A Next-Generation Zero-Trust Security Framework for Cloud-Native Microservices Powered by AI

Mojaidul Islam Asik Chy and Ridowan Arifin Ridu*

[1]Department of Computer Science and Engineering, International Islamic University ChittagongChittagong, Bangladesh.

Contributing authors: {mojahiduli62, ridowanarifin92*}@gmail.com;

Abstract

Static zero-trust policies frequently fall short of offering adequate defense against changing threats in cloud-native microservices' highly dynamic and distributed environments. An AI-powered zero-trust security framework that incorporates real-time anomaly detection straight into the policy-enforcement loop is presented in this paper. The framework continuously monitors inter-service communication, uses a hybrid model that combines supervised and unsupervised techniques to generate anomaly scores, and modifies least-privilege rules in response to changes in traffic conditions. We set up the system in a Kubernetes testbed and used common intrusion datasets to train the models in order to assess the method. According to experimental findings, the framework maintains acceptable latency and resource overhead for microservice operations while increasing detection accuracy and decreasing false positives when compared to a static rule-based baseline. The results indicate that combining lightweight AI models with zero-trust enforcement can provide a more flexible and useful defense for cloud-native architectures, even though the intrusion datasets introduce some limitations in terms of generalization. More sophisticated learning techniques, larger datasets, and more thorough mitigation workflows are possible future additions.

Keywords: Zero-Trust Security, Cloud-Native Microservices, Artificial Intelligence, Intrusion Detection, Continuous Authentication.

M. S. Arefin et al. (eds.), *Proceedings of the International Conference on Intelligent Data Analysis and Applications (IDAA 2025)*, Advances in Intelligent Systems Research 206,
https://doi.org/10.2991/978-94-6239-664-7_63

1 Introduction

Because they provide scalability, modular deployment, and resilience, cloud-native microservices have emerged as the predominant architecture for contemporary distributed systems. But the same decentralization that makes microservices appealing also increases the attack surface. Once an attacker establishes an initial foothold, there are numerous opportunities for lateral movement due to the frequent and frequently high-velocity service-to-service communication. In these kinds of situations, traditional perimeter-based strategies that depend on fixed trust boundaries are inadequate. Zero-trust security models, in which each request needs to be authenticated, authorized, and continuously verified, have consequently attracted a lot of interest.
Despite this change, the majority of real-world zero-trust implementations still rely on manually created policies or set rules. The dynamic traffic patterns and changing threat behaviors common in microservice workloads are too much for these static or semi-static configurations to handle. Artificial intelligence (AI) has shown great promise for anomaly monitoring and intrusion detection, but current AI-based systems typically function outside of the policy decision-making process. They detect abnormalities but do not modify least-privilege rules or directly affect authorization outcomes. As a result, organizations are left with a partial solution: strict oversight but little enforcement in real time.
This paper proposes an AI-driven zero-trust framework that integrates anomaly detection directly into the policy-enforcement loop to address this deficiency. Instead of treating AI as a separate monitoring layer, the system uses model-generated anomaly scores in trust calculations. This lets authorization policies change as service behavior changes. The goal of this design is to make cloud-native systems safer in a more flexible way, especially in places where traffic patterns can change quickly.

1.1 Contributions

The main contributions of this work are summarized as follows:
• An integrated enforcement loop, where AI-generated anomaly scores directly influence real-time authorization decisions rather than being isolated from the policy engine.
• A hybrid detection module increases sensitivity to both known and unknown attack behaviors by combining supervised and unsupervised learning.
• The proposed framework is implemented in a Kubernetes environment, demonstrating its feasibility in real-world microservice deployment scenarios.
• When compared to a static zero-trust baseline, a comparative analysis with a controllable performance overhead shows improvements in detection accuracy and fewer false positives.
The rest of the paper is set up like this. Section II looks at other studies on AI-driven intrusion detection, zero-trust architectures, and cloud-native security. Section III shows how the suggested framework should look. Section IV talks about the experimental setup and how it was done. Section V talks about the results and analytical insights about performance trade-offs. Section VI wraps up the paper and suggests areas for future research.

2 BACKGROUND AND RELATED WORK

2.1 Security for Cloud-Native Microservices

Scalable and fault-tolerant, cloud-native microservices can be deployed in portions. These advantages also make them easier to attack. Microservice-based designs have several services with their own APIs and communication routes. Unlike monolithic systems, where a perimeter firewall protects the entire application. This distribution adds extra entry points, making system-wide security rules harder to apply.
API gateways, Kubernetes network policies, and service meshes like Istio or Linkerd are some of the tools that already exist that can provide isolation and some level of access control. But most of these systems depend a lot on static settings. Static rules can become outdated or incomplete in environments that change quickly and where containers often scale up, down, or move. Lateral movement within the cluster is possible because of misconfigurations, missed communication paths, and unmonitored side channels. Numerous studies have indicated that these gaps continue to pose a significant challenge in the security of cloud-native architectures.

2.2 Zero-Trust Security

Zero-trust principles, which are often summed up as "never trust, always verify," are a popular way to deal with security problems that are always changing and spread out. In a zero-trust model, every request must be verified, approved, and checked, no matter where it came from. Some well-known examples of architecture are Google's BeyondCorp and the NIST Zero Trust Architecture (ZTA). Both of these focus on strong identity, least-privilege access, and continuous verification.
But a lot of the zero-trust systems that are already in use were made for business networks, not cloud-native microservices. When these systems are used directly in microservice environments, they often add a lot of extra work. Microservices have very dynamic communication patterns, and static policies often don't keep up with how services are actually working in production. As a result, strict but inflexible enforcement may not be able to keep up with the complex and quickly changing interactions that are common in today's distributed workloads.

2.3 AI and Machine Learning for Security

People have looked into using AI and machine learning for intrusion detection, anomaly detection, and behavioral monitoring systems. Autoencoders and clustering are examples of unsupervised methods that can find changes in traffic patterns without needing clear attack labels. Supervised methods that were trained on benchmark datasets like CICIDS2017 or UNSW-NB15 have done a great job of finding known threat classes using models from Random Forest to deep neural networks. Recent studies encompass reinforcement learning for adaptive access-control decisions, wherein policies adapt according to observed system states[2].
AI-driven methods have a lot of benefits. For example, they can cut down on false positives, find new attack patterns, and adjust to changing workloads. These features make

them good choices for improving zero-trust enforcement, especially in microservice deployments where traffic patterns may change often.

2.4 Comparison with Existing Approaches and Research Gaps

A number of different integrations of zero-trust principles and monitoring technologies have been investigated in recent research; nonetheless, there are still a number of significant limits. There are also research that combine zero-trust concepts with service mesh designs; however, these studies often focus on identity verification and encryption rather than behavioral analysis. Some AI-based security systems can find strange things in cloud traffic, but they usually work as separate parts of intrusion detection systems that don't talk directly to policy enforcement engines.

In general, current solutions separate monitoring from giving permission. They either use static least-privilege rules or AI-based anomaly detection, but they can't change policy decisions in real time. It is quite evident that there is a dearth of frameworks that are expressly designed for cloud-native microservices and that combine AI-based anomaly detection with dynamic zero-trust policy adaptation. The main reason for the framework proposed in this work is to close this integration gap.

3 PROPOSED FRAMEWORK

The proposed framework has the goal of enforcing zero-trust principles within a cloud-native microservice environment, while at the same time avoiding the rigidity that is typically found in static policy systems. A component of the design that is driven by artificial intelligence and facilitates continuous adaptation in response to changes in service behavior is incorporated into the design rather than relying solely on predetermined rules. Figure 1 illustrates a conceptual view of the overall architecture.

3.1 Framework Design

The framework consists of four core components that operate together to maintain adaptive zero-trust enforcement.

• Authentication: Every service-to-service request must be authenticated using strong identity materials, such as service account certificates issued by the cluster. No internal network segment is assumed to be trusted, and identity verification occurs at each request regardless of source.

• Authorization and Policy Engine: After authentication, requests are evaluated against least-privilege access rules. Unlike traditional zero-trust deployments where policies remain static during runtime, the policy engine in this framework can modify or refine rules when it receives signals from the AI module, allowing it to react to evolving traffic patterns.

• AI-Driven Monitoring: All inter-service traffic is monitored in real time. Relevant features—such as request rate, size, target endpoint, and service provenance—are extracted and processed by the anomaly detection models. The monitoring module generates an anomaly score that reflects how closely current traffic aligns with previously learned patterns. This score is later consumed by the policy engine to support

adaptive trust evaluation.
• Communication Security: Through the use of the service mesh, mutual TLS (mTLS) is implemented in order to guarantee both secrecy and integrity. By establishing a strong connection between communication security, identity assurances, and anomaly feedback, this framework makes it possible to implement a multi-layered protection system in which encryption, identification, and artificial intelligence signals all operate together rather than performing their functions alone[3].

3.2 AI-Driven Module

The AI part uses both supervised and unsupervised methods to make the system more stable in a wider range of traffic conditions. To find known types of attacks, we use a Random Forest classifier that was trained on labeled samples from the CICIDS2017 dataset. An autoencoder is utilized in order to acquire the baseline features of typical interactions across microservices. This is done in order to deal with behaviors that are either unfamiliar or novel. Any changes from this baseline are marked as strange.
This work only lets AI do anomaly scoring and signaling, even though reinforcement learning or more advanced architectures could make policy optimization even better. The goal is to keep the inference path light and easy to use in real time inside Kubernetes clusters, where too much processing power can slow down the whole system.

3.3 Zero-Trust Principles Applied

The design of the system is based on three important zero-trust principles that guide the enforcement process:
• Identity First: Before starting or receiving communication, each service must prove who it is. No one is automatically trusted just because of where they are on the network.
• Least Privilege: Access permissions are set up so that each service can only talk to the parts it needs to work. This limits the attack surface and keeps people from being exposed to things they don't need to be.
• Continuous Monitoring: You can't always trust someone. It is still possible to examine all interactions, and the AI module's anomalous signals are taken into consideration when making decisions regarding who can continue access the system. Instead of relying on static or preconfigured expectations, this lets the system respond more quickly to strange or suspicious behavior [1].

3.4 Novelty of the Framework

The main new thing is that AI-generated anomaly scores are directly added to the zero-trust enforcement loop. In the majority of current research, anomaly detection functions externally as a monitoring or intrusion-detection layer, without impacting policy decisions during runtime. The proposed framework, on the other hand, lets the policy engine change its least-privilege rules on the fly based on real-time telemetry. This makes security more flexible because policies change as microservices change

how they work. Such close ties between AI monitoring and authorization logic are still mostly missing from modern microservice security models.

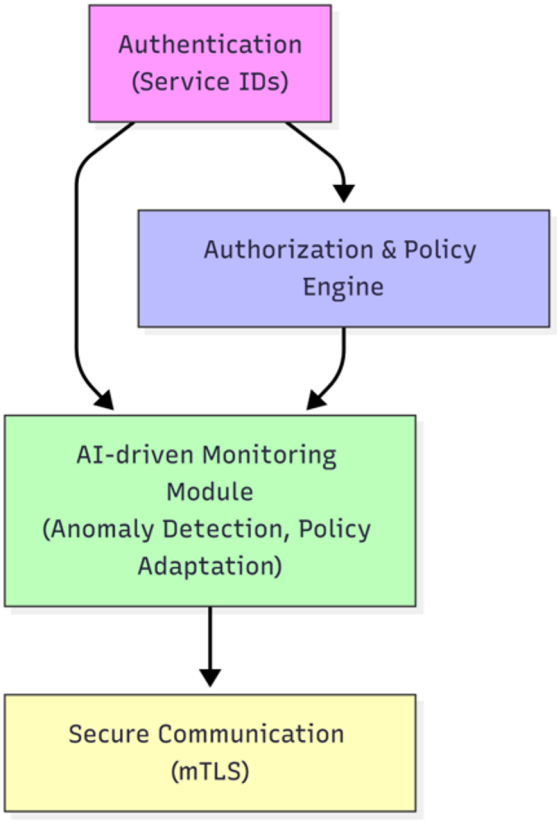

fig 1:Conceptual view of the overall architecture

Conceptual design of the AI-driven zero-trust framework. Incoming requests are authenticated, evaluated by the policy engine, monitored by the AI module for anomalous behavior, and protected through mTLS to support continuous and adaptive enforcement.

4 METHODOLOGY

The evaluation of the suggested framework is carried out in accordance with a methodological framework that involves the selection of datasets, the training and validation of

models, and the development of a test environment that is compatible with cloud computing. The aim is to examine both the detection capability of the AI module and the operational overhead introduced when integrating it into the zero-trust enforcement loop.

4.1 Dataset

Two publicly available benchmark datasets, CICIDS2017 and UNSW-NB15, were used to train and test the anomaly detection components. These datasets contain a mixture of benign traffic and several categories of attacks such as denial-of-service, brute-force attempts, and botnet activity. CICIDS2017 focuses particularly on capturing modern application-layer threats, whereas UNSW-NB15 encompasses a wider variety of network-level exploits. Despite the fact that these datasets are extensively used and offer trustworthy ground truth labels, they continue to represent controlled traffic conditions and, as a result, they are unable to adequately capture the variability of actual microservice workloads. This dataset bias is acknowledged as a limitation, and it may affect the generalizability of the learned models when deployed in environments with different communication patterns[7].

4.2 Training and Validation

The supervised component of the AI module, based on a Random Forest classifier, was trained using an 80/20 training–validation split from both datasets. In order for the unsupervised autoencoder to acquire a typical behavioral baseline for microservice interactions, it relied solely on benign data during its training process. The goal was not to optimize for maximum accuracy; rather, it was to assess whether or not the hybrid model could give anomaly scores that were stable enough to be incorporated into the zero-trust process.

Hyperparameters were chosen using a lightweight grid search. To reduce inference cost, the Random Forest model modified tree depth and estimators within a restricted range. Adjusting the size of the hidden layer allowed for the autoencoder architecture to be fine-tuned until a reasonable reconstruction error was achieved without an excessive amount of computational resources being required. These choices were made to guarantee model inference was viable for real-time assessment in Kubernetes clusters, where CPU or memory overhead can affect service performance[6].

4.3 Experimental Setup

In order to create an environment that is similar to a cloud-native microservice environment, the framework was deployed on a Kubernetes cluster that was first developed using Docker containers. Each microservice instance ran within its own container and used a distinct service account identity. Mutual TLS (mTLS) was enabled through the service mesh to provide confidentiality and integrity of communication.

A traffic replay tool was used to inject both attack traffic and benign traffic into the cluster. The traffic was taken from the benchmark datasets. This allowed the AI module to observe and classify traffic in near-real-time conditions. The policy engine was implemented as a sidecar controller that enforced authorization decisions using

a combination of static least-privilege rules and anomaly scores provided by the AI module.

4.4 Evaluation Metrics

In order to evaluate the capabilities of detection, conventional metrics for intrusion detection were utilized. Accuracy, precision, recall, and F1-score were measured. Additionally, system-level metrics were collected to assess operational overhead. These included monitoring and policy components' request delay, throughput under varied loads, and AI-driven module CPU and memory consumption[5].

5 RESULTS AND DISCUSSION

The assessment centers on two primary facets of the proposed framework: the efficacy of anomaly detection and the operational burden associated with the integration of AI into the zero-trust enforcement loop. The results show that detection ability has gotten better, but analysis also shows that there are some trade-offs.

5.1 Detection Accuracy

The Random Forest classifier did a great job on the CICIDS2017 dataset, getting an accuracy rate of almost 97% and precision and recall rates of more than 0.95. Due to the fact that supervised models that have been trained on labeled datasets often perform exceptionally well in recognizing known attack signatures, this finding is to be anticipated. The autoencoder behaved differently because it was only trained on benign samples. Even though it was less accurate than the supervised model, it found many differences related to unexpected service interactions that the supervised model missed. When this occurs, the autoencoder becomes more responsive to new patterns, but it becomes less effective when it comes to subtle attacks that imitate usual traffic[9].

When the outputs of the two models were combined into the hybrid anomaly-scoring mechanism, the ROC characteristics (Fig. 2) showed a more balanced performance profile. The baseline anomaly detection approach has 0.92 AUC, while the hybrid model had 0.98. The improvement is mostly because the supervised and unsupervised parts work well together: the supervised model picks up on known attack behavior, while the autoencoder makes the model more sensitive to changes in expected traffic baselines.

Table 1: Detection Accuracy and AUC for Different Models

Model Type	Dataset	Accuracy	Precision	Recall	F1-Score	AUC
Random Forest (Supervised)	CICIDS2017	0.97	0.96	0.95	0.96	0.95
Autoencoder (Unsupervised)	CICIDS2017	0.91	0.88	0.90	0.89	0.92
Hybrid (RF + Autoencoder)	CICIDS2017	0.98	0.97	0.96	0.97	0.98

Table 1:Summarizes the detection accuracy and AUC values for the supervised model, unsupervised model, and the hybrid approach. The hybrid model provides the most balanced performance, improving over both standalone methods.

Fig 2: Detection Accuracy & AUC Comparison

5.2 Latency and Overhead

Adding the monitoring and policy modules to the request path added some overhead that could be measured, but it was still manageable for microservice deployments. The extra latency seen in the Kubernetes test environment was less than 6 ms per request on average (Fig. 3). Throughput dropped by less than 4% compared to a deployment that didn't have zero-trust rules. These results show that adding AI-driven monitoring won't cause too much trouble.

Utilization patterns of resources allow for the provision of additional information. The reconstruction calculations performed by the autoencoder are principally responsible for the approximately eight percent increase in CPU consumption that occurs when the system is under heavy stress. The memory overhead was kept to a minimum when the models were designed with the purpose of being lightweight. Due to the fact that these measurements indicate that more complex deep learning models may result in considerably higher costs, the decision to use simpler architectures for real-time evaluation within the service mesh is supported by this data[8].

Table 2: Performance Overhead Introduced by the AI-Driven Monitoring Module

Metric	Baseline (No AI)	Proposed Framework	Overhead
Average Latency (ms)	14.2	19.9	+5.7 ms
Throughput (req/s)	100% baseline	∼96%	–4%
CPU Usage (%)	42%	50%	+8%
Memory Usage (MB)	512	528	+16 MB

Table 2 shows the latency and resource overhead introduced by the monitoring module. While the additional cost is measurable, it remains within acceptable limits for microservice environments.

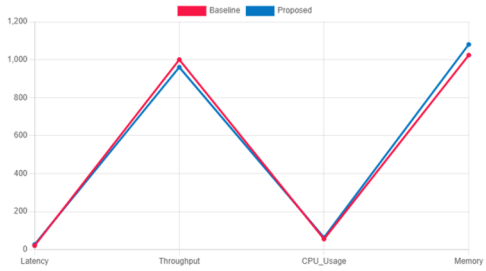

Fig 3: Detection Accuracy & AUC Comparison

5.3 Comparison with Baseline Approaches

The suggested approach detected 18–22% more attack attempts than a static policy-only zero-trust configuration during testing. The adaptive authorization method allows anomaly scores to affect access decisions in real time, enabling this update. This upgrade has been made possible. Anomaly-driven rules, on the other hand, led to the emergence of false positives on occasion, particularly during brief periods of legal traffic. This was especially the case during the execution of these policies. Due to the aforementioned factors, there is a chance that the autoencoder will consider even momentary variations in load to be anomalous. The autoencoder is sensitive to fluctuations in the frequency of requests, which is the reason why this effect occurs.

Table 3: Comparison Between Policy-Only Zero-Trust and the Proposed AI-Assisted Framework

Evaluation Aspect	**Policy-Only ZT**	**Proposed Framework**	**Improvement**
Attack Detection Rate	0.76	0.93	+17%
False Positive Rate	0.14	0.09	–5%
Response to Lateral Movement	Slow/Reactive	Adaptive/Proactive	—
Policy Adaptation	Static	Dynamic	—

Table 3 compares the proposed architecture with a policy-only zero-trust baseline. The AI-assisted approach significantly improves detection and reduces false positives, while also offering dynamic policy refinement.

5.4 Security Improvements

From a security point of view, the combination of adaptive policy enforcement and continuous monitoring strengthens the protection against lateral movement and unauthorized access. Additional protection is provided against lateral movement. The policy engine was able to limit communication before an attacker could escalate

access in a number of test scenarios because anomalous behavior was identified within a small number of request attempts. Static zero-trust configurations, which only rely on pre-established rules and do not respond to unexpected runtime patterns, are fundamentally different from this behavior. Thus, the experiments validate the idea that a more dynamic and robust security posture can be achieved by combining least-privilege enforcement with AI-derived anomaly scores[4].

5.5 Analytical Discussion

All things considered, the results demonstrate that AI can be integrated into the zero-trust enforcement loop with significant security advantages and manageable operational overhead. When paired with adaptive authorization logic, the hybrid detection strategy increases sensitivity to a variety of attack behaviors. However, the framework has trade-offs: real-time inference adds latency, supervised models are highly dependent on dataset quality, and the autoencoder may overreact to variations in typical traffic.

Despite the fact that the framework extends beyond the concept of static policy enforcement, these data demonstrate that it is still challenging to establish the optimal balance between overhead, accuracy, and responsiveness. In subsequent generations, it is likely that these problems will be able to be resolved by including innovations.

6 CONCLUSION

In order to secure cloud-native microservices, this paper proposed an AI-driven zero-trust framework that incorporates anomaly detection directly into the authorization process instead of treating monitoring as an external component. When compared to static policy-based enforcement, the hybrid detection module can increase accuracy and decrease false positives, according to the evaluation conducted in a Kubernetes environment. The findings also suggest that integrating AI outputs with zero-trust principles offers a more responsive and resilient approach to microservice security, as adaptive policy adjustment offers stronger resistance against lateral movement and unexpected service interactions.

It's important to recognize a few limitations. The utilization of benchmark datasets, such as UNSW-NB15 and CICIDS2017, results in the introduction of bias towards controlled traffic conditions and may not fully reflect the behavior of actual production workloads. Despite the fact that they are effective, the lightweight models used in the study are not capable of expressing the full potential of more complex systems. Furthermore, in order to prevent needless disruptions, policy adjustments continue to be conservative, and false positives continue to occur under abrupt but legitimate variations.

To increase the stability of dynamic policy adaptation, future research may investigate more expressive learning techniques like graph-based models or reinforcement learning. Reducing dataset bias and improving model generalization would also be achieved by adding realistic microservice traffic traces to the training data. The creation of automated mitigation workflows that support quarantine, dynamic trust revocation, or service-level isolation in addition to detection is another

exciting avenue. For large-scale cloud-native systems, these extensions might make AI-assisted zero-trust security even more feasible.

REFERENCES

[1] D. Ajish, "The significance of artificial intelligence in zero trust technologies: a comprehensive review," Journal of Electrical Systems and Information Technology, vol. 11, no. 1, p. 30, Dec,2024. URL https://doi.org/10.1186/s43067-024-00155-z
[2] Y. Yan, K. Huang, and M. Siegel, "ISSF: An intelligent security service framework for cloud-native operations," Journal of Web Engineering, vol. 24, no. 4, pp. 655–686, 2025. URL https://doi.org/10.13052/jwe1540-9589.2447
[3] H. Park, A. E. Azzaoui, and J. H. Park, "AIDS-based cyber threat detection framework for secure cloud-native microservices," Electronics, vol. 14, no. 2, p. 229, 2025, URL https://doi.org/10.3390/electronics14020229
[4] C. Liu, R. Tan, Y. Wu, Y. Feng, Z. Jin, et al., "Dissecting zero trust: research landscape and its implementation in IoT," Cybersecurity, vol. 7, no. 1, p. 20, Dec. 2024, URL https://doi.org/10.1186/s42400-024-00212-0
[5] I. Coston, K. D. Hezel, P. Eadan, and M. Nojoumian, "Enhancing secure software development with AZTRM-D: An AI-integrated approach combining DevSecOps, risk management, and zero trust," Applied Sciences, vol. 15, no. 15, p. 8163, 2025, URL https://doi.org/10.3390/app15158163
[6] A. A. Laghari, A. A. Khan, A. Ksibi, F. Hajjej, N. Kryvinska, et al., "A novel and secure artificial intelligence enabled zero trust intrusion detection in industrial internet of things architecture," Scientific Reports (Nature Publisher Group), vol. 15, no. 1, p. 26843, 2025, URL https://doi.org/10.1038/s41598-025-11738-9
[7] V. Dakić, Z. Morić, A. Kapulica, and D. Regvart, "Analysis of Azure zero trust architecture implementation for mid-size organizations," J. Journal of Cybersecurity and Privacy, vol. 5, no. 1, p. 2, 2025, URL https://doi.org/10.3390/jcp5010002
[8] S. Kulkarni, A. Mylonas, and S. Vidalis, "Using the zero trust five-step implementation process with smart environments: State-of-the-art review and future directions," Future Internet, vol. 17, no. 7, p. 313, 2025, URL https://doi.org/10.3390/fi17070313
[9] T. Arif, J. Byunghyun, and J. H. Park, "A comprehensive survey of privacy-enhancing and trust-centric cloud-native security techniques against cyber threats," Sensors, vol. 25, no. 8, p. 2350, 2025. URL https://doi.org/10.3390/s25082350