model Update Frequency on Cumulative Supply-Chain Risk in Enterprise LLM Systems

yet underexamined vulnerability in enterprise LLM gateways. Drawing on decades of research in hardware–software co-design, CPS security, and system-level design, this section demonstrates that LLM supply-chain security must be treated as an architectural concern rather than a peripheral operational issue. By integrating trust, provenance, and verification mechanisms into the design of model acquisition pathways, enterprises can significantly reduce systemic risk while enabling scalable and resilient AI deployment.

Cross-Domain Insights from CPS, Automotive, and IoT Security

The increasing complexity of enterprise LLM gateways, particularly in terms of exposure through misconfigured proxies and unsecured endpoints, necessitates learning from other domains with similar security and architectural challenges. Cyber-Physical Systems (CPS), automotive electronics, and Internet of Things (IoT) ecosystems face analogous issues of system-level integration, real-time constraints, and safety–security trade-offs. This section explores insights from these domains to inform the design of resilient and secure LLM infrastructure.

Lessons from Cyber-Physical Systems Security

CPS integrate computation, networking, and physical processes, creating intricate interdependencies between system components. These systems prioritize both functional safety and security, often requiring co-design approaches where hardware, software, and communication protocols are jointly optimized (Seshia et al., 2016; Dunnihoo, 2015). Key CPS security principles applicable to enterprise LLM gateways include:

- Semantic validation of system interactions: Ensuring that data flows between components adhere to pre-defined trust and safety constraints (Dantas & Nigam, 2023).
- Early-stage co-design: Embedding security and safety measures at the design phase rather than as retrofitted controls (Teich, 2012; Abdallah et al., 2010).
- Real-time monitoring and anomaly detection: Continuous observation of system behavior to detect deviations that could indicate exploitation of misconfigured endpoints (Cauwenberghs et al., 2023).

By adopting CPS-inspired co-design frameworks, enterprise LLM systems can implement systemic resilience rather than relying solely on perimeter defense.

Automotive Systems as a Model for Resilient Gateway Design

Modern automotive systems incorporate heterogeneous electronics, software-defined functions, and AI-based decision-making. Security and functional safety are tightly integrated through fail-operational architectures and secure communication protocols (Wang et al., 2023; Kohn et al., 2016). Lessons for enterprise LLM gateway security include:

- Segmentation of functional domains: Critical systems are isolated from less sensitive ones, reducing the impact of potential breaches.
- Redundancy and fail-operational capabilities: Systems maintain functionality even when components fail or are compromised (Pesé et al., 2017).
- Policy-driven access enforcement: Role-based and cryptographic policy frameworks ensure only authorized processes can access critical modules (Chen, 2008).

These approaches can inform the design of multi-tiered LLM gateways, where API endpoints and model download hooks are segmented based on sensitivity.

IoT Security Practices for Federated and Distributed Architectures

IoT ecosystems present extreme scale and heterogeneity, often with constrained devices and distributed communication paths. Security strategies include lightweight cryptography, device attestation, federated monitoring, and trust graph validation (Kandah et al., 2019; Suhail et al., 2020; Gogineni et al., 2023). For enterprise LLM gateways, these strategies highlight the importance of:

- Distributed monitoring of access events: Detecting anomalous gateway requests in real time.
- Provenance verification of model artifacts: Ensuring integrity and authenticity of AI models downloaded through hooks.
- Identity and trust binding at scale: Cryptographic enforcement of permissions across heterogeneous system components.

Integration of Cross-Domain Insights

Enterprise LLM gateway designers can leverage the combined insights from CPS, automotive, and IoT domains to develop a layered security and resilience framework.

Core principles include:

- Early-stage co-design: Security should be a design constraint, not an afterthought (Teich, 2012).
- Segmentation and domain isolation: Critical APIs and model artifacts must be isolated to reduce risk propagation (Wang et al., 2023).
- Continuous monitoring and verification: Behavioral and semantic anomaly detection ensure timely detection of misconfiguration exploits (Cauwenberghs et al., 2023).
- Redundancy and fail-safe mechanisms: Maintain operational continuity in the event of exposure or compromise (Kohn et al., 2016).
- Trust-centric identity management: Cryptographic binding and provenance verification prevent unauthorized model downloads or API misuse (Suhail et al., 2020; Kandah et al., 2019).

Future Directions in Cross-Domain Security Application

Looking forward, enterprise LLM gateway frameworks could benefit from:

- Dynamic co-design tools that integrate runtime telemetry with architecture validation (Seshia et al., 2016).
- AI-driven policy enforcement to adaptively detect and mitigate misconfigurations.
- Cross-domain security standardization, drawing on CPS, automotive, and IoT compliance models to ensure scalable, robust deployment of LLMs in enterprise environments (Zardini, 2023; Wan, 2018).

In sum, Cross-domain insights from CPS, automotive, and IoT security provide actionable principles for securing enterprise LLM gateways. By integrating co-design strategies, segmentation, continuous monitoring, redundancy, and trust-based identity management, organizations can mitigate the risks of exposed endpoints, misconfigured proxies, and compromised model download hooks. This layered, systemic approach transforms LLM gateway security from reactive patching to proactive, architecture-driven resilience (Dantas & Nigam, 2023; Sangiovanni-Vincentelli, 2007).

Research and Practice Implications

The increasing adoption of large language models (LLMs) within enterprise environments has created a paradigm shift in organizational AI capabilities. While LLM gateways and proxy services enable scalable deployment, orchestration, and multi-tenant access, they also introduce novel security challenges, particularly misconfigured endpoints, exposed model download hooks, and weak identity verification (Dantas & Nigam, 2023; Kandah et al., 2019). Understanding the research and practice implications of these emerging vulnerabilities is critical for bridging the gap between theoretical co-design principles and actionable enterprise security strategies. This section examines the key areas where research can advance, identifies practical guidelines for organizations, and draws upon cross-domain insights from embedded systems, IoT, and cyber-physical systems (CPS) security (Seshia et al., 2016; Chaduvula et al., 2018).

Standardization of LLM Gateway Architectures

A central research implication is the need for standardized reference architectures for enterprise LLM gateways. Existing deployments are often heterogeneous, with variations in API gateways, reverse proxies, authentication mechanisms, and model storage solutions (Cauwenberghs et al., 2023; Wesling, 2020). Standardized frameworks can facilitate reproducibility, interoperability, and security benchmarking. In practice, organizations can adopt architectural blueprints informed by CPS and automotive embedded system co-design, ensuring that safety, reliability, and security are integrated at design time rather than retrofitted (Teich, 2012; Kohn et al., 2016).

Automated Configuration Verification Tools

Automated tools capable of analyzing gateway and proxy configurations are critical for detecting misconfigurations before they result in security incidents. These tools can perform semantic validation of access control policies, endpoint exposure analysis, and runtime traffic monitoring (Dantas & Nigam, 2023; Seshia et al., 2016). Incorporating design automation principles from micro/nano circuits and IoT/CPS domains can enhance detection efficiency and reduce human error (Cauwenberghs et al., 2023; Gogineni et al., 2023).

8.3 Integration of Supply-Chain Security Practices

Research implications extend to securing the supply chain for LLM model artifacts. Model download hooks,

Table 6: Cross-Domain Security Lessons for Enterprise LLM Gateways

Domain	*Key Security Principle*	*Application to LLM Gateways*	*Example Mechanism*
CPS	Co-design of safety and security	Integrate endpoint configuration checks during design	Semantic policy validation (Dantas & Nigam, 2023)
Automotive	Segmentation & fail-operational systems	Multi-tiered gateway with critical endpoint isolation	Redundant API routing & failover controls (Pesé et al., 2017)
IoT	Federated monitoring & identity management	Distributed detection of proxy misconfigurations	Cryptographic device attestation & trust graphs (Suhail et al., 2020)

plugin updates, and external repositories present risks analogous to embedded system firmware or IoT device supply chains (Gossner & Duvvury, 2015; TUD et al., 2022). Establishing provenance verification, signed artifacts, and controlled update mechanisms can mitigate threats such as model poisoning and unauthorized modifications (Chaduvula et al., 2018; Suhail et al., 2020). Practitioners should implement automated integrity checks alongside CI/CD pipelines to ensure consistent security enforcement.

Cross-Domain Learning from Cyber-Physical Systems

The LLM gateway security problem benefits from lessons learned in automotive, aerospace, and CPS domains, where fail-operational designs and co-design principles are well-established (Abdallah et al., 2010; Zardini, 2023; Pesé et al., 2017). Applying these principles can help organizations design resilient LLM infrastructures that anticipate misconfigurations, provide graceful degradation during attacks, and enforce strong trust boundaries. In practice, this involves embedding security controls into gateway logic, deploying redundant validation mechanisms, and continuously monitoring traffic flows.

Human Factors and Organizational Policies

A less explored but significant implication involves human factors and policy governance. Misconfigurations often arise due to inconsistent practices, insufficient training, or inadequate operational guidelines (Soriano et al., 2019; Shankar, 2021). Research should focus on creating decision-support tools that integrate with architectural verification platforms, guiding system administrators to implement secure configurations. Practically, enterprises should establish continuous education programs, security playbooks, and audit mechanisms to maintain compliance with zero-trust principles.

In sum, the research and practice implications highlight that securing enterprise LLM gateways requires a multidimensional approach, combining architectural standardization, automated verification, supply-chain integrity, cross-domain co-design insights, and robust organizational policies. Advancing research in these areas not only strengthens the resilience of AI deployments but also bridges the gap between theoretical security frameworks and operationally feasible solutions. By integrating insights from CPS, IoT, and embedded systems, organizations can proactively mitigate misconfigured proxy exposure and enhance the overall security posture of their enterprise LLM infrastructure (Sangiovanni-Vincentelli, 2007; Dantas & Nigam, 2023; Wang et al., 2023).

Conclusion and Future Research Directions

The deployment of large language models (LLMs) in enterprise environments offers significant opportunities for automation, scalability, and intelligent decision-making. However, misconfigured proxies, exposed endpoints, and insecure model download hooks present substantial security risks. Securing LLM gateways requires a system-level approach that integrates co-design principles, embedded systems architecture, and cyber-physical system security strategies (Teich, 2012; Seshia et al., 2016; Dantas & Nigam, 2023).

This study highlights the importance of standardizing LLM gateway architectures, implementing automated configuration verification tools, securing supply chains for model artifacts, leveraging lessons from CPS, automotive, and IoT systems, and addressing human and organizational factors (Cauwenberghs et al., 2023; Kohn et al., 2016; Gogineni et al., 2023; Gossner & Duvvury, 2015; Soriano et al., 2019). By combining these approaches, organizations can reduce the risk of exposed endpoints, enforce strong trust mechanisms, and improve resilience against both internal misconfigurations and external threats.

Future research should focus on adaptive security mechanisms for dynamic LLM environments, federated verification frameworks across hybrid deployments, explainable security controls to guide administrators, integration with regulatory and compliance frameworks, and the development of simulation platforms to test complex threat scenarios (Wang et al., 2023; Mohsan & Li, 2023; Shankar, 2021; Kandah et al., 2019; Soriano et al., 2019).

Securing enterprise LLM gateways is a multifaceted challenge that intersects AI, cybersecurity, system co-design, and organizational governance. Integrating

Table 7: Example LLM Gateway Security Verification Metrics

Metric	*Description*	*Measurement Approach*	*Example Threshold*
Endpoint Exposure Score	Measures number of publicly accessible endpoints	Automated scan of API routes	<5 exposed endpoints
Proxy Rule Compliance	Validates adherence to zero-trust and segmentation rules	Policy checker tool	100% rule compliance
Authentication Strength	Evaluates identity verification mechanisms	Password/Key analysis & MFA audit	MFA enforced on all accounts
Artifact Integrity	Verifies hash signatures for downloaded models	Cryptographic validation	100% valid checksums
Traffic Anomaly Detection	Identifies unusual API call patterns	Runtime monitoring	<1% anomaly rate per day

design-time security, automated verification, cross-domain insights, and human-centric policies ensures resilient and trustworthy LLM deployments while enabling organizations to fully realize the benefits of AI without compromising security, trust, or operational continuity (Sangiovanni-Vincentelli, 2007; Dantas & Nigam, 2023).

REFERENCES

1. Dantas, Y. G., & Nigam, V. (2023). Automating safety and security co-design through semantically rich architecture patterns. ACM Transactions on Cyber-Physical Systems, 7(1), 1-28.
2. Zardini, G. (2023). Co-design of complex systems: From autonomy to future mobility systems (Doctoral dissertation, ETH Zurich).
3. Teich, J. (2012). Hardware/software codesign: The past, the present, and predicting the future. Proceedings of the IEEE, 100(Special Centennial Issue), 1411-1430.
4. Abdallah, A., Feron, E. M., Hellestrand, G., Koopman, P., & Wolf, M. (2010). Hardware/software codesign of aerospace and automotive systems. Proceedings of the IEEE, 98(4), 584-602.
5. Kovalchuk, Y. (2024). Improving the Accuracy of Artificial Intelligence Models in Nutrition and Health Research Through High-Quality Data Processing. SAMRIDDHI: A Journal of Physical Sciences, Engineering and Technology, 16(01), 48-59.
6. Wang, Y., Xiao, J., Wei, Z., Zheng, Y., Tang, K. T., & Chang, C. H. (2023). Security and functional safety for AI in embedded automotive system—a tutorial. IEEE Transactions on Circuits and Systems II: Express Briefs, 71(3), 1701-1707.
7. Seshia, S. A., Hu, S., Li, W., & Zhu, Q. (2016). Design automation of cyber-physical systems: Challenges, advances, and opportunities. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 36(9), 1421-1434.
8. Cauwenberghs, G., Cong, J., Hu, X. S., Joshi, S., Mitra, S., Porod, W., & Wong, H. S. P. (2023). Micro/nano circuits and systems design and design automation: challenges and opportunities [point of view]. Proceedings of the IEEE, 111(6), 561-574.
9. Kandah, F., Cancelleri, J., Reising, D., Altarawneh, A., & Skjellum, A. (2019, July). A hardware-software codesign approach to identity, trust, and resilience for iot/cps at scale. In 2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) (pp. 1125-1134). IEEE.
10. Chaduvula, S. C., Dachowicz, A., Atallah, M. J., & Panchal, J. H. (2018). Security in cyber-enabled design and manufacturing: A survey. Journal of Computing and Information Science in Engineering, 18(4), 040802.
11. Maheshkar, J. A. (2023). Automated code vulnerability detection in FinTech applications using AI-Based static analysis. Academic Social Research, 9(3), 1–24. https://doi.org/10.13140/RG.2.2.32960.80648
12. Sangiovanni-Vincentelli, A. (2007). Quo vadis, SLD? Reasoning about the trends and challenges of system level design. Proceedings of the IEEE, 95(3), 467-506.
13. Wesling, P. (2020, February). The heterogeneous integration roadmap: enabling technology for systems of the future. In 2020 Pan Pacific Microelectronics Symposium (Pan Pacific) (pp. 1-4). IEEE.
14. Maheshkar, J. A. (2023). AI-Assisted Infrastructure as Code (IAC) validation and policy enforcement for FinTech systems. Academic Social Research, 9(4), 20–44. https://doi.org/10.13140/rg.2.2.26249.92002
15. TUD, G. F., Koszescha, J., Neyer, M., Cogez, P., NXP, P. P., Wambacq, P., & TUD, V. R. (2022). European Core Technologies for future connectivity systems and components. COREnect.
16. Chen, X. (2008). Requirements and concepts for future automotive electronic architectures from the view of integrated safety. KIT Scientific Publishing.
17. Maheshkar, J. A. (2024). Intelligent CI/CD Pipelines Using AI-Based Risk Scoring for FinTech Application Releases. Acta Sci, 25, 1.
18. Pesé, M. D., Schmidt, K., & Zweck, H. (2017). Hardware/software co-design of an automotive embedded firewall (No. 2017-01-1659). SAE Technical Paper.
19. Morshed, B. I., & Morshed. (2021). Embedded Systems-A Hardware-Software Co-Design Approach. Springer International Publishing.
20. Wouters, H., & Martinez, W. (2023). Bidirectional onboard chargers for electric vehicles: State-of-the-art and future trends. IEEE Transactions on Power Electronics, 39(1), 693-716.
21. Suhail, S., Hussain, R., Khan, A., & Hong, C. S. (2020). On the role of hash-based signatures in quantum-safe internet of things: Current solutions and future directions. IEEE Internet of Things Journal, 8(1), 1-17.
22. Carter, J., Feddema, J., Kothe, D., Neely, R., Pruet, J., Stevens, R., ... & Dietrich, E. M. (2023). Advanced research directions on ai for science, energy, and security: Report on summer 2022 workshops.
23. Akan, O. B., & Arik, M. (2020). Internet of radars: Sensing versus sending with joint radar-communications. IEEE Communications Magazine, 58(9), 13-19.
24. Maheshkar, J. A. (2024b, September 20). AI-Driven FinOps: Intelligent Budgeting and Forecasting in Cloud Ecosystems. https://eudoxuspress.com/index.php/pub/article/view/4128
25. Shankar, S. S. (2021, September). Lessons from nature for computing: looking beyond moore's law with special purpose computing and co-design. In 2021 IEEE High Performance Extreme Computing Conference (HPEC) (pp. 1-8). IEEE.
26. Karim, R., Iftikhar, A., Ijaz, B., & Mabrouk, I. B. (2019). The potentials, challenges, and future directions of on-chip-antennas for emerging wireless applications—A comprehensive survey. IEEE Access, 7, 173897-173934.
27. Mohsan, S. A. H., & Li, Y. (2023). A contemporary survey on 6G wireless networks: Potentials, recent advances, technical challenges and future trends. arXiv preprint arXiv:2306.08265.
28. Gogineni, V. C., Werner, S., Gauthier, F., Huang, Y. F., & Kuh, A. (2023). Personalized online federated learning for IoT/CPS: Challenges and future directions. IEEE Internet of Things Magazine, 5(4), 78-84.
29. Soriano, A., Ponce, P., & Molina, A. (2019, May). A novel design of virtual laboratory. In 2019 20th International Conference on Research and Education in Mechatronics (REM) (pp.

1-6). IEEE.

30. Dunnihoo, J. C. (2015). Co-Design Trade-Offs: Balancing Robustness, Performance, and Cost. System Level ESD Co-Design, 353-388.
31. Annapareddy, V. N., Preethish Nandan, B., Kommaragiri, V. B., Gadi, A. L., & Kalisetty, S. (2022). Emerging Technologies in Smart Computing, Sustainable Energy, and Next-Generation Mobility: Enhancing Digital Infrastructure, Secure Networks, and Intelligent Manufacturing. Venkata Bhardwaj and Gadi, Anil Lokesh and Kalisetty, Srinivas, Emerging Technologies in Smart Computing, Sustainable Energy, and Next-Generation Mobility: Enhancing Digital Infrastructure, Secure Networks, and Intelligent Manufacturing (December 15, 2022).
32. Sachar, D. P. S. (2023). Time Series Forecasting Using Deep Learning: A Comparative Study of LSTM, GRU, and Transformer Models. Journal of Computer Science and Technology Studies, 5(1), 74-89.
33. Kohn, A., Schneider, R., Vilela, A., Roger, A., & Dannebaum, U. (2016). Architectural concepts for fail-operational automotive systems (No. 2016-01-0131). SAE Technical Paper.
34. Sen, P., Harutyunyan, A., Umar, M., & Kamal, S. (2023). Joint communication and radar sensing: RF hardware opportunities and challenges A circuits and systems perspective. Sensors, 23(18), 7673.
35. Gossner, H., & Duvvury, C. (2015). Future Applications of SEED Methodology. System Level ESD Co-Design, 334-352.
36. Kommaragiri, V. B., Preethish Nanan, B., Annapareddy, V. N., Gadi, A. L., & Kalisetty, S. (2022). Emerging Technologies in Smart
37. Computing, Sustainable Energy, and Next-Generation Mobility: Enhancing Digital Infrastructure, Secure Networks, and Intelligent Manufacturing. Venkata Narasareddy and Gadi, Anil Lokesh and Kalisetty, Srinivas.
38. Rony, M. M. A., Soumik, M. S., & Akter, F. (2023). Applying Artificial Intelligence to Improve Early Detection and Containment of Infectious Disease Outbreaks, Supporting National Public Health Preparedness. Journal of Medical and Health Studies, 4(3), 82-93.
39. Messaoud, S., Amdouni, R., Albouchi, A., Hajjaji, M. A., Mtibaa, A., & Atri, M. (2023). Future internet of things: connecting the unconnected world and things based on 5/6G networks and embedded technologies. In Internet of Things-New Trends, Challenges and Hurdles. IntechOpen.
40. Rony, M. M. A., Soumik, M. S., & SRISTY, M. S. (2023). Mathematical and AI-Blockchain Integrated Framework for Strengthening Cybersecurity in National Critical Infrastructure. Journal of Mathematics and Statistics Studies, 4(2), 92-103.
41. Siddique, M. T., Hussain, M. K., Soumik, M. S., & SRISTY, M. S. (2023). Developing Quantum-Enhanced Privacy-Preserving Artificial Intelligence Frameworks Based on Physical Principles to Protect Sensitive Government and Healthcare Data from Foreign Cyber Threats. British Journal of Physics Studies, 1(1), 46-58.
42. Soumik, M. S., Sarkar, M., & Rahman, M. M. (2021). Fraud Detection and Personalized Recommendations on Synthetic E-Commerce Data with ML. Research Journal in Business and Economics, 1(1a), 15-29.
43. Wan, J. (2018). Modeling and Co-Design of Multi-domain Cyber-Physical Systems (Doctoral dissertation, University of California, Irvine).
44. Kumar, S. (2007). Patterns in the daily diary of the 41st president, George Bush (Doctoral dissertation, Texas A&M University).
45. Uppuluri, V. (2019). The Role of Natural Language Processing (NLP) in Business Intelligence (BI) for Clinical Decision Support. ISCSITR-INTERNATIONAL JOURNAL OF BUSINESS INTELLIGENCE (ISCSITR-IJBI), 1(2), 1-21.
46. Abraham, U. I. (2020). Deforestation, Air Quality Degradation and Increased Cardiopulmonary Diseases. SRMS JOURNAL OF MEDICAL SCIENCE, 5(02).
47. Azmi, S. K., Vethachalam, S., & Karamchand, G. (2022). The Scalability Bottleneck in Legacy Public Financial Management Systems: A Case for Hybrid Cloud Data Lakes in Emerging Economies.
48. Abraham, U. I. (2022). Immigration Positive Impact in Modifying, Prevention of Genetically Induced Diseases “Obesity, Cancer”. SRMS JOURNAL OF MEDICAL SCIENCE, 7(01).
49. Uppuluri, V. (2020). Integrating behavioral analytics with clinical trial data to inform vaccination strategies in the US retail sector. J Artif Intell Mach Learn & Data Sci, 1(1), 3024-3030.
50. OKAFOR, C., VETHACHALAM, S., & AKINYEMI, A. A DevSecOps MODEL FOR SECURING MULTI-CLOUD ENVIRONMENTS WITH AUTOMATED DATA PROTECTION.
51. Maheshkar, J. A. (2023). Automated code vulnerability detection in FinTech applications using AI-Based static analysis. Academic Social Research, 9(3), 1–24. https://doi.org/10.13140/RG.2.2.32960.80648
52. Abraham, U. I. (2023). Sleep Health as an Economic Asset: Evaluating Roles of adequate sleep in global labor efficiency. Multidisciplinary Innovations & Research Analysis, 4(4), 71-85.
53. Syed, K. A., Vethachalam, S., Karamchand, G., & Gopi, A. (2023). Implementing a Petabyte-Scale Data Lakehouse for India’s Public Financial Management System: A High-Throughput Ingestion and Processing Framework.
54. Maheshkar, J. A. (2023). AI-Assisted Infrastructure as Code (IAC) validation and policy enforcement for FinTech systems. Academic Social Research, 9(4), 20–44. https://doi.org/10.13140/rg.2.2.26249.92002
55. Goel, Nayan. (2024). CLOUD SECURITY CHALLENGES AND BEST PRACTICES. Journal of Tianjin University Science and Technology. 57. 571-583. 10.5281/zenodo.17163793.
56. AZMI, S. K. JVM OPTIMIZATION TECHNIQUES FOR HIGH-THROUGHPUT AI AND ML SYSTEMS.
57. Jaykumar Ambadas Maheshkar. (2024). Intelligent CI/CD Pipelines Using AI-Based Risk Scoring for FinTech Application Releases. Acta Scientiae, 25(1), 90–108. https://www.periodicos.ulbra.org/index.php/acta/article/view/532
58. Rehan, H. (2024). Scalable Cloud Intelligence for Preventive and Personalized Healthcare. Pioneer Research Journal of Computing Science, 1(3), 80-105.
59. Azmi, S. K. (2024). From Reactive Reporting to Proactive Governance: The Impact of a Real-Time Analytics Engine on India’s Direct Benefit Transfer Schemes.

60. Goel, Nayan. (2024). ZERO-TRUST AI SECURITY: INTEGRATING AI INTO ZERO-TRUST ARCHITECTURES. Journal of Tianjin University Science and Technology. 57. 158-173. 10.5281/zenodo.17149652.
61. Azmi, S. K., Vethachalam, S., & Karamchand, G. Predictive Analytics for National Budgeting and Expenditure: Leveraging AI/ML on the PFMS 2.0 Data Ecosystem.
62. Maheshkar, J. A. (2024b, September 20). AI-Driven FinOps: Intelligent Budgeting and Forecasting in Cloud Ecosystems. https://eudoxuspress.com/index.php/pub/article/view/4128
63. Lima, S. A., & Rahman, M. M. (2024). Algorithmic fairness in HRM balancing AI-driven decision making with inclusive workforce practices. Journal of Information Systems Engineering and Management, 9(4s), 10-52783.
64. Kumar, S., Loo, L., & Kocian, L. (2024, October). Blockchain Applications in Cyber Liability Insurance. In 2nd International Conference on Blockchain, Cybersecurity and Internet of