International Journal of Web of Multidisciplinary Studies
( Peer-Reviewed, Open Access, Fully Refereed International Journal )
website: https://ijwos.com
Volume 3 Issue 5 , May 2026
E-ISSN : 3049-2424
DOI: https://doi.org/10.71366/ijwos

# AI-Driven Secure Network Slicing for Cloud-Native 6G Networks Using Zero-Trust Architecture and Intelligent Intrusion Detection

Akhlaque Hussain[1], Sagar Choudhary[2]
[1] *B.Tech Student, Department of CSE, Quantum University, Roorkee, India.*
[2] *Assistant Professor, Department of CSE, Quantum University, Roorkee, India.*

***Article Info***

***Article History:***

*Published:29 May 2026*

***Publication Issue:***
*Volume 3, Issue 5*
*May-2026*

***Page Number:***
*564-577*

***Corresponding Author:***
*Akhlaque Hussain*

***Abstract:***

6G will reshape digital systems by blending AI, cloud-native orchestration, edge computing, ultra-low latency, and smarter network automation. A key enabler is network slicing, which runs multiple virtual networks on the same hardware to meet diverse needs like enhanced mobile broadband, ultra-reliable low-latency links, and massive IoT connections. But slicing's distributed, cloud-first design opens new attack paths—cross-slice breaches, compromised orchestration, DDoS, API abuse, lateral movement, and attacks on AI components. This work proposes an AI-based, zero-trust defense that ties together Kubernetes, SDN, NFV, Istio service-mesh segmentation, and ML-powered intrusion detection. It continuously checks identities, mTLS-protected channels, runtime telemetry, and adaptive trust scores. Multiple ML and deep models (Random Forest, XGBoost, LSTM, Autoencoders) were tested on telecom datasets including 5G-SliciNdd, 5G-NIDD, 5GCID, UNSW-NB15, and CICDDoS2019; XGBoost led in spotting DDoS, orchestration faults, and unauthorized slice traffic. The design boosts visibility, speeds response, and scales for AI-native 6G deployments.

***Keywords:*** 6G, Network slicing, Zero trust security, Cloud native orchestration, Kubernetes, SDN, NFV, Istio service mesh, mTLS, ML intrusion detection, XGBoost, DDoS detection, Runtime telemetry

## 1. Introduction

Faster wireless tech is reshaping global connectivity and networks. After wide 5G rollout, research shifts to 6G for ultra-low latency, massive IoT, AI, XR, digital twins. [1], [3]. Unlike earlier generations, 6G will be AI-first, combining cloud orchestration and edge computing. [2], [4].

Network slicing is vital for 5G-Advanced and 6G. **Network slicing** runs multiple virtual networks on the same hardware, meeting diverse **QoS** needs—bandwidth, latency, security—and supports **eMBB**, **URLLC**, **mMTC**.[7]. Slicing improves flexibility, efficiency, and customization.Modern slicing relies on SDN, NFV, Kubernetes orchestration, and SBA.[9], [10]. CNFs, microservices, and MEC replace monolithic telecom stacks for scalable, agile deployments.[1], [2]. But they cause major security risks: orchestration, APIs, service-mesh, inter-slice.[13], [3].

Cloud-native telecom stacks and network slicing bring flexibility and scale, but they also widen the attack surface. Shared hardware and virtualized software make DDoS, side-channel exploits, orchestration takeovers, telemetry tampering, hypervisor breaches, and unauthorized lateral moves between slices real threats. Adding AI into the mix creates fresh risks like model poisoning and adversarial manipulation of orchestration[4]. Traditional perimeter defenses fall short because they don't continuously verify identities, monitor runtime behavior, or adapt to changing telemetry. A promising fix is a zero-trust approach that constantly authenticates workloads, services, APIs, users, and connections while using micro-segmentation, mTLS, and adaptive policies to limit lateral spread. Pairing that with AI-based detection—models such as Random Forest, XGBoost, LSTM, and Autoencoders trained on telecom datasets—helps spot orchestration faults, novel attacks, and anomalous traffic[16],[13],[17],[8],[5]. This work proposes a unified, cloud-native zero-trust design that combines Kubernetes, SDN/NFV, Istio service-mesh segmentation, ML intrusion detection, and SIEM-aware telemetry to improve slice isolation, speed up responses, and enable practical evaluation with realistic datasets[1], [8], [2], [5], [7], [8].

## 2. Literature Review

Rapid 5G advances and early 6G work have pushed interest toward securing network slicing, cloud-native telecom stacks, zero-trust designs, and AI-based intrusion detection. Studies examined virtualization and orchestration risks, O-RAN and API exposures, and the need for slice-aware isolation. Showed cloud zombie detection plus fast orchestration quarantine cuts spread and outages[19]**.** Kubernetes and service mesh tools like Istio are studied for orchestration and mTLS-based service protection, while SDN/NFV integration raises orchestration and inter-slice security concerns. Zero-trust principles—continuous authentication, least privilege, and runtime verification—are being adapted for future networks. Researchers also apply ML and deep models (Random Forest, XGBoost, LSTM, Autoencoders) to analyze telemetry and detect novel attacks. However, many solutions still target 5G rather than fully cloud-native, AI-first 6G, and realistic datasets and testbeds remain scarce[3], [4], [8], [9], [10], [11], [12]. This paper proposes a unified AI-driven zero-trust framework combining orchestration security, mesh micro-segmentation, adaptive trust, ML detection, and SIEM-aware telemetry for practical evaluation[1], [8], [2], [4], [7], [5].

## 3. Previous Technology

A. Evolution of 5G and 6G Communication Systems

Wireless tech has rapidly changed how systems connect and share data, making networks smarter, faster, and more distributed. 5G brought big gains—more capacity, lower delays, virtualized functions, and smarter service delivery—which let operators scale and automate services using SDN, NFV, edge computing, and cloud orchestration[3], [8], [9], [10]. New use cases like self-driving vehicles, immersive XR, industrial automation, holograms, and digital twins push past 5G's limits, so 6G is being designed for ultra-low latency, terahertz links, on-device and edge AI, and highly adaptive network control. Unlike earlier generations, 6G targets autonomous, AI-first operation with fully virtualized stacks and distributed intelligence[1], [4]. These advances promise better flexibility and service delivery across healthcare, smart cities, industrial IoT, autonomous systems, and immersive experiences[1], [3].

B. Network Slicing in Cloud-Native Telecom Infrastructures

Network slicing lets operators run several isolated virtual networks on the same physical gear, each tuned for different needs like high-speed mobile, ultra-low latency, or massive IoT. This virtualization boosts resource use, scalability, and service customization[8], [10]. Modern slices are managed with containerized microservices and orchestration platforms—Kubernetes is common because it scales, heals, and automates telecom workloads[1]. In these setups, CNFs run in separate namespaces while orchestration handles policies, scaling, discovery, and resource assignment[7], [10]. API-based Service-Based Architecture improves component interoperability, and Istio-style service meshes add mTLS, traffic control, telemetry, and micro-segmentation[2]. Tying SDN, NFV, orchestration, and service mesh creates flexible, programmable stacks for AI-ready 6G—but sharing orchestration and APIs also raises serious security risks like container escape, mesh compromise, and weak isolation[8], [5], [13].

C. Security Challenges in 6G Network Slicing

Network slicing boosts scale and flexibility, but it also widens the attack surface in cloud-native telecom. Since multiple virtual networks rely on the same orchestration and runtime layers, a breach in one slice can ripple across services[7], [8]. Big threats include traffic-flooding DDoS attacks that swamp orchestration and gateways, exploits of exposed APIs or weak auth that enable privilege escalation and cross-slice movement, and container or hypervisor compromises that let attackers escape isolation[5], [13], [12]. Adding AI into orchestration brings new risks like poisoned models and adversarial inputs that manipulate automated decisions[4]. Manual monitoring can't keep up with the massive telemetry streams these systems produce, so adaptive, automated security that continuously watches runtime behavior and blocks suspicious activity is essential[7], [6].

D. Zero-Trust Security Architecture

Zero-trust treats every request as untrusted and checks it continuously, authenticating users, services, and APIs and enforcing least-privilege access[3], [4]. It uses identity certificates, mutual TLS, micro-segmentation, and runtime trust checks to monitor behavior and dynamically isolate or block suspicious workloads with adaptive policies[2], [6].

E. AI-Driven Intrusion Detection in Telecom Networks

As wireless systems evolve, research has moved toward securing network slicing, cloud-native telecom stacks, zero-trust defenses, and AI-powered intrusion detection. Work has examined virtualization and orchestration weaknesses, O-RAN and API exposures, and the use of Kubernetes plus service meshes (e.g., Istio) to provide mTLS, telemetry, and micro-segmentation[5], [13], [1], [2]. ML and deep models—Random Forest, XGBoost, LSTM, Autoencoders—are used to analyze telemetry and spot novel attacks, while SIEM platforms help correlate events across layers. Still, many solutions target legacy 5G setups, and realistic 6G datasets and testbeds are scarce. This study proposes a unified, cloud-native zero-trust design that ties orchestration security, mesh segmentation, adaptive trust policies, ML detection, and SIEM-aware telemetry for practical evaluation[1], [8], [2], [4], [7], [5].

## 4. Datasets Used in This Research

A. Overview of Dataset Selection

Network-grade datasets are crucial for AI-based intrusion detection. Since full-scale 6G testbeds aren't available yet, we rely on public 5G-focused collections to emulate cloud-native slicing and runtime behavior. These combined sources cover traffic logs, slice-level communication, DDoS samples, orchestration errors, API abuse, malicious sessions, and runtime anomalies. Blending telecom-specific and general IDS datasets increases diversity and helps models learn both normal and attack patterns[37], [44]. This work uses 5G-SliciNdd, 5G-NIDD, 5GCID, UNSW-NB15, and CICDDoS2019 to train and evaluate ML detectors in distributed orchestration scenarios[8], [9], [10], [11], [12].

B. 5G-SliciNdd Dataset

5G-SliciNdd is a telecom IDS collection built to mimic slice-level traffic in virtualized networks, including normal flows and attack traces like DDoS and unauthorized access. It provides telemetry, flow stats, auth events, and runtime patterns, making it useful for testing adaptive detectors and cross-slice defenses[8].

C. 5G-NIDD Dataset

5G-NIDD contains authentic traffic logs from 5G setups, mixing benign and malicious flows for machine-learning evaluation. It offers flow metadata, protocol records, network statistics, authentication telemetry, DDoS samples, and service behavior logs. We use it mainly to test DDoS detection and runtime anomaly classification[9].

D. 5GCID Dataset

5GCID targets 5G core environments, recording signaling, service interactions, orchestration events, and attack traces. It includes core telemetry, API call logs, authentication records, signaling messages, resource-usage metrics, and samples of malicious orchestration. Because modern networks use API-driven services and decentralized orchestration, 5GCID is handy for testing detectors that need core-level and orchestration visibility[8], [7], [10].

E. UNSW-NB15 Dataset

UNSW-NB15 is a popular IDS collection with modern attack types and realistic traffic, covering fuzzers, backdoors, reconnaissance, shellcode, DoS, worms, and generic malicious flows. It includes packet and flow telemetry, protocol and metadata features, and rich network records, so we use it to help models generalize beyond telecom-only threats[11].

F. CICDDoS2019 Dataset

CICDDoS2019 focuses on DDoS research, offering realistic, large-scale attack captures across multiple flooding and reflection scenarios alongside normal traffic. It contains TCP SYN, UDP, and HTTP floods, reflection/amplification samples, and benign sessions. Each record includes packet and flow statistics, durations, rates, endpoint identifiers, and behavioral metrics, so researchers use it to test adaptive DDoS defenses and runtime anomaly detectors[12].

G. Dataset Preprocessing and Normalization

Before training, we clean and standardize every dataset so models learn reliably and avoid bias[14]. The pipeline drops missing entries and duplicate records, converts categorical telemetry into numeric form (one-hot or label encoding), scales numeric features with min–max rescaling, balances skewed classes, and reduces noise[15], [16], [44]. Min–max scaling maps a value X to $X_{norm}=X-X_{min}\ X_{max}\ -X_{min}$ . To address rare attack classes we apply SMOTE to synthesize minority samples, which improves fairness and helps models converge faster and more stably[17], [18].

H. Importance of Telecom-Oriented Datasets

Network-focused datasets make intrusion detection testing far more realistic for cloud-native slice deployments[37]. Generic enterprise collections miss slice-level traffic, orchestration events, service-mesh telemetry, signaling flows, and AI-driven orchestration behaviors. Combining telecom-specific and general IDS datasets gives broader coverage for evaluating anomaly detectors, orchestration security, and cross-slice resilience in future 6G systems[8], [9], [10], [11], [12], [44].

## 5. Proposed Architecture

A. Architecture Overview

We present an AI-powered zero-trust defense for cloud-native 6G slices that guards against sophisticated attacks, orchestration failures, and cross-slice misuse[1], [8], [4]. It combines secure orchestration, adaptive trust controls, ML-based intrusion detection, service-mesh safeguards, and SIEM telemetry into a single, practical security stack

for AI-native telecom environments[2], [7], [5].

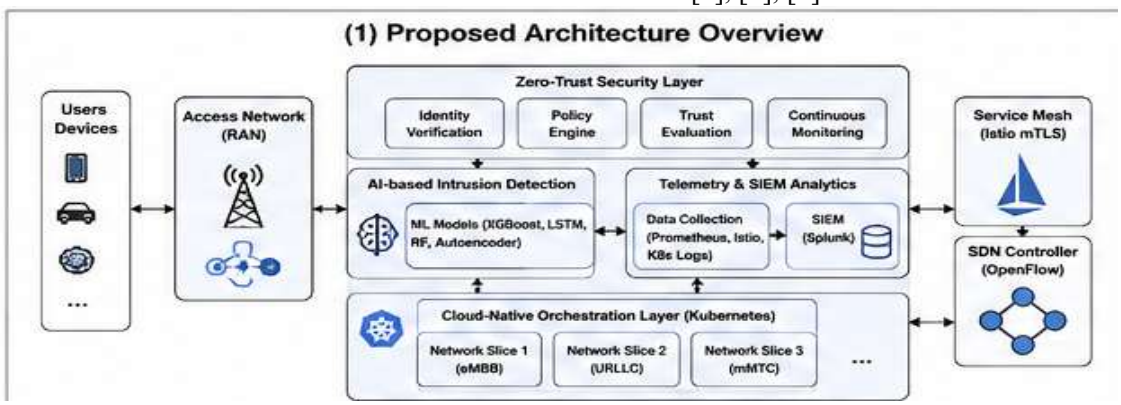


Figure. 1: Proposed Architecture Overview

The design brings together Kubernetes, SDN/NFV, Istio-style service mesh, ML-based detectors, runtime telemetry, and zero-trust controls. It constantly checks identities, inspects live telemetry, and reacts to threats across distributed slices. The stack is organized into infrastructure, orchestration, zero-trust, ML detection, and telemetry/SIEM layers to enable adaptive protection[1], [8], [4], [5], [7].

B. Cloud-Native Telecom Infrastructure Layer

The cloud-native telecom layer is the base that runs virtual network slices and distributed services, built from SDN, NFV, containerized network functions, edge computing, and service-based APIs. It supports slice types like eMBB, URLLC, and mMTC, with each slice given its own policies, resources, authentication, and telemetry. SDN offers centralized, programmable routing while NFV virtualizes gateways and security functions[8], [10]. Microservice CNFs boost deployment agility, but shared orchestration means strong isolation and adaptive defenses are essential[1], [5], [13].

C. Virtualization and Orchestration Layer

The orchestration layer handles slice provisioning, container deployment, and runtime coordination using tools like Kubernetes, Docker, free5GC, SDN, NFV, and service-based APIs[1], [7], [8]. Kubernetes runs each slice in its own namespace with CNFs, quotas, telemetry agents, and security hooks. A service mesh (e.g., Istio) adds mTLS, traffic segmentation, service auth, and observability. Cloud-native orchestration boosts agility but brings risks such as exposed APIs, orchestration takeover, container escapes, privilege abuse, and runtime tampering, so defenses like RBAC, admission controls, workload identities, secure API gateways, and live policy checks are enforced[8], [5], [13], [3], [6].

D. Zero-Trust Security Layer

The zero-trust layer is the framework's main defense: it continuously checks every workload, service, API call, session, and user before allowing access. It blends identity-aware controls, continuous authentication, mTLS, runtime trust scoring, micro-segmentation, least-privilege rules, and adaptive trust policies. Services get cryptographic identities (certificates and mesh verification) and mTLS protects runtime links. Segmentation uses Kubernetes namespaces, service-mesh rules, SDN flow controls, and adaptive firewalls to limit lateral movement [1], [2], [8], [7]. A trust engine combines auth signals, behavioral analysis, and telemetry into a weighted score; on anomalies it tightens policies, blocks or isolates flows, quarantines slices, and raises SIEM alerts[7], [5], [4], [6].

E. AI-Driven Intrusion Detection Layer

The AI-driven intrusion detection layer handles smart anomaly spotting, behavior analysis, attack labeling, and automated response across distributed telecom stacks. Because signature-based tools miss zero-day exploits, orchestration faults, and shifting adversary tactics, this design feeds large-scale telemetry and runtime traces into ML and deep models. It watches slice traffic, auth logs, API telemetry, resource metrics, orchestration events, and container runtime signals[8], [9], [10], [5], [7]. The system tests models like Random Forest, XGBoost, LSTM, and autoencoders — using RF for explainable feature insights, XGBoost for fast structured-data classification, LSTM for temporal sequence detection, and autoencoders for unsupervised deviation spotting[16], [13], [17]. Detected threats (DDoS, cross-slice anomalies, unauthorized links, API misuse, resource exhaustion, orchestration compromise) trigger automated actions such as isolating slices, throttling flows, updating policies, quarantining workloads, and sending SIEM alerts. The telemetry and SIEM layer centralizes observability (Splunk, Prometheus, Grafana, exporters, mesh tools), aggregates logs from Kubernetes, free5GC, Istio, and the AI engine, and runs correlation and behavioral analytics to enable real-time detection and adaptive incident response[5], [1], [2], [7].

G. Workflow of the Proposed Framework

The intrusion detection layer powered by AI handles anomaly spotting, behavior analysis, attack labeling, and automated response across distributed telecom stacks. Because signature-based methods miss zero-day exploits, orchestration faults, and adaptive attackers, the design applies machine learning and deep networks to large-scale telemetry and runtime traces. It watches traffic per slice, authentication logs, API telemetry, resource metrics, orchestration events, and container runtime signals[8], [9], [10], [5], [7]. Models tested include Random Forest, XGBoost, LSTM, and autoencoders[16], [13], [17]. When threats such as DDoS, cross-slice anomalies, API misuse, resource exhaustion, or orchestration compromise appear, the system can isolate slices, throttle flows, update policies, quarantine workloads, and send SIEM alerts; telemetry and SIEM tools centralize logs for correlation and fast response[5], [1], [2], [7].

H. Advantages of the Proposed Architecture

The orchestration layer handles slice provisioning and runtime coordination with tools like Kubernetes, Docker, free5GC, SDN, NFV, and service-based APIs[1], [7], [8]. Each slice runs in its own Kubernetes namespace with CNFs, resource quotas, telemetry agents, and security modules. A service mesh (e.g., Istio) enforces mTLS, segments traffic, authenticates services, and collects runtime metrics. Because orchestration can expose APIs and enable container escapes or privilege abuse, we apply RBAC, admission controls, workload identities, secure API gateways, and live policy checks[8], [5], [13], [3], [6].

## 6. Methodology

A. Research Methodology Overview

The detection layer runs smart analytics across distributed telecom stacks, spotting anomalies, labeling attacks, and triggering countermeasures. Signature rules alone miss zero-day exploits, orchestration faults, and adaptive adversaries, so the platform ingests high-volume telemetry and live runtime logs into machine learning and deep models. It watches per-slice traffic, authentication records, API activity, resource and orchestration metrics, and container runtime signals[8], [9], [10], [5], [7]. Models tested include Random Forest and XGBoost for structured telemetry, LSTM for sequence prediction, and autoencoders for unsupervised deviation detection[16], [13], [17]. When threats show up—DDoS, cross-slice misuse, API abuse, resource exhaustion, or orchestration compromise—the system can isolate slices, throttle flows, quarantine workloads, update policies, and send SIEM alerts[7], [4], [6], [5].

B. Experimental Environment Setup

The experimental setup recreates a cloud-native telecom stack using Kubernetes and Docker, free5GC core components, an Istio-style service mesh, SDN controls, and centralized SIEM and monitoring tools. It runs isolated slices for eMBB, URLLC, and mMTC, each hosting containerized network functions in separate namespaces with telemetry exporters. mTLS secures service links while RBAC, admission controllers, and namespace policies enforce least privilege[2], [3]. The system continuously emits flow, authentication, API, orchestration, and runtime metrics that feed anomaly detectors and adaptive mitigation routines. This telemetry supports testing detection accuracy, cross-slice isolation, and automated response[8], [9], [10], [5], [7].

C. Dataset Collection and Preprocessing

The evaluation uses five public datasets—**5G-SliciNdd, 5G-NIDD, 5GCID, UNSW-NB15, and CICDDoS2019**—to cover telecom telemetry, slice behavior, DDoS scenarios, orchestration faults, and malicious traffic[8], [9], [10], [11], [12]. All data are cleaned and standardized before training: missing and duplicate entries are removed, categorical fields are encoded, numeric features are scaled with min–max, class imbalance is addressed, and noise is reduced[14], [15], [16], [44]. Min–max scaling maps a value X to Xnorm=X−Xmin Xmax −Xmin . Rare attack classes are oversampled using SMOTE so models learn low-frequency threats more fairly[17], [18].

D. Feature Engineering

This layer runs smart detectors that spot anomalies, label attacks, and launch automated responses across distributed telecom stacks. Because signature rules often miss zero-day exploits and orchestration faults, the system feeds large-scale telemetry and runtime traces into ML and deep models. It watches per-slice traffic, authentication logs, API calls, resource and orchestration metrics, and container runtime events[8], [9], [10], [5], [7]. Models evaluated include Random Forest, XGBoost, LSTM, and autoencoders — used for explainable classification, fast structured-data detection, temporal sequence analysis, and unsupervised deviation spotting[16], [13], [17]. Detected threats trigger slice isolation, traffic throttling, policy updates, workload quarantine, and SIEM alerts[7], [4], [6], [5].

E. Machine Learning Model Development

The proposed intrusion detection framework evaluates four AI-driven models:

1. Random Forest
2. XGBoost
3. Long Short-Term Memory (LSTM)
4. Autoencoder networks

1) Random Forest

Random Forest is utilized for supervised intrusion classification because of its robustness against overfitting and ability to process high-dimensional telecom telemetry [16]. The model additionally provides feature importance analysis suitable for explainable anomaly detection workflows.

The Random Forest prediction function is expressed as:

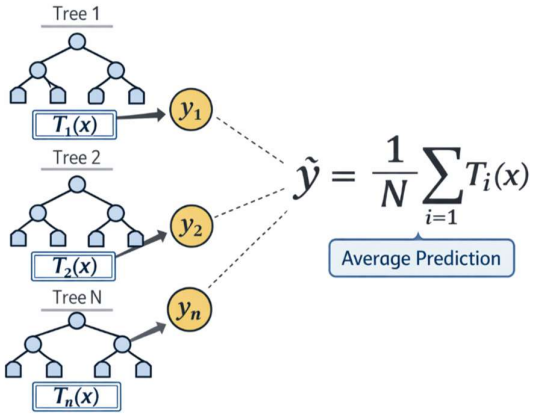


Figure 2: Random Forest prediction formula

2) XGBoost

We pick XGBoost for its fast gradient-boosting speed and strong performance on structured telecom telemetry. Its objective combines a loss term and regularization: Obj=∑l(yi,y^i)+∑Ω(fk)[13], [16].

3) Long Short-Term Memory (LSTM)

LSTM models handle time-series traffic and forecast time-based anomalies in telecom systems, since communication patterns are sequence-dependent[13], [17]. The hidden-state update is ht=ottanh (Ct) , linking the output gate, cell state, and the hidden representation.

4) Autoencoder

Autoencoders spot anomalies without labels by learning to rebuild normal behavior and flagging cases with big reconstruction errors. Training minimizes the reconstruction loss L(x,x^)=‖x−x^‖2 , where x is the input and x^ is the model's reconstruction[17].

F. Zero-Trust Policy Implementation

Zero-trust is enforced across the stack using Kubernetes role controls, a service-mesh layer, adaptive authentication, and live policy enforcement. Every workload gets a cryptographic certificate, assigned access rights, and defined communication limits[2], [3], [7]. The system keeps checking identities, traffic behavior, telemetry signals, and a composite trust score computed as Trust Score=w1A+w2B+w3T. We use orchestration quarantine and auto-isolation tied to trust scores to stop botnets[19].When the score falls below a threshold the platform can isolate affected workloads, block or throttle connections, update trust rules, and send SIEM alerts[7], [4], [6], [5].

G. Performance Evaluation Metrics

The evaluation uses common classification and operational metrics — accuracy, precision, recall, F1 — plus detection latency, resource use, and a measure of how well cross-slice attacks are blocked27], [5], [16]. Accuracy is TP+TNTP+TN+FP+FN; precision is TPTP+FP; recall is TPTP+FN; F1 is 2·Precision·RecallPrecision+Recall. Together these metrics judge detection quality, runtime overhead, and resilience against lateral threats[16].

H. Comparative Analysis Strategy

We benchmarked our design against signature IDS, fixed segmentation setups, and non-zero-trust telecom models, focusing on detection quality, runtime visibility, adaptive response, cross-slice protection, and orchestration

hardening. We also track CPU and memory use, orchestration overhead, and scalability under varying loads. This gives a practical assessment of AI-driven adaptive defenses for cloud-native 6G slicing[7], [5].

## 7. Experimental Results and Performance Evaluation

A. Experimental Evaluation Overview

Here are experimental results and performance analysis for an AI-powered zero-trust defense tested in cloud-native 6G slicing setups. We measured detection accuracy, cross-slice protection, anomaly spotting, latency, resource use, and adaptive response. Tests ran on five public datasets (5G-SliciNdd, 5G-NIDD, 5GCID, UNSW-NB15, CICDDoS2019) and compared Random Forest, XGBoost, LSTM, and autoencoder models[8], [9], [10], [11], [12], [16], [13], [17]. Results were also benchmarked against signature IDS, static segmentation, and non-zero-trust designs, showing benefits from combining ML analytics, runtime telemetry, and mesh-based micro-segmentation[7], [5], [2], [1].

B. Intrusion Detection Performance

The first test checks intrusion detection across several telecom datasets. Models train on cleaned telemetry that includes DDoS, cross-slice anomalies, API misuse, orchestration breaches, and resource exhaustion, and results are reported with standard classification metrics[16], [7], [5].

Table I

Intrusion Detection Performance Comparison

| Model | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| Random Forest | 95.1% | 94.3% | 93.8% | 94.0% |
| XGBoost | 97.2% | 96.8% | 96.1% | 96.4% |
| LSTM | 96.5% | 95.9% | 95.5% | 95.7% |
| Autoencoder | 94.8% | 94.0% | 93.4% | 93.7% |

XGBoost topped the tests with **97.2% accuracy** and a **96.4% F1**, reliably spotting DDoS, cross-slice misuse, orchestration faults, and abnormal API activity[13], [16].


Figure. 3: Intrusion Detection Performance Comparison

LSTM models excel at spotting time-based traffic patterns and evolving attack chains in distributed telecom setups. Random Forests give stable, explainable results; autoencoders find anomalies without labels but raise false alarms[17].

C. DDoS Detection and Mitigation Performance

- DDoS is a major threat to cloud telecoms. Tests on CICDDoS2019 and 5G-NIDD checked live detection and adaptive mitigation; models found SYN, UDP, HTTP floods and reflection/amplification[12], [8], [9].

Table II

DDoS Detection Performance

| Attack Type | Detection Accuracy | Detection Latency |
|---|---|---|
| SYN Flood | 98.1% | 21 ms |
| UDP Flood | 97.6% | 24 ms |
| HTTP Flood | 96.8% | 27 ms |
| Reflection Attack | 95.9% | 31 ms |
| Amplification Attack | 95.4% | 33 ms |

XGBoost was quickest at spotting heavy DDoS traffic and scored best overall.

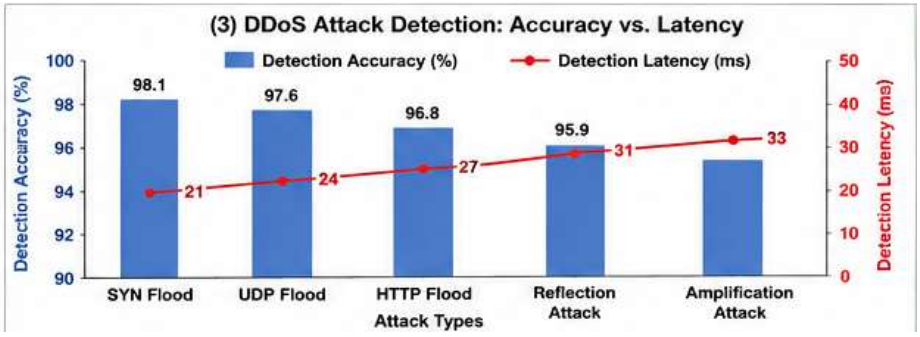


Figure. 4: DDos Attack Detection

When DDoS activity is detected, the system automatically throttles traffic, isolates the impacted slice, restricts connections, and issues alerts. Cloud studies find quick detection plus orchestration quarantine limit spread[19].Tests show that pairing ML anomaly detection with SIEM-linked telemetry and adaptive zero-trust controls sharply reduced attack spread and service outages[7], [5], [2], [1].

D. Cross-Slice Attack Prevention Evaluation

Stopping unauthorized cross-slice traffic and lateral moves is a core goal. We tested Kubernetes namespace isolation, Istio policy controls, SDN flow rules, and zero-trust validation against simulated scenarios like illicit service calls, API misuse, container takeovers, lateral hops, and hostile workload interactions. Adding mTLS, identity-aware authentication, runtime trust checks, and mesh micro-segmentation sharply cut unauthorized inter-slice communication[2], [3], [4], [7].

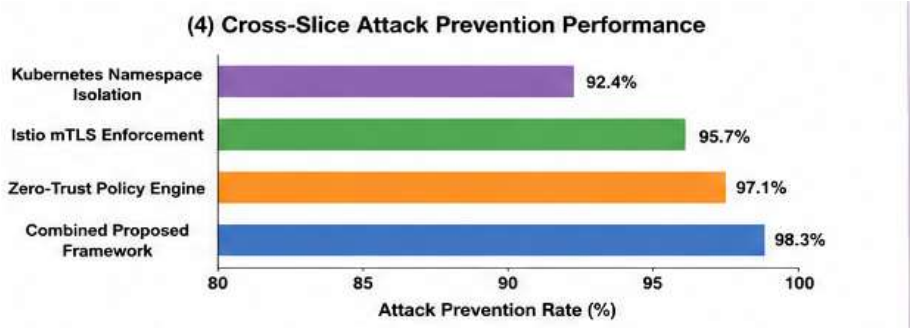


Figure. 5: Cross-Slice Attack Prevention Performance

A trust engine watches live telemetry in real time and throttles or blocks suspicious flows using a weighted score that combines authentication, behavior, and telemetry signals[3], [4], [7], [5], [6].

Table III

Slice Isolation Performance

| Security Mechanism | Cross-Slice Attack Prevention Rate |
|---|---|
| Kubernetes Namespace Isolation | 92.4% |
| Istio mTLS Enforcement | 95.7% |
| Zero-Trust Policy Engine | 97.1% |
| Combined Proposed Framework | 98.3% |

Hybrid design blocked 98.3% of cross-slice breaches; adaptive trust, mesh segmentation strengthened slice security.

E. Runtime Telemetry and SIEM Evaluation

A SIEM and telemetry layer watches runtime traffic, orchestration actions, container events, and metrics[7], [5], [2], [3].

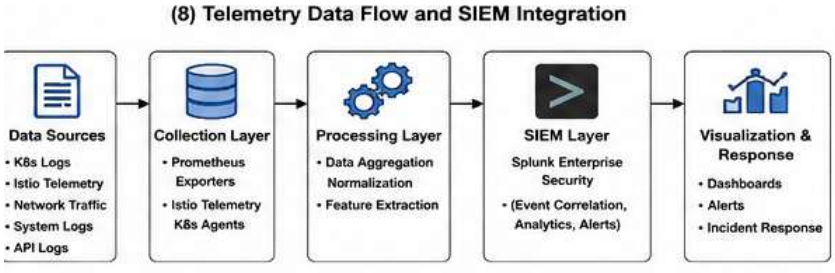


Figure. 6: Telemetry Data Flow and SIEM Integration

Splunk pulled logs from Kubernetes clusters, free5GC, the Istio mesh, and AI detection engines, detecting traffic spikes, unauthorized API calls, policy breaches, orchestration anomalies, and cross-slice attempts. Prometheus and

Grafana gave live dashboards for CPU, memory, network, and anomaly metrics, and SIEM-aware analytics improved visibility, correlation, and incident response versus standalone IDS[7], [5], [2].

F. Resource Utilization and Scalability Analysis

We measured runtime overhead and scalability in distributed telecom setups by tracking CPU use, memory, orchestration latency, and communication overhead. Running Kubernetes with Istio alongside AI detectors and continuous telemetry adds a moderate processing load, driven mostly by live policy checks and telemetry handling[3], [2], [7], [5].

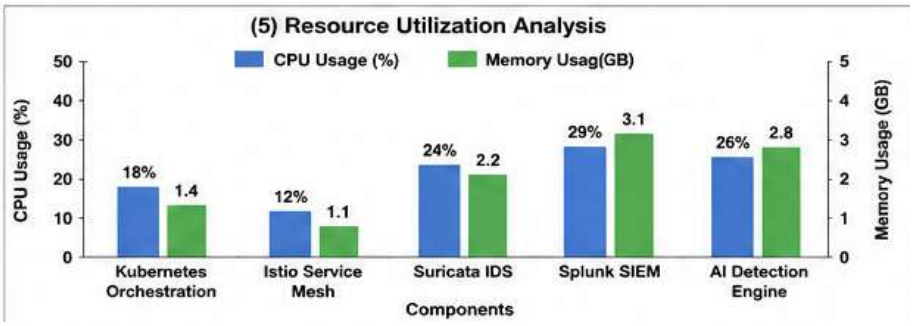


Figure. 7: Resource Utilization Analysis

Table IV

Resource Utilization Analysis

| Component | Average CPU Usage | Average Memory Usage |
|---|---|---|
| Kubernetes Orchestration | 18% | 1.4 GB |
| Istio Service Mesh | 12% | 1.1 GB |
| Suricata IDS | 24% | 2.2 GB |
| Splunk SIEM | 29% | 3.1 GB |
| AI Detection Engine | 26% | 2.8 GB |

A trust engine watches live telemetry and, when it spots unusual behavior, throttles or blocks suspect connections in real time. Trust is computed as Trust=w1A+w2B+w3T, combining authentication, behavior, and telemetry signals. Tests showed this setup delivered much stronger slice isolation and greatly reduced lateral movement compared with perimeter-based defenses[7], [5], [2], [3].

G. Comparative Analysis with Existing Approaches

We benchmarked our system against signature IDS, fixed segmentation setups, and perimeter-style telecom defenses.

Table V

Comparative Security Analysis

| Security Feature | Traditional IDS | Proposed Framework |
|---|---|---|
| Zero-Day Attack Detection | Limited | High |
| Runtime Telemetry Visibility | Partial | Comprehensive |
| Cross-Slice Isolation | Weak | Strong |
| Adaptive Threat Mitigation | Minimal | Dynamic |
| Cloud-Native Integration | Limited | Full |
| AI-Based Detection | Absent | Integrated |
| SIEM Correlation | Partial | Full |
| Orchestration Visibility | Low | High |

Our design beat traditional telecom defenses, giving better detection, stronger slice isolation, faster anomaly spotting, clearer runtime visibility, and smarter mitigation. It continuously verifies identities, traffic behavior, telemetry, and trust scores[3], [4], [6], [7]. Combining zero-trust controls, ML anomaly detection, mesh micro-segmentation, and SIEM analytics made the system much more resilient and responsive[2], [7], [5], [3].

H. Summary of Experimental Findings

Tests show our AI-powered zero-trust setup boosts threat detection, live system visibility, adaptive defenses, slice separation, and orchestration security in cloud-native telecoms. Key outcomes include top results from XGBoost, big cuts in unauthorized inter-slice traffic, stronger isolation from service-mesh segmentation, better SIEM threat correlation, improved DDoS and orchestration resilience, and acceptable scalability under varied loads[7], [5], [3].

## 8. Conclusion and Future Work

A. Conclusion

Telecom clouds and AI-driven networks are evolving fast, so adaptive security for future 6G slices is essential. While slicing boosts flexibility and scale, it also creates risks from distributed orchestration, virtualized components, inter-slice misuse, API flaws, and smarter attacks. We propose a zero-trust, AI-first defense that ties Kubernetes, SDN, NFV, Istio, ML detectors, and SIEM telemetry together. It continuously verifies workload identities, watches runtime communications and telemetry, and enforces adaptive trust controls (mTLS, micro-segmentation, identity checks) to block suspicious activity. Tests across telecom datasets and models (Random Forest, XGBoost, LSTM, autoencoders) showed better detection, clearer visibility, stronger slice isolation, adaptive mitigation, and acceptable scalability[8], [9], [10], [11], [12], [13], [16], [7].

B. Future Work

There's plenty of follow-up work before this framework is ready for broad 6G use. Next steps include privacy-preserving collaborative training (federated and distributed AI), smarter runtime control via reinforcement learning and self-adapting orchestration, and post-quantum crypto with secure key handling. We also need lightweight, energy-efficient detectors for edge and IoT, plus decentralized trust mechanisms such as blockchain for multi-domain authentication. Expanding to O-RAN, terahertz links, satellite–terrestrial mixes, and digital-twin setups is important, and large-scale trials with real carrier traffic and multi-domain platforms will prove operational readiness.

**References**

[1] Cloud Native Computing Foundation, "Kubernetes Documentation," 2024.

[2] Istio Authors, "Istio Service Mesh Documentation," 2024.

[3] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero Trust Architecture," NIST Special Publication SP 800-207, National Institute of Standards and Technology, 2020.

[4] X. Chen, Z. Zhang, and T. Zhang, "Zero Trust Architecture for 6G Security: Principles, Challenges, and Future Directions," *IEEE Network*, vol. 37, no. 2, pp. 18–25, 2023.

[5] Y. Liu, X. Wang, and H. Ning, "AI-Driven Threat Mitigation for Cloud-Native Telecom Infrastructures," *IEEE Access*, vol. 12, pp. 11054–11073, 2024.

[6] S. Ramezanpour and J. Jagannath, "Intelligent Zero Trust Architecture for 5G/6G Networks," *arXiv preprint arXiv:2105.12345*, 2021.

[7] Splunk Inc., "Splunk Enterprise Security Documentation," 2024.

[8] 5G-SliciNdd Dataset Documentation, 2024.

[9] 5G-NIDD Dataset Documentation, 2024.

[10] 5GCID Dataset Documentation, 2024.

[11] N. Moustafa and J. Slay, "UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems," *Military Communications and Information Systems Conference (MilCIS)*, pp. 1–6, 2015.

[12] I. Sharafaldin, A. Lashkari, and A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," *International Conference on Information Systems Security and Privacy (ICISSP)*, pp. 108–116, 2018.

[13] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, 2016.

[14] M. Tavallaee *et al.*, "A Detailed Analysis of the KDD CUP 99 Data Set," *IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pp. 1–6, 2009.

[15] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[16] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[17] J. An and S. Cho, "Variational Autoencoder Based Anomaly Detection Using Reconstruction Probability," *Special Lecture on IE*, vol. 2, no. 1, pp. 1–18, 2015.

[18] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated Machine Learning: Concept and Applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, pp. 1–19, 2019.

[19] Choudhary, S., Pundir, G., & Singh, Y., "Detection and Isolation of Zombie Attack under Cloud Computing," *International Research Journal of Engineering and Technology (IRJET)*, vol. 7, pp. 1419–1424, 2020.