Enhancing SDN-enabled Network Access Control with Adversarial Robust IoT Device Identification

ARTICLE HISTORY
Compiled September 25, 2025

ABSTRACT
The rapid proliferation of Internet of Things (IoT) devices in Software-Defined Networking (SDN) environments presents significant security challenges, particularly at the Network Access Control (NAC) layer. Accurate monitoring and identification of IoT devices are essential for enforcing network security and detecting unauthorized access attempts that may bypass NAC mechanisms through impersonation attacks. This paper proposes a Machine Learning based approach to classify IoT devices using network flow data provided through SDN APIs to detect impersonated devices. Experimental results show high classification accuracy under benign conditions, demonstrating the effectiveness of our method in dynamic and heterogeneous environments. However, adversarial attacks that introduce subtle perturbations can significantly impair classification performance. To address this, we evaluate the impact of such attacks and propose robust countermeasures designed to detect adversarial instances, mitigate their effects, and restore accurate device identification. Our findings emphasize the combined importance of precise IoT device classification and resilient defense mechanisms, offering practical guidance for securing SDN-IoT ecosystems against evolving threats.

KEYWORDS
IoT-SDN, Software-Defined Networking, IoT Security, Network Access Control, Adversarial Attacks, Impersonation attack.

1. Introduction

The Internet of Things (IoT) has rapidly evolved into a transformative technology, reshaping how physical objects interact with the digital world. By equipping everyday devices with communication capabilities, IoT enables seamless interactions among people, processes, and objects-unlocking new possibilities for automation, efficiency, and intelligent decision-making. From smart homes to industrial automation, IoT is a cornerstone of modern innovation, fostering interconnected ecosystems across diverse sectors (Chaudhary, 2022; Wang et al., 2021a).

A key challenge in securing IoT networks is the accurate identification and authentication of devices demanding network access (Elaoudi et al., 2024). In traditional architectures, Network Access Control (NAC) is enforced at the network edge, typically on gateways or routers, using static device-specific rules to manage access. While effective in small-scale deployments, this approach lacks the flexibility and scalability needed for the growing variety and volume of IoT devices (Ragothaman et al., 2023).

Software-Defined Networking (SDN) offers a robust and scalable alternative by centralizing NAC at the SDN controller. This latter applies access policies based on real-time flow data, classifying devices and granting or denying access accordingly.

Although SDN-based NAC benefits from a global network view and programmable policy enforcement, it introduces new vulnerabilities, namely, the SDN controller becomes a single point of failure and a high-value target for attackers (Arevalo-Herrera et al., 2025). Furthermore, both edge-based and SDN-based NAC systems remain vulnerable to access control bypass techniques, particularly impersonation attacks, where malicious devices mimic legitimate ones to gain unauthorized access. These spoofing attacks can result in data breaches, privacy violations, and critical service disruptions (Yong et al., 2024).

To address this limitation, we propose an enhanced NAC architecture in which a Machine Learning (ML) or Deep Learning (DL) based Intrusion Detection System (IDS) is deployed directly within the NAC layer. This IDS monitors the behavior of authenticated devices using network flow data to perform post-access monitoring. If a device's behavior deviates from its expected profile, its access can be revoked. This approach enables the detection and mitigation of impersonation attacks that successfully bypass NAC, thereby strengthening the overall access control mechanism through layered defense.

However, this additional layer introduces new challenges. ML/DL-based IDS models are susceptible to adversarial attacks (Debicha et al., 2023), where subtle perturbations in network traffic can manipulate flow signatures and mislead classifiers into incorrect predictions. Attackers can exploit this vulnerability to evade detection or misclassify malicious devices as benign, compromising the IDS's reliability. To counteract this vulnerability, we investigate defense strategies that aim to enhance the robustness of IDS models against adversarial examples. Techniques such as **adversarial training** and **anomaly detection** based on flow pattern analysis are explored to maintain classification accuracy even in the presence of adversarial manipulation.

In summary, this paper addresses two critical security challenges in SDN-enabled IoT environments: (1) the inability of traditional NAC systems to detect device impersonation and (2) the vulnerability of ML/DL-based IDS models to adversarial manipulation. By integrating a layered NAC-IDS architecture and implementing robust defense mechanisms, our approach aims to ensure accurate device identification and enhance overall network resilience against evolving threats.

The rest of the paper is organized as follows. Section 2 presents background material, covering the IoT ecosystem, SDN, NAC, and adversarial attacks affecting device identification. Section 3 reviews related work on IoT device identification and security within SDN environments, emphasizing ML/DL-based approaches and their limitations. Section 4 describes the proposed methodology, including the device identification phase and adversarial robustness techniques. Section 5 details the experimental setup and results, including device identification performance, adversarial impact analysis, and robustness evaluation.

2. Background

NAC, SDN, and IoT together form a cohesive security framework for modern networks (Zangaraki et al., 2025). NAC verifies that only authorized and compliant IoT devices gain access by identifying, authenticating, and evaluating their security posture. SDN enhances this process by centralizing and dynamically managing network policies, enabling real-time enforcement of access rules and fine-grained segmentation of IoT devices according to security requirements (Sahana and Brahmananda, 2024). Given the inherent vulnerabilities of many IoT devices, SDN-based NAC can promptly

isolate or restrict any compromised device, thereby mitigating potential threats. In combination, NAC and SDN provide an adaptive, scalable foundation for securing increasingly complex IoT deployments.

2.1. Network Access Controller

NAC is a foundational security mechanism designed to regulate and enforce network access policies, ensuring that only authenticated, authorized, and policy-compliant devices can interact with network resources (Golightly et al., 2023). The NAC process is generally divided into three fundamental stages:

- Identification: The NAC system detects and collects relevant information about devices attempting to connect, which includes attributes such as MAC addresses, device type, and operating system fingerprinting.
- Authentication: Device or user identity is verified through mechanisms such as credentials, digital certificates, or authentication protocols (e.g., IEEE 802.1X, RADIUS).
- Posture Assessment: A crucial stage that evaluates the device's security state, checking for the presence of antivirus software, recent security patches, appropriate firewall configurations, and compliance with organizational security policies.

If a device fails any of these stages, NAC systems may either restrict its access, isolate it in a remediation VLAN, or completely block it from the network. This ensures that only secure and trusted devices operate within the network perimeter.

Beyond the initial admission control, modern NAC solutions provide continuous monitoring of device behavior throughout the session. This includes detecting abnormal activity, policy violations, or evidence of compromise after access has been granted. Advanced NAC implementations may integrate behavioral analytics, endpoint detection and response (EDR), and anomaly detection modules to proactively manage threats posed by compromised or malicious devices.

In SDN environments, NAC systems take on a more dynamic and centralized role. By integrating with the SDN controller, NAC policies can be applied and enforced network-wide in real-time, rather than at individual switches or access points. This **global visibility** and **programmable enforcement capability** are especially critical for IoT networks, where a large number of heterogeneous and potentially insecure devices attempt to connect. The SDN-based NAC can dynamically reconfigure network paths, segment traffic, and isolate devices based on real-time posture evaluation and flow data, offering a more scalable, automated, and adaptive security posture for next-generation networks.

2.2. SDN-powered Internet of Things

The IoT comprises a network of physical devices, such as smart home appliances, wearable sensors, and industrial actuators, embedded with sensors, software, and communication modules to collect and transmit data over the Internet (Laghari et al., 2021). Data gathered at the device (perception) layer is sent via various network technologies to gateways or edge servers, which may perform preprocessing before forwarding to cloud or edge platforms. In the cloud, analytics and ML models generate insights and control commands, enabling applications like predictive maintenance in industry, remote patient monitoring in healthcare, and automated energy management in smart

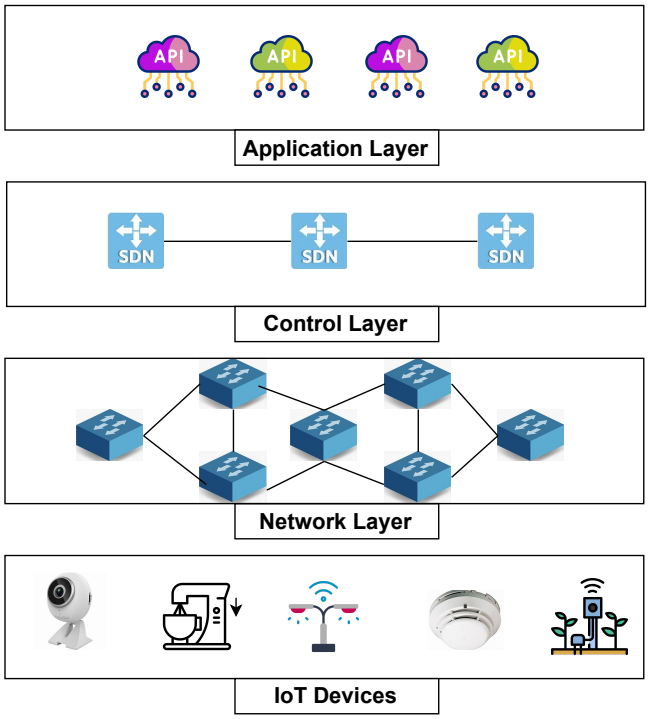

Figure 1. IoT-SDN Environments.

buildings.

As illustrated in Figure 1, sensors collect the necessary information, which is then transmitted to applications through technologies like Ethernet or WiFi, forming part of the network layer. These applications, typically hosted in the cloud, analyze the data to extract valuable insights and control various physical devices. IoT networks are deployed in businesses, institutions, and households.

IoT devices use various methods to connect to the network, often relying on traditional network infrastructure. However, the rapid growth in the number of IoT devices and the resulting increase in traffic have rendered traditional networks inefficient, hindering innovation and scalability.

Moreover, deploying IoT systems faces challenges in scalability, security, and resource constraints. The rapid growth and heterogeneity of IoT devices strain traditional network infrastructures, requiring dynamic, programmable solutions to manage diverse data flows efficiently. Limited onboard security in many IoT devices increases their vulnerability to unauthorized access and data breaches. Additionally, resource-constrained devices cannot run complex security or analytics algorithms locally. Consequently, approaches such as SDN and edge computing have become essential, providing centralized control, fine-grained flow management, and localized data processing to enhance performance, security, and reliability in large-scale IoT networks (Al-Shareeda et al., 2024).

3. Related works

The identification of IoT devices based on network traffic characteristics has become a central research focus, enabling improved access control, anomaly detection, and context-aware management in increasingly heterogeneous environments. Numerous studies have proposed identification techniques that leverage either packet-level metadata or flow-level statistical patterns, often in combination with supervised ML algorithms.

Early efforts in this domain focused on header-based and packet-derived features. As shown in Table 1, Chowdhury et al. (2023) extracted 218 features from packet headers and achieved 83.4% classification accuracy using classical classifiers. Jiang et al. (2024) advanced this further by fusing hardware identification with network features, enabling device classification with 99% accuracy in mobile IoT environments. Li et al. (2024b) proposed a line-rate device identification technique using programmable data planes in ISP-scale deployments, successfully classifying 77 devices without sacrificing performance. Maali et al. (2025) investigated the spatio-temporal drift of device fingerprints and demonstrated how feature stability varies over time and deployment contexts, revealing important insights into longitudinal robustness.

Moreover, Yedilkhan and Smakova (2024), Yao et al. (2023), Gu et al. (2025), and Pashamokhtari et al. (2023) proposed diverse approaches spanning feature engineering, hierarchical models, spoofing resistance, and cross-domain learning. For instance, Yedilkhan and Smakova (2024) benchmarked traditional ML classifiers on preprocessed flow data using dimensionality reduction and statistical filtering. Yao et al. (2023) introduced passive event-driven fingerprinting of device actions across Zigbee, Bluetooth, and Wi-Fi protocols, demonstrating significant gains in identifying control events. Gu et al. (2025) excluded easily spoofed attributes such as MAC and IP addresses and instead used robust flow-level statistics to improve spoofing resistance on datasets like UNSW and IoT Sentinel. Pashamokhtari et al. (2023) contributed with two key efforts: one exploring generalization across 12 smart homes under spatial/temporal variation and another releasing the public IPFIX Records dataset, which captures device activity via flow-level telemetry from 26 devices over 3 months.

Finally, Sivanathan et al. (2018) made similar contributions by collecting a 6-month dataset of IoT traffic traces covering 28 devices. They demonstrated the value of hierarchical classification combining Naive Bayes, Random Forest (RF), and behavioral features such as activity periodicity and service port entropy.

Despite these advancements, most existing solutions are tailored to offline analysis or constrained testbeds and do not consider real-time classification under dynamic SDN-based NAC architectures. Moreover, limited attention has been given to adversarial resilience, even though flow manipulation can critically degrade identification accuracy.

Adversarial Attacks & Defenses

Recent studies have revealed that IoT device identification systems are vulnerable to adversarial attacks, as shown in Table 2, where carefully crafted perturbations to network traffic can cause misclassification, enabling malicious devices to bypass network access controls. These attacks compromise both the security and reliability of behavioral device authentication.

Liu et al. (2018) presented the EPIC framework, which integrates encryption, pseudonym generation, and padding in smart home gateway traffic to thwart side-

Table 1. IoT device identification approaches.

Study	Focus	Technique & Features	Results & Limitations
Chowdhury et al. (2023)	Packet-header fingerprinting	218 header-based features; classical ML classifiers	83.4% accuracy; limited to packet-level features; weak cross-domain generalization
Jiang et al. (2024)	MIoT device fusion	Hardware and network fingerprint fusion	99% accuracy in mobile IoT lab; requires device-side hardware access
Li et al. (2024b)	ISP-scale device ID	Line-rate programmable switch; embeddings of key packets	Classified 77 devices at line speed; scalability concerns for switch deployment
Maali et al. (2025)	Temporal drift in device behavior	Spatio–temporal fingerprint stability analysis	Identified feature drift over time; lacks downstream classifier integration
Yedilkhan and Smakova (2024)	Hierarchical ML benchmarking	Dimensionality reduction pipeline	High precision/recall; offline processing only
Yao et al. (2023)	Event-driven fingerprinting	Protocol-aware passive event extraction; ML classifiers	Improved event identification; protocol-specific method with limited scope
Gu et al. (2025)	Spoof-resistant flow features	TTL, duration, volume; RF classification	High spoofing resilience; reporting focused on robustness over broad comparatives
Pashamokhtari et al. (2023)	Cross-home generalization	IPFIX flow telemetry; 28 features; ML classifiers	96% accuracy; no real-time SDN integration
Sivanathan et al. (2018)	Smart-environment profiling	Flow volume, periodicity, port usage; hierarchical Naive Bayes + RF	97% accuracy on 28 devices; dataset may not reflect NAT traffic variability

channel analysis attacks, effectively hiding device events and improving privacy. Similarly, Hafeez et al. (2019) introduced traffic morphing at the DNS level to camouflage IoT device behavior, sending synthetic traffic during idle periods to maintain constant throughput. Zhang et al. (2023) further developed SHTProtector, embedding realistic user-behavior traces into network flows to conceal identifiable patterns with minimal overhead.

Tramèr et al. (2017) pioneered Ensemble Adversarial Training (EAT), augmenting model robustness by blending adversarial examples crafted from multiple algorithms into the training set. This approach was later applied to environments such as IDS and IoT networks. Peng et al. (2020) proposed ASD, utilizing a generator–encoder–discriminator design to adapt classifiers on adversarial data distributions continually. Wang et al. (2021b) combined generative adversarial data augmentation with multi-source adversarial training (MAT) to bolster intrusion detection robustness. Pawlicki et al. (2020) demonstrated that augmenting training data with both clean and adversarial instances significantly improves detection accuracy under adversarial IDS scenarios.

Alhussien et al. (2024) examined semantic-preserving perturbations that subtly alter flow metadata (e.g., packet size, inter-arrival times) to evade identification while maintaining legitimate functionality. Li et al. (2024a) introduced BANADV, which exploits banner-layer fingerprinting to evade device scanners with over 80% success. Bao et al. (2021) demonstrated practical flow perturbation attacks on SDN-based NAC systems, showing how adversaries can manipulate flow features to impersonate benign devices.

Sánchez et al. (2024) applied adversarial training and knowledge distillation to counter hardware fingerprinting efforts, yielding improved robustness in device authentication. Sagduyu and Erpek (2023) tested FGSM attacks and mitigation on LoRaWAN device authentication, illustrating that even low-complexity adversarial attacks can compromise lightweight IoT systems. Venturi et al. (2024) demon-

Table 2. Adversarial attacks and defenses in IoT device identification.

Study	Attack / Defense	Target	Effectiveness / Limitation
Liu et al. (2018)	Traffic padding; pseudonyms	Smart home gateways	Protects against side-channel attacks; adds gateway overhead
Hafeez et al. (2019)	DNS-based traffic morphing	Device activity inference	Masks idle periods; ineffective against active timing analysis
Zhang et al. (2023)	Simulated user-behavior injection	Smart home traffic classifiers	Preserves behavioral realism; needs user-simulation logic
Tramèr et al. (2017)	Ensemble adversarial training (EAT)	General ML classifiers	Improves robustness; requires multiple adversarial samples
Peng et al. (2020)	Generator–Encoder Discriminator training	Adversarial detection	Adaptive defense; generator quality is critical
Wang et al. (2021b)	MGAN + multi-source training	Deep IDS models	Enhances resistance; relies on data augmentation
Pawlicki et al. (2020)	Adversarial + clean-data training	CICIDS-2017 IDS	Better resilience under attack; dataset-dependent
Alhussien et al. (2024)	Semantic-preserving perturbations	Flow-based classifiers	Evades detection; maintains functional semantics
Li et al. (2024a)	Banner-layer spoofing	Fingerprinting scanners	$>$ 80% spoofing success; depends on banner exposure
Bao et al. (2021)	Flow-level perturbation	SDN-based NAC systems	Practical evasion; exploits statistical fingerprinting
Sánchez et al. (2024)	Adv. training + distillation	Hardware fingerprinting	Increases authentication robustness
Sagduyu and Erpek (2023)	FGSM adversarial examples	LoRa authentication	Breaks lightweight classifiers; mitigated by retraining
Venturi et al. (2024)	Structural GNN attacks	GNN-based intrusion detection	Formalized threat model; GNNs vulnerable to topological shifts
Namvar et al. (2023)	Discretization + ensemble stacking	IoT classifier robustness	Defends against white-box and black-box attacks

strated structural adversarial attacks against GNN-based NIDS, revealing vulnerabilities posed by graph-based feature dependencies.

Namvar et al. (2023) combined discretization with ensemble stacking to reduce model sensitivity to adversarial noise in IoT device identification, reporting enhanced robustness against white-box and black-box attacks.

Tables 1 and 2 provide a summary of the core-related works in the field. However, several key limitations can be identified. First, there is a notable lack of real-time integration between SDN and NAC, despite the critical importance of such coordination for dynamic and adaptive network defense. Second, the generation and analysis of domain-constrained adversarial flows, those that preserve semantic validity within the network context, remain under-explored, limiting the robustness of current detection mechanisms. Third, most approaches fail to implement post-access validation, which is essential for detecting identity spoofing attacks that occur after initial NAC enforcement.

To address these gaps, our proposed framework introduces a combination of flow-level behavioral fingerprinting, semantic-aware adversarial defenses, and SDN-based real-time enforcement. This integrated approach enhances both the precision and responsiveness of network security mechanisms, making them better equipped to handle evolving threats in dynamic environments.

4. Methodology

In SDN-enabled IoT environments, ensuring that only legitimate devices gain access to the network is a critical security requirement. To this end, a NAC mechanism is deployed at the application layer of the SDN architecture to enforce access policies. The NAC system is responsible for inspecting devices attempting to connect to the network by identifying their declared type, such as cameras, lamps, or smartwatches, and authorizing or denying access based on a predefined list of approved device categories.

When an IoT device initiates a connection, the NAC utilizes metadata from the initial connection request, often analyzed through packet-level inspection, to classify the device and decide whether it should be granted network access. This decision is made before any extensive communication between the device and the network occurs, thereby serving as the first layer of defense. The SDN controller, which maintains a global view of the network, facilitates this process by forwarding relevant flow statistics to the NAC module in real-time.

However, traditional NAC systems face a significant vulnerability: they rely heavily on static or initial identity information provided by devices, which can be easily spoofed by malicious actors. An attacker can craft a device to mimic the behavior or identity signature of an authorized device during the initial access request, thereby bypassing the NAC's verification. Once such a device is granted access, it may engage in harmful activities such as eavesdropping, data exfiltration, or inference attacks aimed at uncovering user behavior and system operations.

4.1. Device monitoring

To enhance the security of SDN-enabled IoT environments beyond the initial access control phase, this work introduces a continuous monitoring strategy that verifies the identity of connected devices over time. While the NAC module initially grants access based on device-declared identities, this approach recognizes that attackers can exploit this mechanism through identity spoofing as legitimate devices to gain unauthorized access. To counteract this risk, a post-authentication verification phase is integrated into the system architecture.

Once a device is granted access to the network by the NAC, its network behavior is systematically monitored. Specifically, the SDN controller collects flow-level statistics from the data plane, including features such as packet transmission rates, flow duration, communication intervals, destination diversity, protocol usage, and other behavioral metrics. These flow features are collected over a predefined observation window to allow sufficient behavioral patterns to emerge, which are then aggregated into feature vectors suitable for ML analysis, as shown in Figure 2.

These behavioral profiles are subsequently processed by a dedicated ML-based classification module, referred to as the NAC Enhancer. This component is trained to identify the true device type based on observed network behavior. Unlike the NAC, which classifies devices at the moment of connection using static indicators, the NAC Enhancer analyzes dynamic and contextual flow features that evolve with the device's real usage.

Once classification is complete, the predicted device type from the NAC Enhancer is cross-validated against the identity that was declared during the initial NAC access request. A match confirms the legitimacy of the device, while a mismatch indicates a potential impersonation attack. For instance, a device initially claiming to be a smart

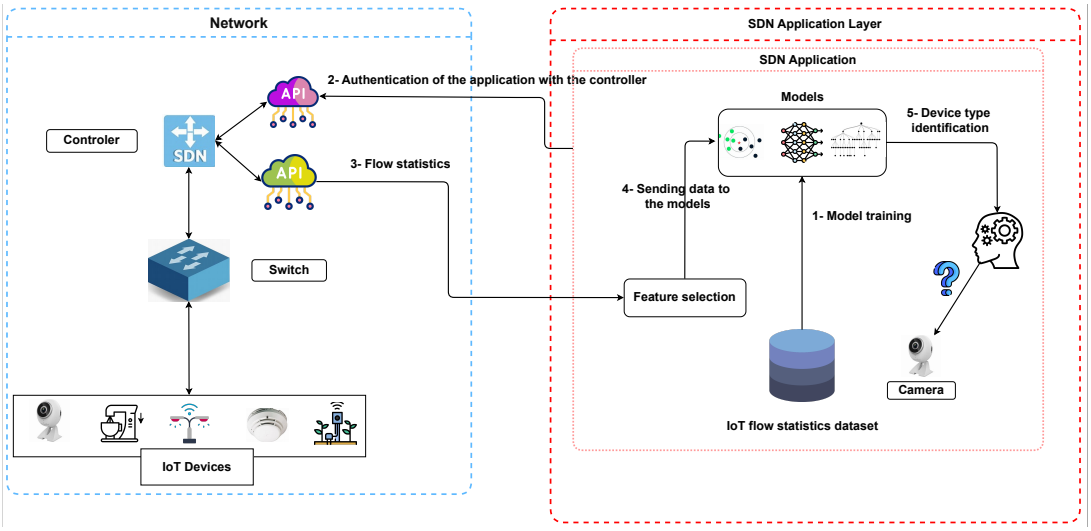

Figure 2. Device type identification in SDN-IoT environments.

thermostat but later exhibiting traffic characteristics typical of a surveillance camera will trigger an alert.

This continuous monitoring and re-identification framework establishes a second line of defense that supplements the static protections of traditional NAC systems. By leveraging flow analytics and behavioral classification, the system improves its ability to detect stealthy intrusions and impersonation-based threats. This not only mitigates unauthorized access but also strengthens the overall trustworthiness of the IoT environment in scenarios where large-scale deployments and device diversity make static access control insufficient.

To perform this behavioral re-identification, the proposed system relies on a set of ML algorithms that serve as the core of the NAC Enhancer. These algorithms play a fundamental role in analyzing the aggregated flow statistics to classify IoT devices based on their real-time behavior. To evaluate the feasibility and effectiveness of this approach, four widely used classification algorithms were employed: RF, k-Nearest Neighbors (KNN), XGBoost, and Deep Neural Networks (DNN).

Given that model performance and robustness are highly sensitive to the choice of hyperparameters, careful tuning was conducted to optimize each algorithm's accuracy and generalization. For instance, the number of neighbors in KNN, the number of hidden layers and neurons per layer in DNN, and the number of estimators in RF and XGBoost were systematically selected through empirical evaluation, as shown in Table 3.

Table 3. Used ML models' hyperparameter (Device identification)

Model	**Hyperparameters**
DNN	Hidden layers: 2; neurons per layer: 256; optimizer: Adam; activation: ReLU
RF	Estimators: 100; criterion: Gini
KNN	Neighbors: 3
XGBoost	Estimators: 100; eval. metric: `merror`

4.2. Impersonation attack

Using the information collected from the SDN controller, an ML/DL model can effectively monitor and detect the exact type of devices connected to the network. However, these models are inherently vulnerable to *adversarial attacks*, where carefully crafted perturbations are applied to input features to mislead the classifier, resulting in misidentification. Such attacks can allow malicious devices to impersonate legitimate ones and bypass post-authentication verification mechanisms such as the NAC Enhancer.

To assess the vulnerability of our system and to simulate realistic attack scenarios, we implemented an adversarial evasion algorithm designed to generate adversarial instances from genuine traffic flows, as shown in Figure 3. The goal is to modify a flow's statistical profile just enough to alter the model's prediction without violating domain constraints or triggering anomaly detection. The attack methodology is iterative and targeted to minimize both the number of modified features (L_0 distance) and the extent of modification (L_2 distance).

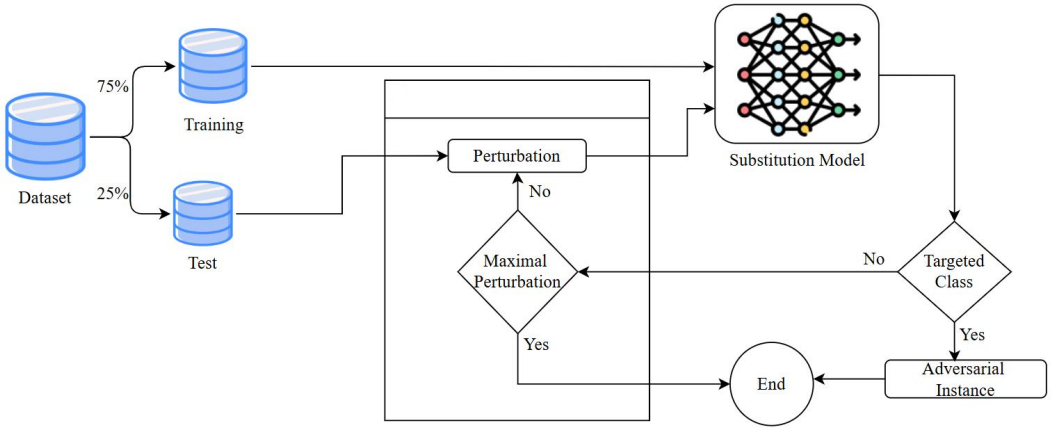

Figure 3. Workflow of generating adversarial instances

The generation of adversarial examples is governed by the following formulation:

$$x_{adv} = \text{Projection}\left[x_0 + c \cdot t \cdot \text{mask} \cdot \text{sign}(\mu_{target} - x_0) \cdot |\text{Difference}(\mu_{target}, x_0)|\right]$$

where x_0 is the original feature vector, x_{adv} the adversarial version, μ_{target} the mean feature vector of the target class, and t the current iteration. The binary mask selects the features to be perturbed, while the projection function ensures that modified values stay within valid semantic and syntactic bounds.

To remain realistic and valid for SDN application usage, feature perturbations must preserve traffic integrity. Therefore, features were categorized as: (i) *independent* (directly modifiable), (ii) *dependent* (must change consistently with others), and (iii) *non-modifiable* (e.g., protocol type). Perturbations were applied only to the first two categories, and necessary semantic corrections were enforced post-modification (e.g., rounding integers, preserving valid ranges).

To further reduce the impact of perturbations and maintain stealthiness, we adopted two key strategies:

- **Mask-based L_0 minimization:** Binary masks were used to restrict the number of altered features. Out of 2^{11} possible masks, only the most frequent and

impactful ones were retained based on empirical trials over test data.

- **Target class diversity for L_2 minimization:** Instead of attacking a single target class, we identified the three closest classes (based on K-means clustering of class centroids) and selected the optimal one per instance using Euclidean distance.

The overall attack pipeline includes training a substitute model at the middleware level above the SDN controller. During evaluation, this model determines whether a generated sample has successfully deceived the system. The perturbation process begins by choosing the nearest target class and iteratively increasing the perturbation until the adversarial instance is misclassified as a member of the target class or the perturbation budget is exhausted.

This methodology demonstrates the high feasibility of adversarial manipulation in IoT-SDN environments, where even minimal modifications can significantly degrade classification accuracy. These adversarial examples bypass post-NAC behavioral validation, highlighting the urgent need for robust defense mechanisms in real-time traffic classification systems.

4.3. Adversarial Robustness

To mitigate the impersonation attack and improve the resilience of IoT device identification in SDN environments, we propose a two-tiered defense architecture integrated into the NAC Enhancer. As shown in Figure 4, this architecture combines both **adversarial detection** and **adversarial training** to proactively identify and neutralize manipulated traffic flow instances.

The first layer of defense is a *pre-classification adversarial detector* (Model-1 in Figure 4) designed to identify whether a given network flow sample has been adversarially perturbed before it is passed to the IoT classifier. This binary classifier is trained using both legitimate flow data and adversarial instances generated through controlled perturbations. As shown in Table 4, the adversarial detector uses different

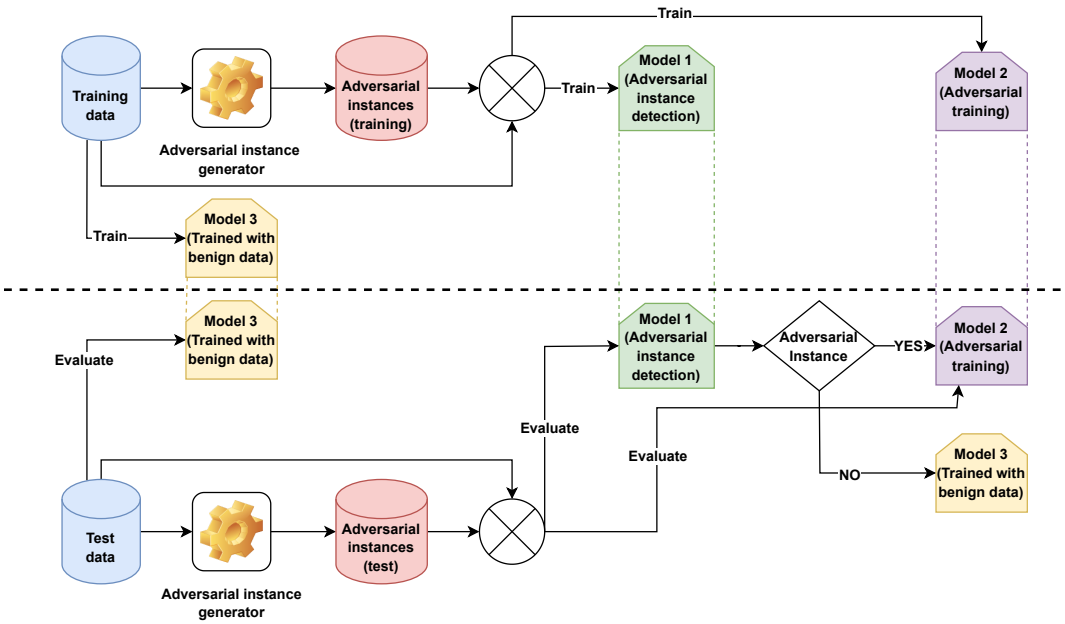

Figure 4. Proposed countermeasure process.

ML models, including RF, KNN, XGBoost, and DNN, to ensure generalization across attack vectors. Each input to this detector includes statistical and temporal features (e.g., entropy, variance, outlier scores), and the output is a binary decision: normal or adversarial. Once a flow is flagged as adversarial, the system enforces the policy by **rejecting and isolating** the associated device to prevent malicious behavior.

Table 4. Used ML models' hyperparameter (Adversarial instance detection)

Model	Hyperparameters
DNN	loss: `sparse_categorical_crossentropy`; hidden activation: ReLU; output activation: Softmax; neurons: 256; layers: 3; optimizer: Adam
RF	min samples split: 5; max features: `sqrt`; max depth: `None`; estimators: 300
KNN	neighbors: 5; weights: `uniform`; algorithm: brute-force
XGBoost	max depth: 3; estimators: 300; learning rate: 0.2

The second layer of defense is *adversarial training* (Model-2 in Figure 4), which aims to enhance the classifier's robustness even when adversarial examples bypass detection. This method involves training the classifier with a mixture of clean and adversarial instances, where adversarial examples are labeled with their true device class. This enlarges the model's decision boundaries and increases tolerance to small perturbations. Although this may slightly reduce the model's accuracy on benign data, it substantially improves resilience to adversarial manipulation.

Defense Pipeline:

(1) **Adversarial data synthesis:** Starting from the clean training set, we generate adversarial counterparts.
(2) **Clean baseline fit:** Train a reference (clean-only) device identifier to quantify non-adversarial performance.
(3) **Joint corpus construction:** Form a combined training set by merging benign and synthesized adversarial samples with preserved class balance.
(4) **Two-stage modeling:**
   (a) *Stage 1—Adversarial detector (binary):* learn to discriminate benign vs. adversarial instances.
   (b) *Stage 2—Device identifier (multi-class):* learn device classes using the joint corpus (adversarial training).
(5) **Hold-out evaluation:** Assess both models on a disjoint test set containing unseen benign and adversarial instances to avoid leakage and bias.

In practice, an incoming flow first passes through the adversarial detector (Model-1 in Figure 4, using the hyperparameters of Table 4); if flagged "adverserial", the system invokes the adversarially trained identifier (Model-2 in Figure 4, using the hyperparameters of Table 5) under a conservative policy; otherwise, the identifier operates normally (Model-3 in Figure 4, using the hyperparameters of Table 6). This two-stage design isolates attack detection from device classification while leveraging adversarial training to harden the classifier against residual perturbations.

This layered approach significantly enhances the security posture of SDN-based IoT networks by introducing proactive filtering and robust classification mechanisms. It complements the behavioral validation step already embedded in the NAC Enhancer and helps maintain classification reliability even in the presence of sophisticated adversarial attacks.

Table 5. Used ML models' hyperparameter (Adversarial training)

Model	Hyperparameters
DNN	activation: ReLU; neurons per layer: 256; layers: 3
RF	bootstrap: `True`; max depth: `None`; max features: `sqrt`; estimators: 200
KNN	algorithm: `auto`; neighbors: 3; weights: `distance`
XGBoost	learning rate: 0.2; max depth: 7; estimators: `None`

Table 6. Used ML models' hyperparameter (Benign data)

Model	Hyperparameters
DNN	activation: ReLU; neurons per layer: 256; layers: 3
RF	bootstrap: `True`; max depth: `None`; estimators: 300
KNN	weights: `distance`; neighbors: 5; algorithm: `auto`
XGBoost	learning rate: 0.1; max depth: `None`; estimators: 200

5. Results

This section presents the experimental evaluation of the proposed approach, beginning with the description of the datasets and the data preprocessing techniques applied. Two flow-level datasets were used and adapted to reflect the constraints of SDN environments by selecting only features accessible through SDN controllers. The data was then cleaned, normalized, and standardized to ensure reliable and consistent input for the machine-learning models used in this work.

5.1. Data Processing

Data preprocessing is crucial in transforming raw data into a format suitable for ML models. This process includes normalization, standardization, and data cleaning to improve model performance. In this work, we use two datasets: IPFIX Records and IoT IPFIX Home.

(1) **IoT IPFIX Home:** This dataset was collected from 12 households in Japan, comprising 24 types of IoT devices, including two cameras, four power switches, two humidifiers, one air sensor, two speakers, two multimedia streaming devices, three hubs, two bulbs, one scale, one tablet, one printer, one sleep sensor, a smart remote, and a vacuum cleaner. These data were collected over 47 days (Pashamokhtari et al., 2023).

(2) **IPFIX Records:** This dataset, collected by Pashamokhtari et al. (Pashamokhtari et al., 2022) over 03 months from January to April 2020, from a residential testbed with 26 IoT devices, contains over nine million records. The data was characterized using 28 relevant flow attributes. The data collection involved acquiring PCAP files, which were then converted into binary files using the YAF (Yet Another Flowmeter) tool. Subsequently, the Super Mediator tool was used to transform these binary files into JSON format.

Each dataset is partitioned into 75% for training and 25% for testing. Due to class imbalance, adjustments were made to remove underrepresented classes to avoid any adverse effects on the results.

Table 7. Device names in IoT IPFIX Home vs. IPFIX Records.

No.	IoT IPFIX Home	IPFIX Records
1	Eclear	Qrio Hub
2	Sleep	Philips Hue Light Bulb
3	Esensor	Planex Pan–Tilt Camera 1
4	Hub Plus	JVC Kenwood Camera
5	Humidifier	Planex Pan–Tilt Camera 2
6	Home Unit	Google Home
7	Ink Jet Printer	Apple HomePod
8	Smart Wi-Fi Plug Mini	Sony Bravia TV
9	Smart Power Strip	Wansview Camera
10	Echo Dot	Qwatch Camera
11	Fire 7 Tablet	Fredi Camera
12	Google Nest Mini	Planex Outdoor Camera
13	Google Chromecast	Powerlec Wi-Fi Plug
14	Atom Cam	LINE Clova Speaker
15	Kasa Camera Pro	Sony Smart Speaker
16	Kasa Smart LED Lamp	Amazon Echo
17	Fire TV Stick 4K	Amazon Echo Show
18	Qrio Hub	–

In our study on the feasibility of identifying IoT device types using SDN network flow statistics, the lack of a public dataset specific to SDN led us to use a traditional network dataset. We restricted the features to those provided by SDN controllers, which reduced model performance. To optimize results, we decided to exclude certain device classes from the datasets that yielded unsatisfactory classification outcomes, reducing the total number of classes to 18 in IoT IPFIX Home and 17 in IPFIX Records, as shown in Table 7. The removed classes were those with a limited number of instances, which directly impacted classification performance.

In the following, we explain the main steps of the dataset preprocessing.

(1) **Normalization** Data is rescaled to a range of [0, 1] using the formula:

$$X_{\text{norm}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

where X_{norm} is the normalized value, $x_{\min}$ and $x_{\max}$ are the minimum and maximum values of the dataset, respectively.

(2) **Standardization** Data is transformed to have a mean of 0 and a standard deviation of 1 using the formula:

$$X_{\text{standard}} = \frac{X - \mu}{\sigma}$$

where X_{standard} is the standardized value, μ is the mean, and σ is the standard deviation of the dataset.

(3) **Duplicate Removal** Duplicate entries were identified and removed using Python's Pandas library. This step ensures data integrity by avoiding redundant records that could bias the analysis.

(4) **Missing Value Handling** Instances with missing values were excluded to maintain data quality and ensure the accuracy of the results. Removing missing values helps to prevent misleading outcomes that could arise from incomplete data.

Adapting Datasets for SDN Networks

In the context of our study, access to datasets containing flow characteristics specific to Software-Defined Networks (SDNs) is essential. However, these datasets are often derived from traditional networks and do not directly align with our requirements. To address this gap, we undertook the task of adapting these datasets by removing certain features and retaining only those pertinent to SDN networks.

This adaptation necessitates a careful selection of data to ensure alignment with the information available through SDN APIs. To achieve this, we systematically selected the relevant features, eliminating those such as source and destination IP addresses, as well as source and destination ports, among others. Our focus was on preserving only the key features provided by SDN APIs, specifically the average inter-arrival time, the average size of incoming and outgoing packets, and the IP protocol.

5.2. Device Identification

The evaluation was conducted using the IoT IPFIX Home dataset by four ML models: KNN, RF, DNN, and XGBoost.

In Table 8, the weighted average values of precision, recall, and F1-score are reported for each classification model across the two datasets used in this study. It can be observed that higher accuracy and better generalization are achieved by ensemble models such as RF and XGBoost in both datasets. In contrast, greater variability is exhibited by DNN and KNN, particularly on devices characterized by sparse or ambiguous traffic patterns.

Table 8. Weighted average performance metrics for IoT device identification.

	IoT IPFIX Home			**IPFIX Records**		
	Precision	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**
KNN	83.2%	83.1%	83.0%	97.4%	97.4%	97.4%
RF	87.0%	86.9%	86.9%	97.7%	97.7%	97.7%
DNN	83.2%	81.5%	82.1%	94.6%	93.8%	93.9%
XGBoost	88.3%	88.1%	88.1%	97.7%	97.7%	97.7%

Among the classification models evaluated on the IoT IPFIX Home dataset (Figure 5), the XGBoost algorithm consistently exhibited the highest performance. It achieved an average F1-score of 88.1% across all device classes and surpassed 93% for several high-traffic devices such as the Google Nest Mini, Fire TV Stick, and Kasa Camera Pro.

This strong performance highlights XGBoost's effectiveness in capturing non-linear feature interactions and handling class variability, even under conditions of device diversity and flow heterogeneity. In multiple instances, both precision and recall exceeded 95%, confirming not only accurate predictions but also a high degree of stability and reliability.

The RF classifier followed closely, with an average F1-score of 87.9%. It performed particularly well on devices characterized by consistent and high-volume communication patterns, such as multimedia streamers and voice-controlled assistants. Its ensemble-based structure allowed it to maintain balanced precision-recall tradeoffs across most device types. This can be attributed to its higher sensitivity to data imbalance and noise, which may lead to overfitting in certain classes.

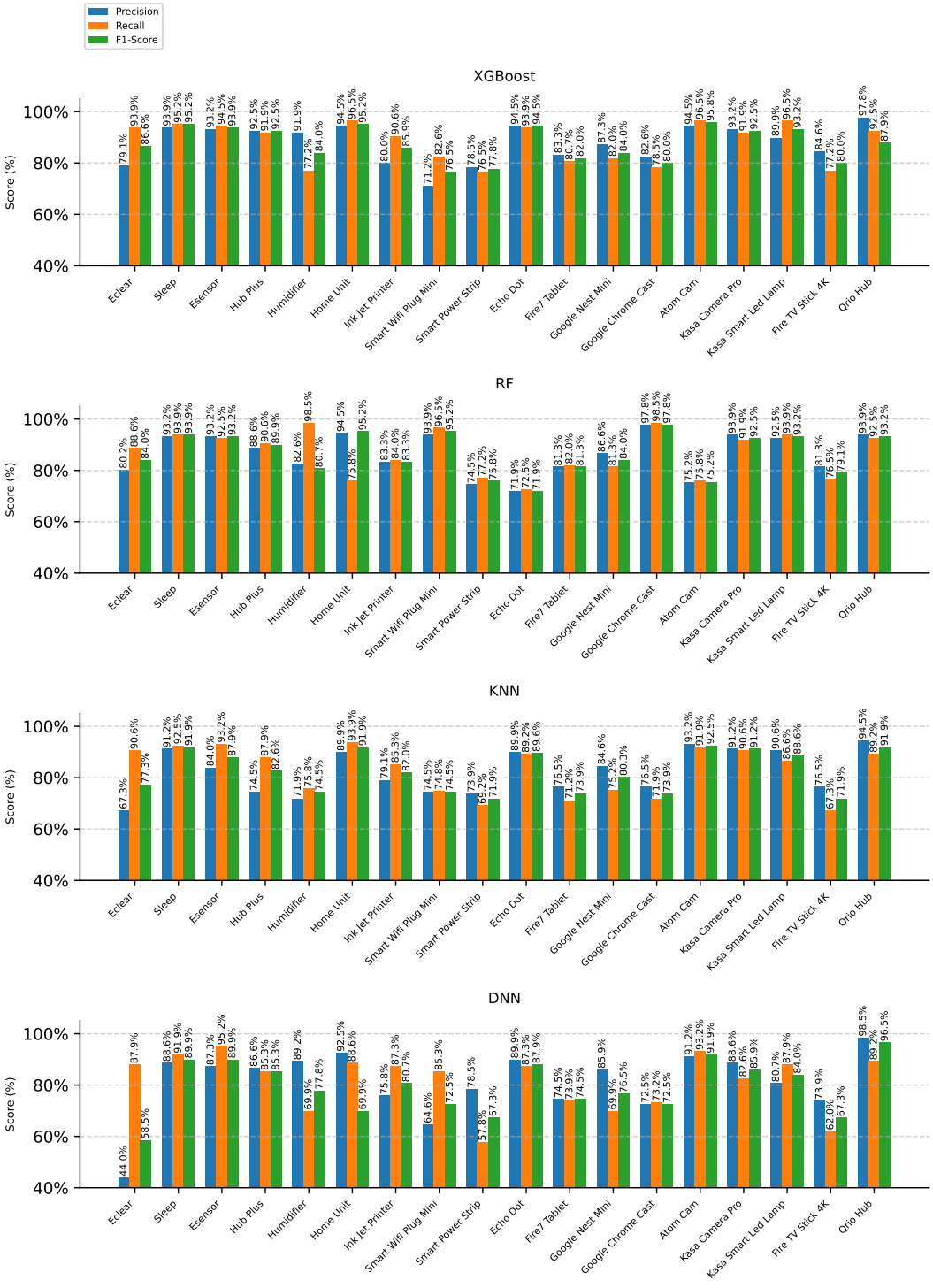

Figure 5. IoT Device Identification Scores in IoT IPFIX Home

The KNN model offered a moderate performance level, reaching an average F1-score of 82.6%. It was effective in classifying devices with distinct traffic signatures but struggled with those exhibiting overlapping statistical characteristics. Devices such as smart plugs and humidifiers, which generate sparse or periodic traffic, were more likely to be misclassified due to the model's reliance on local distance-based decision boundaries, which are susceptible to noise and feature similarity.

The DNN model recorded the lowest overall performance among the tested models, with an average F1-score of 82.1%. While competitive in well-represented and high-volume device classes, it showed substantial weaknesses in low-traffic or intermittently active devices. For instance, the Eclear health monitoring device was poorly identified, with an F1-score falling to 59%. These results suggest that deep learning models may require larger training sets, better regularization, or domain-specific feature engineering to achieve robust generalization in IoT environments where traffic is often sparse and highly variable.

To further validate the robustness of the proposed approach, the four ML models KNN, RF, DNN, and XGBoost-were also trained and evaluated on the IPFIX Records dataset.

When evaluated on the IPFIX Records dataset (Figure 6), the RF classifier emerged as the top-performing model, achieving an impressive average F1-score of 97.57%. It demonstrated near-perfect accuracy in identifying a wide range of devices, including high-traffic categories such as IP cameras and smart assistants. Several devices, such as the Wansview camera, Qwatch camera, and Amazon Echo Show, were classified with F1-scores exceeding 99%, indicating both strong generalization and consistent predictive stability. This performance highlights RF's ability to handle heterogeneous flow features while maintaining robustness against overfitting.

XGBoost followed closely, with an average F1-score of 97.1%. It demonstrated highly competitive results, occasionally outperforming RF on devices with structured and repetitive traffic behaviors, such as smart TVs and voice-controlled speakers. However, slight drops in recall were observed on certain device classes, notably some camera variants and light bulbs, which led to minor declines in its overall ranking. These variations suggest a higher sensitivity of XGBoost to subtle feature overlaps within specific traffic classes.

The KNN model also performed well, with an average F1-score of 96.6%. It maintained stable classification across most devices but exhibited less resilience in those with ambiguous or overlapping flow patterns. For example, surveillance cameras and environmental sensors showed decreased identification scores, reflecting KNN's sensitivity to proximity-based misclassifications in high-dimensional spaces.

The DNN model achieved an average F1-score of 93.4%, which, while lower than its ensemble counterparts, remains respectable considering the diversity and complexity of the dataset. The DNN showed competitive performance on high-volume devices but struggled on classes with sparse or irregular activity. Devices such as the Philips Hue bulb and certain smart speakers yielded F1-scores below 85%, highlighting the challenges of learning generalized representations from imbalanced or weakly discriminative traffic patterns.

Across both datasets, devices characterized by high bandwidth and continuous communication, such as IP cameras, voice assistants, hubs, and streaming systems, were consistently identified with high accuracy. In contrast, devices with intermittent or low-volume traffic, such as smart plugs, bulbs, and sleep sensors, proved more challenging to classify due to their limited and less distinctive flow features. Ensemble learning models, particularly RF and XGBoost, demonstrated superior performance

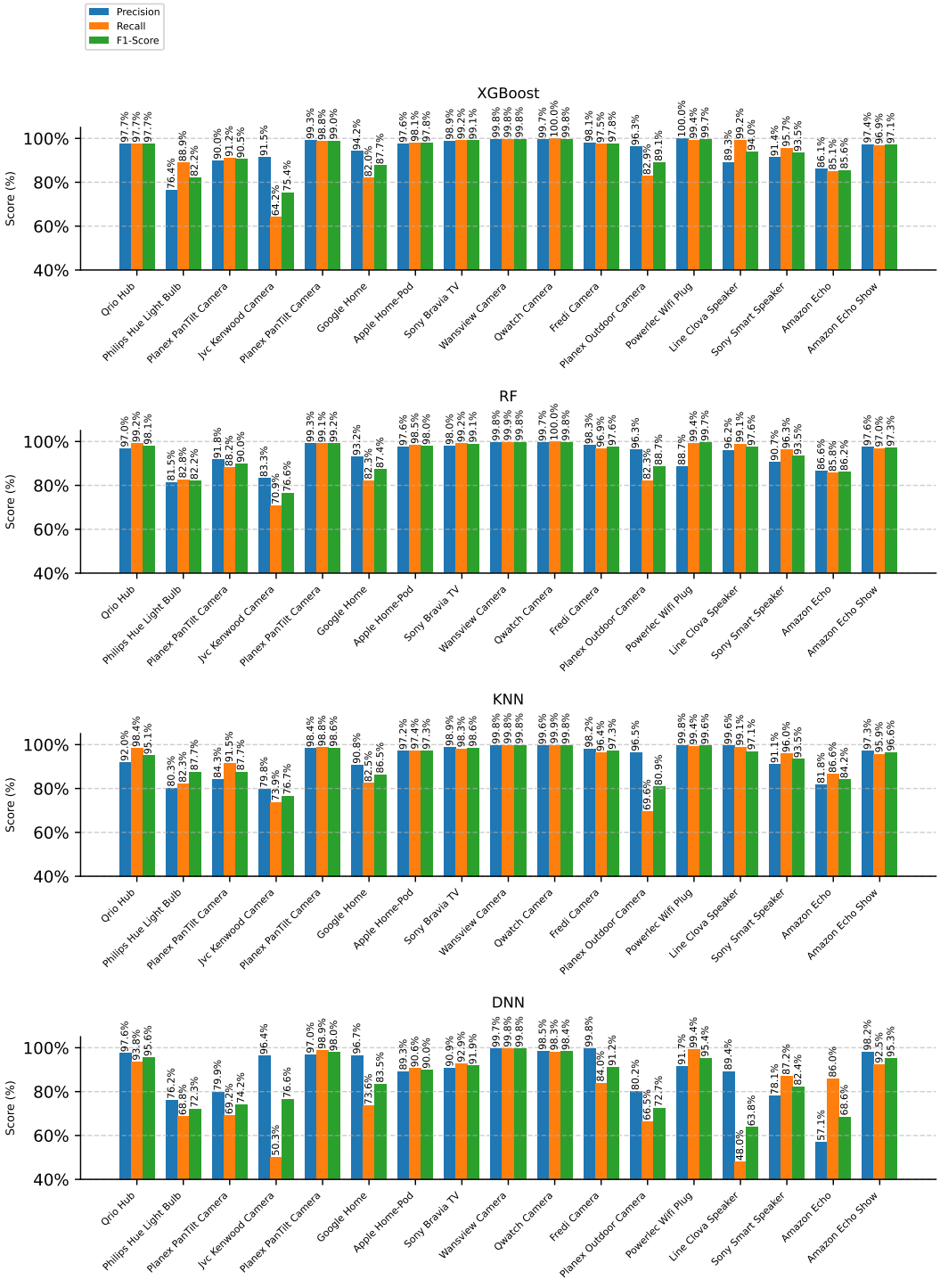

Figure 6. IoT Device Identification Scores in IPFIX Records

and robustness across device types, confirming their effectiveness in handling heterogeneous traffic patterns in SDN-enabled IoT environments. Conversely, deep learning models showed limitations when applied to sparse or noisy traffic without additional feature engineering or data augmentation. These results validate the generalizability of the proposed identification framework and highlight its suitability for real-world network access control applications.

5.3. Impact of Adversarial Perturbations on Device Identification

To evaluate the vulnerability of IoT device identification systems to adversarial manipulation, adversarial evasion attacks were applied to both the IoT IPFIX Home and IPFIX Records datasets. Figures 7 and 8 present a comparative analysis of precision, recall, and F1-score before and after adversarial perturbations for the four classification models.

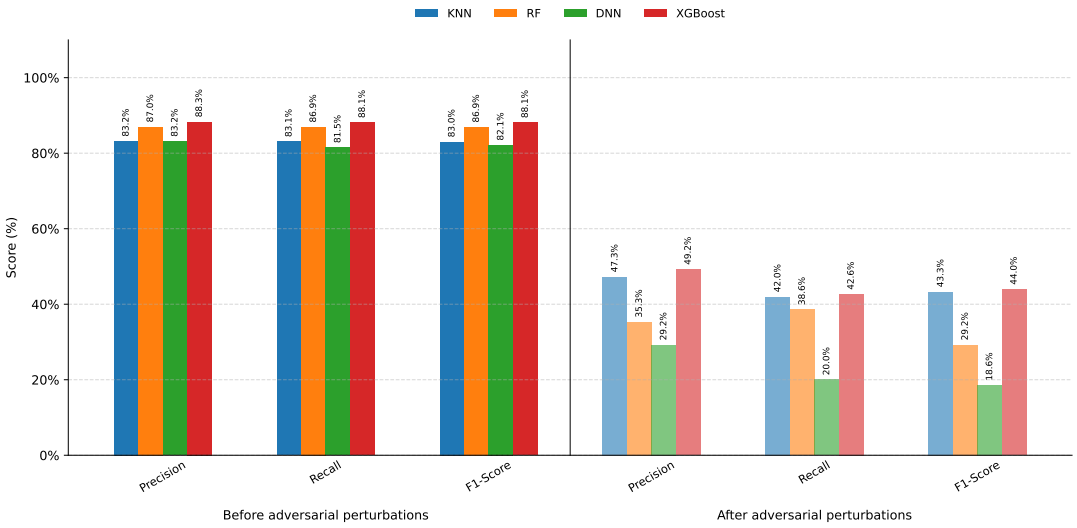

Figure 7. Adversarial effect on IoT IPFIX Home identification device

The results clearly demonstrate that all models experienced significant performance degradation under adversarial conditions. On the IoT IPFIX Home dataset (Figure 7), average F1-scores dropped by more than 40 percentage points for most models. For instance, the DNN model saw its F1-score decrease from 82.1% to just 43.3%, while XGBoost dropped from 88.1% to 44.0%. Even RF, which previously showed strong performance on clean data, was highly affected, with F1-scores falling below 50%.

A similar trend was observed on the IPFIX Records dataset (Figure 8), the impact was severe. The RF model, for example, dropped from an F1-score of 97.6% to 27.5%, while the DNN fell to 28.6%. XGBoost, which initially achieved 97.7%, saw its performance decline to just under 20%. These results confirm the high sensitivity of all models to carefully crafted adversarial inputs, regardless of their baseline accuracy on benign data.

From a broader perspective, the findings underscore the critical need for robust defense mechanisms in SDN-based IoT device identification pipelines. The severity of the accuracy drop reveals that adversarial instances can completely undermine classification reliability, allowing malicious devices to evade behavioral detection and

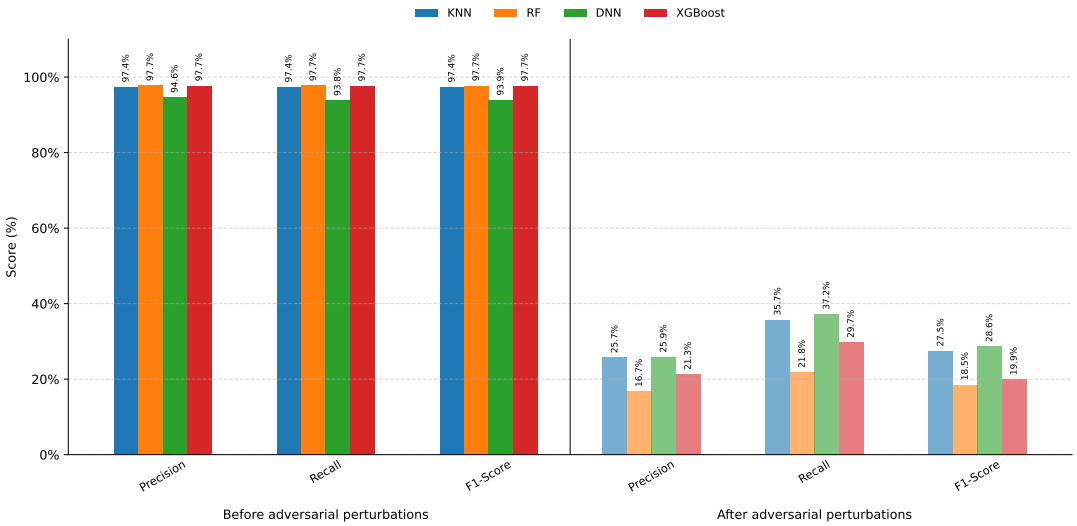

Figure 8. Adversarial effect on IPFIX Records identification devices

potentially gain unauthorized access to the network. Therefore, integrating adversarial training, detection models, or flow validation mechanisms becomes essential to ensure secure and resilient network access control in practical deployments.

5.4. Adversarial Robustness

To counteract the significant performance degradation observed under adversarial conditions, we implemented a two-layer defense strategy to enhance the robustness of IoT device identification in SDN-enabled environments. The first layer is an **adversarial instance detector**, designed to distinguish between clean and adversarial network flows before classification. This binary classifier leverages statistical and temporal features of the input and aims to filter or block manipulated inputs before device identification. The second layer is **adversarial training**, in which the classification models are trained on a combination of benign and adversarial instances. This approach enhances the resilience of the classifier by enlarging decision boundaries and improving tolerance to small perturbations. Together, these mechanisms aim to proactively defend the classification system and restore reliable device identification in the presence of adversarial traffic.

Adversarial Instance Detection

Figure 9 presents the performance of four ML models RF, XGBoost, DNN, and KNN in detecting adversarial instances for the IoT IPFIX Home and IPFIX Records datasets. All models achieved high precision and recall, with DNN and RF showing the best results overall.

On the IoT IPFIX Home dataset, DNN reached an F1-score of approximately 91%, while RF and XGBoost both exceeded 89%, confirming their effectiveness at identifying perturbed flows. KNN performed slightly lower but still maintained an F1-score above 86%. Similar trends were observed on the IPFIX Records dataset, where DNN and RF maintained performance above 90%, demonstrating generalization across datasets.

These findings show that the adversarial detector successfully distinguishes between genuine and manipulated flow data in a wide range of scenarios. This component acts as an effective pre-classification filter, enabling early mitigation of adversarial threats. However, its success depends on the availability of representative adversarial examples during training and may be limited against novel attack patterns or adaptive adversaries.

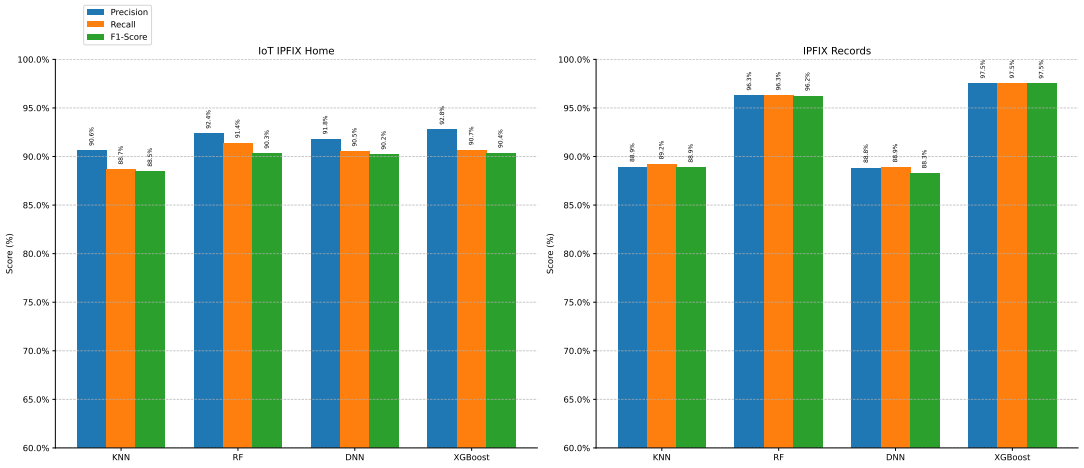

Figure 9. Adversarial Instance Detection

Adversarial Training

Figures 10 and 11 illustrate the effect of adversarial training on the identification accuracy of IoT devices under attack. Four models-XGBoost, RF, KNN, and DNN-were retrained using a mixture of benign and adversarial instances, with the goal of improving their robustness.

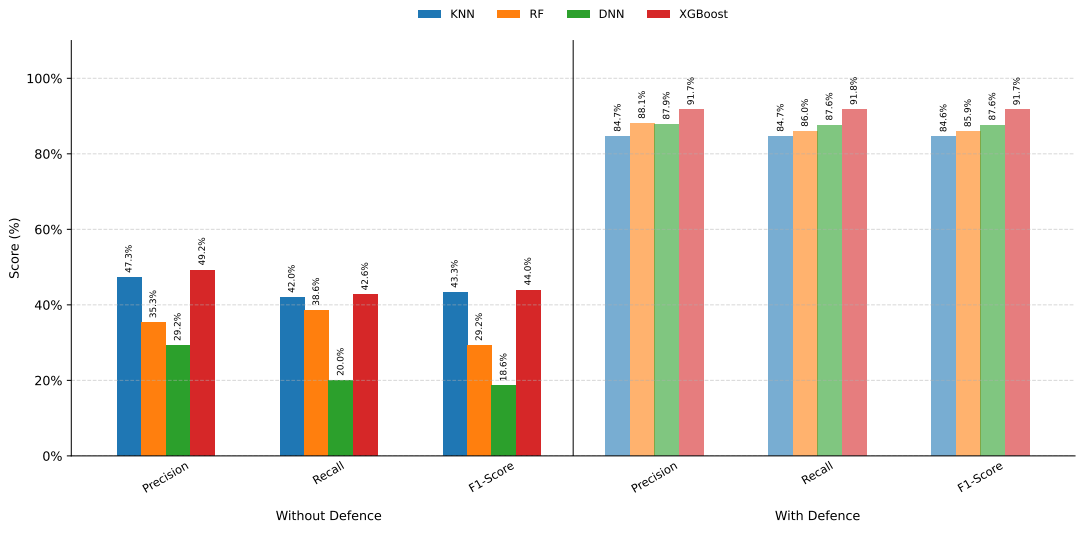

Figure 10. Performance Evaluation of Device Identification in IoT IPFIX Home With Robustness Measures

On the IoT IPFIX Home dataset, the results show a significant recovery in F1-score after applying adversarial training. For example, the F1-score of the XGBoost model increased from 44.0% under attack to over 80% after adversarial training. RF also exhibited similar gains, regaining up to 85% of his original performance on clean data. DNN, despite its lower baseline, improved from 43.3% to over 78%, confirming the benefit of including adversarial examples in the training phase.

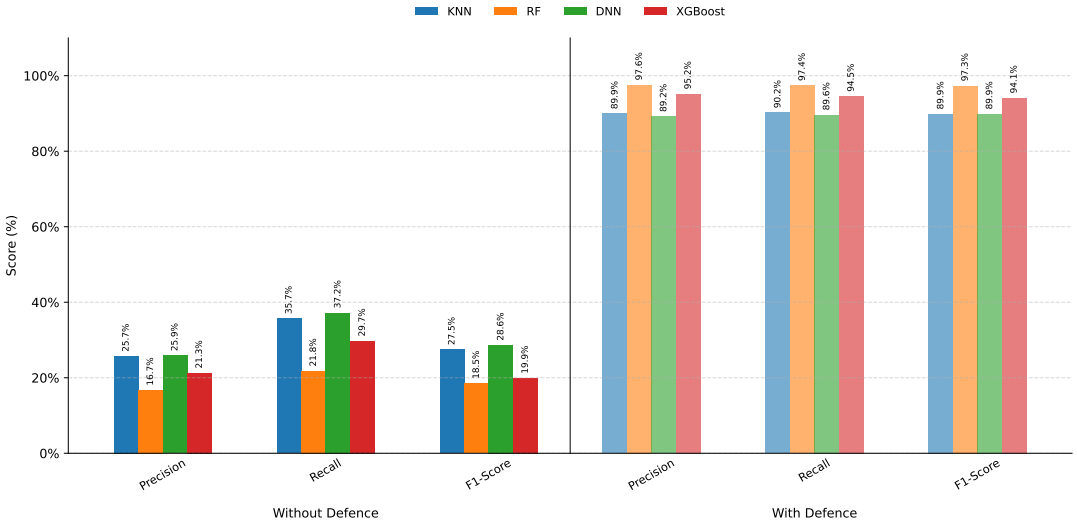

Figure 11. Performance Evaluation of Device Identification in IPFIX Records with Robustness Measures

The same improvement pattern is observed for the IPFIX Records dataset. XGBoost, which dropped to 19.9% under attack, recovered to nearly 87% with adversarial training. RF achieved a similar recovery rate, while DNN climbed from 28.6% to 84.0%.

These results confirm that adversarial training significantly enhances the robustness of device identification models. By exposing the model to crafted perturbations during training, it learns to generalize better and becomes more tolerant to real-world adversarial scenarios. Combined with the adversarial detector, this two-layer defense strategy forms a practical and effective solution for securing IoT access control in SDN environments.

6. Conclusion

This work investigated the feasibility of identifying IoT devices in home environments using only network flow statistics accessible via SDN controllers. By integrating the device classification process into the NAC framework, we demonstrated that ML models particularly ensemble methods, such as RF and XGBoost can achieve high accuracy in distinguishing diverse IoT devices based solely on flow-level metadata. Our experiments, conducted on two real-world datasets (IoT IPFIX Home and IPFIX Records), validated the generalizability and scalability of this approach, especially for devices with consistent traffic patterns such as hubs, speakers, and streaming platforms.

However, our findings also revealed a critical vulnerability: ML models are highly susceptible to adversarial evasion attacks, where subtle perturbations introduced into network flows can significantly degrade classification performance. These attacks pose

a tangible threat, allowing malicious devices to mimic legitimate ones and evade post-authentication behavioral checks. Our empirical results showed that F1-scores could drop by more than 70 percentage points under adversarial conditions, even for the best-performing models.

To mitigate this risk, we proposed a two-layer defense strategy consisting of (1) a dedicated adversarial instance detector that flags suspicious traffic prior to classification, and (2) adversarial training to enhance model robustness during learning. Both mechanisms proved effective in restoring classification accuracy under attack, with ensemble models recovering up to 90% of their original performance. This demonstrates that a layered defense approach can significantly reinforce the resilience of NAC systems against adversarial threats.

In conclusion, while flow-based IoT device identification in SDN environments offers a promising path for scalable and automated network access control, its deployment in real-world scenarios must account for adversarial risks. The proposed countermeasures provide a practical foundation for achieving both accurate and robust device identification, thereby enhancing the overall security posture of SDN-IoT ecosystems.

While the proposed framework demonstrates strong potential for robust IoT device identification in SDN environments, several avenues remain open for future investigation. First, the current approach could be extended to incorporate temporal and contextual flow features using sequence-based models such as LSTMs or Transformers, which may improve detection for low-traffic or behaviorally similar devices. Second, evaluating the system in real-time testbed environments or on dynamic enterprise networks would offer valuable insights into deployment scalability and latency overhead. Additionally, future research should explore adaptive adversarial strategies, where attackers evolve perturbation patterns over time, requiring the development of continuous learning and self-adaptive defense mechanisms. Finally, integrating flow-based identification with hardware or physical-layer fingerprints could lead to more comprehensive multi-modal device authentication, enhancing the resilience of NAC systems against sophisticated impersonation attacks.

References

Mahmood A. Al-Shareeda, Abeer Abdullah Alsadhan, Hamzah H. Qasim, and Selvakumar Manickam. Software defined networking for internet of things: Review, techniques, challenges, and future directions. *Bulletin of Electrical Engineering and Informatics*, 13(1): 638–647, 2024.

Nour Alhussien, Ahmed Aleroud, Abdullah Melhem, and Samer Y. Khamaiseh. Constraining adversarial attacks on network intrusion detection systems: Transferability and defense analysis. *IEEE Transactions on Network and Service Management*, 21(3):2751–2772, 2024.

Juliana Arevalo-Herrera, Jorge Camargo Mendoza, Jose Ignacio Martínez Torre, Tatiana Zona-Ortiz, and Juan M. Ramirez. Assessing SDN controller vulnerabilities: A survey on attack typologies, detection mechanisms, controller selection, and dataset application in machine learning. *Wireless Personal Communications*, pages 1–37, 2025.

Zhida Bao, Yun Lin, Sicheng Zhang, Zixin Li, and Shiwen Mao. Threat of adversarial attacks on DL-based IoT device identification. *IEEE Internet of Things Journal*, 9(11):9012–9024, 2021.

Anit Chaudhary. Internet of things (IoT): Research challenges and future applications. *International Journal of Emerging Trends in Science and Technology*, 2022.

Rajarshi Roy Chowdhury, Azam Che Idris, and Pg Emeroylariffion Abas. Internet of things: Digital footprints carry a device identity. In *AIP Conference Proceedings*, volume 2643. AIP

Publishing, 2023.

Islam Debicha, Benjamin Cochez, Tayeb Kenaza, Thibault Debatty, Jean-Michel Dricot, and Wim Mees. Adv-Bot: Realistic adversarial botnet attacks against network intrusion detection systems. *Computers & Security*, 129:103176, 2023.

Sanâ Elaoudi, Marouane Sebgui, and Slimane Bah. Survey on IoT device authentication protocols from hash-based schemes to blockchain-based schemes. In *International Conference on Internet of Everything*, pages 12–29. Springer, 2024.

Lewis Golightly, Paolo Modesti, Rémi Garcia, and Victor Chang. Securing distributed systems: A survey on access control techniques for cloud, blockchain, IoT and SDN. *Cyber Security and Applications*, 1:100015, 2023.

Dinglin Gu, Jian Zhang, Zhangguo Tang, Qizhen Li, Min Zhu, Hao Yan, and Huanzhou Li. IoT device identification based on network traffic. *Wireless Networks*, 31(2):1645–1661, 2025.

Ibbad Hafeez, Markku Antikainen, and Sasu Tarkoma. Protecting IoT-environments against traffic analysis attacks with traffic morphing. In *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 196–201. IEEE, 2019.

Yu Jiang, Yufei Dou, and Aiqun Hu. Identification of IoT devices based on hardware and software fingerprint features. *Symmetry*, 16(7):846, 2024.

Asif Ali Laghari, Kaishan Wu, Rashid Ali Laghari, Mureed Ali, and Abdullah Ayub Khan. A review and state of the art of internet of things (IoT). *Archives of Computational Methods in Engineering*, pages 1–19, 2021.

Haocong Li, Yaxin Zhang, Long Cheng, Wenjia Niu, Haining Wang, and Qiang Li. Obfuscating IoT device scanning activity via adversarial example generation. *arXiv*, 2024a.

Ruoyu Li, Qing Li, Tao Lin, Qingsong Zou, Dan Zhao, Yucheng Huang, Gareth Tyson, Guorui Xie, and Yong Jiang. DeviceRadar: Online IoT device fingerprinting in ISPs using programmable switches. *IEEE/ACM Transactions on Networking*, 2024b.

Jianqing Liu, Chi Zhang, and Yuguang Fang. EPIC: A differential privacy framework to defend smart homes against internet traffic analysis. *IEEE Internet of Things Journal*, 5 (2):1206–1217, 2018.

E. Maali, O. Alrawi, and J. McCann. Evaluating machine learning-based IoT device identification models for security applications. In *Network and Distributed System Security (NDSS) Symposium*, 2025.

Anahita Namvar, Chandra Thapa, and Salil S. Kanhere. Discretization-based ensemble model for robust learning in IoT. In *International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services*, pages 353–367. Springer, 2023.

Arman Pashamokhtari, Norihiro Okui, Yutaka Miyake, Masataka Nakahara, and Hassan Habibi Gharakheili. Combining stochastic and deterministic modeling of IPFIX records to infer connected IoT devices in residential ISP networks. *IEEE Internet of Things Journal*, 10(6):5128–5145, 2022.

Arman Pashamokhtari, Norihiro Okui, Masataka Nakahara, Ayumu Kubota, Gustavo Batista, and Hassan Habibi Gharakheili. Dynamic inference from IoT traffic flows under concept drifts in residential ISP networks. *IEEE Internet of Things Journal*, 10(17):15761–15773, 2023.

Marek Pawlicki, Michał Choraś, and Rafał Kozik. Defending network intrusion detection systems against adversarial evasion attacks. *Future Generation Computer Systems*, 110: 148–154, 2020.

Ye Peng, Guobin Fu, Yingguang Luo, Jia Hu, Bin Li, and Qifei Yan. Detecting adversarial examples for network intrusion detection system with GAN. In *2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS)*, pages 6–10. IEEE, 2020.

Kaushik Ragothaman, Yong Wang, Bhaskar Rimal, and Mark Lawrence. Access control for IoT: A survey of existing research, dynamic policies and future directions. *Sensors*, 23(4): 1805, 2023.

Yalin E. Sagduyu and Tugba Erpek. Adversarial attacks on LoRa device identification and

rogue signal detection with deep learning. In *MILCOM 2023 – IEEE Military Communications Conference (MILCOM)*, pages 385–390. IEEE, 2023.

D. S. Sahana and S. H. Brahmananda. Authentication-centric and access-controlled architecture for edge-empowered SDN-IoT networks. *Journal of The Institution of Engineers (India): Series B*, 105(6):1497–1509, 2024.

Pedro Miguel Sánchez, Alberto Huertas Celdrán, Gérôme Bovet, and Gregorio Martínez Pérez. Adversarial attacks and defenses on ML- and hardware-based IoT device fingerprinting and identification. *Future Generation Computer Systems*, 152:30–42, 2024.

Arunan Sivanathan, Hassan Habibi Gharakheili, Franco Loi, Adam Radford, Chamith Wijenayake, Arun Vishwanath, and Vijay Sivaraman. Classifying IoT devices in smart environments using network traffic characteristics. *IEEE Transactions on Mobile Computing*, 18 (8):1745–1759, 2018.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv*, 2017.

Andrea Venturi, Dario Stabili, and Mirco Marchetti. Problem space structural adversarial attacks for network intrusion detection systems based on graph neural networks. *arXiv*, 2024.

Jianxin Wang, Ming K. Lim, Chao Wang, and Ming-Lang Tseng. The evolution of the internet of things (IoT) over the past 20 years. *Computers & Industrial Engineering*, 155:107174, 2021a.

Jianyu Wang, Jianli Pan, Ismail AlQerm, and Yuanni Liu. Def-IDS: An ensemble defense mechanism against adversarial attacks for deep learning-based network intrusion detection. In *2021 International Conference on Computer Communications and Networks (ICCCN)*, pages 1–9. IEEE, 2021b.

Yunhao Yao, Jiahui Hou, Sijia Zhang, Zhengyuan Xu, and Xiang-Yang Li. Traffic processing and fingerprint generation for smart home device event. In *2022 IEEE 28th International Conference on Parallel and Distributed Systems (ICPADS)*, pages 9–16. IEEE, 2023.

Didar Yedilkhan and Saida Smakova. Machine learning approaches for smart home device recognition from network traffic. *Procedia Computer Science*, 231:709–714, 2024.

Hongliang Yong, Le Yu, Tian Dong, Yan Meng, Guoxing Chen, and Haojin Zhu. DevDet: Detecting IoT device impersonation attacks via traffic-based identification. In *International Conference on Wireless Artificial Intelligent Computing Systems and Applications*, pages 439–451. Springer, 2024.

Shahrbanoo Zangaraki, Meghdad Mirabi, Seyed Hossein Erfani, and Amir Sahafi. SecShield: An IoT access control framework with edge caching using software defined network. *Peer-to-Peer Networking and Applications*, 18(1):1–17, 2025.

Shuo Zhang, Fangyu Shen, Yaping Liu, Zhikai Yang, and Xinyu Lv. A novel traffic obfuscation technology for smart home. *Electronics*, 12(16):3477, 2023.