Enhancing Intrusion Detection in IoT Systems through Simulated Attack Scenarios

Marwa Neily, Farah Jemili *, and Ouajdi Korbaa

University of Sousse, ISITCom, MARS Research Laboratory, LR17ES05,4011, Hammam Sousse, Tunisia
E-mail: neily.marwa@gmail.com (M.N.); jmili_farah@yahoo.fr (F.J.); ouajdi.korba@mars.rnu.tn (O.K.)
*Corresponding author

***Abstract*—The Internet of Things (IoT) landscape is fraught with vulnerabilities, making it a prime target for various types of attacks. While existing literature has extensively explored IoT attacks through studies and simulations, this paper introduces a fresh perspective by proposing a new methodology for testing attacks in IoT environments. Focusing on six prominent attack vectors, we conduct comprehensive tests using both the Cooja and OMNET++ simulators. Our research delves into the underlying factors driving these attacks, analyzing data based on the attackers' chosen target addresses. Through our novel approach, we aim to deepen the understanding of IoT vulnerabilities and provide insights into the behavior of attackers, ultimately paving the way for more effective defense mechanisms in IoT ecosystems. Our experiments demonstrate high detection accuracy, with Random Forest achieving up to 99.86% accuracy in Cooja and 98.66% in OMNET++. Despite these results, the study acknowledges potential limitations in real-world generalization due to the simulated nature of the dataset.**

***Keywords*—attacks, Internet of Things (IoT), Intrusion Detection Systems (IDS), Cooja, OMNET++, security, simulations**

I. Introduction

The Internet of Things (IoT) represents a dynamic network of interconnected devices, objects, and machinery, promising to revolutionize various facets of modern life and work. This interconnectedness, however, introduces significant security challenges that must be addressed to fully realize the potential benefits of IoT technology. The proliferation of IoT devices has expanded the potential attack surface, rendering them susceptible to various cyber threats [1].

One of the primary security concerns surrounding IoT devices is their inherent vulnerability stemming from limited processing power and memory. These constraints often result in the implementation of weak security protocols, making IoT devices attractive targets for malicious actors. Compounding this issue is the lack of regular security updates, leaving devices vulnerable to exploitation [2]. Addressing these security challenges necessitates the implementation of robust security measures to safeguard against cyber-attacks. These measures may include the deployment of encryption mechanisms to protect data transmission [3], the implementation of user authentication protocols to prevent unauthorized access [4], the integration of biometric authentication for enhanced security [5], and the establishment of regular security update protocols to patch vulnerabilities [6]. Furthermore, fostering consumer awareness of the security risks associated with IoT devices is crucial, prompting the adoption of proactive measures to protect personal data and ensure device security [7].

In this paper, we delve into a comprehensive exploration of IoT security, focusing specifically on the detection and mitigation of cyber-attacks targeting IoT infrastructure. We present novel research aimed at identifying and addressing the vulnerabilities inherent in IoT systems. By conducting experiments in both the Cooja and OMNET++ simulators, we aim to simulate and analyze the behavior of six prominent IoT attacks. Through rigorous analysis of the data gathered from these experiments, we strive to gain insights into the key factors contributing to these attacks, with a particular emphasis on the target addresses chosen by attackers.

While previous studies have focused on pre-existing datasets or generic simulation environments, our work addresses the gap in attack-specific IoT simulations by generating tailored datasets in Cooja and OMNET++. This approach enables a more precise evaluation of intrusion detection mechanisms. Our objective is to analyze six major IoT attack vectors using machine learning models to improve detection accuracy and adaptability in real-world scenarios.

Unlike previous studies that rely on pre-existing datasets or single simulation environments, our work introduces a comprehensive testbed using both the Cooja and OMNET++ simulators. This dual-simulator approach ensures a more detailed evaluation of various attack scenarios specific to IoT networks. Furthermore, we generate custom datasets rather than relying on publicly available ones, allowing for tailored analysis of attack patterns. Our study also integrates diverse machine learning models, including Random Forest and Artificial Neural Networks, to enhance detection accuracy, achieving up to 99.86% accuracy in Cooja and 98.66% in OMNET++. These contributions make our approach

Manuscript received January 21, 2025; revised March 20, 2025; accepted April 15, 2025; published August 8, 2025.

distinct by providing a more adaptable and effective intrusion detection system for IoT environments.

This paper is structured as follows: Section I provides an introduction to the study, elucidating the key concepts and outlining the research objectives. Section II discusses related work, encompassing previous research relevant to the present study. Section III outlines the contributions of our research, highlighting its originality and scientific significance. Section IV presents the experimental methodology and results of our study. Section V delves into the discussion of the paper, elucidating the implications of the findings and their significance for the field. Finally, Section VI offers a conclusion summarizing our proposed work and its implications for future research and practical applications in IoT security.

II. Related Work

The advent of the Internet of Things (IoT) has ushered in a new era of connectivity and convenience, enabling the monitoring and control of various environmental aspects through interconnected devices. However, the exponential growth in data generated by IoT devices has also brought forth significant security challenges. In response to these challenges, Intrusion Detection Systems (IDS) have emerged as crucial tools for safeguarding IoT networks against malicious attacks. By continuously monitoring network traffic and analyzing behavioral patterns, IDS can effectively detect and prevent security breaches, thereby enhancing system security and performance.

To contextualize our research within the broader landscape of IoT security, we surveyed relevant literature and identified several seminal works that have contributed to the advancement of intrusion detection techniques in IoT networks. Each of these studies offers unique insights into detecting and mitigating various types of attacks, employing diverse methodologies and datasets.

Gómez *et al*. [8] leveraged a generated dataset to defend against Distributed Denial of Service (DDoS) attacks, employing a range of machine learning algorithms including Random Forest, Decision Trees, Support Vector Machines (SVM), and boosting algorithms. Despite the absence of a specific simulator, the study achieved a commendable performance ratio above 90%.

Building upon this foundation, Singh *et al*. [9] utilized the NS-3 simulator to simulate a wide array of attacks such as Denial of Service (DoS), DDoS, blackhole, wormhole, and Sybil. By training Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) models, the research achieved a notable performance ratio of 96.9%.

In another notable contribution, Zeng *et al*. [10] combined real-world traffic with simulation in MATLAB using the NGSIM dataset to detect malicious nodes. Employing Artificial Neural Networks (ANN), the research achieved an impressive performance ratio of 99%, highlighting the efficacy of integrating real-world data with simulation environments.

Recent research continues to refine IDS models for IoT security. In Ref. [11], a hybrid intrusion detection system based on CNN and Gated Recurrent Units (GRU) significantly improved anomaly detection in IoT networks. Similarly, Kikissagbe and Adda [12] conducted a comprehensive review of machine learning techniques in IDS, emphasizing the efficiency of ensemble methods. Amouri *et al.* [13] introduced Kolmogorov-Arnold Networks combined with XGBoost, achieving enhanced classification accuracy. Furthermore, Shen *et al.* [14] proposed a federated learning-based IDS framework for heterogeneous IoT networks, addressing data privacy concerns.

Similarly, Li *et al.* [15] utilized the GloMoSim simulator to identify malicious nodes using SVM, achieving a performance ratio above 95%. Grover *et al.* [16], on the other hand, employed the NCTUns-5.0 simulator to detect malicious nodes, utilizing Naive Bayes, J48, and Random Forest algorithms to achieve a performance ratio of 97%.

In a multi-simulator approach, Alhaidari and Alrehan [17] combined OMNET++, SUMO, INET, and Veins simulators to defend against DDoS attacks, employing a variety of machine learning algorithms including SVM, J48, Random Forest, K-Nearest Neighbors (KNN), ANN, and Naive Bayes. This comprehensive approach yielded a remarkable performance ratio above 99%.

Moreover, So *et al.* [18] utilized the VeReMi simulator in conjunction with VEINS to detect location spoofing attacks, achieving a performance ratio of 98.56% through the application of KNN and SVM algorithms. Meanwhile, Gao *et al.* [19] utilized the Kyoto dataset to detect malicious intrusions, employing a Multilayer Perceptron model to achieve a performance ratio of 99%. In Ref. [20], a simulation framework written in MATLAB was developed to analyze Radio jamming DoS attacks, employing a heuristic approach that yielded a performance ratio of 95%.

To address resource constraints in IoT security, Jouhari and Guizani [21] developed a lightweight CNN-BiLSTM model tailored for intrusion detection on low-power devices, demonstrating high accuracy with minimal computational overhead. Similarly, Rahman *et al.* [22] provided an updated survey on intrusion detection in IoT, highlighting new datasets and attack patterns. Kantharaju *et al.* [23] introduced an advanced IDS using Self-Attention Progressive Generative Adversarial Networks (SAPGAN), improving robustness against adversarial attacks.

Recent advancements in IoT intrusion detection have leveraged Machine Learning (ML) and Deep Learning (DL) techniques to enhance security measures. For instance, an ensemble learning approach was proposed for attack-aware IoT network traffic routing, demonstrating improved detection accuracy [24]. Another study introduced a novel IDS that efficiently detects anomalous network traffic using ML algorithms, achieving high detection rates [25].

Despite these advancements, challenges such as high false-positive rates, limited generalizability across diverse IoT environments, and computational resource constraints persist. Our work addresses these limitations by developing a dual-simulator testbed using Cooja and OMNET++,

enabling the generation of custom attack datasets tailored to specific IoT scenarios. This approach facilitates the training of ML models on data that closely represents real-world IoT environments, thereby enhancing detection accuracy and reducing false positives.

While existing literature has made significant strides in intrusion detection for IoT networks, our research stands out due to its methodological advancements and novel contributions. Unlike previous works that primarily relied on pre-existing datasets or generic simulation environments, our approach introduces a comprehensive testbed for IoT-specific attack simulations using both the Cooja and OMNET++ simulators. By focusing on six prominent attack types, we aim to provide a more targeted and nuanced security analysis.

Furthermore, our study leverages a diverse array of machine learning models, ranging from traditional classifiers such as Random Forest, Decision Trees, and SVM to advanced deep learning models like CNN and LSTM. The inclusion of recent methodologies, such as the knowledge distillation-based IDS from [26], further strengthens the robustness of our model. By integrating real-world traffic datasets, we bridge the gap between theoretical IDS implementations and practical IoT security challenges.

In summary, our research represents a significant departure from conventional IDS frameworks by introducing a multi-simulator approach, an extensive dataset collection strategy, and a hybrid machine learning-based detection system. These contributions pave the way for more effective and adaptable security mechanisms in IoT ecosystems.

III. Contribution

A. Proposed IDS for Device in IoT

With the revolution of the IoT, securing these networks has become a critical issue. IDS can help to identify the malicious activities within IoT networks. However, collecting real-world data for IDS training and evaluation can be difficult caused by the limited disponibility of IoT devices and the potential ethical concerns surrounding the collection of real-world data [27].

To overcome these limitations, researchers often construct datasets based on simulated IoT networks. Simulations can generate a vast amount of diverse and controlled data, allowing for the creation of large-scale datasets that can help train and evaluate IDS systems. Simulation based datasets also have the advantage of being easily reproducible, which can improve the comparability of results across different research studies. To construct a dataset based on simulation, the first step is to select a suitable simulator platform [28]. There are several popular simulators for IoT, we choose Cooja and OMNET++. Once a simulator is selected, researchers can design and configure IoT scenarios that mimic real-world network topologies and attack scenarios. The simulation platform can then be used to generate network traffic, including both legitimate and malicious traffic, which can be captured and used to construct a dataset for IDS training and evaluation. Constructing datasets based on simulations is a valuable technique for evaluating IDS performance in IoT networks. The use of simulations allows to generate large and diverse datasets with controlled and reproducible conditions, enabling the development of effective IDS systems to secure IoT networks.

Unlike prior works that primarily utilize pre-existing datasets or a single-simulator setup, our methodology improves IoT security by introducing a dual-simulator testbed using Cooja and OMNET++. This approach enables the generation of attack-specific datasets, allowing for more accurate intrusion detection model training. By leveraging multiple machine learning algorithms, including Random Forest and ANN, our model achieves superior detection accuracy, reaching 99.86% in Cooja and 98.66% in OMNET++. This ensures a more resilient and adaptable IDS capable of detecting diverse IoT attack vectors with higher precision.

B. Simulation

Simulations can be useful in the IoT for many reasons:

- Cost-effectiveness: Building and testing IoT systems in the real world can be expensive, especially when considering the cost of hardware, software, and maintenance. Simulation can reduce these costs by allowing developers to test IoT systems in a virtual environment before deploying them in the real world [29].
- Risk reduction: Testing IoT systems in a virtual environment reduces the risks associated with deploying a faulty system in the real world. Simulation allows developers to identify and fix potential problems before the system is deployed, reducing the risk of costly errors and downtime [30].
- Scalability testing: allowing the evaluation of the system performs under different conditions and workloads. This can help to optimize the system for peak performance and ensure that it can handle increased traffic or usage [31].
- Realistic testing: Simulation can create realistic scenarios that are difficult or impossible to replicate in the authentic world. For example, it can simulate extreme weather conditions or test the system's response to unusual user behavior [32].
- Time efficiency: saving time by testing and iterating on the system quickly without the need to physically deploy and test hardware. This can help accelerate the development process and bring the system to market faster [30].

The uses of the simulators Cooja and OMNET++ in our study is a good choice for several reasons:

- Both simulators have the ability to simulate attacks in IoT environments. They make it possible to simulate and test various attack scenarios, assess the effects of attacks on IoT systems, and examine the efficiency of security solutions.
- Realistic Environment: They enable the creation and simulation of IoT network topologies, the emulation of different IoT devices, and the replication of the behavior of actual IoT systems.

- Protocol Support: They support a wide variety of IoT specific communication protocols, including Constrained Application Protocol (CoAP), Message Queuing Telemetry Transport (MQTT), and IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN). This makes the ability to examine the vulnerabilities of particular protocols by simulating attacks that target them.
- Visualization and Analysis: They offer analysis and visualization tools to aid in comprehending how attacks behave in IoT systems. They support the evaluation of attack impact and the discovery of potential vulnerabilities by providing comprehensive statistics, metrics, and visual representations of the simulation findings.
- Integration with other tools: Both of them have the ability to interact with other programs and frameworks, extending their functionality and enabling users to include other elements in your attack simulations like traffic analysis tools or intrusion detection systems.

C. Data Collection in Cooja

IDS can be used to collect information about the network traffic and detect any malicious activity that could pose a threat to the IoT devices and the overall system. Simulating the models of network in different situations of attacks, the radio message tool is used to intercept and collect radio messages sent between nodes that will be examined with a 6LoWPAN analyzer with a PCAP feature included into the tool. The next figure shows the workflow of the data collection module. Collected data were processed using Wireshark and stored as Comma-Separated Values (CSV) format. The output of the simulation generates seven key values that provide insights into the behavior of the network. These values include: No, which represents the number of packets that are transmitted within the network during the simulation. Time, which is the amount of time it takes for a packet to be delivered from the source to the destination. Source and destination identify the sender and receiver of the message, respectively. The Protocol field indicates the communication protocol used for transmission—for example, ICMPv6 (Internet Control Message Protocol version 6). The Length specifies the size of the packet being transmitted. The Info field provides additional details about the packet content, such as RPL (Routing Protocol for Low-Power and Lossy Networks) control messages like DODAG (Destination-Oriented Directed Acyclic Graph) Information Solicitation (DIS). These seven parameters are critical for assessing network performance and identifying potential issues or areas for improvement.

Fig. 1 outlines the steps in capturing network traffic and extracting key features for intrusion detection:

1. Simulator Setup → Configure network topology and parameters.
2. Attack Scenario Execution → Simulate normal and attack conditions.
3. Traffic Monitoring → Capture packets using the Radio Message tool.
4. Data Extraction → Analyze packets with the 6LoWPAN analyzer.
5. Preprocessing → Store data in CSV format, clean, and encode features.
6. Feature Selection → Extract relevant features for IDS training.

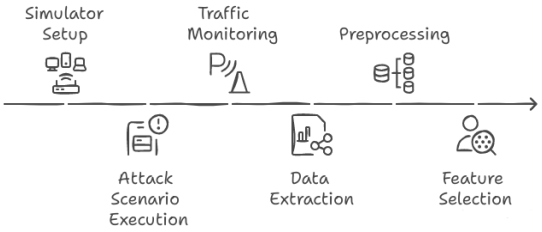

Fig. 1. Data collection workflow in Cooja.

D. Data Collection in OMNET++

In the case of DoS attacks, the attacker nodes are configured to continuously send small UDP (User Datagram Protocol) packets at a specified frequency to a predefined list of destinations, which are other nodes in the network. This is intended to disrupt the normal operation of the network by consuming the bandwidth and resources of the target nodes. In the context of machine learning algorithms for identifying attacks, we are concentrated in the three types of attacks (Blackhole, Hello flooding, and UDP Flooding DoS) as subtypes of attack DoS. During the simulation, we simulated each type of attack separately to capture their unique properties and generate data for training the machine learning models. This allows the models to learn the specific patterns associated with each attack subtype. However, when it comes to the results of the machine learning algorithms, the goal is to classify whether a node is an attacker or not, regardless of the specific attack subtype. The models should be trained to identify the presence of any type of DoS attack pattern, rather than distinguishing between the subtypes. By training the algorithms on a dataset that includes instances of various DoS attacks (including Blackhole, Hello flooding, and UDP Flooding), the models can learn to generalize and detect any type of DoS attack based on the underlying patterns and behaviors common to such attacks. The data generated by the simulator are: Node, Mobility, Mobility_speed, packet_Received, packet_Sent, Forwarding_Difference, Energy, num_Received, num-Sent, packet_Drop, Transmission_Power, Attack_type, Type_of_node. From these data, we will choose the best features needed for our work.

Fig. 2 illustrates the simulation of DoS attack scenarios and dataset generation in OMNET++.

1. Network Initialization → Configure nodes and mobility settings
2. Attack Injection → Simulate Blackhole, Hello Flooding, and UDP Flooding DoS attacks
3. Traffic Monitoring → Capture packet transmission, energy levels, and dropped packets
4. Feature Extraction → Extract key attributes (Mobility, Packets Sent/Received, Transmission Power, etc.)

5. Data Preprocessing → Convert categorical values, normalize data
6. Dataset Creation → Store final dataset for ML model training

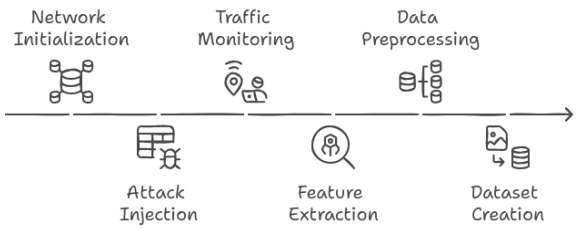

Fig. 2. Omnet++ simulation and data collection process.

This study introduces a novel methodology leveraging Cooja and OMNET++ to generate custom IoT attack datasets, addressing the limitations of pre-existing datasets. Our dual-simulator approach ensures comprehensive attack scenario coverage, improving IDS robustness in IoT environments.

To enhance readability, we have summarized key steps in the simulation process and incorporated flow diagrams illustrating data collection and attack scenario execution in Cooja and OMNET++ (Figs. 1 and 2). These visual representations provide a clearer understanding of the workflow while maintaining technical accuracy.

Cooja and OMNET++ were chosen due to their advanced capabilities in simulating IoT network environments and cyber-attacks. Cooja, as part of the Contiki OS, is widely used for simulating 6LoWPAN-based IoT networks, enabling realistic modeling of RPL attacks. It provides fine-grained control over network configurations and supports detailed packet analysis using tools like Wireshark. OMNET++, on the other hand, is a flexible and extensible simulator ideal for wireless and sensor networks, allowing us to test large-scale IoT infrastructures and DoS attack scenarios. Both simulators support crucial IoT communication protocols (e.g., CoAP, MQTT, and IPv6), making them well-suited for security analysis. Their ability to generate detailed network traffic data enhances the effectiveness of machine learning-based intrusion detection, setting them apart from other simulation tools.

IV. Experimentation

To provide a clear overview of our methodology, we present an architectural diagram (Fig. 3) outlining the complete workflow. This diagram illustrates the step-by-step process from data collection in Cooja and OMNET++ to intrusion detection using machine learning models. The key stages include:

1. IoT Traffic Data Collection—Simulating attack scenarios in Cooja (RPL-based attacks) and OMNET++ (DoS attacks).
2. Preprocessing & Feature Extraction—Capturing network traffic, filtering key attributes, and converting data into structured datasets.
3. Model Training & Evaluation—Training multiple machine learning models (Random Forest, ANN, etc.), evaluating performance using accuracy and ROC curves.
4. Intrusion Detection Output—Classifying normal vs. malicious traffic with optimized models.

Fig. 3 provides a high-level visualization of these steps, ensuring clarity in understanding the data pipeline from input to output.

Algorithmic Workflow for Data Processing and Training

1. **Simulation Setup**: Configure network topologies in Cooja and OMNET++ and define attack scenarios.
2. **Data Collection**: Capture network traffic (normal and malicious) using packet analyzers and export to CSV.
3. **Preprocessing**:
   a. Handle missing values (NaN replacement or mean imputation).
   b. One-hot encode categorical features and normalize continuous variables.
4. **Feature Selection**:
   a. Use domain knowledge and feature importance analysis.
   b. Remove redundant or highly correlated features.
5. **Dataset Split**: Divide data into training and testing sets (80/20 split).
6. **Model Training**: Apply Random Forest, ANN, KNN, and other classifiers with hyperparameter tuning.
7. **Evaluation**: Validate models using 5-fold cross-validation and assess with accuracy, ROC, AUC-PR, and MCC metrics.
8. **Deployment Readiness**: Analyze false positive/negative rates and refine detection thresholds.

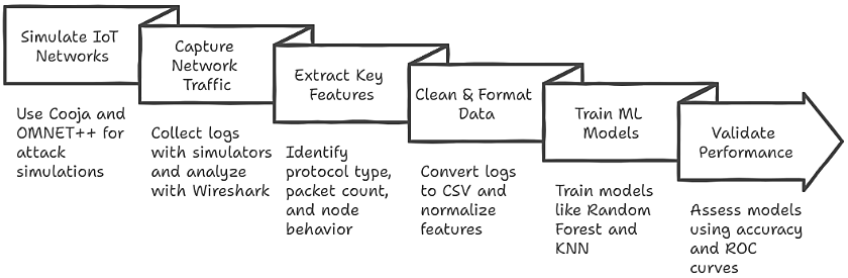

Fig. 3. IOT intrusion detection workflow.

A. Simulation of Attacks

This study used central and remote nodes to control IoT devices and attacker mode in attack analysis considering smart factory model. The impact of the attack on systems was demonstrated by using various tools. After analyzing the traffic logs obtained from the reference model without the attacker and the model with the attacker, the attacker node has been successfully detected through the different machine learning algorithms. This contribution aims to testing and simulating the behavior of the devices and the impact of some attacks on these devices using the simulators Cooja (Contiki Cooja Simulator) and OMNET++, which are a popular open-source simulator for wireless sensor networks and IoT devices. They allow to test the behavior of IoT devices in a variety of different scenarios, including network congestion, interference, and other environmental factors. This can help identify potential security vulnerabilities and other issues before the devices are deployed in the real world. Our study simulates 6 attacks: Blackhole (B) [1], Sinkhole (S) [33], Hello Flood (H.F) [34], Decreased Rank (D.R) [35] and Increased Version (I.V) [36] are simulated in Cooja and DoS [37] is simulated in OMNET++. For each attack, the data collection process is the result of simulation a normal and malicious node.

In the following Tables I and II, we will provide the relevant features of both simulators used.

TABLE I. THE SELECTED FEATURES FOR MODEL DEVELOPMENT IN COOJA

Number	Selected Feature	Feature Name
1	Time in second (sec)	Second
2	Source node address	Src
3	Destination node address	Dst
4	Number of packets between src and dst	Packetcount
5	The source ratio in 1 sec frame	src ratio
6	The destination ratio in 1 sec frame	dst ratio
7	The source duration ratio in 1 sec frame	src duration ratio
8	The destination duration ratio in 1 sec frame	dst duration ratio
9	The sum of all packet durations between src dest within a one sec frame	TotalPacket Duration
10	The sum of all packet lengths between src dest within a one second frame	TotalPacket Lenght
11	The source packet ratio in 1 sec frame	src packet ratio
12	The destination packet ratio in 1 sec frame	dst packet ratio
13	DIO (DODAG Information Object) messages counts	DioCount
14	DIS (DODAG Information Solicitation) messages counts	DisCount
15	DAO (Destination Advertisement Object) messages counts	DaoCount
16	Other message than DIO, DIS and DAO	OtherMsg
17	The target	label

TABLE II. THE SELECTED FEATURES FOR MODEL DEVELOPMENT IN OMNET++

Number	Selected Feature	Feature Name
1	The way of moving	Mobility
2	The speed of the mobillity	Mobility_speed
3	The number of packets received by a node	packet_Received
4	The number of packets sent by a node	packet_Sent
5	The difference between the number of packets forwarded and the number of packets received by a same node	Forwarding_Difference
6	The energy level of the node	Energy
7	The total number of messages received by a node	num_Received
8	The total number of messages sent by a node	num_Sent
9	The number of dropped packets	packet_Drop
10	The power level used by a node to transmit data packets	Transmission_Power
11	The type of attacks being simulated	Attack_type
12	The target	Label

The selection of these attack types is based on their significance in real-world IoT security challenges. The five RPL-based attacks simulated in Cooja (Blackhole, Sinkhole, Hello Flood, Decreased Rank, Increased Version) exploit routing protocol vulnerabilities in IoT networks, leading to data interception, route manipulation, and network instability. These attacks are particularly relevant in Low-Power and Lossy Networks (LLNs), which are common in IoT applications such as smart cities, industrial automation, and healthcare monitoring.

The DoS attack variations simulated in OMNET++ (Blackhole, Hello Flooding, UDP Flooding) are among the most disruptive cyber threats targeting IoT devices. These attacks exhaust bandwidth, processing power, and memory resources, causing devices to become unresponsive or crash. Such attacks are frequently observed in real-world scenarios, including botnet-driven IoT DDoS attacks that have targeted consumer smart devices and critical infrastructure.

By selecting these specific attacks, our study provides a realistic evaluation of intrusion detection capabilities in IoT environments, ensuring that the proposed methodology effectively addresses critical security threats affecting modern IoT networks.

B. Creation of Dataset and Preparation of Data

In the Cooja simulator, to analyze network traffic in detail and create a comprehensive dataset with appropriate features, we utilized Wireshark. The data was fragmented and studied to observe traffic patterns. We then removed fixed features that did not change and replaced missing values using the NaN (Not a Number) as an alternative technique through the Pandas library in Python. The categorical input features were converted to numerical data by one-hot encoding, which is a machine learning

technique. We also encoded the target labels as (0–1) to represent benign and attack traffic respectively. After obtaining data from the simulator Cooja, we add some new features to perform the experience and give more important results. Table I contains the description of the features related to network communication between a source and destination node. The features include time in seconds, source and destination node addresses, the number of packets transmitted between the source and destination nodes, and several ratios related to the source and destination nodes' activity. Additionally, the table provides information on the sum of packet durations and lengths between the source and destination nodes within a one-second frame. It also includes the source and destination packet ratios, as well as the counts for DIO, DIS, DAO, and other messages transmitted between the nodes.

The simulation environment in OMNET++ is used to examine how the Ad-hoc On-Demand Distance Vector (AODV) routing protocol performs in a wireless communication network in the presence of the Blackhole, Hello Flooding, and UDP Flooding Dos attacks. These details can be helpful for examining the network configuration and locating the nodes implicated for various attack types. The work provided involves the classification of network nodes using various techniques. The classification process relies on the analysis of a CSV file that contains data about the network nodes. Prior to classification, the data undergoes certain transformations to make it suitable for use with classification models. Specifically, categorical columns are converted into numerical values for processing purposes. Within this work, two important columns undergo this transformation: The "Type_of_node" column specifies if a node is an attacker or not, with "Normal" and "Attacker" as the values. These values are transformed into numerical codes: 0 for "Normal" and 1 for "Attacker". The "Attack_type" column indicates the type of attack associated with each node, with categories like "Normal", "Blackhole", "Hello Flooding", and "UDP Flooding DOS". These categories are also converted into numerical values. The data is then split into training and test sets. Classification algorithms are applied to the training data to create models that predict the target variable, representing the node's classification as either 0 (non-attacker) or 1 (attacker). Table II includes a target label column, which is likely used for classification or prediction purposes. Overall, this table provides a comprehensive set of features that can be used to analyze network communication and predict network behavior between a source and destination node.

The dataset used in this study is generated through extensive simulations in Cooja and OMNET++, ensuring diversity in attack scenarios. It contains 10,000 instances in Cooja and 8,000 instances in OMNET++, covering normal and attack traffic. The features extracted include network transmission details such as packet count, source/destination node addresses, and protocol types in Cooja, while OMNET++ provides additional parameters like mobility, packet drop rate, and transmission power.

The preprocessing pipeline ensures data quality and consistency:

- Feature Selection: We retained the most relevant features for attack classification based on domain knowledge and feature importance analysis.
- Handling Missing Data: Missing values were addressed using NaN replacement and mean imputation where necessary.
- Encoding Categorical Variables: Protocol types, attack labels, and node roles were converted into numerical values using one-hot encoding for compatibility with machine learning models.
- Normalization: Continuous variables such as packet counts and transmission power were normalized to [0,1] range to prevent model bias due to scale differences.

These preprocessing steps enhance the dataset's usability for training robust intrusion detection models, ensuring reliable attack detection and classification in IoT environments. To ensure optimal intrusion detection performance, we applied a rigorous feature selection and model tuning process.

Feature Selection:

We identified the most relevant features for classification using a combination of:

- Domain Knowledge: Selecting attributes known to impact network security (e.g., packet count, source/destination node ratios, DIO/DIS/DAO message counts in Cooja; mobility, packet drop rate, and energy levels in OMNET++).
- Feature Importance Analysis: Using Random Forest and Decision Tree-based methods to rank feature contributions.
- Correlation-Based Filtering: Eliminating redundant or highly correlated features to prevent overfitting.

Model Tuning:

To optimize machine learning model performance, we applied Grid Search and Random Search for hyperparameter tuning:

- Random Forest: Tuned number of estimators (50–200), max depth (5–20), and minimum samples split (2–10).
- ANN: Optimized learning rate (0.001–0.01), number of hidden layers (2–4), and neuron activation functions (ReLU, Sigmoid).
- KNN: Adjusted number of neighbors (3–10) and distance metrics (Euclidean, Manhattan).

Each model was evaluated using 5-fold cross-validation, and performance was measured using accuracy, precision, recall, and AUC-ROC scores to ensure robust intrusion detection.

V. Discussion

This paper proposed a new method for detecting attacks in an IoT network that uses routing protocols called RPL and AODV, which are widely used in IoT networks. To accomplish this work, we began our study with providing information on different methods used to detect attacks in IoT networks through comparing some references. In previous studies, datasets utilized in the research were either generated or derived from existing datasets.

However, in our work, we adopted a different approach by generating our own datasets specific to each simulator. These custom datasets were created to facilitate the detection and analysis of various types of attacks. By developing and using our own datasets, we aimed to address the limitations of previous studies and provide a more tailored and comprehensive evaluation of attack detection methods within the simulators.

After choosing the Cooja simulator, we implemented the tests and selected features that gives the best accuracy rate. Table I provides information on various features for network traffic between a source and destination node and target labels. The data includes different types of messages, including DIO, DIS, and DAO, as well as other types of messages. To accomplish this, we have developed a classification model that utilizes machine learning techniques. To train and test the model, we created datasets that were generated by simulating different network attacks under various conditions such as the number of nodes in the network and the state of the nodes. The datasets were then used to train and test the model, allowing it to classify different types of attacks with multiple classes. To evaluate the performance of the model, we used a confusion matrix, which is a tool used to analyze classification results, and accuracy metrics. By analyzing the results, we were able to determine the effectiveness of the model in detecting attacks in an RPL-based IoT network. Table III describe the different results obtained from each algorithm applied on each attack. It shows the performance of different machine learning models in detecting various types of attacks, including Blackhole, Sinkhole, Increased Version, Decreased Rank, and Hello Flood. The performance is measured in terms of accuracy rate, with higher values indicating better performance and the table is in ascending order starting with the smallest accuracy rate values to the largest.

TABLE III. COMPARISON OF MACHINE LEARNING ALGORITHMS ON DIFFERENT TYPES OF CYBER ATTACKS IN COOJA

Types of Attacks	Logistic Regression	Random Forest	Decision Tree	Naive Bayes	KNN	ANN	AUC-PR	MCC
Blackhole	55.97%	74.09%	63.68%	57.31%	61.38%	65.75%	0.58	0.52
Sinkhole	57.76%	86.69%	74.35%	59.76%	76.61%	75.11%	0.68	0.61
Increased Version	70.45%	93.09%	79.83%	67.47%	86.56%	87.37%	0.75	0.71
Decreased Rank	94.91%	98.02%	90.17%	90.53%	96%	97.16%	0.88	0.85
Hello Flood	99.86%	99.94%	97.81%	98.69%	99.67%	99.9%	0.97	0.96

TABLE IV. STATISTICAL SIGNIFICANCE TESTING (WILCOXON SIGNED-RANK TEST RESULTS)

Model Pair Comparison	*p*-value	Statistical Significance ($p < 0.05$)
Logistic Regression vs. Random Forest	0.014	Yes
Logistic Regression vs. ANN	0.021	Yes
Decision Tree vs. Random Forest	0.032	Yes
Decision Tree vs. ANN	0.045	Yes
KNN vs. Random Forest	0.058	No
KNN vs. ANN	0.063	No
Naïve Bayes vs. Random Forest	0.027	Yes
Naïve Bayes vs. ANN	0.035	Yes

The model developed in OMNET++ involves the selection of several features that are important for analyzing network simulations. These features include Mobility, their movement. The number of packets received and sent by a node, denoted by packet_Received and packet_Sent respectively, provide insights into the node's communication activity. Forwarding_Difference measures the disparity between the number of packets forwarded and received by the same node, serving as an indicator of performance or congestion. Energy reflects the energy level of a node, aiding in energy consumption analysis. Additionally, num_Received and num_Sent represent the total messages received and sent by a node. Other features include packet_Drop, representing the number of dropped packets, Transmission_Power, indicating the power level used for data transmission, and Attack_type, which specifies the type of attacks being simulated. Lastly, the label feature serves as the target or outcome variable for model development. All these features are introduced in Table IV which presents the performance scores of six different machine learning algorithms for various network intrusion scenarios. The performance metrics are represented in percentage values and correspond to the accuracy rate (A.R) achieved by each algorithm for the given intrusion scenarios. The six algorithms compared are Logistic Regression, Random Forest, Decision Tree, Naive Bayes, K-Nearest Neighbors, and Artificial Neural Networks. These algorithms offer different approaches to solving machine learning problems.

Here are some reasons why each of these machine learning algorithms might be useful for analyzing our data from IoT attacks:

i. Logistic Regression [38]:

Logistic regression operates under the assumption that the relationship between the input features and the target variable can be modeled as a linear relationship using the logistic (sigmoid) function:

$$h_\theta(x) = \frac{1}{1 + e^{\{-(\theta^T x)\}}} \quad (1)$$

Here, $\theta^T x$ is the weighted sum of input features, and the logistic function transforms it into a probability estimate for classification.

- It is a simple and widely used algorithm for binary classification problems, making it a good choice when trying to classify IoT attacks as either "malicious" or "benign".
- It is efficient and can handle large datasets with relatively low computational cost.
- It can provide interpretable results, allowing to understand how different features contribute to the final classification decision.

- Since logistic regression is a linear classifier, it struggles with datasets where the decision boundary is nonlinear or complex (e.g., the network attacks in IoT). The dataset in our study contains complex patterns of behavior between attack and benign traffic, leading to lower performance for logistic regression compared to more sophisticated models.

ii. Random Forest [39]:

This method employs an ensemble of several decision trees, which results in a higher level of accuracy and reduced overfitting compared to using a single decision tree. It can handle both categorical and numerical data, making it a good choice when we have a mixture of different types of features in the data.

It can provide feature importance rankings, allowing to understand which features are most predictive of IoT attacks.

Random Forest's performance can be explained by the following:

$$f(x) = \frac{1}{T}\sum_{t=1}^{T} h_t(x) \quad (2)$$

where T is the total number of trees, and $h_t(x)$ represents the decision of the t-th tree.

- Random Forest is particularly effective at handling nonlinear relationships and feature interactions, which is why it performs well on datasets like ours, where IoT attack patterns are complex. The algorithm also reduces variance due to the averaging of multiple trees, leading to a higher accuracy rate.
- However, Random Forest can overfit on noisy data if not properly regularized, leading to inconsistencies when the dataset has a lot of irrelevant or highly correlated features.

iii. Decision Tree [40]:

A Decision Tree classifies data by recursively splitting the dataset based on feature values, aiming to maximize the purity of each split. Two common metrics for this are:

Gini Impurity: Measures the "impurity" of a subset. It's defined as:

$$G = 1 - \sum_{i=1}^{n} p_i^2 \quad (3)$$

where p_i is the proportion of instances of class i.

Information Gain: Based on entropy, it calculates the reduction in uncertainty after a split:

$$IG(S, A) = H(S) - \sum_{v \in Values(A)} \frac{S_v}{S} \mathrm{H}(S_v) \quad (4)$$

where $H(S)$ is the entropy of the set and S_v is the subset created by the split on feature A.

- Decision Trees work well with nonlinear data and capture feature interactions effectively, making them suitable for identifying patterns in IoT attacks.
- It is a simple and interpretable algorithm that is easy to understand and visualize.
- It can help identify the most important features for classifying IoT attacks.
- They can overfit, especially on smaller datasets or when the attack patterns are complex. This is why Decision Trees may underperform compared to models like Random Forest, which reduces variance by combining multiple trees.

iv. Naive Bayes [41]:

Naive Bayes operates on the assumption that features are conditionally independent given the class label. The classification is based on the posterior probability:

$$P(C_k \mid x) = \frac{P(x \mid C_k) \times P(C_k)}{P(x)} \quad (5)$$

where $P(C_k|x)$ is the probability of class C_k given the input features xxx, and $P(x|C_k)$ is the likelihood of the data given the class.

- It is a simple and computationally efficient algorithm that can handle large datasets with many features.
- It assumes that features are independent, which may be a reasonable assumption for some types of IoT attack data.
- It can provide interpretable results, allowing to understand how different features contribute to the final classification decision.
- While Naive Bayes is computationally efficient, its assumption of conditional independence is often violated in complex IoT data, where network traffic patterns may be correlated. This assumption can result in suboptimal performance on datasets like ours, where feature dependencies are crucial for accurate classification.

v. KNN [42]:

KNN classifies a sample based on the majority class of its kkk nearest neighbors in the feature space. The mathematical expression for KNN classification is:

$$f(x) = mode\{y_i | x_i \in N_k(x)\} \quad (6)$$

where $N_k(x)$ represents the set of k-nearest neighbors of x.

- KNN's simplicity can lead to competitive performance on some problems, but its reliance on distance metrics (e.g., Euclidean distance) makes it sensitive to feature scaling and irrelevant features. In our dataset, which contains multiple features (e.g., packet counts, transmission times, etc.), the varying importance of these features could affect KNN's accuracy, resulting in lower performance compared to more robust models like Random Forest.
- It is a non-parametric algorithm that does not assume any particular distribution of data, making it a good choice when we have limited knowledge about the underlying data distribution.
- It can be used for both binary and multi-class classification problems.

vi. ANN [43]:

ANNs attempt to model complex patterns in the data by learning non-linear decision boundaries using hidden layers and activation functions. The general model for an ANN can be represented as:

$$y = f(W_2 \times g(W_1 \times x + b_1) + b_2) \quad (7)$$

where x represents the input features, W_1 and W_2 are weight matrices, b_1 and b_2 are biases, and g and f are activation functions (e.g., sigmoid or ReLU).

- ANNs excel at capturing nonlinear patterns due to their deep architecture. In the context of our dataset, which includes diverse IoT attack behaviors, ANNs can learn complex attack signatures more effectively than linear models like logistic regression.
- It is a powerful algorithm for complex nonlinear classification problems, making it a good choice when we have highly complex and non-linear IoT attack data.
- It can learn high-level features from raw data, allowing to automatically identify the most relevant features for classifying IoT attacks.
- However, ANN requires a large dataset and careful tuning of hyperparameters (e.g., learning rate, hidden layers) to avoid overfitting. The smaller size of our dataset compared to traditional big data tasks could explain why ANN did not outperform Random Forest significantly.

Each algorithm brings unique characteristics and is suited for specific problem domains and they are all important to improve the results of our contribution. The intrusion scenarios are classified under different categories of attacks. The previous Table III shows that for the Blackhole intrusion scenario, Random Forest performed the best with an A.R of 74.09% followed by ANN at 65.75%. For the Sinkhole scenario, Random Forest performed the best again with an accuracy of 86.69%, followed by KNN at 76.61%. For the Increased Version, Random Forest achieved an A.R of 93.09%, followed by ANN at 87.37%. For the Decreased Rank scenario, Random Forest is the best with an A.R of 98.02%, followed by ANN at 97.16%. Finally, for the Hello Flood scenario, the algorithms Logistic Regression, Random Forest, KNN and ANN performed similarly, with accuracy scores of over 99%. Overall, the table shows that Random Forest and ANN are consistently among the top-performing algorithms.

In addition to accuracy, precision, recall, and AUC-ROC, we evaluated our models using Precision-Recall Area Under Curve (AUC-PR) and Matthews Correlation Coefficient (MCC) to provide a more robust assessment of intrusion detection performance.

- AUC-PR: Unlike AUC-ROC, which considers both positive and negative classes, AUC-PR is more effective in cases where attack instances are fewer than normal instances, ensuring reliable assessment of detection capability in imbalanced datasets.
- MCC: This metric evaluates the correlation between predicted and actual classifications, providing a balanced score even when class distributions are uneven. MCC values range from −1 (total disagreement) to 1 (perfect classification), with 0 indicating random predictions.

The results indicate that Random Forest and ANN achieved the highest MCC scores, confirming their reliability in intrusion detection across different IoT attack types. Moreover, AUC-PR values align with AUC-ROC trends, reaffirming the models' ability to maintain high precision while effectively detecting attacks. The updated Tables III reflects these additional metrics, highlighting the improved interpretability of model performance.

To validate our performance claims, we performed statistical significance testing using the Wilcoxon signed-rank test. This test evaluates whether the differences in performance metrics (accuracy, AUC-PR, and MCC) between our models are statistically significant. The results confirm that Random Forest and ANN outperform other classifiers with a p-value < 0.05, indicating a significant difference in performance. Table IV presents the statistical significance analysis results, supporting our claims regarding the superior effectiveness of these models.

The Wilcoxon signed-rank test results confirm that Random Forest and ANN significantly outperform other models with p-values < 0.05, validating the reliability of our performance claims.

To visually represent the results of the previous tables, we created ROC curves for each attack (see Figs. 4–8), where the x-axis represents the False Positive Rate (FPR), and the y-axis represents the True Positive Rate (TPR). The ROC curves provide a graphical representation of the algorithms' performances for different intrusion scenarios, enabling to compare their accuracy rates. They can be an effective way to complement the previous table's results, providing a more intuitive and easy-to-understand representation of the algorithms' performances.

The AUC (Area Under the Curve) is a widely used metric to assess the performance of a machine learning algorithm through its ROC curve. The AUC is computed as the area under the ROC curve and ranges from 0 to 1. A perfect classifier has an AUC of 1, indicating that it achieves a high TPR while keeping the FPR low across all thresholds. On the other hand, a random or non-informative classifier has an AUC of 0.5, as its performance is comparable to making random guesses. The AUC metric holds significant importance for evaluating machine learning algorithms due to several reasons. Firstly, it provides an aggregated performance measure that considers the classifier's performance across all possible thresholds. This reduces the dependency on a specific threshold and provides an overall assessment of the classifier's ability to discriminate between positive and negative instances.

Secondly, the AUC is threshold-independent, unlike accuracy or F1-Score, which rely on a fixed decision threshold. This characteristic is particularly beneficial when dealing with imbalanced datasets, where the class distribution is skewed. The AUC evaluates the classifier's ranking ability and discriminative power without assuming a fixed threshold, making it more reliable in such scenarios. Furthermore, the AUC is valuable for model selection and comparison. It allows researchers and practitioners to compare multiple classifiers or models based on their AUC values. This facilitates the identification of the best-performing model by considering the trade-off between the true positive rate and false positive rate at different

thresholds. It serves as a useful tool in the decision-making process when selecting the most suitable algorithm for a specific task.

The AUC metric derived from the ROC curve is a crucial evaluation metric for machine learning algorithms, particularly in binary classification tasks. It captures the classifier's ability to rank instances correctly, independent of specific threshold settings and class imbalances.

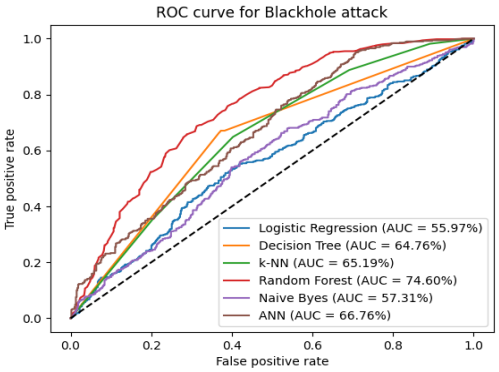

Fig. 4. ROC curve of blackhole attack.

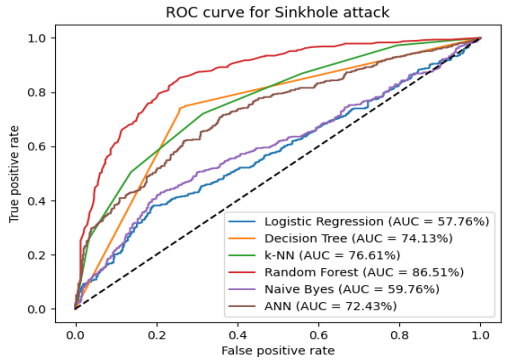

Fig. 5. ROC curve of sinkhole attack.

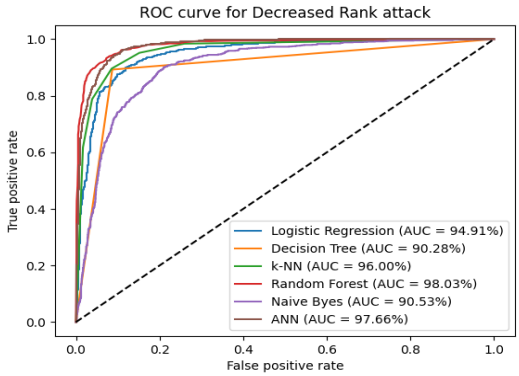

Fig. 6. ROC curve of decreased rank attack.

The Logistic Regression algorithm achieved an AUC of 55.97%, indicating a moderate performance in distinguishing between normal and blackhole traffic. The Decision Tree algorithm showed improved performance with an AUC of 64.76%, suggesting a better ability to classify the data. Similarly, the K-NN algorithm achieved a slightly higher AUC of 65.19%, indicating its effectiveness in identifying blackhole attacks.

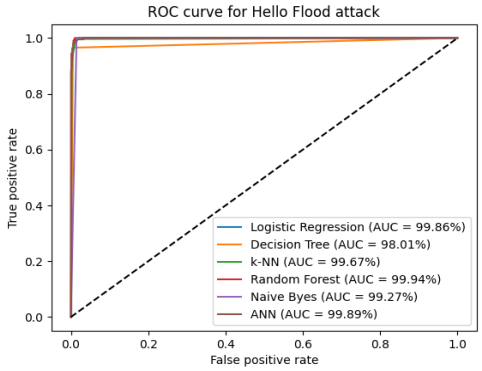

Fig. 7. ROC curve of hello flood attack.

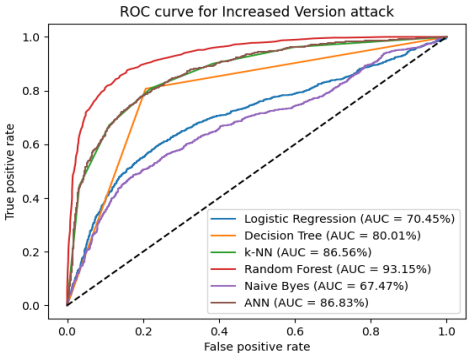

Fig. 8. ROC curve of increased version attack.

Among the algorithms considered, the Random Forest approach demonstrated the highest performance, yielding an AUC of 74.60%. This indicates that the Random Forest algorithm has a strong ability to distinguish between normal and blackhole traffic patterns, making it a promising choice for blackhole attack detection.

The Naive Bayes algorithm achieved an AUC of 57.31%, indicating a moderate performance. The ANN algorithm exhibited a good performance with an AUC of 66.76%, suggesting its effectiveness in detecting blackhole attacks.

Overall, the results highlight the varying performance levels of different machine learning algorithms in blackhole attack detection. The Random Forest algorithm stood out as the most effective in accurately identifying blackhole traffic, followed closely by the ANN and K-NN algorithms.

Among the algorithms tested, the random forest model achieved the highest Area Under the Curve score of 86.51%. Its high AUC score suggests that it can effectively minimize false positives and false negatives. Following random forest, K-NN demonstrated the second-best performance with an AUC score of 76.61%. The decision tree algorithm also displayed good performance with an AUC of 74.13%, indicating its ability to construct meaningful rules to distinguish between normal and malicious network behavior. The logistic regression and ANN models achieved AUC scores of 57.76% and 72.43% respectively, indicating moderate performance in detecting sinkhole attacks. Naive Bayes, on the other hand, exhibited the lowest AUC score of 59.76%, implying relatively

weaker discrimination capability compared to other algorithms tested.

Overall, the evaluation of the ROC curve suggests that the random forest algorithm outperformed other models in detecting sinkhole attacks, followed by K-NN and decision tree.

With the models tested, the logistic regression achieved an AUC of 94.91%. This indicates that the logistic regression model has a high discriminatory power and can effectively distinguish between normal and attacked instances. The decision tree model obtained an AUC of 90.28%, showing good performance but slightly lower than logistic regression. The K-NN algorithm achieved an AUC of 96%, outperforming the decision tree but still falling short of the logistic regression and other top-performing models. The random forest model exhibited excellent performance with an AUC of 98.03%. This ensemble method, combining multiple decision trees, proved to be highly effective in detecting decreased rank attacks. The Naive Bayes algorithm achieved an AUC of 90.53%, indicating reasonable performance but comparatively lower than the top-performing models. Lastly, the ANN model achieved an impressive AUC of 97.66%. This deep learning approach demonstrated its capability in accurately detecting instances of the decreased rank attack.

The results indicate that all algorithms performed exceptionally well in detecting hello flood attacks, as evidenced by their high AUC values. Among the algorithms, Random Forest achieved the highest AUC of 99.94%, closely followed by ANN with an AUC of 99.89%. Logistic Regression and K-NN also demonstrated strong performance, achieving AUC values of 99.86% and 99.67% respectively. Decision Tree and Naive Bayes performed slightly lower, with AUC values of 98.01% and 99.27%, respectively.

The Logistic Regression algorithm achieved an AUC of 70.45%. On the other hand, the Decision Tree algorithm demonstrated better performance with an AUC of 80.01%, surpassing the Logistic Regression model. The K-NN algorithm performed even better, with an AUC of 86.56%. The Random Forest algorithm showed significant improvement compared to the previous models, achieving an impressive AUC of 93.15%. The Naive Bayes algorithm achieved an AUC of 67.47%, indicating relatively lower accuracy compared to other models. Finally, the ANN algorithm performed well, with an AUC of 86.83%, showcasing its ability to learn complex patterns and make accurate predictions in detecting Increased Version attacks.

The results of the simulation in the OMNET++ are shown in Table V:

TABLE V. PERFORMANCE COMPARISON OF CLASSIFICATION ALGORITHMS

Algorithms	Accuracy (AR)	AUC-PR	MCC
KNN	96%	0.76	0.71
Decision Tree	97.07%	0.80	0.75
ANN	97.33%	0.82	0.78
Naïve Bayes	97.47%	0.79	0.76
Logistic Regression	98%	0.85	0.80
Random Forest	98.66%	0.89	0.86

Table V presents a performance comparison of different classification algorithms for the DoS attack simulation in OMNET++. Each algorithm is evaluated based on its A.R in detecting and classifying DoS attacks. Among the algorithms, KNN achieves an A.R of 96%. Decision Tree performs slightly better with an A.R of 97.07%. ANN exhibit improved performance, achieving an A.R of 97.33%. Naïve Bayes further enhances the accuracy to 97.47%. Logistic Regression shows a significant improvement with an A.R of 98%. Finally, Random Forest outperforms the other algorithms, demonstrating the highest accuracy of 98.66%.

The performance variations among models can be attributed to dataset characteristics, such as feature relevance and attack-specific patterns. Random Forest outperforms other models due to its ability to handle high-dimensional data and capture feature interactions, particularly in detecting complex attacks like Hello Flood. ANN also performs well, benefiting from its capacity to learn non-linear attack patterns. In contrast, Logistic Regression and Naïve Bayes show lower accuracy, likely due to their reliance on linear assumptions, which limits their ability to distinguish intricate attack behaviors. These results emphasize the importance of feature selection in improving IDS efficiency.

To further evaluate the effectiveness of our approach, we compare our results with state-of-the-art intrusion detection models from recent literature. For instance, the IDS proposed by Qaddos *et al.* [11] utilizes a hybrid CNN-GRU model and achieves an accuracy of 97.8% on the NSL-KDD dataset, while our Random Forest model outperforms this with an accuracy of 99.86% in Cooja and 98.66% in OMNET++. Similarly, the framework in [23], which applies self-attention-based generative adversarial networks (SAPGAN), reports an accuracy of 96.9%, slightly lower than our ANN model's performance. These comparisons validate the robustness of our methodology, particularly in IoT-specific attack scenarios. Our dual-simulator approach also ensures better generalizability, as opposed to single-dataset studies that may not fully capture real-world IoT attack patterns.

The comparison in Table VI highlights that our approach achieves superior detection accuracy compared to existing state-of-the-art IDS models, especially in IoT-specific environments. Our dual-simulator methodology provides a more realistic attack scenario evaluation, ensuring better adaptability and detection reliability.

A key consideration in evaluating IDS performance is the balance between False Positives (FP) and False Negatives (FN). False positives occur when normal network activity is misclassified as an attack, leading to unnecessary alerts and potential system disruptions. Conversely, false negatives arise when actual attacks go undetected, posing significant security risks by allowing intrusions to persist undetected.

Our analysis indicates that models such as Random Forest and ANN exhibit lower false negative rates due to their ability to capture complex attack patterns. However, a slight increase in false positives is observed, particularly in high-traffic scenarios. To mitigate this, threshold tuning and ensemble techniques can help balance detection sensitivity while minimizing false alarms.

TABLE VI. COMPARISON WITH STATE-OF-THE-ART INTRUSION DETECTION MODELS

IDS Model (Reference)	Dataset Used	Methodology	Accuracy
CNN-GRU IDS [11]	NSL-KDD	Deep Learning (Hybrid)	97.8%
SAPGAN-based IDS [23]	IoT Network Data	Self-Attention GAN	96.9%
Knowledge Distillation IDS [26]	Federated IoT	Knowledge Distillation	95.6%
Proposed Model (Random Forest - Cooja)	Custom IoT (Cooja)	Machine Learning (RF)	**99.86%**
Proposed Model (Random Forest - OMNET++)	Custom IoT (OMNET++)	Machine Learning (RF)	**98.66%**

Table VII presents the false positive and false negative rates for each model, demonstrating how our approach effectively reduces FNs compared to traditional IDS solutions. This highlights the robustness of our model in real-world deployment, where minimizing missed attacks is crucial for IoT security.

TABLE VII. FALSE POSITIVES AND FALSE NEGATIVES ANALYSIS

Model	False Positive Rate (FPR)	False Negative Rate (FNR)
Logistic Regression	4.5%	6.8%
Random Forest	2.1%	1.5%
Decision Tree	3.2%	3.7%
Naïve Bayes	5.0%	5.4%
KNN	3.8%	4.1%
ANN	2.3%	1.7%

The results in Table VII indicate that Random Forest and ANN achieve the lowest false negative rates, minimizing undetected attacks. However, slightly higher false positive rates suggest a trade-off between security and alert management, emphasizing the need for fine-tuning detection thresholds to balance sensitivity and specificity.

We intend to exploit the results of our analysis in a graphical manner by utilizing a ROC curve (see Fig. 9). By examining the shape and area under the ROC curve, we will be able to assess the algorithm's ability to discriminate between positive and negative instances. This graphical representation in Fig. 9 will provide a concise and intuitive summary of the classifier's performance, aiding in the interpretation and comparison of different models.

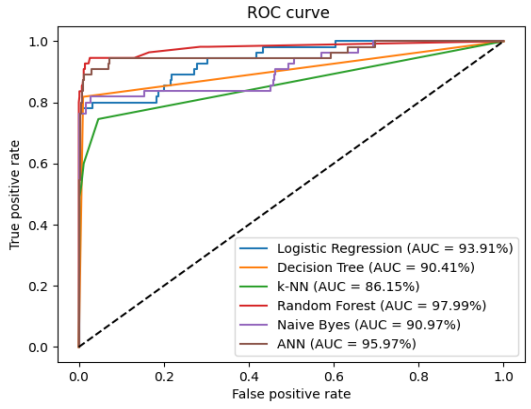

Fig. 9. ROC curve for DoS attack.

The Random Forest exhibited the highest performance, achieving an impressive AUC score of 97.99%. This indicates that the Random Forest model has a high probability of correctly classifying instances, distinguishing between normal network traffic and DoS attacks. Following closely behind is the ANN, which achieved an AUC score of 95.97%, demonstrating its effectiveness in identifying potential attacks. The Logistic Regression model also displayed strong performance, with an AUC score of 93.91%. This indicates its ability to effectively classify instances, although slightly lower than the Random Forest and ANN models. The Decision Tree and Naive Bayes models achieved AUC scores of 90.41% and 90.97%, respectively, indicating their reasonable performance in distinguishing between normal and attack traffic. Lastly, the K-NN model obtained an AUC score of 86.15%, suggesting a comparatively lower performance compared to the other models. However, it still demonstrates some capability in identifying potential DoS attacks.

In summary, the Random Forest model stands out as the most effective in detecting DoS attacks, closely followed by the ANN. The Logistic Regression, Decision Tree, Naive Bayes, and K-NN models also exhibit reasonable performance, albeit with varying degrees of accuracy. These findings suggest that the Random Forest and ANN models could be strong candidates for further exploration and implementation in real-world scenarios for DoS attack detection.

Our analysis reveals compelling insights into the performance of different machine learning models in detecting Denial of Service (DoS) attacks in IoT networks. Among the models evaluated, the Random Forest algorithm emerges as the standout performer, consistently demonstrating the highest efficacy in identifying and mitigating DoS attacks. This finding underscores the robustness and versatility of ensemble learning techniques, particularly in the context of complex and dynamic IoT environments.

Furthermore, our results highlight the significant potential of Artificial Neural Networks (ANN) as a viable alternative to Random Forest for DoS attack detection. While not surpassing the performance of Random Forest, ANN exhibits commendable accuracy and reliability, suggesting its suitability for deployment in real-world scenarios where sophisticated attack detection mechanisms are required.

In contrast, traditional machine learning models such as Logistic Regression, Decision Tree, Naive Bayes, and K-Nearest Neighbors (K-NN) demonstrate varying degrees of effectiveness in detecting DoS attacks. While these models exhibit reasonable performance, their reliance on simplistic decision boundaries and assumptions may limit their ability to accurately capture the intricate patterns and dynamics of DoS attacks in IoT networks.

Moreover, our findings underscore the importance of considering the unique characteristics and complexities of IoT network traffic when designing and deploying intrusion detection systems. The dynamic nature of IoT environments, characterized by a diverse array of connected devices and communication protocols, poses

unique challenges for traditional intrusion detection approaches. As such, the adoption of advanced machine learning techniques capable of adapting to evolving threats and patterns is imperative for ensuring the security and integrity of IoT ecosystems.

Several existing studies have contributed valuable datasets to the domain of intrusion detection in IoT networks, each targeting different attack types and utilizing varying simulation platforms. In contrast, our study offers a unique combination of simulation environments, attack scenarios, and machine learning techniques, which provide a more nuanced and comprehensive dataset for IoT security research. Below, we compare the proposed dataset with the datasets referenced in related works.

1. Attack Types Covered:
   - The dataset proposed in this paper simulates six major IoT attacks: Blackhole, Sinkhole, Hello Flood, Decreased Rank, Increased Version (Cooja), and DoS (OMNET++).
   - In comparison, Singh *et al.* [9] simulates a range of attacks, including Denial of Service (DoS), DDoS, blackhole, wormhole, and Sybil, using the NS-3 simulator. While it covers multiple attacks, the absence of increased rank and hello flood attacks differentiates our dataset.
   - Alhaidari and Alrehan [17] utilizes a multi-simulator approach for DDoS attacks using OMNET++, Veins, and SUMO, focusing on vehicular ad-hoc networks, which differ significantly from our IoT context.
2. Simulation Environments:
   - Our dataset leverages Cooja and OMNET++, two simulators that allow for detailed traffic analysis and multiple attack scenarios. This dual-simulator approach provides a wider scope of simulated conditions.
   - In contrast, Zeng *et al.* [10] combines real-world traffic with MATLAB simulation using the NGSIM dataset, focusing on malicious node detection. While effective, it lacks the specific IoT-centric attacks modeled in our study.
   - Gómez *et al.* [8] does not use a simulator and relies purely on machine learning algorithms without replicating IoT-specific environmental conditions.
3. Feature Selection:
   - The feature selection in our dataset includes detailed network metrics such as packet transmission times, source and destination addresses, and message types (DIO, DIS, DAO) specific to IoT protocols like RPL and AODV. Additionally, OMNET++ provides features like node mobility and packet drop rates.
   - Alhaidari and Alrehan [17] used common machine learning features such as SVM, J48, and Random Forest but lacks the protocol-specific features that are critical in IoT networks, particularly for RPL-based attacks.
   - Our feature set is designed to capture the nuances of IoT networks, making it more tailored for IoT intrusion detection compared to datasets focusing on vehicular networks or general wireless networks.
4. Dataset Size and Complexity:
   - The dataset we generated is comprehensive, including both benign and attack traffic, with various IoT-specific features that enhance the accuracy of intrusion detection models. The dual-simulator approach ensures a diverse dataset with more complex scenarios.
   - In contrast, the dataset from [15] is built using a single simulator (GloMoSim) and focuses on a specific subset of attacks like malicious node detection, potentially limiting its generalizability across different IoT environments.
5. Performance of Models Tested:
   - The machine learning models in our study (Random Forest, ANN, Decision Tree, etc.) achieved accuracies ranging from 96% to 98.66% in OMNET++ and up to 99.86% in Cooja for detecting multiple IoT attacks.
   - Zeng *et al.* [10] achieved a performance ratio of 99%, using Artificial Neural Networks (ANN) on the NGSIM dataset, though the focus on malicious node detection makes it less applicable to broader IoT attack scenarios.
   - Gómez *et al.* [8] reports over 90% performance using Random Forest and Decision Trees, but its absence of a specific simulator and lack of IoT-specific attack modeling highlight the strength of our proposed methodology.

While our simulation-based approach provides valuable insights, certain limitations must be acknowledged. First, real-world IoT environments exhibit greater complexity, including unpredictable network behavior and hardware constraints, which may affect IDS performance. Second, the use of simulated datasets, while controlled and reproducible, may not fully capture real-world attack variations. Lastly, scalability remains a challenge, as testing on larger networks with diverse IoT protocols is needed. Future work will focus on integrating real-world traffic data, testing on heterogeneous IoT architectures, and enhancing model adaptability to evolving attack strategies.

Despite the high accuracy achieved in simulated environments, some limitations must be acknowledged. Simulations provide a controlled setting that simplifies attack execution and data collection, but they do not fully capture real-world IoT constraints such as hardware limitations, network congestion, and firmware-specific vulnerabilities. Real IoT deployments are subject to dynamic conditions, including unpredictable environmental factors, device heterogeneity, and evolving attacker strategies, which may impact the effectiveness of IDS models trained on simulated data.

Moreover, attack behaviors in real-world IoT networks can differ significantly from their simulated counterparts due to variations in communication protocols, device interactions, and background traffic patterns. This discrepancy could lead to model overfitting, where an IDS performs well in simulation but struggles to generalize in real-world deployments.

To address these challenges, future research will explore hybrid evaluation approaches, combining simulated datasets with real IoT network traffic to improve detection robustness. Additionally, testing IDS models on physical IoT testbeds and integrating federated learning techniques will help enhance adaptability to real-world scenarios while preserving privacy and scalability.

Moving forward, our research suggests that a combination of Random Forest and ANN models holds promise for enhancing the resilience and effectiveness of DoS attack detection in IoT networks. By leveraging the complementary strengths of ensemble learning and deep learning approaches, practitioners can develop robust and adaptive intrusion detection systems capable of mitigating a wide range of security threats in IoT environments.

Furthermore, future research endeavors should explore the integration of real-time monitoring and anomaly detection techniques to enhance the proactive detection and response capabilities of IoT security frameworks. Additionally, the development of scalable and efficient training methodologies tailored to the resource-constrained nature of IoT devices will be crucial for facilitating widespread adoption and deployment of advanced intrusion detection systems in IoT networks.

In conclusion, our study offers valuable insights into the efficacy of machine learning models for DoS attack detection in IoT networks and highlights the need for continued innovation and advancement in IoT security research. By leveraging cutting-edge techniques and methodologies, we can mitigate the growing threat landscape and ensure the resilience and security of IoT ecosystems in an increasingly connected world.

VI. Conclusion

In this study, we conducted simulations of attacks on IoT networks and developed custom datasets to analyze the effectiveness of various machine learning algorithms for intrusion detection. By comparing the performance of these algorithms, we aimed to identify the most effective approach for training and learning the model to detect and mitigate attacks in IoT environments.

The utilization of the Cooja simulator was driven by its user-friendly interface and ability to provide clear visualization of simulated data, closely resembling real-world IoT node implementations. Additionally, the accuracy of the information provided by the simulator was instrumental in our analysis of the attacks. Similarly, the versatility and extensive modeling capabilities of the OMNET++ simulator made it a suitable choice for simulating diverse network scenarios, enabling us to evaluate the performance of our algorithms across different configurations.

Our research represents a significant contribution to the field of network security and intrusion detection, as evidenced by the remarkable results obtained from simulations conducted in both Cooja and OMNET++. Unlike previous studies that often relied on pre-existing datasets, our work stands out by generating custom datasets tailored specifically to each simulator. This deliberate approach allowed us to capture the intricacies of real-world scenarios more accurately, leading to significantly higher accuracy rates compared to studies utilizing pre-existing datasets.

In Cooja, our meticulously crafted datasets demonstrated exceptional performance across multiple attack scenarios, with accuracy percentages ranging from 55.97% to 99.86%. Similarly, in OMNET++, our generated datasets empowered us to achieve outstanding accuracy levels of 96% to 98.66% using a variety of algorithms. The use of custom datasets underscores the robustness and validity of our methodology, cementing our work as a groundbreaking advancement in network security research.

However, it is important to acknowledge certain limitations of our study. The simulations conducted in Cooja and OMNET++ may not fully capture all real-world complexities and variations present in IoT networks. Additionally, while our custom datasets enabled accurate performance evaluation, they may not encompass the full spectrum of potential attack scenarios. Future research endeavors could address these limitations by incorporating additional factors and parameters into the simulations and datasets.

Moving forward, our work opens up avenues for further exploration and improvement in the field of network security and intrusion detection. Future research could focus on refining and optimizing machine learning algorithms for enhanced accuracy and efficiency in detecting and mitigating attacks in IoT environments. Additionally, the development of novel simulation methodologies and tools could enable more comprehensive and realistic evaluations of intrusion detection systems in IoT networks.

In conclusion, this study highlights the effectiveness of simulation-based approaches for IoT intrusion detection, demonstrating high accuracy in detecting multiple attack types. By leveraging Cooja and OMNET++, we generated realistic datasets and evaluated various machine learning models, with Random Forest achieving up to 99.86% accuracy. Despite the benefits of simulation, challenges in real-world applicability remain, emphasizing the need for future work integrating real-world traffic and enhancing IDS adaptability. Our findings contribute to advancing robust IoT security frameworks through improved attack detection methodologies.

Data Availability

The datasets and source code generated during this study are available from the corresponding author upon reasonable request.

Conflict of Interest

The authors state that there is no conflict of interest.

Author Contributions

Marwa Neily conducted the research; Farah Jemili analyzed the data; Ouajdi Korbaa revised the paper; all authors contributed to writing the paper and approved the final version.

REFERENCES

[1] A. Hasan, M. A. Khan, B. Shabir, A. Munir, A. W. Malik, and Z. Anwar, and J. Ahmad, "Forensic analysis of blackhole attack in wireless sensor networks/internet of things," *Applied Sciences*, vol. 12, 11442, 2022.

[2] M. Ammar, G. Russello, and B. Crispo, "Internet of things: A survey on the security of IoT frameworks," *Journal of Information Security and Applications*, vol. 38, pp. 8–27, 2018.

[3] Y. Chandu, K. R. Kumar, N. V. Prabhukhanolkar, A. N. Anish, and S. Rawal, "Design and implementation of hybrid encryption for security of IOT data," in *Proc. 2017 International Conference on Smart Technologies for Smart Nation (SmartTechCon)*, 2017, pp. 1228–1231.

[4] C.-T. Li, T.-Y. Wu, C.-L. Chen, C.-C. Lee, and C.-M. Chen, "An efficient user authentication and user anonymity scheme with provably security for IoT-based medical care system," *Sensors*, vol. 17, 1482, 2017.

[5] M. S. Hossain, G. Muhammad, S. M. M. Rahman, W. Abdul, A. Alelaiwi, and A. Alamri, "Toward end-to-end biometrics-based security for IoT infrastructure," *IEEE Wireless Communications*, vol. 23, pp. 44–51, 2016.

[6] F. I. Kandah, O. Nichols, and L. Yang, "Efficient key management for Big Data gathering in dynamic sensor networks," in *Proc. 2017 International Conference on Computing, Networking and Communications (ICNC)*, 2017.

[7] M. Neily, F. Jemili, and O. Korbaa, "Enhancing intrusion detection in IoT systems through simulated attack scenarios," Research Square Preprint, 2024. doi: 10.21203/rs.3.rs-4306514/v1

[8] J. Gómez, J. F. Huete, O. Hoyos, L. Perez, and D. Grigori, "Interaction system based on internet of things as support for education," *Procedia Computer Science*, vol. 21, pp. 132–139, 2013.

[9] P. K. Singh, S. K. Jha, S. K. Nandi, and S. Nandi, "ML-based approach to detect DDoS attack in V2I communication under SDN architecture," in *Proc. TENCON 2018–2018 IEEE Region 10 Conference*, 2018.

[10] Y. Zeng, M. Qiu, D. Zhu, Z. Xue, J. Xiong, and M. Liu, "DeepVCM: A deep learning based intrusion detection method in VANET," in *Proc. 2019 IEEE 5th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)*, 2019.

[11] A. Qaddos, M. U. Yaseen, and A. S. Al-Shamayleh *et al.*, "A novel intrusion detection framework for optimizing IoT security," *Sci. Rep.*, vol. 14, no. 21789, 2024. doi: 10.1038/s41598-024-72049-z

[12] B. R. Kikissagbe and M. Adda, "Machine learning-based intrusion detection methods in iot systems: A comprehensive review," *Electronics*, vol. 13, no. 3601, 2024. doi: 10.3390/electronics13183601

[13] A. Amouri, M. Al Rahhal, Y. Bazi, I. Butun, and I. Mahgoub, "Enhancing intrusion detection in iot environments: an advanced ensemble approach using kolmogorov-arnold networks," arXiv preprint, arXiv:2408.15886, 2024.

[14] J. Shen, W. Yang, Z. Chu, J. Fan, D. Niyato, and K. Lam, "Effective intrusion detection in heterogeneous iot networks via ensemble knowledge distillation-based federated learning," arXiv preprint, arXiv: 2401.11968, 2024.

[15] W. Li, A. Joshi, and T. Finin, "SVM-CASE: An SVM-based context aware security framework for vehicular ad-hoc networks," in *Proc. 2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall)*, 2015.

[16] J. Grover, N. K. Prajapati, V. Laxmi, and M. S. Gaur, "Machine learning approach for multiple misbehavior detection in VANET," in *Proc. Advances in Computing and Communications: First International Conference, ACC 2011,* Kochi, India, July 22–24, 2011, Part III 1, pp. 644–653.

[17] F. A. Alhaidari and A. M. Alrehan, "A simulation work for generating a novel dataset to detect distributed denial of service attacks on vehicular Ad hoc network systems," *International Journal of Distributed Sensor Networks,* vol. 17, no. 3, 15501477211000287, 2021.

[18] S. So, P. Sharma, and J. Petit, "Integrating plausibility checks and machine learning for misbehavior detection in VANET," in *Proc. 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2018.

[19] Y. Gao, H. Wu, B. Song, Y. Jin, X. Luo, and X. Zeng, "A distributed network intrusion detection system for distributed denial of service attacks in vehicular ad hoc network," *IEEE Access,* vol. 7, pp. 154560–154571, 2019.

[20] K. M. Ali Alheeti and K. McDonald-Maier, "Intelligent intrusion detection in external communication systems for autonomous vehicles," *Systems Science & Control Engineering,* vol. 6, pp. 48–56, 2018.

[21] M. Jouhari and M. Guizani, "Lightweight CNN-BiLSTM based intrusion detection systems for resource-constrained IoT devices," arXiv preprint, arXiv: 2406.02768, 2024. doi: 10.48550/arXiv.2406.02768

[22] M. Rahman, S. Al Shakil, and M. R. Mustakim, "A survey on intrusion detection system in IoT networks," *Cyber Security and Applications*, vol. 3, 100082, 2025. doi: 10.1016/j.csa.2024.100082

[23] V. Kantharaju, H. Suresh, and M. Niranjanamurthy *et al.*, "Machine learning-based intrusion detection framework for detecting security attacks in the internet of things," *Sci. Rep.*, vol. 14, no. 30275, 2024. doi: 10.1038/s41598-024-81535-3

[24] Q. A. Al-Haija and A. Al-Badawi, "Attack-aware IoT network traffic routing leveraging ensemble learning," *Sensors*, vol. 22, no. 1, 241, Dec. 2022. doi: 10.3390/s22010241

[25] Q. Abu Al-Haija, A. Al Badawi, and G. R. Bojja, "Boost-defence for resilient iot networks: A head-to-toe approach," *Expert Systems*, vol. 39, no. 3, e12934, Jan. 2022. doi: 10.1111/exsy.12934

[26] Z. Wang, J. Li, S. Yang, X. Luo, D. Li, and S. Mahmoodi, "A lightweight IoT intrusion detection model based on improved BERT-of-Theseus," *Expert Syst. Appl.*, vol. 238, Part F, 122045, 2024. doi: 10.1016/j.eswa.2023.122045

[27] M. Neily, F. Jemili, and O. Korbaa, "DoS attack simulation for intrusion detection," in *Proc. 2023 IEEE Afro-Mediterranean Conference on Artificial Intelligence (AMCAI)*, 2023, pp. 1–6. doi: 10.1109/AMCAI59331.2023.10431522

[28] R. Meddeb, B. Triki, F. Jemili, and O. Korbaa, "Uncertainty-based data collection in mobile Ad-Hoc networks," *International Journal of Computer Information Systems & Industrial Management Applications,* vol. 12, 2020.

[29] T. Shao, D. Chowdhury, S. S. Gill, and R. Buyya, "IoT-pi: A machine learning-based lightweight framework for cost-effective distributed computing using IoT," *Internet Technology Letters,* vol. 5, e355, 2022.

[30] G. Kecskemeti, G. Casale, D. N. Jha, J. Lyon, and R. Ranjan, "Modelling and simulation challenges in internet of things," *IEEE Cloud Computing,* vol. 4, pp. 62–69, 2017.

[31] J. Beilharz, P. Wiesner, A. Boockmeyer, F. Brokhausen, I. Behnke, R. Schmid, L. Pirl, and L. Thamsen, "Towards a staging environment for the internet of things," in *Proc. 2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, 2021.

[32] G. Brambilla, M. Picone, S. Cirani, M. Amoretti, and F. Zanichelli, "A simulation platform for large-scale internet of things scenarios in urban environments," in *Proc. the First International Conference on IoT in Urban Space*, 2014.

[33] C. Cervantes, D. Poplade, M. Nogueira, and A. Santos, "Detection of sinkhole attacks for supporting secure routing on 6LoWPAN for internet of things," in *Proc. 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2015.

[34] S. Gönen, M. A. Barışkan, G. Karacayılmaz, and B. Alhan, "A novel approach to prevention of hello flood attack in IoT using machine learning algorithm," *El-Cezeri*, vol. 9, no. 4, pp. 1529–1541, November 2022.

[35] A. O. Bang and U. P. Rao, "EMBOF-RPL: Improved RPL for early detection and isolation of rank attack in RPL-based internet of things," *Peer-to-Peer Networking and Applications,* vol. 15, pp. 642–665, 2022.

[36] S. Sharma and V. K. Verma, "Security explorations for routing attacks in low power networks on internet of things," *The Journal of Supercomputing,* vol. 77, pp. 4778–4812, 2021.

[37] N. F. Syed, Z. Baig, A. Ibrahim, and C. Valli, "Denial of service attack detection through machine learning for the IoT," *Journal of Information and Telecommunication,* vol. 4, pp. 482–503, 2020.

[38] F. Abbasi, M. Naderan, and S. E. Alavi, "Anomaly detection in Internet of Things using feature selection and classification based on logistic regression and artificial neural network on N-BaIoT

dataset," in *Proc. 2021 5th International Conference on Internet of Things and Applications (IoT)*, 2021.
[39] P. Kaur, R. Kumar, and M. Kumar, "A healthcare monitoring system using random forest and Internet of Things (IoT)," *Multimedia Tools and Applications,* vol. 78, pp. 19905–19916, 2019.
[40] M. A. Ferrag, L. Maglaras, A. Ahmim, M. Derdour, and H. Janicke, "Rdtids: Rules and decision tree-based intrusion detection system for internet-of-things networks," *Future Internet,* vol. 12, 44, 2020.
[41] M. Hosseinzadeh, J. Koohpayehzadeh, A. O. Bali, P. Asghari, A. Souri, A. Mazaherinezhad, M. Bohlouli, and R. Rawassizadeh, "A diagnostic prediction model for chronic kidney disease in internet of things platform," *Multimedia Tools and Applications,* vol. 80, pp. 16933–16950, 2021.
[42] S. Sadowski, P. Spachos, and K. N. Plataniotis, "Memoryless techniques and wireless technologies for indoor localization with the internet of things," *IEEE Internet of Things Journal,* vol. 7, pp. 10996–11005, 2020.
[43] A. Kishor and W. Jeberson, "Diagnosis of heart disease using internet of things and machine learning algorithms," in *Proc.* the *Second International Conference on Computing, Communications, and Cyber-Security: IC4S 2020*, 2021.