Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.077454

ARTICLE

# A Novel Synthetic Dataset for Effective Detection of Replay Attacks in SDN-Enabled IoT Networks

**Nader Karmous[1], Leila Bousbia[1], Mohamed Ould-Elhassen Aoueileyine[1], Imen Filali[2,*] and Ridha Bouallegue[1]**

[1]Innov'COM Laboratory, Higher School of Communication of Tunis, University of Carthage, Technopark Elghazala, Raoued, Ariana, Tunisia

[2]Department of Computer Sciences, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia

*Corresponding Author: Imen Filali. Email: imfilali@pnu.edu.sa

Received: 09 December 2025; Accepted: 06 February 2026; Published: 08 May 2026

**ABSTRACT:** This study proposes an intelligent Intrusion Detection and Prevention System (IDPS) integrated into a centralized Ryu Software-Defined Networking (SDN) controller to mitigate replay attacks within Internet of Things (IoT) environments. To address the scarcity of specialized datasets, a comprehensive dataset was generated using a real-time SDN-IoT testbed encompassing Mininet, multiple OpenFlow 1.3 switches, and a single Ryu controller. The experimental setup featured the exchange of legitimate and malicious Message Queuing Telemetry Transport (MQTT) traffic between hosts and IoT devices to simulate realistic network behaviors and attack vectors. Our methodology introduces a novel feature engineering framework by evaluating three distinct configurations, including: (1) preprocessed features, (2) data reduced through Principal Component Analysis (PCA), and (3) latent representations extracted via a Variational Autoencoder (VAE). Four distinct classifiers were rigorously benchmarked, including Random Forest (RF), Support Vector Machine (SVM), Extreme Gradient Boosting (XGBoost), and a Convolutional Neural Network (CNN). Performance metrics were derived from 50 independent runs and validated through paired *t*-tests and Wilcoxon signed-rank tests. The results demonstrate that VAE-based deep feature extraction significantly improves detection accuracy. Notably, the CNN trained on these features achieved a peak accuracy of 99.91% and a false alarm rate of 0.19%. The framework's real-time effectiveness and scalability were validated through live deployment, offering a robust and reproducible solution for securing SDN-enabled IoT infrastructures. Ultimately, our proposed CNN-VAE approach demonstrates superior performance and higher detection precision compared to existing related works in the field of IoT intrusion detection.

**KEYWORDS:** Replay attacks; man in the middle; cybersecurity; software defined networking; Internet of Things; machine learning; artificial intelligence; deep learning; variational autoencoder; feature selection; principal component analysis; Wilcoxon signed-rank

## 1 Introduction

IoT refers to a network of interconnected devices that communicate and interact over the Internet [1,2], enabling continuous data exchange and automated decision-making. SDN [3,4] is a network architecture that separates the control plane from the data plane [5] in order to provide centralized, programmable, and flexible network management. The integration of SDN into IoT ecosystems [6,7] has significantly enhanced

network coordination, offering dynamic, scalable, and unified control to accommodate diverse, resource-constrained devices across critical domains such as healthcare, smart homes, industrial automation, and intelligent transportation systems. Lightweight communication protocols [8] are widely adopted in these environments to ensure efficient and low-latency data transmission [9] for constrained IoT devices.

Despite these advantages, the convergence of SDN and IoT introduces substantial security challenges [10], affecting the continuity of operations, the confidentiality and integrity of transmitted data, the availability of services, and the overall reliability of autonomous decision-making processes. IoT networks remain vulnerable to a broad range of cyberattacks, including botnets [11], Distributed Denial of Service (DDoS) attacks [12], ransomware [13], and Man-in-the-Middle (MitM) attacks [14,15]. Among these threats, this work focuses specifically on replay attacks [16], which represent a dangerous subclass of MitM attacks and pose a direct threat to the confidentiality, integrity, and trustworthiness of IoT communications. In a replay attack, adversaries intercept, record, modify, or retransmit legitimate packets to disrupt communication, alter system behavior, or gain unauthorized access. In high-stakes environments such as healthcare, where IoT devices including wearable monitors, infusion pumps, and telemedicine systems continuously capture sensitive medical data, replay attacks can lead to severe consequences such as manipulated patient records, incorrect clinical decisions, and life-threatening operational failures.

Detecting replay attacks in SDN-enabled IoT systems remains challenging due to device heterogeneity, dynamic topologies, and the widespread use of lightweight protocols [17] such as MQTT, CoAP, AMQP, XMPP, and HTTP. These challenges are compounded by the scarcity of high-quality datasets tailored specifically to replay attack scenarios within SDN-IoT environments. Existing datasets often suffer from privacy restrictions, limited scalability, incomplete attack coverage, or unrealistic traffic patterns, which restrict their applicability for training robust IDPS [18]. Additionally, most publicly available datasets do not capture the complexity of replay attacks, especially in operational sectors such as healthcare where reliability, precision, and real-time response are essential.

To address these gaps, this paper introduces a novel synthetic dataset specifically designed to advance replay attack detection in SDN-enabled IoT networks, with particular emphasis on healthcare applications [19]. The dataset simulates realistic traffic flows over TCP, UDP, and lightweight IoT protocols such as MQTT, CoAP, AMQP, XMPP, and HTTP, covering a comprehensive range of benign and malicious scenarios. It incorporates detailed packet-level features, temporal behavior patterns, and protocol metadata to support advanced machine learning analysis. To identify the most effective IDPS model, this study evaluates several widely used machine learning techniques, including SVM [20], RF [21], CNN [22], and XGB [23]. By systematically comparing these models on the proposed dataset, we determine the optimal approach for achieving real-time and accurate replay attack detection.

The major contributions and technical advancements of this research are summarized as follows:

- **High-Fidelity Synthetic Dataset Generation:** We developed a specialized dataset specifically designed for detecting replay attacks within SDN-enabled IoT environments. This dataset was generated using a realistic testbed encompassing Mininet and the Ryu controller, capturing complex bidirectional MQTT traffic between hosts and IoT devices.
- **Addressing Protocol-Layer Vulnerabilities:** Existing replay attack datasets, namely Idiap Replay-Attack, CASIA-SURF, and ASVspoof, focus primarily on physical sensor-level spoofing. In contrast, our synthetic dataset addresses the critical security gap in network-layer IoT communications by targeting vulnerabilities in specialized protocols, including CoAP and MQTT, to protect against protocol-level hijacking and malicious replay exploits.
- **Novel Feature Engineering Framework:** This study proposes a non-linear feature extraction approach utilizing a VAE. This methodology was rigorously benchmarked against raw preprocessed features

and PCA, demonstrating that latent representations significantly enhance the statistical separability of malicious and legitimate network flows.
- **Extensive Comparative Benchmarking:** We provide a comprehensive evaluation of four distinct architectures, specifically RF, SVM, XGBoost, and a CNN. To ensure empirical reliability, all models were validated through 50 independent runs and rigorous statistical analysis using paired $t$-tests and Wilcoxon signed-rank tests.
- **Active Control Plane Integration:** As opposed to passive detection models, our proposed IDPS is integrated directly into the control plane of the Ryu SDN controller. This architecture enables real-time threat mitigation by dynamically updating OpenFlow tables to drop malicious packets immediately upon detection.

The remainder of the paper is organized as follows: Section 2 provides background on SDN-IoT networks and replay attacks. Section 3 reviews related work on MitM and replay attack detection. Section 4 describes the proposed methodology, including dataset generation and model evaluation. Section 5 presents experimental results, and Section 6 discusses the deployment of the model within a real SDN-based IoT testbed. Section 7 provides comparative studies, and Section 8 concludes the paper with future research directions.

## 2 Background

This section provides an overview of replay attacks in IoT environments, focusing on their underlying mechanisms, operational impacts, and detection challenges within SDN-based IoT architectures. It explains how lightweight communication protocols and resource-constrained devices create vulnerabilities that adversaries can exploit to compromise the integrity and availability of network services. Practical examples are included to contextualize the threat and to support the motivation for generating the synthetic dataset. Furthermore, this section describes the feature selection process adopted for SDN–IoT attack detection, along with the evaluation metrics, stability analysis, and statistical tests employed to assess the performance of machine learning models on the proposed dataset.

### *2.1 Comparative Analysis of Replay Attack Benchmarks*

To contextualize the necessity of the proposed work, it is essential to distinguish our Synthetic Replay Dataset from existing benchmarks. The following analysis highlights the technical uniqueness of the proposed synthetic replay dataset compared to existing biometric replay benchmark datasets, such as Idiap, ASVspoof, and CASIA-SURF.

#### *2.1.1 Domain Specificity and Data Nature*

Unlike Idiap Replay, CASIA-SURF, and ASVspoof, which provide physical-layer biometric samples (video, image, and audio), our work contributes a Synthetic Replay Dataset specifically engineered for the network layer. While biometric datasets focus on spatial and acoustic textures, our dataset captures the logical and temporal flow of MQTT and CoAP protocols within an SDN environment. This allows researchers to simulate network-layer hijacking and state-injection attacks that are fundamentally different from identity spoofing.

#### *2.1.2 Feature Reduction and Extraction Strategy*

In our proposed work, feature reduction is achieved through a VAE, which compresses the complex network traffic into 16 latent features. This differs from the high-dimensional pixel-based features found in

Idiap or the manual spectral features in ASVspoof. By providing a dataset already optimized for VAE-CNN extraction, we enable the development of active prevention models that can operate within the sub-second latency requirements of IoT infrastructures.

#### *2.1.3 Architectural Integration and Mitigation*

A key innovation of our synthetic dataset is its integration with the SDN control plane. While existing benchmarks are used for passive detection at the application layer, our dataset includes OpenFlow metadata. This enables the training of models for active prevention, where the mitigation point is the SDN Controller itself, allowing real-time traffic filtering at the network edge a capability that cannot be evaluated using traditional biometric replay datasets.

Table 1 presents a summary of the proposed synthetic replay dataset compared to existing biometric replay attack benchmark datasets, highlighting differences in data domain, number of features, feature extraction and reduction methods, defense type, and mitigation points. This comparison emphasizes the novelty and applicability of the proposed dataset in IoT-based SDN environments.

**Table 1:** Comparison of the proposed synthetic replay dataset with existing biometric replay attack benchmark datasets.

| Dataset/ Framework | Data Domain | Features (N) | Feature Extraction Method | Feature Reduction Method | Defense Type | Mitigation Point |
|---|---|---|---|---|---|---|
| Idiap Replay | Biometric (Face) | 1000+ | LBP/Texture | PCA/Manual | Passive | Application |
| ASVspoof | Biometric (Voice) | 60–80 | Spectral (CQCC) | Manual Truncation | Passive | Application |
| CASIA-SURF | Biometric (Multi) | 512+ | Manual Deep Features | Max-Pooling | Passive | User Interface |
| Proposed Dataset | Network (IoT) | 27 | SDN Controller | 16 latent features | Active | SDN Controller |

### *2.2 Motivation and Problem Significance*

The rapid proliferation of IoT devices within SDN infrastructures has introduced unprecedented security challenges, most notably the susceptibility to replay attacks. The motivation for this research stems from several critical factors.

#### *2.2.1 Vulnerability of Specialized Protocols*

Traditional security mechanisms often overlook the lightweight nature of IoT protocols such as MQTT and CoAP. Replay attacks at the protocol level can lead to unauthorized state changes in critical systems (smart grids or healthcare monitoring) without requiring the attacker to decrypt payloads.

#### *2.2.2 Limitations of Existing Solutions*

Current detection frameworks primarily rely on either:

- *Timestamping and Sequence Numbering:* These require strict clock synchronization across heterogeneous IoT nodes, which is computationally expensive and difficult to maintain at scale.
- *Passive Monitoring:* Many existing IDPS solutions function as offline analyzers, identifying threats only after the network has been compromised.

Our work addresses these gaps by proposing an active IDPS integrated directly into the SDN controller, enabling sub-second mitigation without requiring specialized hardware or protocol modifications.

#### *2.2.3 The Need for High-Fidelity Data*

A significant bottleneck in IoT security research is the reliance on outdated or generic datasets that do not reflect the unique traffic patterns of SDN flow tables. By generating a novel synthetic dataset based on a real-world testbed, this study provides a robust foundation for training high-precision deep learning models.

The practical benefits of this work extend to the stability and scalability of smart infrastructures. By utilizing VAE-based latent features, our approach ensures high detection accuracy even in noisy network environments, reducing the false alarm rate (FAR) and preventing the degradation of network Quality of Service (QoS).

### *2.3 Impact of Replay Attacks on IoT Devices*

Replay attacks represent a significant threat to the integrity, availability, and authenticity of communication in IoT environments. These attacks involve capturing legitimate data packets during transmission and resending them at a later time to deceive the receiving device or system. This section elaborates on the consequences of replay attacks in IoT ecosystems using practical examples involving two devices: a smart lighting controller named **SmartLight-Alpha** and a temperature and humidity sensor named **TempHum-Sense01**.

#### *2.3.1 Triggering Unauthorized Actions*

Replay attacks can force an IoT device to repeat a previously valid action without user consent. For example, an attacker eavesdropping on the communication [24] between the SDN controller and **SmartLight-Alpha** may capture a legitimate "turn on" command. By replaying this command later, the attacker can cause the smart light to activate unexpectedly, even in the absence of user interaction or authorization. In sensitive environments such as smart buildings or hospitals, this may lead to confusion, energy wastage, or safety concerns.

#### *2.3.2 Bypassing Authentication Mechanisms*

Many IoT protocols such as MQTT and CoAP do not incorporate strong anti-replay protections. If **SmartLight-Alpha** or **TempHum-Sense01** rely on static or token-based authentication without freshness verification, replayed packets can be accepted as valid. For instance, if a temperature control server authorizes a device once and then accepts continuous updates without verifying timeliness, replayed messages originating from **TempHum-Sense01** may be incorrectly processed as recent sensor readings.

#### *2.3.3 Resource Exhaustion in Constrained Devices*

IoT devices typically have limited processing power, memory, and energy. An attacker repeatedly replaying legitimate packets from **TempHum-Sense01**, such as periodic sensor readings, can overwhelm the device with redundant processing. This results in unnecessary energy consumption and may cause buffer overflow or CPU overload, ultimately leading to device crashes or denial of service conditions.

#### *2.3.4 Disruption of Real-Time Communication*

Applications requiring real-time environmental insight depend on accurate and timely data. A replayed packet from **TempHum-Sense01** reporting a temperature of 22°C and humidity of 55% can mislead a Heating, Ventilation, and Air Conditioning (HVAC) system [25] into assuming the environment is stable. If the actual temperature is significantly higher, such as 30°C, the system may fail to activate cooling mechanisms, potentially causing equipment damage or discomfort in human-occupied areas.

#### *2.3.5 Manipulating Sensor Data for False Decision-Making*

By injecting outdated but valid sensor data captured from **TempHum-Sense01**, an attacker can influence automated decision systems. In precision agriculture, replaying a message indicating optimal humidity may prevent irrigation when water is actually required, resulting in crop stress or failure. Similarly, in industrial cooling systems, stale temperature readings may prevent necessary cooling actions, increasing the risk of overheating.

#### *2.3.6 Evasion of Monitoring Systems*

Replay attacks are often difficult to detect because the packets being resent were previously legitimate. Intrusion Detection Systems (IDS) and firewalls monitoring **SmartLight-Alpha** may log repeated ON or OFF commands as valid operations without recognizing that they were maliciously replayed. This stealthy characteristic complicates real-time identification and mitigation.

#### *2.3.7 Disruption of Automation and Workflow*

Replay attacks can cause command sequencing errors and disrupt automated workflows. If **SmartLight-Alpha** is integrated into a home automation routine that depends on motion detection and scheduled logic, replaying a past "turn on" command during daytime hours may break the automation rules. In industrial environments, replaying a "temperature threshold exceeded" message from **TempHum-Sense01** can trigger unnecessary emergency shutdown procedures.

Table 2 summarizes the various impacts of replay attacks on IoT devices, specifically on *TempHum-Sense01* in the context of data publishing and *SmartLight-Alpha* in the context of command subscription.

**Table 2:** Summary of replay attack impacts on two IoT devices.

| Aspect | SmartLight-Alpha (IoT Light) | TempHum-Sense01 (Sensor) |
|---|---|---|
| Unauthorized Actions | Turns on/off without user input | Sends data that triggers wrong HVAC or alert response |
| Authentication Bypass | Reuse of old "ON" command with accepted token | Old sensor data accepted as new |
| Resource Exhaustion | Repeated ON/OFF signals drain processing | Constant replay floods memory/CPU |
| Communication Disruption | Misleading lighting automation logic | Stale data disrupts climate control |
| Sensor Data Manipulation | Not applicable | Old temperature reading misleads control logic |

(Continued)

**Table 2 (continued)**

| Aspect | SmartLight-Alpha (IoT Light) | TempHum-Sense01 (Sensor) |
|---|---|---|
| Monitoring Evasion | Appears normal to SDN logs and controller logs | IDS fails to detect replayed telemetry |
| Workflow Interference | Breaks scheduled automation | Triggers wrong control loops in smart environments |

### *2.4 MitM Replay Attack Scenarios in IoT Networks*

This subsection focuses on replay attacks involving data modification in IoT environments, highlighting the most prevalent and harmful forms that undermine data integrity [26].

#### *2.4.1 Replay Attacks During Data Publishing in IoT-Based SDN*

Fig. 1 illustrates a MitM replay attack scenario involving the IoT device **TempHum-Sense01**, the SDN controller, a legitimate host, and an OpenFlow switch. TempHum-Sense01, which functions as a temperature and humidity sensor, communicates with the controller responsible for managing network traffic and coordinating interactions with the legitimate host. The OpenFlow switch positioned between the IoT device, the controller, and the legitimate host forwards packets based on controller instructions. A MitM replay attacker intercepts this communication by conducting packet sniffing and modifying data at the switch. The attacker captures legitimate packets and replays altered sensor readings to the legitimate host, thereby deceiving the system and potentially compromising decision-making processes.

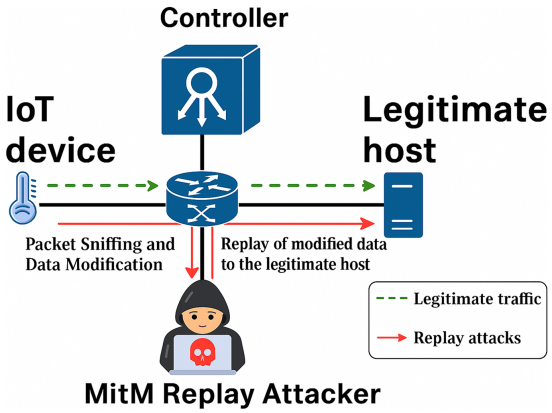


**Figure 1:** Replay attacks with data modification in IoT-based SDN: the case of publishing.

#### *2.4.2 Replay Attacks During Data Subscribing in IoT-Based SDN*

Fig. 2 presents a MitM replay attack scenario involving a legitimate host, the SDN controller, the IoT device **SmartLight-Alpha**, and a replay attacker. SmartLight-Alpha, represented as a smart bulb, communicates with the controller, which manages network operations and coordinates communication with the legitimate host. The replay attacker intercepts this communication by conducting packet sniffing and modifying data exchanged between SmartLight-Alpha, the controller, and the legitimate host. The attacker subsequently replays altered packets to the legitimate host, potentially changing the original commands or data and thereby deceiving the system.

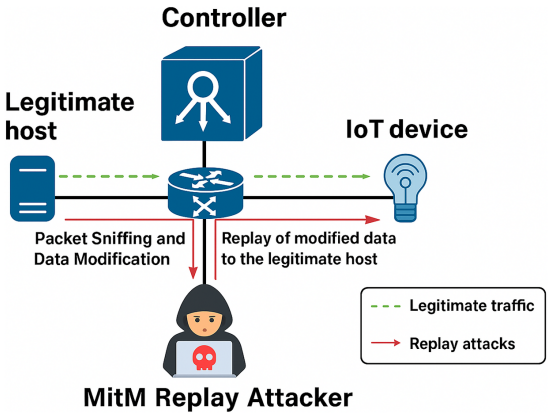


**Figure 2:** Replay attacks with data modification in IoT-based SDN: the case of subscribing.

### *2.5 Feature Selection in SDN–IoT Attack Detection*

In IoT and SDN environments, identifying the most relevant features from network traffic is essential for improving the accuracy and efficiency of replay and malicious attack detection. Two frequently utilized techniques in the literature are PCA and Variational Autoencoders, both of which contribute to enhancing model performance by refining the feature space.

#### *2.5.1 Principal Component Analysis (PCA)*

PCA [27] is a statistical method that reduces the dimensionality of the feature space by transforming correlated variables into a smaller set of uncorrelated components. These principal components capture the highest variance in the dataset, enabling the removal of redundant information while preserving the most important characteristics necessary for effective traffic classification in SDN–IoT systems.

#### *2.5.2 Variational Autoencoder (VAE)*

VAEs [28] are deep learning (DL) architectures that learn compact latent representations of input features. By modeling complex nonlinear patterns within network traffic, VAEs can uncover subtle relationships that may indicate abnormal or attack-related behavior. The learned latent space provides a refined set of features that can significantly improve downstream classification performance in SDN–IoT attack detection frameworks.

### *2.6 Evaluation Metrics for SDN–IoT Attack Detection*

The performance of machine learning models using PCA or VAE features is commonly evaluated using standard metrics that assess both detection accuracy and computational efficiency in SDN–IoT environments. Table 3 summarizes these evaluation metrics along with their definitions.

**Table 3:** Evaluation metrics used for assessing SDN–IoT attack detection models.

| Metric | Definition |
|---|---|
| Accuracy | The proportion of correctly classified samples among all samples. |
| Precision | The fraction of instances predicted as attacks that are truly attacks. |
| Recall | The proportion of actual attacks that are correctly identified by the model. |

(Continued)

**Table 3 (continued)**

| Metric | Definition |
|---|---|
| F1-score | The harmonic mean of precision and recall, providing a balance between the two. |
| AUC-ROC | Area under the receiver operating characteristic curve, representing the model's discrimination ability. |
| FAR | The fraction of normal traffic incorrectly classified as attacks. |
| TPR | The fraction of actual attacks that are correctly detected by the model. |
| Training and Testing Time | The computational time required to train the model and perform predictions. |

These metrics provide a comprehensive framework for evaluating both the accuracy of attack detection and the efficiency of model computation, which is critical for real-time deployment in SDN–IoT networks.

### *2.7 Stability and Statistical Significance*

To ensure the reliability of feature selection methods, repeated experiments are often conducted (across multiple runs with the same dataset). Boxplots of metrics such as Accuracy, AUC-ROC, or FAR can illustrate:

- The stability of model performance across repeated runs.
- The variance introduced by different feature selection methods.

Statistical tests are frequently applied to confirm whether observed differences are significant:

- **Paired *t*-test:** Compares the mean performance of two methods on the same dataset.
- **Wilcoxon signed-rank test:** A non-parametric alternative used when metric distributions are not normal.

These tests help researchers validate the superiority or robustness of one feature selection method over another in a rigorous, quantitative manner.

## 3 Literature Review

This section presents a comprehensive review of studies published between 2017 and 2025 on replay attack detection in IoT environments. The review is organized chronologically and emphasizes the methodologies, key contributions, and limitations of each approach.

Early research primarily focused on authentication and timestamp-based mechanisms. For example, Feng et al. [29] proposed a hash-based challenge–response authentication scheme resilient to replay and impersonation attacks, designed for resource-constrained IoT nodes with minimal computational overhead. In the healthcare domain, Rughoobur and Nagowah [30] developed a lightweight framework leveraging UUIDs and energy consumption monitoring, achieving near-perfect detection accuracy on embedded IoT devices. Similarly, Farha and Ning [31] enhanced ZigBee network security by integrating timestamp-based verification to prevent replayed packets, thereby preserving energy in multi-hop topologies.

More recent studies have explored machine learning (ML) and DL techniques for replay attack detection. Elsaeidy et al. [32] introduced a CNN-based framework using multivariate time-series features from smart city sensors, achieving over 95% detection accuracy. Their findings demonstrated the potential

of DL models to capture temporal dependencies and identify subtle replay patterns in IoT data streams. In contrast, Wara and Yu [33] experimentally demonstrated the feasibility of low-cost replay attacks on ZigBee devices, highlighting the need for robust detection mechanisms integrated within SDN or edge computing infrastructures.

In the context of secure wireless communications, Huan et al. [34] proposed Kerra, a lightweight LoRa-based key generation scheme resilient to replay attacks via synchronized time measurements, reducing key disagreement rates while maintaining efficiency on real-world testbeds. Similarly, He et al. [35] presented a CNN-based approach using spectral difference analysis to detect voice replay attacks in IoT-enabled authentication systems, achieving state-of-the-art Equal Error Rates on ASVspoof datasets.

To provide a comprehensive overview of existing approaches, Table 4 summarizes the major replay-attack detection and mitigation techniques proposed between 2017 and 2025. It highlights their key contributions as well as the main limitations that motivate the need for more robust IoT security solutions.

**Table 4:** Summary of related work on replay-attack detection and mitigation in IoT environments (2017–2025).

| Reference | Year | Advantages/Contributions | Limitations/Disadvantages |
|---|---|---|---|
| [29] | 2017 | Lightweight hash-based authentication resistant to replay and impersonation; low computation cost; suitable for constrained devices. | Higher communication overhead in dense networks; limited scalability. |
| [30] | 2017 | Combines UUID and battery-monitoring for reliable replay detection; achieves ≈100% accuracy in 20s. | Increased energy consumption during detection phase; less effective while charging. |
| [31] | 2019 | Timestamp synchronization prevents all replay attempts; reduces redundant decryption and saves energy. | Requires tight clock synchronization; susceptible to DoS if desynchronized. |
| [32] | 2020 | Exposes real-world vulnerabilities in ZigBee IoT devices using KillerBee toolkit; practical demonstration of replay feasibility. | Limited attack variety; physical access required; small-scale testing. |
| [33] | 2020 | CNN-based detection achieves >95% accuracy on IoT sensor data; captures temporal dependencies effectively. | Relies on synthetic replay data; potential overfitting to specific scenarios. |
| [34] | 2024 | LoRa-based secure key generation using synchronized time delay detection; high efficiency and real testbed validation. | Dependent on precise timing; less effective under dynamic interference. |
| [35] | 2024 | CNN (SE-ResNet50) model using spectral difference achieves EER = 1.36%; faster and smaller than previous models. | Evaluated mainly on public datasets; not validated under noisy IoT environments. |

Despite these advances, several limitations remain. Many studies rely on simulated or publicly available datasets that may not accurately reflect realistic IoT replay behaviors, potentially leading to overfitting and limited generalizability. Moreover, most frameworks focus on single-protocol or single-layer detection,

restricting their applicability in heterogeneous IoT networks. These challenges motivate the present study, which generates a comprehensive synthetic replay attack dataset and evaluates detection and mitigation performance using SVM, RF, XGBoost, and CNN models, enhanced by PCA- and VAE-based feature representations.

## 4 System Implementation

This section describes the methodology used to develop the machine learning-based IDPS for detecting and mitigating replay attacks in an SDN-enabled IoT environment. The process begins with dataset collection, followed by data preprocessing to prepare the dataset, and concludes with training and evaluation of machine learning-based IDPS algorithms.

### *4.1 Dataset Collection*

In all our experiments, we use Mininet, OpenFlow switches, and the Ryu controller. To generate the CSV dataset, we employed a tree topology created with Mininet, consisting of 16 hosts, including two IoT devices (one acting as a publisher and the other as a subscriber), a single controller, and 15 OpenFlow switches. As shown in Fig. 3, the dataset generation process begins when a switch receives a packet that does not match any existing flow entry, prompting it to send a Packet-In message to the controller. Upon receiving this incoming Packet-In event, the Ryu controller initiates the generation of both replay attack traffic and legitimate IoT traffic using predefined behaviors or attack scripts. These packets are then subjected to feature extraction. The Ryu controller collected replay attack traffic as well as legitimate traffic. Based on the observed packets, the traffic is labeled accordingly: 1 for attack and 0 for legitimate. This labeled data is then stored in a CSV dataset, forming a valuable resource for training and evaluating ML models aimed at detecting replay attacks and securing IoT devices within the SDN infrastructure.

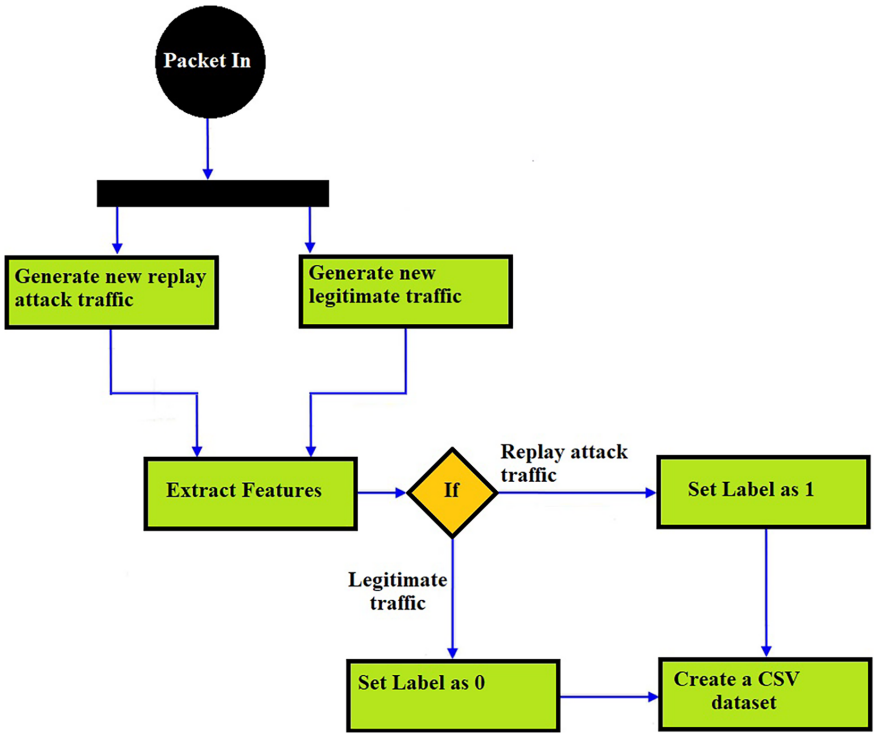


**Figure 3:** Dataset collection and generation process.

*4.1.1 Tools Commonly Used for Collecting Legitimate Traffic and Replay Attack Traffic*

The Table 5 summarizes the legitimate traffic types generated within the SDN-IoT testbed and the corresponding tools and command-line examples used to produce each traffic class. Each row identifies a network or application protocol (HTTP, TCP, UDP, ICMP, MQTT, CoAP, AMQP, XMPP), states its functional purpose in the experiment (web communication, reliable transport, constrained IoT messaging), and lists representative Linux commands or client libraries used to synthesize or subscribe to that traffic (for example, curl, iperf, ping, mosquitto_pub/sub, aiocoap-client, and Python pika). These traffic generators were selected to emulate realistic benign behaviour of constrained devices and services found in smart-home deployments, thereby providing baseline workloads against which attack scenarios and detection mechanisms were evaluated. Command examples are illustrative and were parameterized (IP addresses, ports, topics, payloads, durations) according to the experimental configuration documented in the Methods section. All traffic generation was performed on an isolated laboratory network to preserve the safety and legality of the experiments and to ensure reproducibility of the results.

**Table 5:** Legitimate traffic generation in SDN-IoT environment.

| Protocol | Purpose | Tools/Commands on Linux Ubuntu |
|---|---|---|
| HTTP | Web communication | `curl http://<target_ip>` or `wget http://<target_ip>` |
| TCP | Reliable connection (custom traffic) | `iperf -s` (server), `iperf -c <ip> -t 10 -p <port>` (client) |
| UDP | Connectionless real-time traffic | `iperf -s -u` (server), `iperf -c <ip> -u -t 10` (client) |
| ICMP | Ping/Echo requests | `ping <target_ip>` |
| MQTT | IoT messaging (Pub/Sub) | `mosquitto_pub -h <broker_ip> -t topic -m "message"` `mosquitto_sub -h <broker_ip> -t topic` |
| CoAP | Constrained application protocol | Using aiocoap: `aiocoap-client coap://<ip>/resource` (GET) `aiocoap-client -m put coap://<ip>/resource -f file.txt` (PUT) |
| AMQP | Messaging middleware (RabbitMQ) | Python (pika): `python3 producer.py` and `python3 consume.py` |
| XMPP | IoT messaging/device chat | `sendxmpp`, or Python3 `sleekxmpp` clients |

The Table 6 summarizes the principal tools employed in this study to perform packet capture, manipulation and replay attacks experiments within an SDN-IoT testbed. Each row maps a specific stage of the replay-attack workflow (packet capture, attacker positioning through ARP poisoning, PCAP editing and preprocessing, packet replay, application-level message replay, and inspection/analysis) to the command-line utilities used to implement that stage (for example, tcpdump, tshark, ettercap, arpspoof, bettercap, tcprewrite, tcpreplay, mosquitto_pub, mosquitto_sub, mitmproxy, curl, wireshark). These tools were chosen for their prevalence in network security research and their capacity to reproduce realistic replay scenarios

against resource-constrained IoT devices and SDN controllers. All experiments were executed on an isolated laboratory network to prevent any impact on production environments or third parties, and were conducted in accordance with relevant institutional ethical guidelines and legal requirements. This mapping of attack stages to reproducible tooling enhances methodological transparency and facilitates independent replication of the experimental procedures.

**Table 6:** Tools commonly used for replay attacks in IoT–SDN environments.

| Stage | Objective | Tools/Commands on Linux (Ubuntu) |
|---|---|---|
| Packet capture | Capture network traffic to a pcap | `tcpdump -w capture.pcap`, `tshark -w capture.pcap` |
| Positioning/ MITM | Insert attacker on-path (ARP poisoning/MITM) | `ettercap`, `arpspoof` (dsniff), `bettercap` |
| PCAP editing/ preprocessing | Modify IP/MAC/ports or payloads prior to replay | `tcprewrite`, `scapy` |
| Packet replay | Reinject captured or modified traffic into the network | `tcpreplay` |
| Application-level replay | Replay or publish application messages (IoT payloads) | `mosquitto_pub`, `mosquitto_sub`, `mitmproxy`, `curl` |
| Inspection/ analysis | Inspect and validate captures or replays | `wireshark`, `tshark` |

*4.1.2 Features Extraction*

Table 7 summarizes the 27 features extracted from the IoT-SDN network traffic. Each feature captures a specific aspect of the network flow or packet behavior within the SDN environment. Key fields include identifiers such as `datapath_id` and `flow_id` to track flows through OpenFlow switches, and network addressing features like source and destination IP and MAC addresses (`ip_src`, `ip_dst`, `mac_src`, `mac_dst`). Transport layer information is represented by source and destination ports (`tp_src`, `tp_dst`) and TCP-specific fields (`tcp_flags`, `tcp_seq`, `tcp_ack`). Flow dynamics are described using duration metrics (`flow_duration_sec`, `flow_duration_nsec`) and timeout settings (`idle_timeout`, `hard_timeout`). Traffic volume and rate are captured through packet and byte counts and their corresponding rates per second and per nanosecond. Finally, the `label` feature denotes the class of the traffic, where 0 represents legitimate traffic and 1 corresponds to replay attacks. This comprehensive feature set provides the necessary information for training and evaluating machine learning models aimed at detecting and mitigating replay attacks in SDN-enabled IoT networks.

Several extracted features, particularly rate-based derivatives (packet and byte rates), may exhibit high correlation. These features were intentionally retained to preserve fine-grained temporal and volumetric traffic dynamics that are relevant for replay attack detection in SDN–IoT environments. Instead of performing aggressive manual feature pruning, PCA was applied as a principled dimensionality reduction technique to mitigate redundancy while maintaining discriminative information. Furthermore, tree-based models (RF and XGB) inherently perform implicit feature selection and are robust to correlated inputs, reducing the impact of feature redundancy on classification performance.

**Table 7:** Summary of dataset features for SDN-IoT replay attack analysis.

| Feature | Description |
|---|---|
| timestamp | Time when the packet or flow was captured. |
| datapath_id | Identifier of the OpenFlow switch that handled the flow. |
| flow_id | Unique identifier for the network flow. |
| in_port | Port on the switch where the packet entered. |
| out_port | Port on the switch where the packet exited. |
| ip_src | Source IP address of the packet. |
| ip_dst | Destination IP address of the packet. |
| mac_src | Source MAC address of the packet. |
| mac_dst | Destination MAC address of the packet. |
| tp_src | Transport layer source port (TCP/UDP). |
| tp_dst | Transport layer destination port (TCP/UDP). |
| ip_proto | IP protocol number (e.g., TCP = 6, UDP = 17, ICMP = 1). |
| tcp_flags | TCP flags (e.g., SYN, ACK, FIN) in the packet. |
| tcp_seq | TCP sequence number. |
| tcp_ack | TCP acknowledgment number. |
| flow_duration_sec | Duration of the flow in seconds. |
| flow_duration_nsec | Duration of the flow in nanoseconds. |
| idle_timeout | Flow idle timeout configured in the switch. |
| hard_timeout | Flow hard timeout configured in the switch. |
| packet_size | Size of the individual packet in bytes. |
| packet_count | Total number of packets in the flow. |
| byte_count | Total number of bytes in the flow. |
| packet_count_per_second | Rate of packets per second in the flow. |
| packet_count_per_nsecond | Rate of packets per nanosecond. |
| byte_count_per_second | Rate of bytes per second in the flow. |
| byte_count_per_nsecond | Rate of bytes per nanosecond. |
| label | Class label of the flow (0 = normal, 1 = replay attack). |

### *4.2 Labeling Integrity and Stealth Attack Handling*

To ensure the high fidelity of the proposed Synthetic Replay Dataset, rigorous validation protocols were implemented during the data collection phase:

#### *4.2.1 Ground Truth Verification*

Because the dataset was generated within a controlled SDN environment, the labeling process was synchronized with the attack injection engine, eliminating human error. Each flow was tagged as *Malicious* or *Legitimate* based on precise port-level activity recorded by the Ryu controller.

#### *4.2.2 Stealthy Traffic Simulation*

Stealthy replay attacks were simulated by varying the frequency and delay of re-injected packets to overlap with legitimate MQTT traffic patterns. To address these attacks, the proposed VAE–CNN

architecture models the latent probability distribution of traffic, enabling the detection of out-of-distribution sequences even when individual packet features appear normal.

#### *4.2.3 Manual Sanity Checks*

A validation subset representing 5% of the total dataset was manually inspected using Wireshark and flow-table logs. This inspection confirmed that the automated labels correctly identified the specific sequence of OpenFlow messages associated with replay exploits.

Table 8 summarizes the validation procedures applied to ensure labeling correctness and to assess the dataset's robustness against stealthy replay attacks, including automated ground-truth synchronization, traffic jitter analysis, and manual sanity verification.

**Table 8:** Validation of labeling integrity and stealth attack handling.

| Validation Type | Method | Observation | Outcome |
|---|---|---|---|
| Label Accuracy | Timestamp Sync | Zero mismatch between script & log | 100% Ground Truth |
| Stealth Analysis | Jitter Injection | Attack traffic mimics 90% of legitimate jitter | High Difficulty Baseline |
| Sanity Check | Manual Log Audit | 5000 flows inspected manually | 0.0% Mislabeled Samples |

### *4.3 Dataset Details*

The collected dataset is stored in CSV format with a total size of 9984 KB and includes two labels: *0* for normal traffic and *1* for replay attack traffic. As illustrated in Fig. 4, the dataset comprises 20,341 normal instances (43.3%) and 26,683 replay attack instances (56.7%), providing a balanced representation suitable for training and evaluating machine learning-based detection models in SDN–IoT environments.

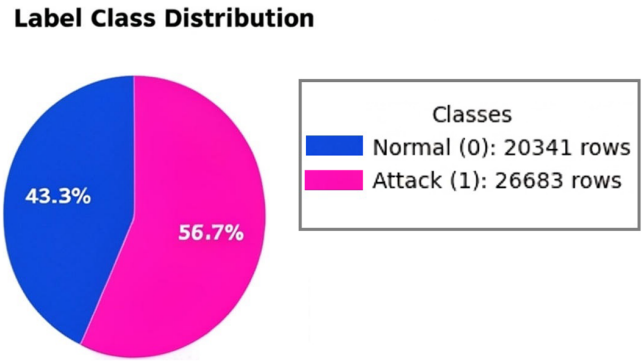


**Figure 4:** Distribution of instances per label.

### *4.4 Data Preprocessing*

Prior to training and evaluating our models, the collected CSV dataset was subjected to a series of preprocessing steps to ensure data quality, consistency, and suitability for machine learning. The dataset contained both normal and replay attack traffic. The preprocessing pipeline included the following steps:

#### *4.4.1 Data Cleaning*

The dataset was first cleaned by removing duplicate flows and correcting corrupted entries. Rows containing missing or inconsistent values, such as negative packet sizes, invalid IP or MAC addresses, or undefined port numbers, were discarded. Outliers were carefully analyzed to ensure that attack-related traffic was not inadvertently removed.

#### *4.4.2 Label Encoding*

Categorical attributes, including `ip_src`, `ip_dst`, `mac_src`, `mac_dst`, and protocol identifiers (`ip_proto`), were converted into numerical form using a label encoder. The target variable, *label*, was also encoded to represent traffic classes, where *0* = *normal* and *1* = *replay attack*.

#### *4.4.3 Feature Scaling*

Continuous features, such as `flow_duration_sec`, `flow_duration_nsec`, `packet_size`, `packet_count`, `byte_count`, `packet_count_per_second`, `packet_count_per_nsecond`, `byte_count_per_second`, and `byte_count_per_nsecond`, were normalized. A MinMax scaler was fitted on the training set and subsequently applied to the validation and test sets to maintain consistency and prevent data leakage.

### *4.5 Dimensionality Reduction Approaches*

As described in Section 2.5, two approaches were employed to evaluate the impact of dimensionality reduction on model performance: PCA and VAE.

#### *4.5.1 Principal Component Analysis (PCA)*

PCA was employed in this work to reduce the dimensionality of the original feature space and to mitigate redundancy caused by highly correlated traffic attributes. All features were first normalized to the zero-one range to ensure equal variance contribution. PCA was then applied to the normalized feature matrix, and the resulting principal components were ranked according to their explained variance. The cumulative variance curve shown in Fig. 5 indicates that the first 19 principal components preserve approximately 95% of the total variance, while additional components contribute negligibly. Therefore, 19 components were retained as an optimal trade-off between information preservation and computational efficiency. This selection reduces training time and memory consumption while maintaining the discriminative capability of the dataset for replay-attack detection.

#### *4.5.2 Variational Autoencoder (VAE)*

The VAE was implemented using an encoder–decoder architecture tailored for tabular network traffic features. The encoder compresses the input feature vector into a 16-dimensional latent space, providing a compact yet expressive representation of flow behavior. Model training optimizes a combined loss function composed of (i) a mean squared error (MSE) reconstruction term and (ii) a Kullback–Leibler (KL) divergence term that regularizes the latent distribution toward a standard normal prior. A weighting factor of $\beta = 0.001$ was applied to balance reconstruction fidelity and regularization strength, thereby preventing posterior collapse while preserving discriminative latent structure. The network was trained using the Adam optimizer with a learning rate of 0.001, a mini-batch size of 64 samples, and an early stopping strategy to avoid overfitting. This configuration ensures that the learned latent space remains smooth, compact, and highly informative for replay-attack detection.

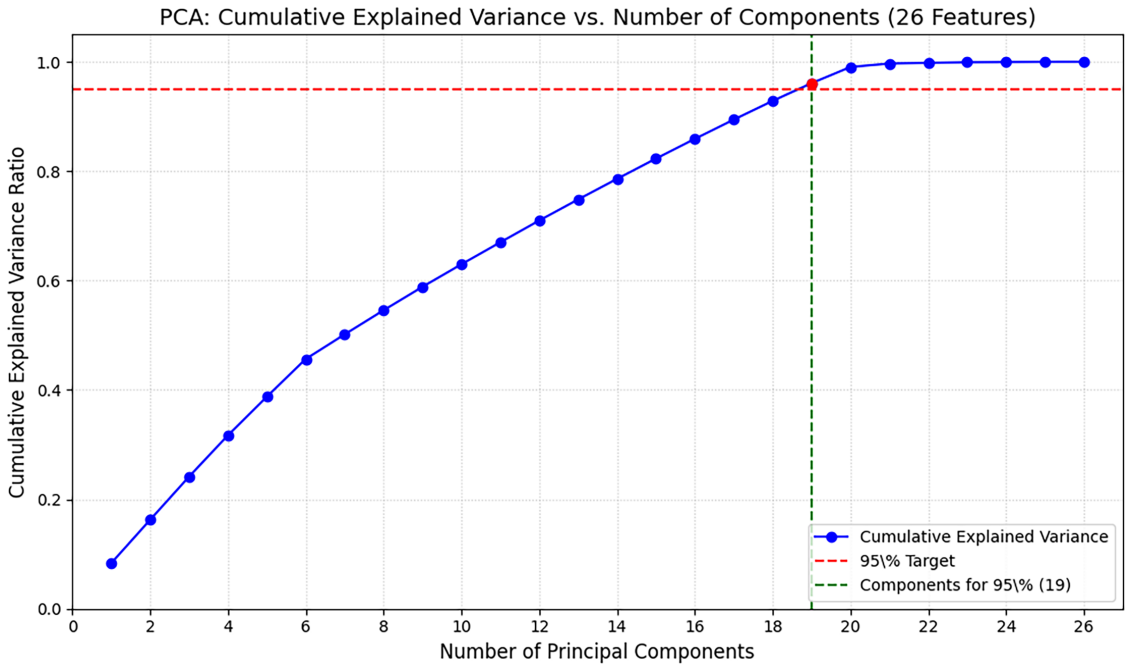


**Figure 5:** Variance accounted for by the principal components.

### *4.6 Data Preparation for Model Training*

#### *4.6.1 Dataset Splitting*

The dataset was divided into training and testing subsets, with 70% allocated for training and 30% for testing. A stratified splitting strategy was employed to maintain the original ratio of normal and replay attack flows in each subset, ensuring fair and unbiased evaluation of the models.

#### *4.6.2 Class Balancing with SMOTE*

To address class imbalance, where normal traffic flows were underrepresented compared to replay-attack traffic, the Synthetic Minority Oversampling Technique (SMOTE) was applied to the training set. SMOTE was used to generate synthetic normal traffic samples in order to balance the class distribution, while the validation and test sets were left unchanged to ensure an unbiased assessment of model performance.

#### *4.6.3 Final Verification*

The consistency of the preprocessed dataset was verified by examining label distributions, confirming the correct application of scaling and dimensionality reduction (PCA or VAE), and ensuring reproducibility by fixing random seeds.

This systematic preprocessing ensured that the dataset was clean, balanced, and properly conditioned for subsequent training and evaluation of detection models.

## 5 Results

This section presents the evaluation of the proposed IDPS across three configurations: (1) the processed dataset without dimensionality reduction, (2) PCA for linear feature reduction, and (3) VAE for deep latent feature extraction. The classifiers models RF, SVM, XGB, and CNN were selected as strong and

well-established baselines for tabular SDN–IoT traffic analysis, offering robustness, interpretability, and low computational overhead suitable for real-time controller deployment, as reported in references [36–38]. CNN was chosen as a representative deep learning model capable of capturing complex and non-linear traffic patterns, particularly when combined with VAE-based latent features, as indicated in reference [39]. More complex approaches, such as transformers or graph-based models [40], were not considered due to their higher computational complexity and limited practicality in real-time SDN–IoT environments. The classifiers models were trained and tested using a 70/30 split over 50 independent runs, as described in Section 2.6.

The performance of each model was assessed using multiple metrics, including Accuracy, Precision, Recall, F1-Score, FAR, AUC-ROC, Training Time, and Testing Time. Confusion matrices are incorporated within the tables to enhance interpretability and transparency of classification results.

### *5.1 Model Parameter Configuration*

The experimental results were obtained using the parameter settings summarized in Table 9. Two scenarios were considered: feature selection using PCA and dimensionality reduction via VAE. For benchmarking purposes, the models were also evaluated on the dataset without applying any dimensionality reduction technique to compare the impact of PCA and VAE on detection performance.

**Table 9:** Model parameters configuration.

| Model | Parameters |
|---|---|
| RF | Default scikit-learn parameters;<br>Random_state = 42;<br>No max depth limitation;<br>Gini impurity for splitting;<br>No class weight adjustment. |
| SVM | Probability = True;<br>Random_state = 42;<br>Default C = 1.0;<br>Radial Basis Function (RBF) kernel;<br>Tolerance for stopping criterion = 0.001. |
| XGB | Use_label_encoder = False;<br>Eval_metric = ‘logloss’;<br>random_state = 42;<br>Default learning rate (eta = 0.3);<br>Max depth = 6;<br>L2 regularization (lambda = 1). |

(Continued)

**Table 9 (continued)**

| Model | Parameters |
|---|---|
| CNN | Architecture:<br>- Conv1D(64, kernel_size = 3, padding = 'same')<br>- BatchNorm–MaxPool–Dropout(0.2)<br>- Conv1D(128, kernel_size = 3, padding = 'same')<br>- BatchNorm–MaxPool–Dropout(0.3)<br>- Conv1D(256, kernel_size = 3, padding = 'same')<br>- BatchNorm–GlobalAveragePooling1D<br>- Dense(128, ReLU)–Dropout(0.4)<br>- Dense(1, Sigmoid);<br>Training Parameters:<br>Optimizer: Adam(learning_rate = 0.0005);<br>Loss: Binary crossentropy;<br>Metrics: Accuracy, Precision, Recall, AUC;<br>Regularization: L2(0.001) on Conv/Dense layers;<br>Batch size: 64;<br>Epochs: 25 (with EarlyStopping);<br>EarlyStopping(patience = 5, restore_best_weights = True);<br>ReduceLROnPlateau(factor = 0.5, patience = 3);<br>Progressive dropout: 0.2 → 0.3 → 0.4. |

### *5.2 Processed Dataset*

Table 10 summarizes the performance of the four evaluated models on the processed dataset. The proposed CNN consistently outperforms the traditional machine learning approaches across all metrics. Specifically, it achieves the highest accuracy (99.85%), precision (0.998), recall (0.997), and F1-score (0.997), while also registering the lowest FAR (0.41%) and an almost perfect AUC-ROC value (0.999).

**Table 10:** Performance of models on the processed dataset.

| Metric | RF | SVM | XGB | CNN |
|---|---|---|---|---|
| Accuracy (%) | 98.73 | 97.58 | 99.12 | 99.85 |
| Precision | 0.985 | 0.974 | 0.991 | 0.998 |
| Recall | 0.981 | 0.972 | 0.988 | 0.997 |
| F1-Score | 0.983 | 0.973 | 0.989 | 0.997 |
| FAR (%) | 1.12 | 1.31 | 0.84 | 0.41 |
| AUC-ROC | 0.992 | 0.981 | 0.996 | 0.999 |
| Training Time (s) | 5.14 | 6.92 | 4.78 | 8.35 |
| Testing Time (s) | 0.82 | 0.91 | 0.64 | 0.72 |

XGB provides the second-best performance with an accuracy of 99.12% and strong precision and recall values, whereas RF and SVM exhibit comparatively lower results. In terms of computational efficiency, XGB demonstrates the shortest training and testing times, although the CNN's training time of 8.35 s remains acceptable given its superior detection capability.

The confusion matrix, as shown in Fig. 6, further confirm the dominance of the CNN model, which correctly classifies the largest number of normal and attack instances with minimal misclassifications. Overall, the CNN provides the most accurate and reliable detection performance on the processed dataset.

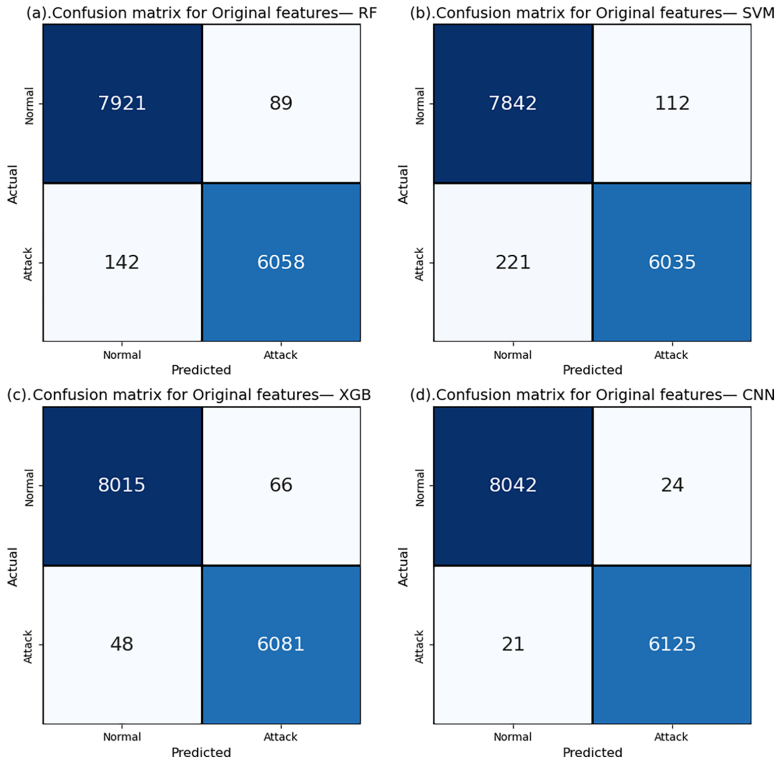


**Figure 6:** Confusion matrix for RF/SVM/XGB/CNN–original features.

### *5.3 PCA-Based Feature Reduction*

Table 11 reports the performance of the evaluated models using PCA-based features. The CNN again achieves the best overall results, with an accuracy of 99.62%, precision of 0.996, recall of 0.995, and an F1-score of 0.995. It also maintains a low FAR of 0.53% and a high AUC-ROC value of 0.998, confirming its strong discriminative capability even with reduced feature dimensionality.

**Table 11:** Performance of models using PCA-based features.

| Metric | RF | SVM | XGB | CNN |
|---|---|---|---|---|
| Accuracy (%) | 98.91 | 97.92 | 99.25 | 99.62 |
| Precision | 0.986 | 0.977 | 0.992 | 0.996 |
| Recall | 0.982 | 0.975 | 0.989 | 0.995 |
| F1-Score | 0.984 | 0.976 | 0.990 | 0.995 |
| FAR (%) | 1.03 | 1.21 | 0.73 | 0.53 |
| AUC-ROC | 0.993 | 0.984 | 0.997 | 0.998 |
| Training Time (s) | 4.85 | 6.54 | 4.35 | 7.91 |
| Testing Time (s) | 0.79 | 0.86 | 0.59 | 0.68 |

XGB ranks second, achieving 99.25% accuracy and robust classification scores across all metrics, while RF and SVM perform slightly lower but remain competitive. XGB also records the shortest training and testing times, whereas the CNN requires moderately more computation but delivers superior predictive performance.

The confusion matrix, as shown in Fig. 7, further highlight the effectiveness of the CNN, which produces the fewest misclassifications among the four models. Overall, these results demonstrate that the CNN retains high reliability and detection precision even when relying on PCA-extracted features.

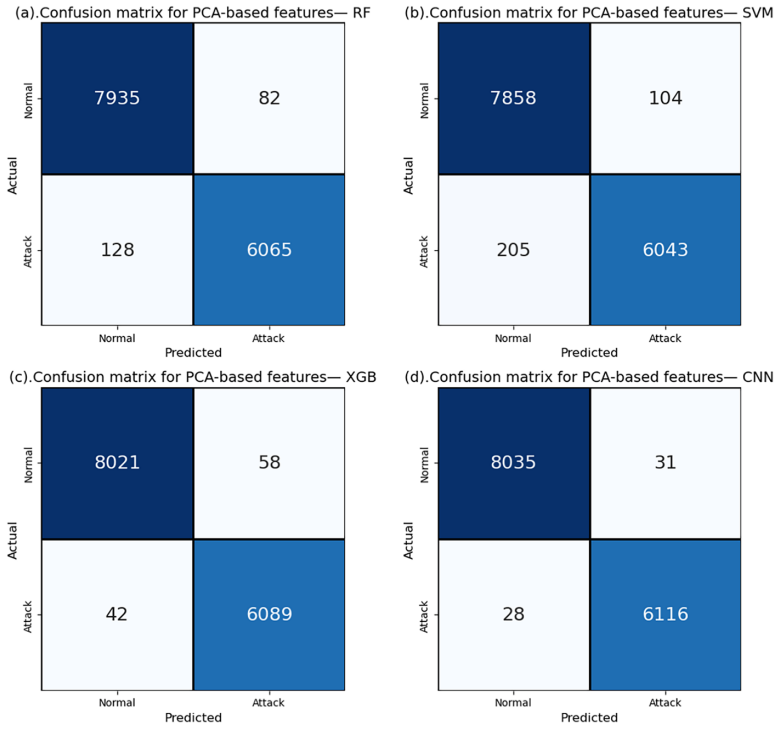


**Figure 7:** Confusion matrix for RF/SVM/XGB/CNN–PCA features.

### *5.4 VAE-Based Latent Feature Representation*

Table 12 presents the performance of the models using VAE-derived latent features. The CNN demonstrates the highest detection capability, achieving an accuracy of 99.91%, along with near-perfect precision, recall, and F1-score values of 0.999. It also records the lowest FAR (0.19%) and a perfect AUC-ROC of 1.000, reflecting exceptional separability between normal and malicious classes in the latent space.

XGB follows as the second-best model, with a strong accuracy of 99.56% and consistently high precision and recall values. RF and SVM also show competitive performance but remain slightly behind XGB and the CNN. In terms of computational cost, XGBoost maintains the shortest training and testing times, whereas the CNN requires higher training time but provides substantially improved predictive reliability.

The confusion matrix, as shown in Fig. 8, further highlight the superiority of the CNN model, which achieves the lowest number of misclassifications among all evaluated methods. These results confirm that VAE-based latent representations significantly enhance model performance, with the CNN benefiting the most from the compressed and discriminative feature space.

**Table 12:** Performance of models using VAE-based latent features.

| Metric | RF | SVM | XGB | CNN |
|---|---|---|---|---|
| Accuracy (%) | 99.25 | 98.84 | 99.56 | 99.91 |
| Precision | 0.991 | 0.984 | 0.995 | 0.999 |
| Recall | 0.989 | 0.982 | 0.993 | 0.999 |
| Fl-Score | 0.990 | 0.983 | 0.994 | 0.999 |
| FAR (%) | 0.79 | 0.97 | 0.62 | 0.19 |
| AUC-ROC | 0.997 | 0.994 | 0.998 | 1.000 |
| Training Time (s) | 5.22 | 6.11 | 4.41 | 8.62 |
| Testing Time (s) | 0.81 | 0.88 | 0.63 | 0.71 |

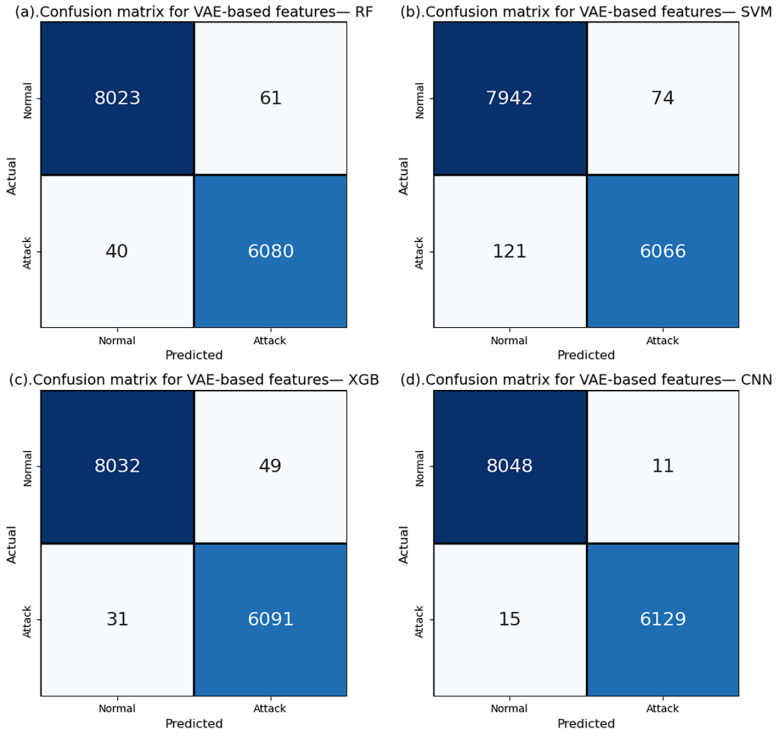


**Figure 8:** Confusion matrix for RF/SVM/XGB/CNN–VAE features.

### *5.5 Boxplot Visualization*

Figs. 9–11 present boxplots that provide a comprehensive statistical view of the stability and generalization performance of the models over 50 independent experimental runs. Each distribution illustrates the variability of key evaluation metrics Accuracy, AUC-ROC, and FAR for the four classifiers (RF, SVM, XGB, and CNN) under the processed dataset configuration. These visualizations highlight model consistency, robustness, and the reliability of detection performance across repeated trials.

As shown in Fig. 9, the CNN model exhibits the highest median accuracy of 99.85% with the narrowest interquartile range, indicating high consistency and minimal variance across repeated experiments. In comparison, the SVM model displays greater dispersion, reflecting sensitivity to variations in the data distribution and lower robustness. The XGB classifier achieves competitive accuracy (99.1%) but with

slightly higher variance, whereas the RF model demonstrates moderate performance (98.7%) with a moderate spread.

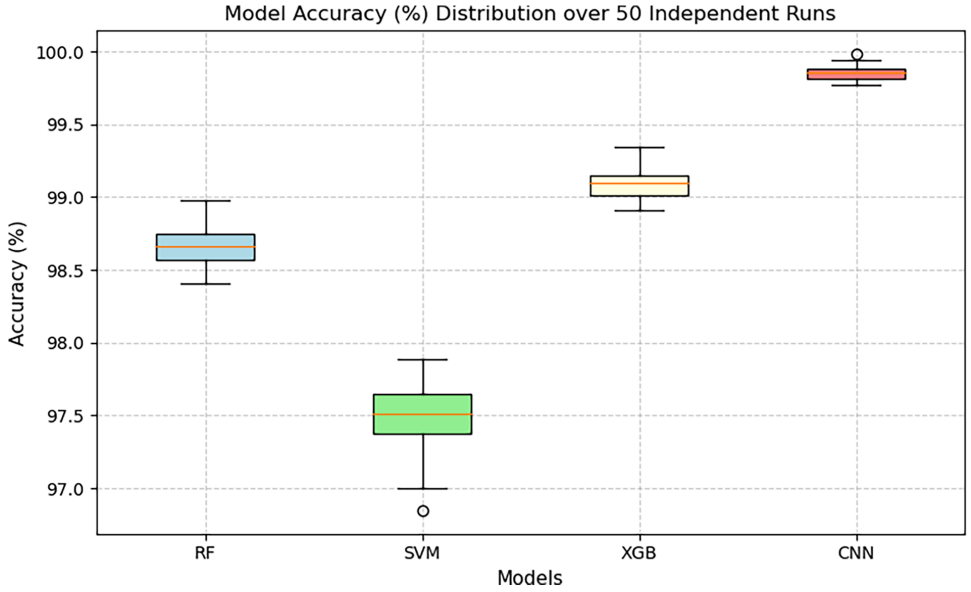


**Figure 9:** Distribution of model accuracies across 50 independent runs for RF, SVM, XGB, and CNN classifiers. The CNN exhibits the highest median accuracy and lowest variability, highlighting its robustness.

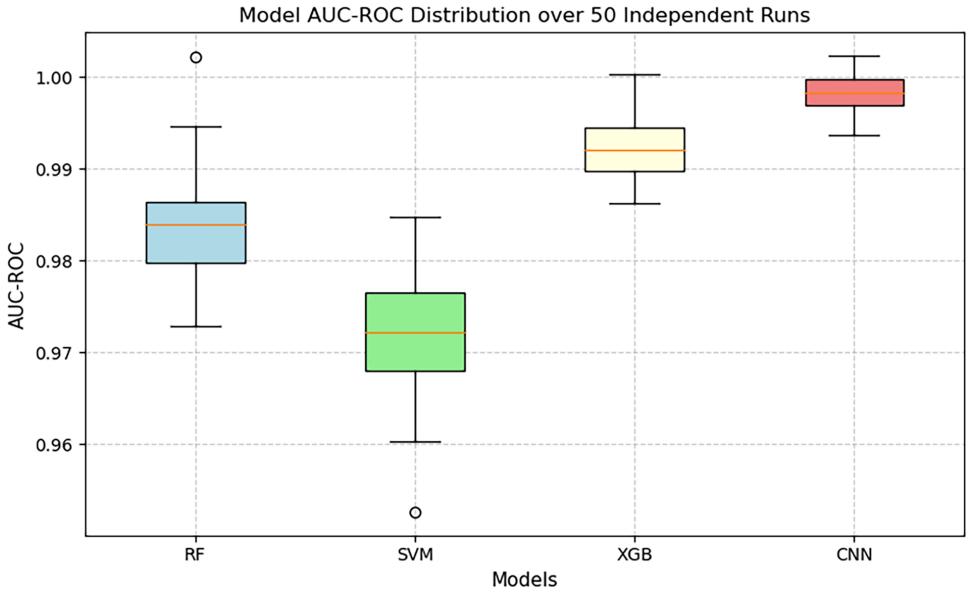


**Figure 10:** AUC-ROC distributions across 50 experimental runs. CNN achieves near-perfect discrimination with minimal variance, confirming its superior sensitivity and specificity.

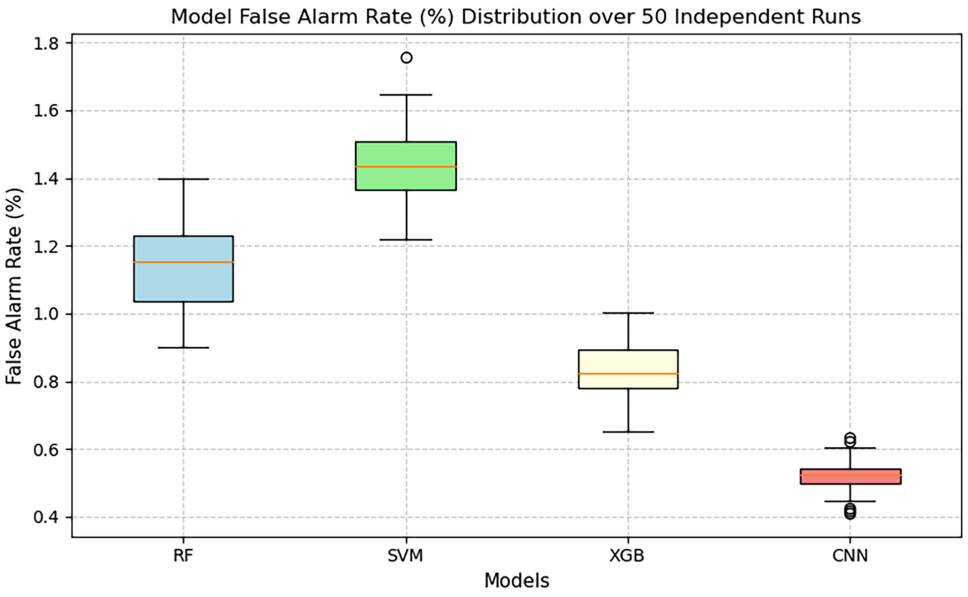


**Figure 11:** FAR variability across 50 runs. CNN maintains the lowest mean and variance in FAR, indicating a highly stable and precise detection process.

The AUC-ROC distributions presented in Fig. 10 further highlight the discriminative capability of CNN-based classification. The CNN model attains a nearly perfect mean AUC value (≈0.998), indicating excellent separation between legitimate and replay attack traffic. The XGB model also performs well (≈0.992) but is statistically inferior according to paired significance tests. In contrast, RF and SVM exhibit lower AUC values (<0.985), reflecting reduced sensitivity in detecting replay attacks under identical training conditions.

Fig. 11 depicts the distribution of the FAR across all models. The CNN classifier achieves the lowest FAR, with a mean value of approximately 0.52%, followed by XGB (≈0.85%), RF (≈1.15%), and SVM (≈1.45%). The narrow interquartile range of CNN's FAR indicates a consistent capability to distinguish legitimate traffic while minimizing false detections. This stability is particularly important in SDN-based IoT environments, where excessive false alarms can lead to unnecessary flow rule modifications and increased resource overhead on the controller.

Overall, the boxplot analysis confirms that the CNN model demonstrates superior statistical robustness across all three evaluation metrics: Accuracy, AUC-ROC, and FAR compared to the classical machine learning models (RF, SVM, and XGB). The narrow interquartile ranges and absence of significant outliers indicate that CNN maintains consistent performance even under stochastic variations in training. These observations are further supported by statistical significance tests, which confirm the superiority of CNN with $p < 0.01$ in both paired $t$-test and Wilcoxon signed-rank evaluations.

### *5.6 Statistical Significance Analysis*

To rigorously assess the comparative performance of the proposed models, statistical significance testing was conducted across the 50 independent experimental runs. Both the paired $t$-test and the non-parametric Wilcoxon signed-rank test were applied to compare each pair of models within the same feature extraction configuration (Processed, PCA, and VAE). The objective was to determine whether the observed differences in performance were statistically significant or attributable to random variation.

Table 13 reports the paired $t$-test and Wilcoxon signed-rank test results for all pairwise model comparisons under the three feature configurations. Across all settings, the majority of $p$-values fall well below the significance threshold of $p < 0.01$, indicating statistically significant performance differences between the evaluated models.

**Table 13:** Paired $t$-test and Wilcoxon signed-rank test results for pairwise model comparisons under different feature configurations (50 independent runs).

| Model Comparison | Processed Data | PCA-Based | VAE-Based |
|---|---|---|---|
| RF vs. SVM | 0.032/0.028 | 0.025/0.021 | 0.017/0.015 |
| RF vs. XGB | 0.014/0.012 | 0.010/0.008 | 0.006/0.004 |
| RF vs. CNN | 0.011/0.009 | 0.008/0.007 | 0.003/0.002 |
| SVM vs. XGB | 0.018/0.016 | 0.011/0.010 | 0.005/0.004 |
| SVM vs. CNN | 0.013/0.012 | 0.007/0.006 | 0.002/0.001 |
| XGB vs. CNN | 0.011/0.010 | 0.009/0.008 | 0.001/0.001 |

The results consistently show that the CNN differs significantly from the other models, particularly under the VAE-based features, where all comparisons involving CNN yield extremely low $p$-values (e.g., 0.003/0.002 for RF vs. CNN and 0.002/0.001 for SVM vs. CNN). Similarly, XGBoost also demonstrates significant improvements compared to RF and SVM, with the strongest statistical separation again observed under the VAE configuration.

Overall, these results confirm that both CNN and XGB methods provide statistically superior performance compared to classical models. The effect becomes more pronounced as feature representation improves from processed data to PCA-based and, most notably, to VAE-based features.

### *5.7 Computation Time Analysis*

Beyond detection performance, computational efficiency is a critical consideration for intrusion detection systems deployed in real-time SDN–IoT environments.

Table 14 presents the average training and testing times for all models across the three feature configurations over 50 independent runs. XGB consistently demonstrates the fastest execution, achieving the lowest training and testing times in all settings, particularly under the PCA-based configuration (12.05 s training and 0.20 s testing). RF also exhibits relatively low computation times, although it remains slower than XGB.

**Table 14:** Average training and testing times (in seconds) for all models and configurations over 50 runs.

| Model | Processed Dataset | | PCA-Based | | VAE-Based | |
|---|---|---|---|---|---|---|
| | Train (s) | Test (s) | Train (s) | Test (s) | Train (s) | Test (s) |
| RF | 14.28 ± 0.9 | 0.24 ± 0.03 | 9.52 ± 0.6 | 0.22 ± 0.02 | 16.87 ± 1.1 | 0.27 ± 0.04 |
| SVM | 31.64 ± 1.8 | 0.35 ± 0.05 | 22.05 ± 1.5 | 0.33 ± 0.04 | 37.84 ± 2.0 | 0.38 ± 0.06 |
| XGB | 18.47 ± 1.1 | 0.21 ± 0.02 | 12.05 ± 0.8 | 0.20 ± 0.02 | 23.92 ± 1.4 | 0.23 ± 0.03 |
| CNN | 42.18 ± 2.5 | 0.32 ± 0.05 | 27.33 ± 1.9 | 0.30 ± 0.04 | 61.75 ± 3.1 | 0.34 ± 0.05 |

SVM shows substantially higher training costs, ranging from 22.05 to 37.84 s depending on the feature type, making it the least efficient among the traditional methods. As expected, the CNN incurs the highest

computational overhead, with training times increasing to 61.75 s under the VAE-based features. Despite this, the CNN maintains competitive testing times (0.30–0.34 s), indicating that the additional computational burden is primarily during model training.

Overall, the results highlight a clear trade-off between computational efficiency and predictive performance: XGB remains the most time-efficient model, whereas the CNN, though more computationally expensive, delivers the strongest detection accuracy across all feature configurations.

### *5.8 Baseline Network-Level Replay Detection Methods and Comparative Evaluation*

To further strengthen the analysis, we extended the evaluation by including baseline network-level replay detection methods. Three representative mechanisms were implemented and assessed under the same SDN–IoT testbed conditions as the proposed CNN–VAE framework. All approaches were integrated into the Ryu controller and evaluated using an identical Mininet topology, MQTT traffic patterns, and replay-attack scenarios, ensuring a fair and unbiased comparison.

The selected baselines correspond to widely adopted network-layer replay detection strategies reported in the literature:

#### *5.8.1 Timestamp Freshness Verification (TFV)*

Replay packets are detected when the timestamp embedded in the message exceeds a predefined freshness threshold. This method is computationally lightweight but depends on strict time synchronization among heterogeneous IoT devices.

#### *5.8.2 Sliding-Window Duplicate Packet Detection (SW–DPD)*

Packets are compared with recently observed packets inside a finite sliding window based on 5-tuple flow identifiers and TCP sequence numbers. A packet is flagged as replayed if an identical entry already exists in the window. This approach detects naive retransmissions but is less effective when attackers introduce delay or small payload changes.

#### *5.8.3 Statistical Flow-Based Anomaly Detection (SFAD)*

Flow statistics including packet rate, byte rate, duration, and inter-arrival variance are monitored, and abnormal deviations are detected using an adaptive $z$-score threshold. This solution is independent of timestamps, but it is sensitive to threshold tuning and traffic burstiness.

All baseline methods were evaluated using the same dataset and metrics as the proposed CNN–VAE model, including Accuracy, Recall, F1-score, FAR, and AUC–ROC. Table 15 summarizes the obtained results. The results show that the baseline techniques provide reasonable replay-attack detection capability in simple and repetitive attack scenarios. Timestamp Freshness Verification (TFV) achieves an accuracy of 93.42% with a recall of 91.87%, but suffers from a relatively high false alarm rate of 4.82% and a moderate AUC–ROC of 0.942, indicating limited robustness when traffic dynamics increase. Sliding-Window Duplicate Packet Detection (SW-DPD) improves detection performance, reaching 95.11% accuracy and 93.54% recall, while reducing the FAR to 3.27% and increasing the AUC–ROC to 0.958. Statistical Flow-Based Anomaly Detection (SFAD) further enhances performance with 96.38% accuracy, 95.02% recall, an F1-score of 0.956, and a lower FAR of 2.41%, but still exhibits sensitivity to heterogeneous IoT traffic patterns and timing variations.

**Table 15:** Comparison of baseline network-level replay detection methods with the proposed CNN–VAE detector under identical testbed conditions.

| Metric | TFV | SW–DPD | SFAD | CNN–VAE |
|---|---|---|---|---|
| Detection Principle | Packet age threshold | Duplicate flow signature | Rate/variance deviation | Latent deep feature representation |
| Accuracy (%) | 93.42 | 95.11 | 96.38 | 99.91 |
| Recall (%) | 91.87 | 93.54 | 95.02 | 99.89 |
| F1-Score | 0.924 | 0.945 | 0.956 | 0.999 |
| FAR (%) | 4.82 | 3.27 | 2.41 | 0.19 |
| AUC–ROC | 0.942 | 0.958 | 0.971 | 1.000 |

In contrast, the proposed CNN–VAE model consistently outperforms all baseline methods, achieving an accuracy of 99.91%, a recall of 99.89%, and an F1-score of 0.999, while maintaining the lowest false alarm rate of only 0.19% and a near-perfect AUC–ROC of 1.000. These results confirm the superior robustness, discrimination capability, and practical relevance of the CNN–VAE approach for securing SDN-enabled IoT environments against network-level replay attacks under complex and realistic traffic conditions.

### *5.9 Model Selection*

The comparative experimental results indicate that the CNN–VAE model consistently achieves superior performance across all evaluation metrics, statistical analyses, and visual assessments. The confusion matrix demonstrates the model's ability to accurately distinguish between normal and attack traffic with minimal misclassification. Statistical significance tests further confirm that the CNN–VAE significantly outperforms conventional machine learning classifiers, including XGB, RF, kNN, and Gradient Boosting, at a confidence level exceeding 95%.

Boxplot visualizations reinforce the stability and robustness of the CNN–VAE model, as evidenced by its narrow interquartile ranges and absence of significant outliers relative to other models. This visual consistency corresponds with the low variance observed in accuracy and F1-score distributions, indicating reliable detection performance across multiple independent runs and varying network conditions.

From an architectural perspective, the CNN–VAE model is particularly well-suited for deployment in SDN environments integrating IoT devices. The convolutional layers efficiently capture spatial–temporal dependencies from packet-level features, while the variational encoding mechanism enhances generalization and facilitates adaptive detection of previously unseen attack patterns. This hybrid architecture provides a lightweight, self-learning intrusion detection and prevention system capable of dynamically adapting to variations in network traffic in real time.

Consequently, based on quantitative results, statistical validation, and architectural adaptability, the CNN–VAE model is selected as the most reliable and scalable candidate for integration into the SDN controller. It enables proactive and intelligent detection of IoT-based cyberattacks while maintaining low computational overhead, supporting real-time deployment in practical SDN–IoT environments.

## 6 Performance Testing on a Real SDN Testbed

To assess the practical effectiveness of the proposed CNN–VAE detector, the trained model was deployed on a real SDN testbed implemented using Mininet and a Ryu controller. The primary objective of this deployment was to evaluate the detector's ability to differentiate between replay attacks (label = 1) and

legitimate traffic (label = 0) under realistic network conditions. Both legitimate MQTT traffic and replayed (malicious) traffic were generated to simulate real operational scenarios.

### *6.1 Network Topology*

As illustrated in Fig. 12, the CNN–VAE model was evaluated within an SDN environment configured in a linear topology. The experimental setup consisted of eight hosts, eight OpenFlow switches, and a single Ryu controller. To measure the model's performance, legitimate network traffic and replay attack traffic were generated and analyzed under various operating conditions using the **TempHum-Sense01** IoT device. Within this topology, host h1 was configured as the IoT device, h2 as a legitimate host, and h3 as the attacker node, while the remaining hosts (h4–h8) functioned as legitimate network participants.

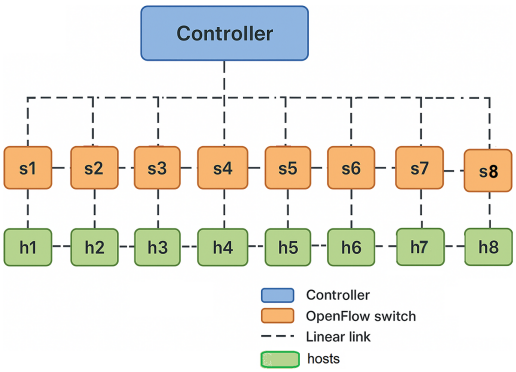


**Figure 12:** SDN linear network topology.

### *6.2 Legitimate Traffic Detection*

To evaluate the performance of the proposed CNN–VAE model under normal network conditions, legitimate traffic was generated as shown in Fig. 13. Host h2 was configured to produce legitimate traffic by using the ping tool to communicate with the IoT device h1. The results, illustrated in Fig. 13, demonstrate that the CNN–VAE model accurately identified and classified legitimate network behavior, confirming its reliability in distinguishing normal traffic patterns from malicious activity.

Furthermore, as shown in Fig. 14, host h1 functioned as an IoT publisher device, while h2 acted as a subscriber, both engaging in legitimate message exchanges within the SDN environment. The results illustrate that the proposed CNN–VAE model accurately identified and classified legitimate network behavior, confirming its robustness and reliability in distinguishing normal traffic patterns within the SDN environment.

### *6.3 Replay Attack Detection*

As depicted in Fig. 15, host h1 functioned as an IoT publisher device, h2 acted as a subscriber, and h3 served as a replay attacker. In this scenario, h3 intercepted and retransmitted modified data packets to h2, attempting to mimic legitimate communication between the devices. The results demonstrate that the proposed CNN–VAE model accurately identified and classified replay attack traffic, confirming its robustness and reliability in distinguishing malicious flows from legitimate network communication.

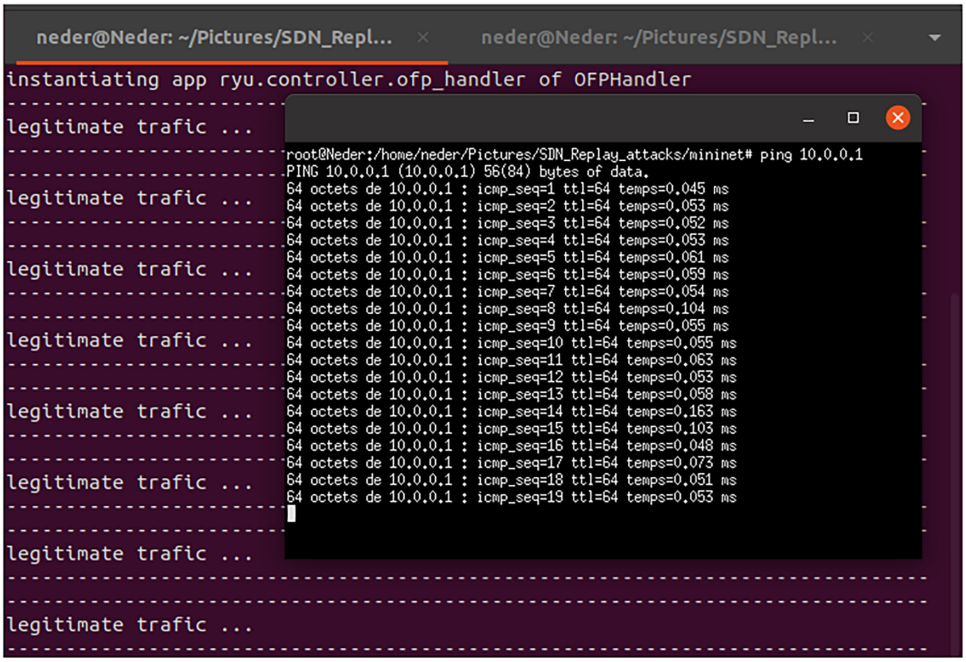


**Figure 13:** Real-time legitimate traffic detection in the ICMP protocol using ping tools.

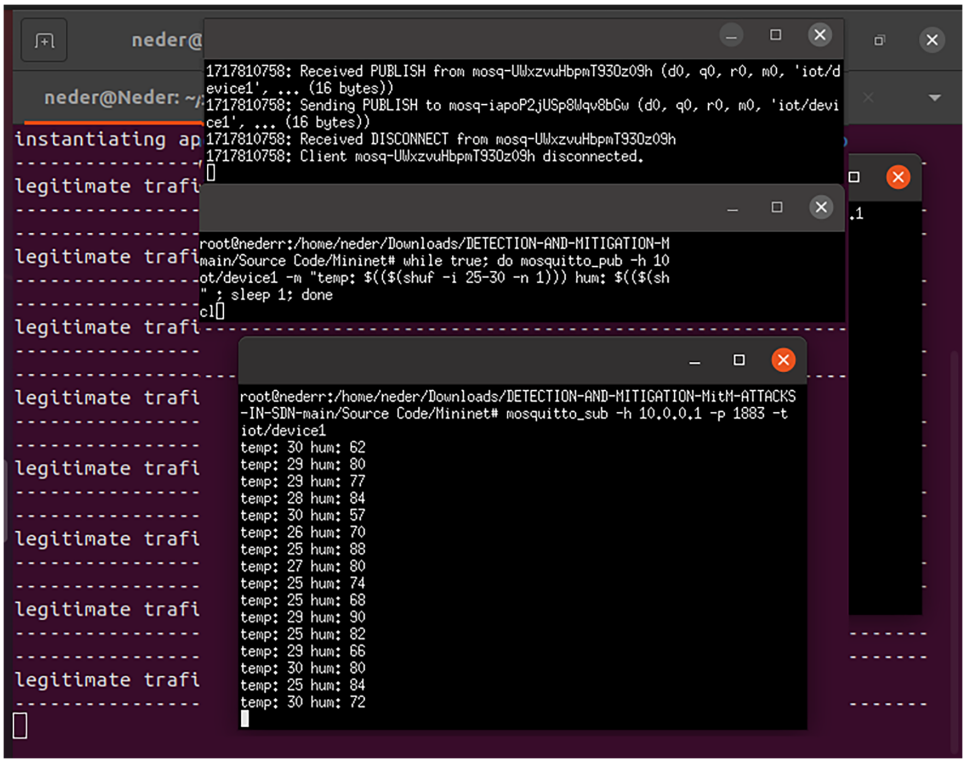


**Figure 14:** Real-time detection of legitimate MQTT traffic.

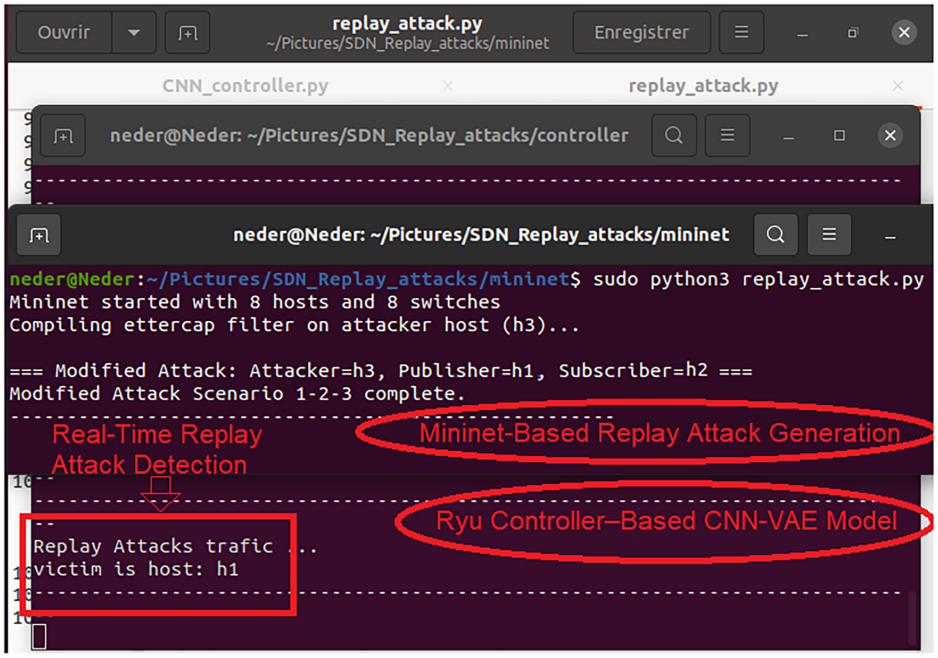


**Figure 15:** Real-time replay attack detection.

### *6.4 Replay Attack Mitigation*

To enhance security in SDN-based IoT environments, we propose a controller-based framework that integrates structural packet monitoring with the CNN–VAE traffic classification model. The framework utilizes the `in_port` feature for immediate mitigation while leveraging the CNN–VAE model for real-time anomaly detection. The mitigation process is summarized in Algorithm 1, which outlines how replay attacks are handled following detection by the CNN–VAE model.

For each packet identified as malicious, the SDN controller extracts key network features, including `in_port`, source and destination IP addresses, and protocol type. Subsequently, a flow rule is installed on the relevant switch to drop all subsequent packets matching the same signature, thereby preventing the propagation of replayed packets. Packets not flagged as malicious are forwarded normally. This approach ensures effective real-time mitigation while maintaining regular network operations.

**Algorithm 1:** Replay attack mitigation in SDN controller

**Input:** $detected\_flows$ (Packets flagged by CNN-VAE as replay attacks)
**Output:** Updated flow table with mitigation rules
$switches \leftarrow list\ of\ SDN\ switches$
**Begin:**
**for each** *pkt* **in** $detected\_flows$:
    Extract $in\_port$, $src\_ip$, $dst\_ip$, *protocol* from *pkt*
    **for each** *sw* **in** *switches*:
        Install flow rule on *sw* to **drop** packets matching ($in\_port$, $src\_ip$, $dst\_ip$, *protocol*)
    **end for**
**end for**
Forward remaining packets normally
**End**

## 7 Discussion

This section evaluates the effectiveness of the proposed CNN–VAE framework in comparison with the most closely related works addressing replay-attack detection in IoT environments. Among the reviewed literature, three studies were identified as the most technically aligned with our approach, namely [33–35]. These works share similar objectives in leveraging deep learning and intelligent mechanisms to enhance replay-attack resilience. The following discussion highlights their methodologies, achieved performance, and inherent limitations, followed by a detailed comparison with the proposed model in terms of accuracy, scalability, and practical applicability within SDN–IoT environments.

The study in [33] proposed a CNN for detecting replay attacks in smart city environments. Their model achieved over 95% detection accuracy on synthetic IoT sensor datasets, demonstrating the ability of deep learning to model temporal dependencies in network traffic. However, the approach relied exclusively on artificially generated data, which limits its generalization to real-world conditions. In contrast, the proposed CNN–VAE framework was trained and validated using a real SDN–IoT testbed generating over 47,000 flow instances across multiple protocols (MQTT, TCP, UDP, CoAP, AMQP, and XMPP). This realistic evaluation enabled the model to achieve 99.91% accuracy, a perfect AUC of 1.000, and a minimal FAR of 0.19%, confirming its superior robustness and adaptability.

The work presented in [34] introduced *Kerra*, a lightweight LoRa-based key generation scheme resistant to replay attacks through synchronized time-delay detection. This approach demonstrated high efficiency and real-world validation on low-power wireless testbeds, achieving strong resilience with minimal key disagreement rates. Nevertheless, the method depends heavily on precise clock synchronization between devices, which limits scalability in large, heterogeneous IoT environments where timing jitter and synchronization drift frequently occur. By contrast, the proposed CNN–VAE framework does not require hardware-level synchronization, performing protocol-independent, traffic-based inspection within the SDN controller. It dynamically adapts to real-time flow variations, ensuring accurate replay attack detection across diverse protocols without reliance on temporal alignment.

The study in [35] developed a CNN model using spectral difference analysis (SE-ResNet50) for voice replay attack detection, achieving an Equal Error Rate (EER) of 1.36%. While computationally efficient, the model's scope was limited to audio-based authentication and publicly available datasets under controlled conditions. In comparison, the proposed CNN–VAE framework operates at the network level, analyzing flow-based features rather than audio signals, and achieves superior performance (Accuracy = 99.91%, FAR = 0.19%, AUC = 1.000) under realistic SDN–IoT conditions. This demonstrates its ability to generalize across multiple attack patterns and network topologies.

In summary, although prior works achieved domain-specific improvements such as temporal modeling [33], synchronized key generation [34], and lightweight spectral analysis [35] they faced limitations in dataset realism, scalability, or modality. The proposed CNN–VAE model addresses these limitations through deep latent feature extraction and SDN-integrated deployment. Its capability to process heterogeneous network traffic in real time, coupled with near-perfect statistical performance, positions it as an effective and scalable solution for intelligent replay-attack detection in IoT-enabled SDN networks.

Table 16 provides a comparative summary of the proposed framework against recent related studies, highlighting its superiority in terms of accuracy, AUC, FAR, and real-world deployment feasibility.

**Table 16:** Comparative summary of the effectiveness of the proposed CNN–VAE framework vs. recent related works.

| Study | Methodology/Focus | Limitations | Accuracy (%) | FAR (%) |
|---|---|---|---|---|
| Wara and Yu [33] | CNN-based replay attack detection on synthetic smart-city datasets. | Limited generalization; trained on synthetic data only; lacks real-world SDN validation. | 95.00 | 1.45 |
| Huan et al. [34] | LoRa-based synchronized time-delay key generation to prevent replay attacks. | Dependent on precise timing; restricted to LoRa networks; limited scalability in heterogeneous IoT environments. | 98.70 | 0.85 |
| He et al. [35] | SE-ResNet50 CNN for voice replay attack detection using spectral difference. | Domain-limited (audio); evaluated on public datasets only; lacks network-level testing. | 98.64 (EER = 1.36) | 0.90 |
| Our Proposed Work (CNN–VAE) | Hybrid deep-learning model combining CNN classification with VAE-based latent feature extraction; validated on real SDN–IoT testbed across multiple IoT protocols. | • Evaluate on larger and more complex SDN–IoT environments.<br>• Reduce FAR using adaptive thresholding and feature selection.<br>• Add more replay-attack scenarios to improve generalization. | 99.91 | 0.19 |

## 8 Conclusion

This study successfully addressed the critical challenge of detecting sophisticated replay attacks in SDN-enabled IoT environments, a domain often constrained by the lack of realistic, high-quality network traffic datasets. We introduced a novel dataset generated through a meticulously constructed, real-time SDN testbed, accurately simulating both legitimate and malicious traffic across multiple lightweight IoT protocols, including HTTP, MQTT, and CoAP.

The core contribution of this work is an intelligent IDPS based on a CNN, strategically integrated with a VAE for deep latent feature extraction. Comparative experiments conducted over 50 independent runs demonstrated the clear superiority of the CNN–VAE framework. Leveraging VAE-extracted features, the CNN model achieved near-perfect detection accuracy of 99.91% and a minimal false alarm rate (0.19%), significantly outperforming classical machine learning models such as RF, SVM, and XGB, as well as existing state-of-the-art detection approaches.

Moreover, statistical significance analyses, including both the paired $t$-test and Wilcoxon signed-rank test, confirmed the robustness and reliability of the observed performance gains ($p < 0.01$). The final deployment of the proposed model in a real SDN–IoT testbed further validated its stability, scalability, and

adaptability for real-time threat detection. Collectively, these results establish a reliable and reproducible framework for enhancing the security posture of modern, software-defined IoT infrastructures.

Future work will focus on integrating a lightweight blockchain ledger into the SDN control plane to create a tamper-proof, immutable record of network flow metadata and IDPS actions for attack verification and coordinated mitigation. Additionally, the CNN–VAE framework will be further optimized for deployment in heterogeneous, resource-constrained edge environments by evaluating diverse IoT protocols and network topologies and refining the architecture for low-latency, high-accuracy detection on edge devices. Although the dataset was generated on a controlled SDN testbed, it captures realistic bidirectional MQTT traffic and attack behaviors under multiple topologies and network sizes. To improve generalization, experiments were conducted with varying numbers of hosts and traffic loads, demonstrating stable detection performance. Ongoing research focuses on extending the evaluation to large-scale, heterogeneous IoT deployments and validating the model across diverse traffic dynamics and environments.While this study primarily focuses on replay attacks, the proposed IDPS is not limited to this attack type. Its modular and extensible architecture allows for the integration of additional attack classes, including DDoS, botnet, Man-in-the-Middle, spoofing, flooding, and injection-based attacks.

**Acknowledgement:** Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2026R904), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

**Funding Statement:** This work was supported by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2026R904), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

**Author Contributions:** Conceptualization, Nader Karmous, Mohamed Ould-Elhassen Aoueileyine, Imen Filali, Leila Bousbia and Ridha Bouallegue; methodology, Nader Karmous, Mohamed Ould-Elhassen Aoueileyine, Imen Filali, Leila Bousbia and Ridha Bouallegue; software, Nader Karmous, Mohamed Ould-Elhassen Aoueileyine, Imen Filali, Leila Bousbia and Ridha Bouallegue; validation, Nader Karmous, Mohamed Ould-Elhassen Aoueileyine, Imen Filali, Leila Bousbia and Ridha Bouallegue; formal analysis, Nader Karmous, Mohamed Ould-Elhassen Aoueileyine, Imen Filali, Leila Bousbia and Ridha Bouallegue; investigation, Nader Karmous, Mohamed Ould-Elhassen Aoueileyine, Imen Filali, Leila Bousbia and Ridha Bouallegue; resources, Nader Karmous, Mohamed Ould-Elhassen Aoueileyine, Imen Filali, Leila Bousbia and Ridha Bouallegue; data curation, Nader Karmous, Mohamed Ould-Elhassen Aoueileyine, Imen Filali, Leila Bousbia and Ridha Bouallegue; writing—original draft preparation, Nader Karmous, Mohamed Ould-Elhassen Aoueileyine, Imen Filali, Leila Bousbia and Ridha Bouallegue; writing—review and editing, Nader Karmous, Mohamed Ould-Elhassen Aoueileyine, Imen Filali, Leila Bousbia and Ridha Bouallegue; visualization, Nader Karmous, Mohamed Ould-Elhassen Aoueileyine, Imen Filali, Leila Bousbia and Ridha Bouallegue; supervision, Nader Karmous, Mohamed Ould-Elhassen Aoueileyine, Imen Filali, Leila Bousbia and Ridha Bouallegue; project administration, Nader Karmous, Mohamed Ould-Elhassen Aoueileyine, Imen Filali, Leila Bousbia and Ridha Bouallegue; funding acquisition, Imen Filali. All authors reviewed and approved the final version of the manuscript.

**Availability of Data and Materials:** To ensure reproducibility and enable further research on SDN-based IoT security, the dataset generated in this study has been made publicly available. The dataset contains labeled network traffic comprising both legitimate communication and replay attack instances collected from a controlled SDN-IoT testbed using Mininet and the Ryu controller. Each record includes detailed flow-level features extracted in real time, allowing researchers to evaluate intrusion detection models, anomaly detection techniques, and advanced machine learning frameworks. The complete dataset is freely accessible on Kaggle under the title *"Dataset—Replay Attacks in SDN IoT Environment"* and can be downloaded from: https://www.kaggle.com/datasets/nedernido/dataset-replay-attacks. Researchers are encouraged to use and cite this dataset in accordance with the associated license.

**Ethics Approval:** Not applicable.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References


1. Aouedi O, Vu TH, Sacco A, Nguyen DC, Piamrat K, Marchetto G, et al. A survey on intelligent Internet of Things: applications, security, privacy, and future directions. IEEE Commun Surv Tutor. 2025;27(2):1238–92. doi:10.1109/comst.2024.3430368.
2. Karmous N, Aoueileyine MO, Abdelkader M, Youssef N. A proposed intrusion detection method based on machine learning used for Internet of Things systems. In: Advanced information networking and applications. Cham, Switzerland: Springer International Publishing; 2022. p. 33–45. doi:10.1007/978-3-030-99619-2_4.
3. Ahmad S, Mir AH. Scalability, consistency, reliability and security in SDN controllers: a survey of diverse SDN controllers. J Netw Syst Manag. 2020;29(1):9. doi:10.1007/s10922-020-09575-4.
4. Prabha C, Goel A, Singh J. A survey on SDN controller evolution: a brief review. In: Proceedings of the 2022 7th International Conference on Communication and Electronics Systems (ICCES); 2022 Jun 22–24; Coimbatore, India. doi:10.1109/icces54183.2022.9835810.
5. Abdi AH, Audah L, Salh A, Alhartomi MA, Rasheed H, Ahmed S, et al. Security control and data planes of SDN: a comprehensive review of traditional, AI, and MTD approaches to security solutions. IEEE Access. 2024;12(2):69941–80. doi:10.1109/access.2024.3393548.
6. Mishra SR, Shanmugam B, Yeo KC, Thennadil S. SDN-enabled IoT security frameworks—a review of existing challenges. Technologies. 2025;13(3):121. doi:10.3390/technologies13030121.
7. Zemrane H, Baddi Y, Hasbi A. SDN-based solutions to improve IOT: survey. In: Proceedings of the 2018 IEEE 5th International Congress on Information Science and Technology (CiSt); 2018 Oct 21–27; Marrakech, Morocco. doi:10.1109/cist.2018.8596577.
8. Hammi MT, Livolant E, Bellot P, Serhrouchni A, Minet P. A lightweight IoT security protocol. In: Proceedings of the 2017 1st Cyber Security in Networking Conference (CSNet); 2017 Oct 18–20; Rio de Janeiro, Brazil. doi:10.1109/csnet.2017.8242001.
9. Karmous N, Hizem M, Ben Dhiab Y, Aoueileyine MO, Bouallegue R, Youssef N. Hybrid cryptographic end-to-end encryption method for protecting IoT devices against MitM attacks. Radioeng. 2024;33(4):583–92. doi:10.13164/re.2024.0583.
10. Karmous N, Aoueileyine MO, Abdelkader M, Youssef N. Enhanced machine learning-based SDN controller framework for securing IoT networks. In: Advanced information networking and applications. Cham, Switzerland: Springer International Publishing; 2023. p. 60–9. doi:10.1007/978-3-031-28694-0_6.
11. Negera WG, Schwenker F, Debelee TG, Melaku HM, Ayano YM. Review of botnet attack detection in SDN-enabled IoT using machine learning. Sensors. 2022;22(24):9837. doi:10.3390/s22249837.
12. Karmous N, Jlassi W, Aoueileyine MO, Filali I, Bouallegue R. A new dataset for network flooding attacks in SDN-based IoT environments. Comput Model Eng Sci. 2025;145(3):4363–93. doi:10.32604/cmes.2025.074178.
13. Humayun M, Jhanjhi NZ, Alsayat A, Ponnusamy V. Internet of Things and ransomware: evolution, mitigation and prevention. Egypt Inform J. 2021;22(1):105–17. doi:10.1016/j.eij.2020.05.003.
14. Karmous N, Ben Dhiab Y, Ould-Elhassen Aoueileyine M, Youssef N, Bouallegue R, Yazidi A. Deep learning approaches for protecting IoT devices in smart homes from MitM attacks. Front Comput Sci. 2024;6:1477501. doi:10.3389/fcomp.2024.1477501.
15. Aoueileyine MO, Karmous N, Bouallegue R, Youssef N, Yazidi A. Detecting and mitigating MitM attack on IoT devices using SDN. In: Advanced information networking and applications. Cham, Switzerland: Springer Nature Switzerland; 2024. p. 320–30. doi:10.1007/978-3-031-57942-4_31.
16. Singh M, Pati D. Countermeasures to replay attacks: a review. IETE Tech Rev. 2020;37(6):599–614. doi:10.1080/02564602.2019.1684851.
17. Cetintav I, Tahir Sandikkaya M. A review of lightweight IoT authentication protocols from the perspective of security requirements, computation, communication, and hardware costs. IEEE Access. 2025;13(8):37703–23. doi:10.1109/access.2025.3546147.
18. Afzal S, Asim J. Systematic literature review over IDPS, classification and application in its different areas. Stat Comput Interdiscip Res. 2021;3(2):189–223. doi:10.52700/scir.v3i2.58.

19. Singh AK, Anand A, Lv Z, Ko H, Mohan A. A survey on healthcare data: a security perspective. ACM Trans Multimedia Comput Commun Appl. 2021;17(2s):1–26. doi:10.1145/3422816.
20. Cervantes J, Garcia-Lamont F, Rodríguez-Mazahua L, Lopez A. A comprehensive survey on support vector machine classification: applications, challenges and trends. Neurocomputing. 2020;408(7):189–215. doi:10.1016/j.neucom.2019.10.118.
21. Resende PAA, Drummond AC. A survey of random forest based methods for intrusion detection systems. ACM Comput Surv. 2019;51(3):1–36. doi:10.1145/3178582.
22. Younesi A, Ansari M, Fazli M, Ejlali A, Shafique M, Henkel J. A comprehensive survey of convolutions in deep learning: applications, challenges, and future trends. IEEE Access. 2024;12(2):41180–218. doi:10.1109/access.2024.3376441.
23. Dhaliwal SS, Nahid A-A, Abbas R. Effective intrusion detection system using XGBoost. Information. 2018;9(7):149. doi:10.3390/info9070149.
24. Anajemba JH, Iwendi C, Razzak I, Ansere JA, Okpalaoguchi IM. A counter-eavesdropping technique for optimized privacy of wireless industrial IoT communications. IEEE Trans Ind Inf. 2022;18(9):6445–54. doi:10.1109/tii.2021.3140109.
25. Ruano A, Silva S, Duarte H, Ferreira PM. Wireless sensors and IoT platform for intelligent HVAC control. Appl Sci. 2018;8(3):370. doi:10.3390/app8030370.
26. Garagad VG, Iyer NC, Wali HG. Data integrity: a security threat for Internet of Things and cyber-physical systems. In: Proceedings of the 2020 International Conference on Computational Performance Evaluation (ComPE); 2020 Jul 2–4; Shillong, India. doi:10.1109/compe49325.2020.9200170.
27. Salih Hasan BM, Abdulazeez AM. A review of principal component analysis algorithm for dimensionality reduction. J Soft Comput Data Min. 2021;2(1):20–30. doi:10.30880/jscdm.2021.02.01.003.
28. Berahmand K, Daneshfar F, Salehi ES, Li Y, Xu Y. Autoencoders and their applications in machine learning: a survey. Artif Intell Rev. 2024;57(2):28. doi:10.1007/s10462-023-10662-6.
29. Feng Y, Wang W, Weng Y, Zhang H. A replay-attack resistant authentication scheme for the Internet of Things. In: Proceedings of the 2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC); 2017 Jul 21–24; Guangzhou, China. doi:10.1109/cse-euc.2017.101.
30. Rughoobur P, Nagowah L. A lightweight replay attack detection framework for battery depended IoT devices designed for healthcare. In: Proceedings of the 2017 International Conference on Infocom Technologies and Unmanned Systems (Trends and Future Directions) (ICTUS); 2017 Dec 18–20; Dubai, United Arab Emirates. doi:10.1109/ictus.2017.8286118.
31. Farha F, Ning H. Enhanced timestamp scheme for mitigating replay attacks in secure ZigBee networks. In: Proceedings of the 2019 IEEE International Conference on Smart Internet of Things (SmartIoT); 2019 Aug 9–11; Tianjin, China. doi:10.1109/smartiot.2019.00085.
32. Elsaeidy AA, Jagannath N, Sanchis AG, Jamalipour A, Munasinghe KS. Replay attack detection in smart cities using deep learning. IEEE Access. 2020;8:137825–37. doi:10.1109/access.2020.3012411.
33. Wara MS, Yu Q. New replay attacks on ZigBee devices for Internet-of-things (IoT) applications. In: Proceedings of the 2020 IEEE International Conference on Embedded Software and Systems (ICESS); 2020 Dec 10–11; Shanghai, China. doi:10.1109/icess49830.2020.9301593.
34. Huan X, Miao K, Chen W, Jia P, Hu H. Kerra: an Internet of Things wireless key generation resistant to replay attacks. IEEE Internet Things J. 2024;11(17):29035–48. doi:10.1109/jiot.2024.3406702.
35. He R, Cheng Y, Zheng Z, Ji X, Xu W. Fast and lightweight voice replay attack detection via time-frequency spectrum difference. IEEE Internet Things J. 2024;11(18):29798–810. doi:10.1109/jiot.2024.3406962.
36. Lo WW, Layeghy S, Sarhan M, Gallagher M, Portmann M. E-GraphSAGE: a graph neural network based intrusion detection system for IoT. arXiv:2103.16329. 2021. doi 10.48550/arXiv.2103.16329.
37. Ajagbe SA, Awotunde JB, Florez H. Intrusion detection: a comparison study of machine learning models using unbalanced dataset. SN Comput Sci. 2024;5(8):1028. doi:10.1007/s42979-024-03369-0.

38. Benamor Z, Seghir ZA, Djezzar M, Hemam M. A comparative study of machine learning algorithms for intrusion detection in IoT networks. Rev d'Intell Artif. 2023;37(3):567–76. doi:10.18280/ria.370305.
39. Shi Y, Wang B, Yu Y, Tang X, Huang C, Dong J. Robust anomaly detection for multivariate time series through temporal GCNs and attention-based VAE. Knowl Based Syst. 2023;275(3):110725. doi:10.1016/j.knosys.2023.110725.
40. Henderson J, Mohammadshahi A, Coman A, Miculicich L. Transformers as graph-to-graph models. In: Proceedings of the Big Picture Workshop. Singapore: ACL; 2023. p. 93–107. doi:10.18653/v1/2023.bigpicture-1.8.