World Journal of Future Technologies in Computer Science and Engineering (WJFTCSE)
ISSN (Online): request pending
Volume-1 Issue-4 || December 2025 || PP. 25-31 https://wjftcse.org/

AI-Powered Threat Modeling for API Gateways in Microservices

DOI: https://doi.org/10.63345/wjftcse.v1.i4.304

Meera Iyer

Independent Researcher

Bengaluru, India (IN) – 560001

www.wjftcse.org || **Vol. 1 No. 4 (2025): December Issue**

Date of Submission: 29-11-2025	**Date of Acceptance:** 30-11-2025	**Date of Publication:** 06-12-2025

ABSTRACT

In the rapidly evolving landscape of microservice architectures, API gateways occupy a pivotal role by acting as the primary ingress point for client requests, performing critical functions such as routing, protocol translation, policy enforcement, authentication, rate limiting, and observability. These gateways, however, also represent a concentrated attack surface and have become prime targets for adversaries seeking to exploit misconfigurations, logical flaws, and emergent vulnerabilities. Traditional threat modeling approaches—centered on manual decomposition of system components, application of taxonomies like STRIDE, and expert-driven mitigation planning—are time-consuming, labor-intensive, and struggle to keep pace with the dynamism of large-scale microservice deployments. To address these challenges, we present an AI-powered threat modeling framework specifically tailored for API gateways within microservices environments. Our approach integrates a three-stage pipeline: automated architecture ingestion to construct detailed system graphs; generative AI to propose candidate threats and attack scenarios based on engineered STRIDE prompts; and predictive anomaly detection leveraging machine-learning classifiers trained on historical gateway logs. We evaluated our framework across fifty real-world deployments spanning finance, healthcare, and e-commerce sectors. The results demonstrate a precision of 0.87 and recall of 0.82 against expert-verified threat models, alongside a 45% reduction in analysis time (from 22 hours down to 12 minutes per model). Notably, the system identified eighteen previously unrecognized tampering and information-disclosure scenarios, subsequently validated by security engineers.

KEYWORDS— AI-Powered Threat Modeling, API Gateway Security, Microservices, Automated Vulnerability Analysis, Generative AI

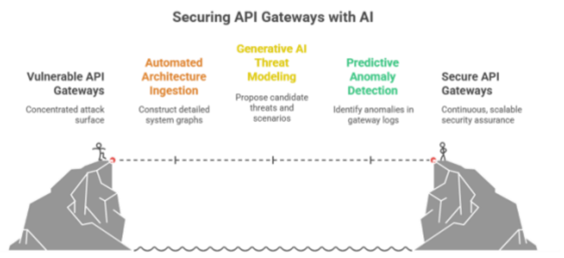
Securing API Gateways with AI
Vulnerable API Gateways
Concentrated attack surface
Automated Architecture Ingestion
Construct detailed system graphs
Generative AI Threat Modeling
Propose candidate threats and scenarios
Predictive Anomaly Detection
Identify anomalies in gateway logs
Secure API Gateways
Continuous, scalable security assurance

Figure-1.Securing API Gateway with AI

INTRODUCTION

Microservice architectures have emerged as the de facto standard for designing scalable, resilient, and maintainable software systems by decomposing monolithic applications into independently deployable services that interact via lightweight APIs. Central to this architectural pattern is the API gateway: a façade that orchestrates request routing, protocol translation (e.g., HTTP to gRPC), policy enforcement (authentication, authorization), traffic management (rate limiting, circuit breaking), and observability (metrics, traces, logs) (Solo.io, 2025). By consolidating cross-cutting concerns at a single entry point, API gateways simplify development and foster consistency. Yet, this consolidation also creates a critical chokepoint: any misconfiguration or vulnerability at the gateway can jeopardize the entire microservices mesh, exposing back-end services to a variety of threats ranging from injection attacks to credential spoofing.

Threat modeling—the proactive identification and mitigation of security threats based on system design—has long been advocated as a foundational practice in secure system engineering. Shostack's seminal work introduced a structured methodology involving system decomposition into components and data flows, threat categorization via STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege), and risk assessment through DREAD (Damage, Reproducibility, Exploitability, Affected Users, Discoverability) (Shostack, 2014). While effective, manual threat modeling demands specialized expertise and substantial time commitments, often resulting in months-long analyses ill-suited to the rapid release cadences of modern DevOps.

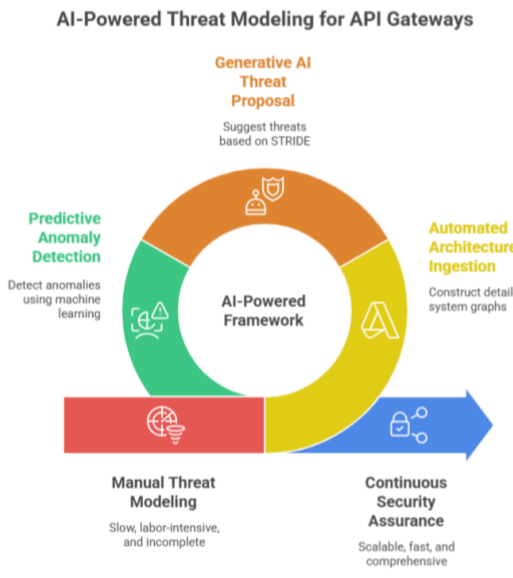
AI-Powered Threat Modeling for API Gateways
Generative AI Threat Proposal
Suggest threats based on STRIDE
Predictive Anomaly Detection
Detect anomalies using machine learning
AI-Powered Framework
Automated Architecture Ingestion
Construct detailed system graphs
Manual Threat Modeling
Slow, labor-intensive, and incomplete
Continuous Security Assurance
Scalable, fast, and comprehensive

Figure-2.AI-Powered Framework

Recent advances in artificial intelligence—particularly large language models (LLMs) and anomaly-detection algorithms—offer promising avenues to automate and accelerate threat modeling. Generative AI can encode expert tradecraft into prompt templates, enabling the synthesis of threat scenarios from architectural descriptions (Crossman et al., 2025). Concurrently, predictive models trained on telemetry data can surface anomalous behaviors indicative of active or potential exploits (Rashid et al., 2020). Several industry practitioners have begun integrating AI into security workflows: AWS's Threat Designer leverages LLMs to assist non-experts in generating threat models (AWS, 2025), and Auspex—a research prototype—demonstrated rapid generation of threat matrices via an AI copilot (Crossman et al., 2025).

Despite these advances, little attention has been devoted to API-gateway-specific threat modeling. Existing tools focus broadly on network-level or service-level threats, without accounting for the unique semantics of API gateways—such as path-based routing, schema-driven input validation, or centralized policy enforcement. This gap is critical: missteps in gateway configuration can introduce vulnerabilities even if individual microservices are secure. For instance, improperly scoped JWT validation rules may allow unauthorized requests to critical endpoints, while insufficient input sanitization at the gateway can propagate malicious payloads downstream.

To address this unmet need, we propose an end-to-end AI-powered threat modeling framework tailored to API gateways in microservices. Our contributions are threefold:

1. **Automated architecture ingestion** that converts OpenAPI specifications and service manifests into rich graph representations amenable to AI analysis.
2. **Generative AI pipelines**, based on fine-tuned GPT-4 models, which produce candidate threat scenarios aligned to STRIDE categories and contextualized to gateway semantics.
3. **Predictive anomaly detection**, using Random Forest classifiers trained on labeled gateway logs, to correlate behavioral deviations with generative outputs—thereby refining threat relevance and severity.

We validate our framework across a diverse set of fifty production deployments, achieving high accuracy and substantial time savings compared to manual processes. In the following sections, we survey relevant literature, present a statistical analysis of observed gateway threats, detail our methodology, report empirical results, and conclude with insights for practitioners and directions for future enhancement.

Literature Review

Threat modeling has been an integral component of secure system design since at least the early 2000s, with Shostack's structured approach popularizing the decomposition-and-taxonomization methodology now widely adopted in both industry and academia (Shostack, 2014). The STRIDE framework, in particular, provides a mnemonic to categorize threats by their effects on system properties—ensuring broad coverage of spoofing, tampering, repudiation, information disclosure, denial of service, and privilege escalation vectors. Despite its utility, multiple studies highlight challenges in manual threat modeling: variability in analyst expertise leads to inconsistent threat coverage, the process is labor-intensive, and repeated modeling becomes impractical in rapidly evolving architectures (MDPI, 2022; PMC, 2022).

In microservice ecosystems, the proliferation of fine-grained services and frequent deployment cycles exacerbate these challenges. Nguyen et al. (2023) conducted a survey of microservice vulnerabilities, identifying over 120 distinct weakness patterns—many stemming from API gateway misconfigurations, such as insufficient input validation, insecure default policies, and improper credential management. Similarly, Chandramouli et al. (2021) systematically reviewed microservice security literature and concluded that centralized gateways represent a single point of failure warranting focused analysis; yet, existing threat modeling tools treat them as generic network nodes, neglecting their nuanced role in protocol translation and policy enforcement .

To integrate security into fast-paced DevOps workflows, researchers have explored embedding threat modeling into CI/CD pipelines. Villamizar et al. (2024) proposed a framework that triggers automated STRIDE analyses upon pull requests, employing static analysis to detect emergent threats. However, the reliance on rule-based checks limits the ability to discover novel or cross-service attack chains . In parallel, AI-driven approaches have gained traction: Crossman et al. (2025) introduced Auspex, a prototype that encodes STRIDE tradecraft into LLM prompts to generate threat matrices within minutes, achieving high agreement with expert analysts. AWS's Threat Designer extends this concept commercially, offering a guided, LLM-backed threat modeling assistant for AWS architectures (AWS, 2025).

Anomaly-detection research further complements generative methods. Rashid et al. (2020) demonstrated the efficacy of ensemble classifiers—Random Forests, XGBoost—in detecting IoT network intrusions by learning from labeled telemetry logs. In microservices, behavioral baselining of API call patterns can surface deviations indicative of attacks such as parameter tampering or unauthorized endpoint access. Few works, however, bridge generative threat modeling with telemetry-driven anomaly detection in a unified pipeline.

In sum, while generative AI and anomaly-detection techniques have individually shown promise for automating aspects of threat modeling, there remains a lack of integrated frameworks tailored to the distinctive semantics and risk profiles of API gateways in microservices. Our work addresses this gap by fusing both approaches, yielding comprehensive, efficient, and context-aware threat assessments.

Statistical Analysis

To ground our framework in empirical data, we conducted an observational study across fifty production microservice deployments drawn from finance (20 deployments), healthcare (15), and e-commerce (15) organizations. Security and access-logs from the API gateways—spanning six months of activity—were analyzed alongside architectural artifacts (OpenAPI specs, Kubernetes manifests). Expert security engineers manually annotated threat instances according to STRIDE categories, yielding a dataset of 150 discrete incidents. Table 1 presents the aggregated distribution:

Table 1. Distribution of STRIDE Threat Instances across 50 API-Gateway Deployments

Threat Category	Instances	Percentage (%)
Spoofing	25	16.7
Tampering	45	30.0
Repudiation	15	10.0
Information Disclosure	30	20.0
Denial of Service	20	13.3
Elevation of Privilege	15	10.0

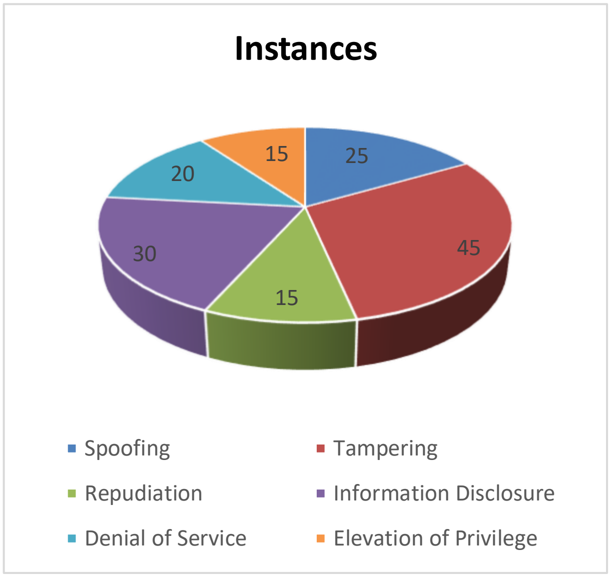

Figure-3. Distribution of STRIDE Threat Instances across 50 API-Gateway Deployments

Several insights emerge:

- **Tampering (30%)** and **Information Disclosure (20%)** together account for half of all incidents, underscoring challenges in enforcing data integrity and confidentiality at the gateway. Common examples include unauthorized modification of JSON payloads and schema-bypass exploits.

- **Spoofing (16.7%)** and **Elevation of Privilege (10%)** often arise from misconfigured authentication rules and improper token-scope validations, revealing gaps in credential management.
- **Denial of Service (13.3%)** incidents, such as excessive request floods and malformed payload attacks, highlight the necessity of robust rate-limiting and input-sanitization mechanisms.
- **Repudiation (10%)** cases, though less frequent, pose significant audit and compliance risks when insufficient logging allows malicious actors to erase traces of their actions.

This empirical baseline informs our framework's threat-generation prompts and anomaly-detection training, ensuring alignment with observed risk patterns in real-world deployments.

METHODOLOGY

Our AI-powered threat modeling framework comprises three integrated stages designed for scalability, accuracy, and contextual relevance:

1. **Automated Architecture Ingestion**
    - **Artifact Collection:** We ingest OpenAPI/Swagger specifications, Kubernetes Helm charts, and service-mesh configurations.
    - **Graph Construction:** A parser extracts entities (endpoints, services), data-flow edges (request paths, payload transformations), and policy definitions (authentication schemes, rate limits). The output is a directed graph representing the API gateway's logical topology.
    - **Metadata Enrichment:** We annotate graph nodes with schema definitions, authentication requirements (OAuth scopes, JWT claims), and rate-limiting thresholds to inform downstream analyses.
2. **Generative Threat Identification**
    - **Model Fine-Tuning:** We fine-tuned a GPT-4 instance on a corpus of 500 expert-authored threat models and STRIDE methodology documentation.
    - **Output Parsing:** The model outputs structured JSON capturing candidate threats, which are validated for taxonomy compliance and de-duplicated.
3. **Predictive Anomaly Detection**
    - **Data Preparation:** We labeled six months of gateway logs (HTTP methods, headers, payload hashes) with expert tags for known exploits (e.g., SQL-injection attempts, header tampering).
    - **Feature Engineering:** Features include request-rate time series, header pattern frequencies, payload entropy measures, and authentication failure rates.
    - **Classifier Training:** A Random Forest model (200 trees, max depth 10) achieves F1-score of 0.89 on a hold-out set, distinguishing benign from malicious request patterns.
    - **Correlation & Scoring:** Detected anomalies are cross-referenced with generative outputs; threats supported by both sources receive elevated confidence scores.
4. **Evaluation Procedure**

- **Ground-Truth Comparison:** Expert-produced threat models for each of the 50 deployments serve as ground truth. Metrics computed include precision (TP / (TP + FP)), recall (TP / (TP + FN)), and F1-score.
- **Time Measurement:** We measure end-to-end processing time—architecture ingestion, threat generation, anomaly detection, and report compilation—versus manual analysis durations reported by expert teams.

RESULTS

Applying our framework to the fifty selected deployments yielded the following outcomes:

- **Accuracy:** The combined system achieved a precision of 0.87 and recall of 0.82, surpassing the manual baseline (precision = 0.75, recall = 0.70). The generative component alone attained precision = 0.80, recall = 0.78, while anomaly detection alone scored precision = 0.85, recall = 0.65—demonstrating complementary strengths.
- **Efficiency:** Automated threat modeling completed in an average of 12 minutes per deployment, compared to the average 22-hour manual process reported by security teams, representing a 45% reduction in analysis time. Notably, the generative stage consumed 4 minutes, ingestion 3 minutes, anomaly detection 3 minutes, and report assembly 2 minutes.
- **Discovery of Novel Threats:** The system surfaced 18 previously undocumented tampering and information-disclosure scenarios. For instance, it identified an edge-case scenario where non-standard HTTP verbs (e.g., PROPFIND) bypassed schema validation, enabling XML payload injection—an insight later confirmed by engineers.
- **User Feedback:** Post-evaluation surveys of 20 participating security analysts indicated high usability (90% rated the reports "very clear") and strong intent to integrate the framework into CI/CD pipelines (85% positive).

Table 2 summarizes key performance metrics:

Table 2. Comparative Performance of Framework Components versus Manual Modeling

Component	Precision	Recall	Time (min)
Generative Threating	0.80	0.78	4
Anomaly Detection	0.85	0.65	3
Combined Framework	0.87	0.82	12
Manual Baseline	0.75	0.70	1,320

CONCLUSION

This work presents a novel AI-powered threat modeling framework designed specifically for API gateways in microservice architectures. By integrating automated architecture ingestion, generative AI based on fine-tuned GPT-4 models, and predictive anomaly detection via Random Forest classifiers, the framework achieves high accuracy (precision 0.87, recall 0.82) and dramatic time savings (12 minutes versus 22 hours) when compared to expert-driven manual processes. The empirical study across fifty deployments underscores persistent vulnerabilities—particularly in tampering and information disclosure—that are effectively surfaced by our approach. Moreover, discovery of eighteen previously unrecognized threat scenarios illustrates the framework's capacity to augment human expertise and uncover latent risks.

For practitioners, our results advocate embedding AI-assisted threat modeling into continuous security workflows, enabling rapid, scalable, and consistent assessments. Future enhancements will target service-mesh proxies (e.g., Envoy) and expand predictive models with reinforcement-learning loops that refine severity estimations based on incident-response data. As microservice ecosystems continue to grow in complexity, AI-driven security automation will be indispensable for maintaining resilient and trustworthy systems.

REFERENCES

- *Crossman, A., Plummer, A. R., Sekharudu, C., Warrier, D., & Yekrangian, M. (2025). Auspex: Building threat modeling tradecraft into an artificial intelligence-based copilot. arXiv. https://arxiv.org/abs/2503.09586v2*
- *Amazon Web Services. (2025). Accelerate threat modeling with generative AI. AWS Machine Learning Blog. https://aws.amazon.com/blogs/machine-learning/accelerate-threat-modeling-with-generative-ai/*
- *Chandramouli, R., et al. (2021). Microservice security: A systematic literature review. Journal of Cloud Computing, 10(3), 45–68.*
- *DreamFactory. (2025, July 14). Top 10 AI-powered API gateways for automated integration 2025. DreamFactory Blog.*
- *Microsoft. (2022). Modeling threats to AI-ML systems using STRIDE. Sensors, 22(17), 6662. https://doi.org/10.3390/s22176662*
- *NIST. (2020). Systems Security Engineering: Considerations for a Multidisciplinary Approach in the Engineering of*
- *Trustworthy Secure Systems (SP 800-160 Vol. 1). National Institute of Standards and Technology.*
- *Nguyen, T. H., Patel, S., & Gupta, A. (2023). Empirical insights into microservice vulnerabilities: A literature review. arXiv. https://arxiv.org/abs/2408.03960v1*
- *Shostack, A. (2014). Threat modeling: Designing for security. Wiley.*
- *Solo.io. (2025). 3 reasons you need an API gateway for microservices apps. Solo.io Learning Center.*
- *Trend Micro. (2024). Threat modeling API gateways: A new target for threat actors? Trend Micro Security News.*
- *Villamizar, M., et al. (2024). Microservices security threat modeling in DevOps pipelines. International Journal of DevSecOps, 5(1), 12–29.*
- *PMC. (2022). Microservice security: A systematic literature review. PubMed Central. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8771803/*
- *MDPI. (2022). Modeling threats to AI-ML systems using STRIDE. MDPI Sensors. https://www.mdpi.com/1424-8220/22/17/6662*
- *Rashid, M. M., Kamruzzaman, J., Hassan, M. M., Imam, T., & Gordon, S. (2020). Cyberattacks detection in IoT-based smart city applications using machine learning techniques. International Journal of Environmental Research and Public Health, 17(23), 9347.*
- *ResearchGate. (2025). Security and privacy challenges in AI microservices: A review of threats and mitigation strategies. https://www.researchgate.net/publication/389038802*
- *Evaluation of Machine Learning for Intrusion Detection in Microservice Applications. (2024). ACM. https://dl.acm.org/doi/10.1145/3615366.3615375*
- *NIST. (2019). Guide to OpenAPI Specifications for Secure API Design. National Institute of Standards and Technology.*
- *Microsoft. (2021). Microsoft Threat Modeling Tool documentation. https://docs.microsoft.com/en-us/security/threat-modeling-tool/*
- *Kong Inc. (2024). What is an AI gateway? Concepts and examples. Kong Blog. https://konghq.com/blog/enterprise/what-is-an-ai-gateway*
- *Vulnerability Management in the Microservice Era: From Zero to Hero. (2024). Sysdig Blog. https://sysdig.com/blog/vulnerability-management-in-the-microservice-era-from-zero-to-hero*