Dicle University
Journal of Engineering
https://dergipark.org.tr/tr/pub/dumf
duje.dicle.edu.tr

Research Article

Detection of RPL-based Routing Attacks Using Machine Learning Algorithms

Burak AYDIN[1*], Hakan AYDIN[2], Sedat GÖRMÜŞ[3], Emin MOLLAHASANOĞLU[4]

[1] Karadeniz Technical University, Computer Engineering Department, burakaydin@ktu.edu.tr, Orcid No: 0000-0002-5377-4590
[2] Karadeniz Technical University, Computer Engineering Department, hakanaydin@ktu.edu.tr, Orcid No: 0000-0002-4057-3693
[3] Karadeniz Technical University, Computer Engineering Department, sedatgormus@ktu.edu.tr, Orcid No: 0000-0003-2172-2144
[4] Karadeniz Technical University, Electrical and Electronics Engineering Department, emin.mollahasanoglu@ktu.edu.tr, Orcid No: 0000-0001-7029-1956

ARTICLE INFO

Article history:

Received 27 May 2024
Received in revised form 14 November 2024
Accepted 25 November 2024
Available online 23 December 2024

Keywords:

Internet of things, RPL, Machine learning, IDS, Routing attacks

Doi: 10.24012/dumf.1490367

* Corresponding author

ABSTRACT

This study analyzes various machine learning techniques for detecting attacks against Routing Protocol for Low-Power and Lossy Networks (RPL), a routing protocol commonly used in Internet of Things (IoT) applications. RPL is often employed in IPv6-based IoT applications that require low power consumption and limited bandwidth. The research reviews recent literature examining attacks on RPL-based networks and utilizes the ROUT-4-2023 dataset for detecting routing attacks. This dataset, created using the Cooja simulator, encompasses four types of routing attacks: Blackhole Attack, Flooding Attack, DODAG Version Number Attack, and Decreased Rank Attack. The attack types are detected using machine learning techniques. In the combined dataset, the Decision Tree and Bagging algorithm exhibited the highest performance with a 99.99% accuracy. To create a more accurate representation of the real world, we incorporate a 10% level of noise into the dataset. On the noisy dataset, Random Forest algorithm performed the best with about 84.80% accuracy. The high accuracy show that the employed methods can be effectively used as an Intrusion Detection System (IDS) to protect IoT networks. As a result, this study demonstrates that machine learning techniques offer a promising approach for detecting routing attacks in the RPL protocol.

Introduction

The concept of the Internet of Things (IoT) refers to the idea of various devices interacting and communicating with each other over the internet to share data. This concept has become highly significant today [1]. As shown in Figure 1, this approach is used in a diverse array of applications, from industrial systems to smart homes, from healthcare services to the agricultural sector [2]. However, this rapid growth and widespread use also introduce significant security challenges for IoT devices and networks. The increasing number of connected devices expands the attack surface, making it crucial to address vulnerabilities and protect sensitive information from potential threats.

Security vulnerabilities in IoT networks can arise from various sources, including weak authentication mechanisms, insufficient encryption, and lack of regular software updates. These security weaknesses can be exploited, potentially leading to unauthorized access, data breaches, and other harmful intrusions. Therefore, implementing robust security measures is essential to safeguarding IoT environments.

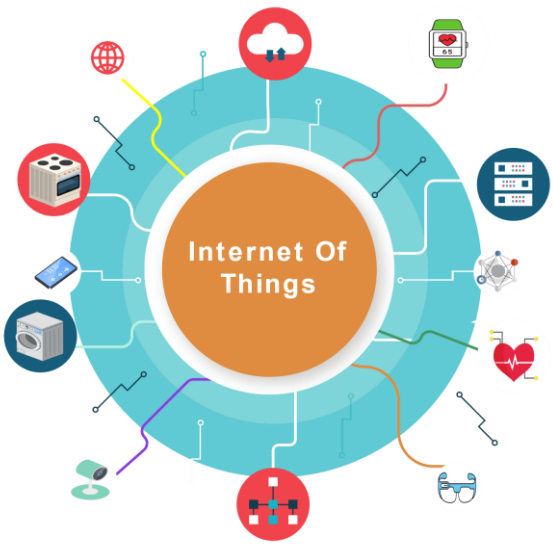

Figure 1. Internet of things domains of use

Given the diverse applications and the critical nature of IoT deployments, ensuring secure and efficient communication is paramount. Among the routing protocols commonly used in IoT networks, the Routing Protocol for Low-Power and Lossy Networks (RPL) stands out [3]. RPL aims to optimize communication in wireless sensor networks while ensuring low power consumption for IoT devices. Nevertheless, security vulnerabilities and attacks in RPL-based networks present a major concern. Intrusion Detection Systems (IDS) become an important tool for detecting and preventing attacks in RPL-based IoT networks. Machine learning-based IDS can provide more effective protection through methods such as network traffic analysis and identifying behavioral patterns [4]. Such systems can protect against known attacks by using predefined attack signatures and can also detect new and unknown attacks.

Applying machine learning in IoT not only enhances the performance of IoT systems but also allows for more efficient management of unpredictable processes. Moreover, it increases the effectiveness of these systems by reducing human intervention and eliminating the challenges of reprogramming [5].

The motivation of this study is to investigate the effects of using (performance of) machine learning algorithms to detect routing attacks in RPL-based IoT networks. In this scope, the ROUT-4-2023 [6] dataset representing IoT network scenarios with diverse attack types has been utilized for the first time to analyze the performance of various algorithms, including AdaBoost, KNN, Random Forest, Decision Tree, Bagging, Logistic Regression, Gaussian NB, Gradient Boosting, Extra Trees, XGBoost, ANN, and CNN. This study aims to analyze the ROUT-4-2023 dataset using machine learning and ANN models to assess their effectiveness in detecting attacks within the dataset. The ROUT-4-2023 dataset, a comprehensive representation of real-world IoT network scenarios with various attack types, serves as the basis for comparing the capabilities of different algorithms in identifying routing attacks. Through this evaluation, we aim to determine the impact of machine learning and neural network models on detecting and analyzing these attacks effectively.

The remainder of this study is organized as follows: Section RPL introduces the RPL routing protocol, detailing its operational mechanisms, core functionalities, and significance in IoT networks. This section also highlights the features that make RPL suitable for low-power and lossy networks. Section Literature reviews related work, analyzing previous studies on RPL-based IoT networks, including methodologies, findings, and limitations. Section Related work focuses on approaches for detecting routing attacks in RPL networks. It discusses various machine learning algorithms used for IoT security, their strengths, and weaknesses. Section Test results present the machine learning test results using the ROUT-4-2023 dataset, detailing the experimental setup, data pre-processing, and performance evaluation criteria. Section Discussion analyzes the results in relation to the dataset characteristics and algorithm performance, comparing machine learning and deep learning models in the context of IoT security needs and computational efficiency. Finally, Section Conclusion summarizes the key contributions and findings, discusses their significance in IoT security, and suggests potential directions for future research.

RPL

Routing Protocol for Low-Power and Lossy Networks is a routing protocol specifically designed for wireless sensor networks. This protocol is of great importance, especially for applications such as Internet of Things and Smart Grid. The goal of the routing protocol is to ensure energy efficiency, low latency, and reliable communication in wireless sensor networks [7]. One of the main goals of the RPL protocol is to route data packets between nodes with low energy consumption. Thus, the battery life of the nodes in the network is extended and energy resources are used more efficiently [8].

RPL creates and manages the network topology at the node level to enable wireless sensor networks to efficiently transmit data. The protocol can create and manage various topology structures, for example, using DODAGs (Destination Oriented Directed Acyclic Graphs), which are hierarchical structures at the node level to ensure efficient data transmission. The routing protocol aims to transmit data packets with low latency and high reliability while minimizing the energy consumption of nodes in the network.

The main features of RPL include the use of node routing tables, the creation of network topology through inter-node routing messages, the mutual exchange of information between nodes, and the ability to determine the functions of nodes (e.g. root, parent, and leaf nodes).

Figure 2 shows the structure for creating DODAGs with RPL:

- DIS (DODAG Information Solicitation) is a type of message that nodes use to request DODAG information. When a node wants to learn or update its DODAG structure, it can request DIO messages from neighboring nodes by broadcasting the DIS message. DIS messages allow nodes in the network to dynamically discover and update the DODAG structure.
- DIO (DODAG Information Object) is one of the message types used in the RPL protocol. DIO messages are used for nodes in the network to create and update the DODAG structure. DIO messages broadcast by the root node at regular intervals enable other nodes to learn and update the DODAG structure. DIO messages contain neighboring nodes and routing metrics (e.g., link quality, latency) of nodes in the network. This information helps nodes determine the best routing path.
- DAO (Destination Advertisement Object) is another type of message used in the RPL protocol. DAO messages allow nodes to notify neighboring nodes of their presence and services, if any. With

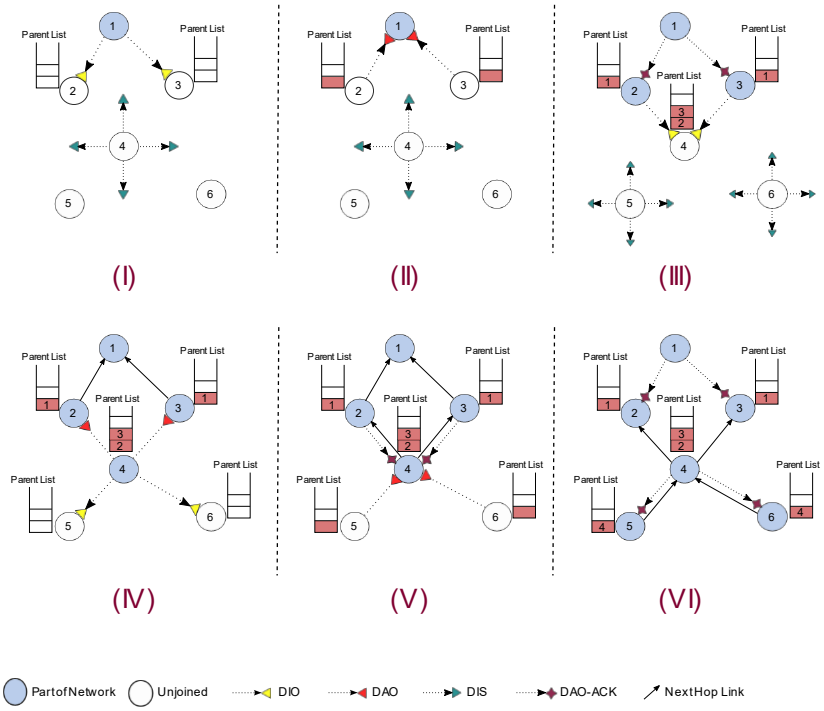

Figure 2. Structure of building DODAG with RPL

DAO messages, a node introduces itself to its parent nodes and indicates that it wants to join the network. The parent nodes receive the DAO messages, update the network structure, and include the node in the DODAG structure. DAO messages can also contain the node's child nodes. This message enables the creation of downward (root to node) routing paths.

- DAO-ACK (DAO Acknowledgment) message type is utilized to confirm the receipt of DAO messages. Once a node transmits a DAO message to its parent node, it awaits a DAO-ACK message to confirm the successful transmission and its inclusion in the DODAG structure.
- Parent List is used to maintain node parents where a default parent is used as the default router. A node can forward data packets to its default parent node, which creates a cascading connection to the Root node of the RPL topology. The Parent List helps nodes make routing decisions and determine the best path.

The common use of RPL in various application areas, such as IoT and WSN has also increased the security threats in these networks. As RPL continues to be widely adopted, the prevalence of security vulnerabilities has grown, necessitating a more robust approach to safeguarding these networks. Various attacks on RPL-based networks can compromise both the efficiency and the security of the entire network infrastructure. These attacks are particularly concerning as they have the potential to disrupt critical applications and services reliant on IoT and WSN technologies. Additionally, these attacks can culminate in node denial of service, where nodes are rendered inoperative, thereby severely degrading network performance and reliability. The taxonomy of attacks against RPL networks is depicted in Figure 3. This figure categorizes the various attack types, illustrating the broad spectrum of threats that can target RPL-based networks.

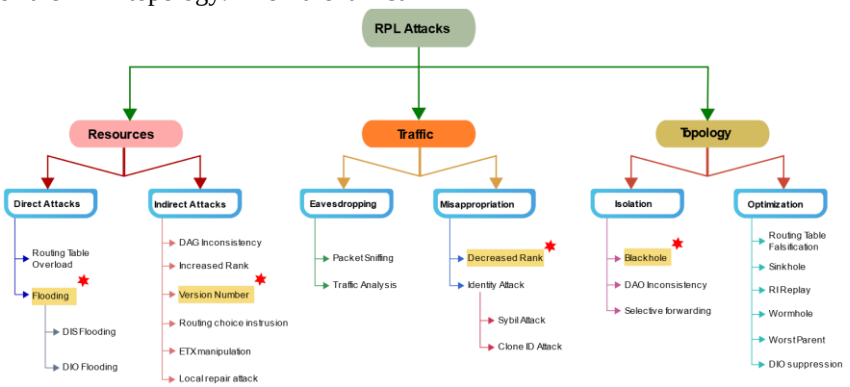

Figure 3. Taxonomy of attacks against RPL networks

The specific attack types examined in this study are clearly marked and highlighted within the figure, providing a visual representation of the focus areas of our research. By identifying and categorizing these threats, we can better understand the potential vulnerabilities and develop more effective countermeasures to protect RPL-based IoT and WSN networks.

Literature

In recent years, with the rapid deployment of IoT devices, attacks on RPL-based networks have increased. Many research and studies have been conducted to detect and prevent these attacks. These studies have been conducted to identify the security vulnerabilities of RPL-based networks, analyze attack patterns, and develop effective defense mechanisms [10]. The results of these studies provide an important resource for understanding the vulnerabilities of RPL-based networks and taking advanced security measures. Table 1 provides summaries of these studies and provides guidance to researchers and network administrators on attack detection and prevention.

In 2019, Verma et al. introduce ELNIDS (Ensemble Learning-based Network Intrusion Detection System) to detect various routing attacks against IPV6 routing protocols [11]. The ELNIDS system is specifically created to identify and identify seven different forms of routing attacks, including Sinkhole (SH), BlackHole (BH), Sybil, Clone ID (CID), Selective Forwarding (SF), Hello Flooding (HF), and Local Repair. This detection is done using the

NIDDS17 dataset, as mentioned in the paper [22]. The ELNIDS methodology is founded on the utilization of ensemble learning, which involves the amalgamation of diverse machine learning classifiers. The researchers utilize four distinct machine learning classifiers: Boosted Trees, Subspace Discriminant, RUSBoosted Tree, and Bagged Trees [23], [24]. The authors assess the individual performance of each classifier applying distinct evaluation and validation measures. Subsequently, they use an ensemble model, namely a voting scheme, to enhance the classification outcomes. Nevertheless, the process of constructing intricate ensemble models may entail significant computational expenses, rendering the system unfeasible.

In 2019, Aydogan et al. propose a new technique for identifying HF and VN attacks in RPL-based IIoT networks. They utilized genetic programming and suggested that this approach is the most appropriate framework for IIoT environments [12]. Their proposed technique uses Genetic Programming in conjunction with a centralized IDS to detect and identify threats. The implemented framework is installed at the main node to oversee the packets of the neighboring nodes. The root consistently analyzes the network data and collects 50 characteristics, which are subsequently utilized to construct genetic programming trees. The top-performing individual from the previous generation is assessed for both HF and DODAG VN attacks, resulting in the development of two detection algorithms. The values are acquired by gathering data at successive intervals of 500 ms and 5000 ms. The

Table 1. Comparative analysis of detection methods for various RPL-based routing attacks

Suggested	Types of Attacks	Methods Used
Verma et al. [11]	Sinkhole, BlackHole, Sybil, Clone ID, Selective Forwarding, Hello Flooding, Local Repair	Boosted Trees, Subspace Discriminant, RUSBoosted Tree, Bagged Trees
Aydogan et al. [12]	Version Number, Hello Flooding, Sinkhole	Genetic programming for attacks
Deshmukh-Bhosale et al. [13]	Wormhole, Hello Flooding	Routing information and received signal strength indicator
Farzaneh et al. [14]	Local Repair Attack	Fuzzy Logic
Agiollo et al. [15]	Sinkhole, Wormhole, Hello Flooding, Version Number, Clone ID, Local Repair, DIS, Selective Forwarding	Signature and behavior-based detection rule
Garcia Ribera et al. [16]	Hello Flood Attack, DIS Attack, DAO Insider Attack, Blackhole Attack, Greyhole Attack	a Hybrid IDS
Alazab et al. [17]	Blackhole, Selective Forwarding, Sybil Attacks	a Decision Tree Classifier and a One-class Support Vector Machine Classifier (a Hybrid Intrusion Detection System)
Azzaoui et al. [18]	CIC-IDS2017 Dataset (DoS, DDoS, Brute Force, XSS, SQL Injection, Infiltration, Portscan, Botnet), WSN-DS Dataset (Blackhole, Grayhole, Flooding, Scheduling)	a Lightweight Artificial Neural Network
Bokka et al. [19]	Blackhole, Sybil, Selective Forwarding, Sinkhole, DIO Suppression, DIS Flooding	analysis
Kiran et al. [20]	DODAG Information Solicitation (DIS) Attack, Version Number Attack, Decreased Rank Attack, Worst Parent Selection (WPS) Attack	analysis
Osman et al. [21]	Version Number, Decreased Rank, DIS Flooding Attacks	Ensemble Learning-based Intrusion Detection System (ELG-IDS)
Our work	Blackhole Attack, Flooding Attack, DODAG Version Number Attack, Decreased Rank Attack	AdaBoost, KNN, Random Forest, Decision Tree, Bagging, Logistic Regression, Gaussian NB, Gradient Boosting, Extra Trees, XGBoost, ANN, CNN

researchers only examine the accuracy of genetic programming in detecting intrusions and reducing false positives. They do not explore how intrusion techniques affect performance metrics like energy consumption and memory usage. In addition, the evaluation of the system does not consider factors such as resource requirements, scalability, extensibility, and mobility support.

Deshmukh-Bhosale et al. present a real-time approach for identifying wormhole and HF attacks in RPL-based IoT networks [13]. Their IDS utilizes the collected signal strength to detect nodes that may be engaging in suspicious activities. Both centralized and distributed IDS are suggested. The findings indicate that the accuracy of correctly identifying positive cases is 90% when the network is small, consisting of 8, 16, or 24 nodes. However, the performance of the method declines noticeably as the network size grows. In addition, the authors solely assess the impact of IDS on a solitary metric and a solitary attack. Furthermore, they disregard the typical issues of mobility associated with networks and fail to acknowledge that the offered solution relies on static topological information.

Farzaneh et al. conduct a study utilizing Fuzzy logic to detect local repair attacks on IoT networks, specifically targeting the RPL routing protocol. The study demonstrates that the suggested strategy, implemented using the Contiki operating system on the Cooja simulator, effectively detects local repair attacks with a significantly high True Positive Rate (TPR). The authors of this study assert that the proposed approach is a highly efficient means of identifying security risks in IoT networks that utilize the RPL protocol [14].

Agiollo et al. introduce the DETONAR model in RPL, which integrates the strengths of anomaly and signature-based IDS models to effectively identify malicious activities in network traffic [15]. This approach utilizes a mechanism known as the 'Detector', which employs signature and behavior-based detection algorithms to identify 14 distinct routing assaults. Additionally, the intrusion detection systems are assessed using the 'Routing Attacks Dataset' (RADAR). Nevertheless, their strategy fails to account for the ever-changing nature of the IoT network. Furthermore, the dataset might have been examined with regards to the privacy and security of Quality of Service (QoS).

Garcia Ribera et al. develop an IDS for IoT networks utilizing the RPL routing protocol. Their objective is to precisely and effectively identify many sorts of assaults, including different routing and denial-of-service (DoS) attacks. The attacks mentioned involve targeted attacks against RPL, such as the version number attack, blackhole attack, and grayhole attack. The effectiveness of the proposed IDS is assessed by a comprehensive analysis of the specified assaults and their consequences. This evaluation also encompasses the ability to forecast the performance of the IDS in terms of its accuracy and effectiveness when confronted with the predetermined attacks. The results obtained demonstrate a high level of accuracy in detection. Moreover, the detected attacks have been determined to have a negligible impact on CPU utilization and power consumption. More precisely, the rise in CPU use is below 2% in every instance, while the average increase in power consumption is limited to 0.5% [16].

In 2023, a study is undertaken by Ammar Alazab et al. that specifically examined the security difficulties associated with routing attacks that target RPL in IoT systems based on 6LoWPAN. The assaults encompass Flooding assaults, Data-DoS/DDoS Attacks, Wormhole Attacks, RPL Rank Attacks, Blackhole Attacks, Version Attacks, and Sinkhole Attacks. To combat these dangers, they suggest implementing a pioneering Hybrid Intrusion Detection System (HIDS) that merges a decision tree classifier with a class Support Vector Machine classifier. The objective of this hybrid technique is to attain a high rate of detecting intrusions while keeping the incidence of false alarms low. This is accomplished by utilizing the advantages of both Signature Intrusion Detection Systems (SIDS) and Anomaly-based Intrusion Detection Systems (AIDS). Experiments conducted on a dataset of network traffic from real IoT devices, including different types of routing attacks, have demonstrated that their suggested Host-based Intrusion Detection System (HIDS) works better than conventional System-based Intrusion Detection Systems (SIDS) and Anomaly-based Intrusion Detection Systems (AIDS). The HIDS exhibits greater detection abilities and a reduced rate of false positives compared to the traditional techniques [17].

In 2024, Azzaoui et al. propose a streamlined cross-layer IDS approach for the IoT called RPL-IDS. This approach utilizes the RPL routing protocol and employs certain routing parents as artificial neural network (ANN) based mitigation agents. The researchers integrate RPL-IDS into the Contiki operating system and conducted a comprehensive evaluation using the Cooja simulator. The empirical findings demonstrate that RPL-IDS is efficient and can be implemented on devices with restricted resources. The work involves the construction of an IDS scheme that use a machine learning technique and may be implemented on devices with limited resources. Additionally, the results demonstrate that RPL-IDS exhibits superior detection rates in comparison to several IDS schemes documented in the literature. Furthermore, the energy overhead associated with RPL-IDS is nearly insignificant [18].

In 2024, Bokka et al. introduce a study on the effects of RPL routing attacks, which cause disruptions to the regular routing operations and structure of IoT networks. The study assessed the network's performance under normal conditions and five different routing attack scenarios, utilizing a range of performance measures. The metrics encompass Link data rate, Number of packets generated (control and data), Sensor data rate, Packet Delivery Rate (PDR), and Packet delivery delay. The text emphasizes the susceptibility of IoT devices connected via the RPL protocol to routing attacks and underscores the significance of protecting them to thwart both internal and external attacks. The study, which utilizes simulation, demonstrates the crucial importance of robust security measures in

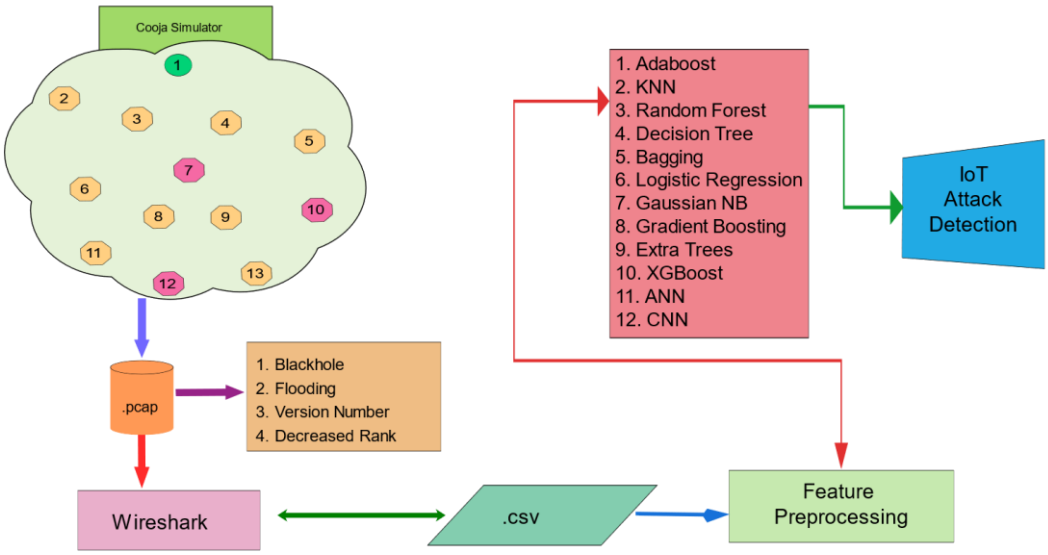

Figure 4. Cooja attack models and feature extraction diagram

safeguarding IoT networks and devices against routing attacks [19].

Kiran et al. examine the effects of several RPL attacks, including DIS attack, version number attack, decreased rank attack, and worst parent selection (WPS) attacks. The Contiki Cooja network simulator is utilized for the purpose of simulation. In addition, a thorough examination of various attacks based on RPL is conducted, revealing that the WPS attack has a substantial influence on network performance in comparison to other assaults [20].

In 2024, Osman et al. present an Ensemble Learning-based Intrusion Detection System (ELG-IDS) using stacking and over-parameter optimization to detect three RPL internal attacks (version number, decreased rank, and DIS flooding attacks). ELG-IDS uses improved feature extraction and genetic algorithm-based feature selection. Their research improves the security of IoT networks by using ELG-IDS and provides better defense against emerging security threats [21].

Upon examining the presented studies, it is emphasized that RPL-based IoT networks face serious security threats, and this underscores the importance of implementing effective protection measures against these threats. Especially considering the limited resources and the impact of attacks on network performance, strong security controls need to be implemented. These studies show the necessity of machine learning methods, new approaches, and technologies to improve the security of IoT networks.

Related Work

Description of the ROUT-4-2023 dataset

The study uses the ROUT-4-2023 dataset, which is made up of ".csv" files with four different routing attacks against the RPL protocol: the Blackhole Attack (blackhole.csv), the Flooding Attack (flooding.csv), the DODAG Version Number Attack (dodag.csv), and the Decreased Rank Attack (rank.csv). It can see these attacks in Table 2. The attack type is given in the Category header. Malicious or normal status is kept as Label. Additionally, as shown in Table 3, the dataset contains 16 features.

Table 2. Number of RPL-based routing attacks

Category	Malicious	Normal	Total
Blackhole Attack	134282	269852	404134
Flooding Attack	135576	263206	398782
DODAG Version Number Attack	170242	297818	468060
Decreased Rank Attack	139891	229108	368999
Total	579991	1059984	1639975

Table 3. Features and descriptions in the ROUT-4-2023 dataset

Name/Abbreviation	Description
TIME	Simulation time
SOURCE	Source Node IP
DESTINATION	Destination Node IP
LENGTH	Packet Length
INFO	Packet Information
TR	Transmission Rate (per 1000 ms)
RR	Reception Rate (per 1000 ms)
TAT	Transmission Average Time
RAT	Reception Average Time
TPC	Transmitted Packet Count (per second)
RPC	Received Packet Count (per second)
TTT	Total Transmission Time
TRT	Total Reception Time
DAO	DAO Packet Count
DIS	DIS Packet Count
DIO	DIO Packet Count

Figure 4 illustrates the sequence of steps involved in simulating attacks on IoT networks and extracting features for analysis using machine learning. The Cooja simulator is utilized to expose a network to different attacks, and the resulting network traffic is recorded in pcap files. The files undergo analysis using Wireshark and are subsequently converted to CSV format for feature preprocessing, which encompasses data cleaning and normalization. The processed data is subsequently utilized to train multiple machine learning models for the purpose of identifying and

categorizing IoT attacks, culminating in the development of an IoT attack detection model.

Types of attacks

Blackhole attack

It is a type of attack that targets data transmission on the network. In this attack, attackers receive messages, but instead of forwarding them to the destination, they destroy them or redirect them to their own devices. Thus, other devices on the network realize that their messages did not reach the destination and experience communication problems [25]. When the Blackhole Attack is carried out by an attacker in a strategic position in the network, it can isolate many nodes from the network and significantly impact network performance [26]. An example Blackhole Attack scenario is shown in Figure 5.

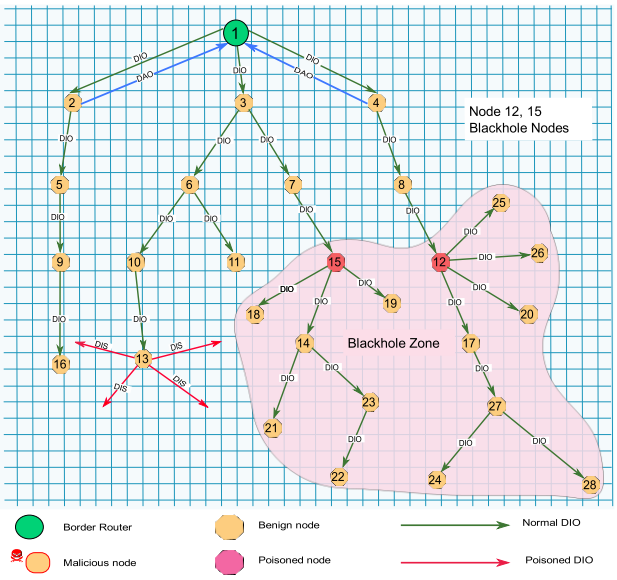

Figure 5. RPL blackhole attack

Flooding attack

It is a type of attack that involves transmitting large amounts of spoofed data to the network to consume communication resources on the network or render target devices non-functional. In this attack, attackers typically send large amounts of traffic to the network at a fast rate, consuming network resources or exceeding the capacity of the target device. Therefore, it becomes difficult for legitimate data packets to be processed and reach their destination. Flooding Attack can be used to damage the network by disrupting or disabling network services [27]. The HELLO Flood attack is a type of Flood attack using the RPL protocol on a network. In this attack, the attacker Floods the network using DIS messages, which are request messages. In RPL networks, the attacker can perform the HELLO Flood attack either by broadcasting DIS messages to neighboring nodes that should reset the trickle timer, or by sending a unicast DIS message to a node that should respond to a DIO message [26]. Figure 6 shows an example of Flooding Attack scenario.

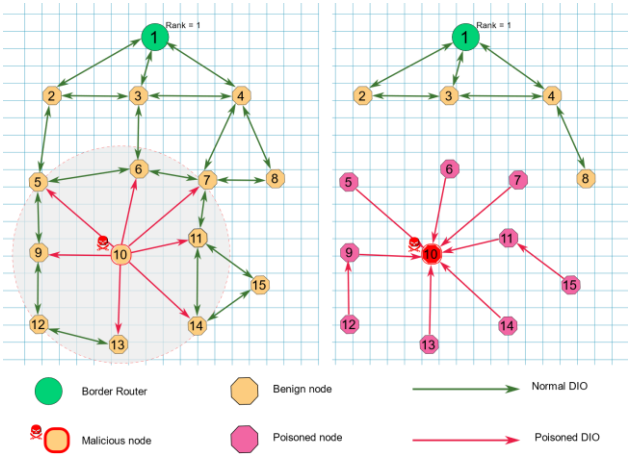

Figure 6. RPL flooding attack

DODAG version number attack

This attack specifically focuses on exploiting version numbers within the Directed Acyclic Graph utilized in the RPL protocol. The DODAG Version Number is a crucial component of every DIO transmission. This value remains constant as it moves down the DODAG graph and is often only increased by the root node when the DODAG requires reconstruction. A node with an outdated version number signifies that it has not been transferred to the new DODAG graph and is not suitable for usage as a parent node. A malicious actor can modify this field by unlawfully increasing the version number of DIO messages as it transmits them to neighboring nodes. This form of attack has the potential to result in the needless reconstruction of the entire DODAG graph. During this attack, the perpetrators can control the routing processes in the network by substituting the version numbers in DODAG with counterfeit or altered values [26], [28]. Figure 7 shows an example DODAG Version Number Attack scenario.

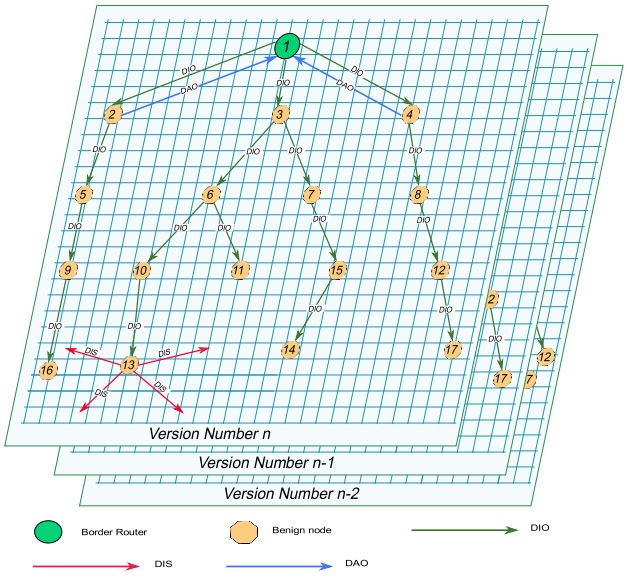

Figure 7. RPL DODAG version number attack

Decreased rank attack

This attack is carried out by manipulating the ranks in the routing tables of resource-constrained devices in the RPL network. As a result of these manipulations, devices with low Rank become favored over other devices in the network. By targeting devices with this low Rank, attackers can direct the data traffic in the network as they wish. The lower the rank in the DODAG graph, the closer the node is to the root and the more traffic that node has to manage. As a result, many legitimate nodes connect to the DODAG network through the attacker. In the RPL protocol, an attacker can change the ranking value through the tampering of DIO messages [26]. This attack results in the inability to perform correct routing operations and can prevent or delay data packets from reaching their destination [29]. Figure 8 shows an example of a decreased rank attack scenario.

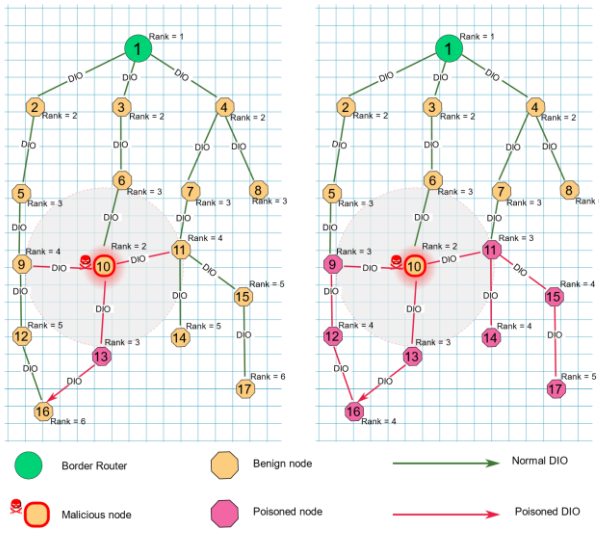

Figure 8. RPL decreased rank attack

Algorithms

This study utilizes a range of machine learning and deep learning algorithms to detect RPL-based routing attacks in IoT networks. The selected algorithms, spanning from simple to complex models, are chosen for their proven effectiveness in classification tasks. By evaluating models such as AdaBoost, K-Nearest Neighbors (KNN), Decision Tree, Random Forest, Bagging, Logistic Regression, Gaussian Naive Bayes (GaussianNB), Gradient Boosting, Extra Trees, and XGBoost, as well as artificial neural networks (ANN) and convolutional neural networks (CNN), we aim to cover a diverse set of approaches for comparison [30-33].

Tree-based models, such as Random Forest, Decision Tree, and Extra Trees, are particularly suitable for classification tasks due to their ability to handle complex feature interactions and provide high accuracy. Ensemble methods like Bagging, AdaBoost, and Gradient Boosting are included to examine their robustness and performance under different boosting strategies. Additionally, neural network models (ANN and CNN) are incorporated to assess their potential in learning complex patterns in the dataset. Logistic Regression and GaussianNB offer simpler, interpretable baseline models to compare against more complex algorithms.

The parameters used in these algorithms are carefully selected to optimize performance, and are listed below:

- AdaBoost was configured with a decision tree as its base classifier, and the number of weak classifiers was set to 100 to create a strong ensemble model. This number of estimators provides a balance between detection accuracy and computational efficiency, ensuring robust performance without excessive complexity.
- K-Nearest Neighbors (KNN), the number of neighbors was set to 1. By focusing on the single nearest neighbor, the model aims to increase precision in distinguishing between well-separated classes within the dataset, which helps improve classification outcomes in a simple yet effective way.
- Random Forest was also set to build 100 decision trees. With this configuration, the model gains both robustness and accuracy, as multiple trees reduce the variance in predictions, providing a stable output that generalizes well to new data. Using 100 trees strikes a balance by ensuring sufficient learning capacity without unnecessary computational demands.
- Decision Tree was used without specific hyperparameter tuning, as this model inherently splits data based on optimal feature values. This simplicity allows Decision Trees to perform well on their own, especially on datasets with clear, interpretable decision boundaries.
- Bagging, the number of estimators was also set to 100. This ensures stability and reduces the variance in predictions by averaging across multiple classifiers, each trained on different subsets of the data. This approach reinforces model robustness and improves accuracy, especially useful for data with potential noise.
- Logistic Regression, the maximum number of iterations was set to 100 to ensure that the model had ample opportunity to converge. This parameter controls the training duration, balancing adequate model tuning with computational efficiency.
- Gaussian Naive Bayes (GaussianNB) was used without specific parameter adjustments, as it relies on probability distributions and assumes feature independence, making it a straightforward yet effective choice for classification tasks in this study.
- Gradient Boosting was configured to include 100 boosting stages, which provides enough stages to gradually correct errors from previous models

while avoiding overfitting. This balanced approach enhances accuracy and helps the model perform better on complex data structures.

- Extra Trees, the number of trees was also set to 100, like Random Forest. This additional randomness in splitting nodes enhances generalization and further stabilizes the model's output by reducing prediction variance.
- XGBoost, the number of trees was set to 100, the learning rate to 0.1, and the maximum depth to 3. These settings ensure that each tree contributes gradually, allowing the model to learn incrementally and prevent overfitting. The depth limit of 3 helps maintain a simpler model that is computationally efficient and performs well on structured data.
- ANN uses the sgd (stochastic gradient descent) optimizer with an initial learning rate of 0.05. This configuration consists of two hidden layers with 64 and 32 nodes, respectively, and employs the ReLU activation function, which helps capture more complex patterns in the data. The adaptive learning rate allows efficient training by adjusting the learning rate as needed throughout the 500 training iterations. With a batch size of 32, this setup ensures a controlled and gradual learning process, making it well-suited for datasets where the complexity of decision boundaries benefits from a deeper network.
- For comparison purposes, this Convolutional Neural Network (CNN) model was tested to evaluate its effectiveness in binary classification of RPL-based routing attacks, leveraging both convolutional and fully connected layers to capture key patterns in the data. The architecture begins with a 1D convolutional layer with 32 filters and a kernel size of 2, using ReLU activation to extract local features from the input time series data. A subsequent max pooling layer with a pool size of 2 reduces dimensionality, focusing on the most relevant features while minimizing computation and overfitting. The output is flattened and passed through a dense layer of 50 neurons with ReLU activation, enabling the model to learn complex relationships. Finally, a single neuron with a sigmoid activation outputs a probability score for binary classification, and the model is compiled with the adam optimizer and binary_crossentropy loss function, ensuring effective training and accuracy in distinguishing attack instances from normal activity.

Test results

The tests are performed on the Visual Studio Code platform, Python version 3.12, AMD Ryzen 5 5500 3.60 GHz processor, and 32 GB RAM. Different files containing 4 different attack types in the ROUT-4-2023 dataset are merged. This merged file is analyzed.

In the process of feature selection, all 13 features, excluding "time, source, destination", are utilized for feature extraction. Because these parameters will not be meaningful for the nodes that may join the network. The dataset is randomly divided into training and test data using 5-fold cross-validation (k=5). The results are presented by averaging 5 different cross-validation iterations for each model.

Performance metrics

Explanation of the expressions used in the measurement parameters:

- True Positive (TP): This statistic corresponds to the number of instances in which the IDS accurately identified and detected attacks.
- True Negative (TN): Refers to the count of instances where the IDS accurately identified that no attacks were present.
- False Positive (FP): This metric quantifies the number of instances where an IDS erroneously identified an attack when no attack really occurred. This encompasses instances where IDS erroneously detected and reported an attack despite the absence of an actual attack.
- False Negative (FN): Refers to the instances where IDS incorrectly indicated that no attack had a place, despite the presence of an actual attack.
- Accuracy: Refers to the proportion of correctly classified instances. This metric represents the proportion of events to total events that IDS correctly detected. It is often the most fundamental metric used to evaluate the performance of a model. The calculation of Accuracy is given in Equation 1.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100 \quad (1)$$

- Recall: The ratio of true positives (predicted positives) to all positives. Also known as sensitivity. The calculation of Recall is given in Equation 2. Recall is an important metric that evaluates the ability of a model to accurately detect the positive class. A high recall value means that the model correctly detected most of the positive samples, while a low recall value means that the model missed some of the positive samples or classified them as false negatives.

$$Recall = \frac{TP}{TP + FN} \times 100 \quad (2)$$

Table 4. Performance Evaluation of Data

Algorithm	Accuracy (%)	Recall (%)	Precision (%)	F1 Score (%)	Training Time (s)	Prediction Time (s)	AUC	Log Loss
AdaBoost	81.5297	72.2550	74.6953	73.4403	84.5643	1.8047	0.7943	6.6574
KNN	99.8008	99.7338	99.7030	99.7184	3.5782	24.6636	0.9979	0.0718
Random Forest	99.9783	99.9678	99.9709	99.9693	113.0783	1.8342	0.9998	0.0078
Decision Tree	99.9920	99.9872	99.9899	99.9886	4.0960	0.0252	0.9999	0.0029
Bagging	**99.9945**	99.9907	99.9937	99.9922	294.4614	2.2958	0.9999	0.0020
Logistic Regression	72.4614	50.9102	63.8869	56.6651	1.0133	0.0078	0.6758	9.9258
Gaussian NB	71.7257	53.6537	61.4907	57.3055	0.2468	0.0546	0.6763	10.1910
Gradient Boosting	90.6458	90.6384	84.1389	87.2671	170.6360	0.2920	0.9064	3.3715
Extra Trees	99.9617	99.9405	99.9512	99.9458	58.3624	2.9720	0.9995	0.0138
XGBoost	89.9186	90.0350	82.9247	86.3333	1.5107	0.0531	0.8994	3.6336
ANN	95.6652	93.5362	94.1708	93.8456	3396.4120	0.1137	0.9518	1.5623
CNN	87.9658	80.0843	85.1612	82.4647	313.6185	7.0707	0.8618	4.3375

- Precision: The ratio of predicted positives to true positives (true positives and false positives). That is, the percentage of data points that the model predicts as positive is correct. The calculation of Precision is given in Equation 3.

$$Precision = \frac{TP}{TP + FP} \times 100 \quad (3)$$

- F1 Score: A measure that balances precision and recall. It is more useful for low precision and low-recall models. The calculation of F1 Score is given in Equation 4.

$$F1\,Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \times 100 \quad (4)$$

- Training Time: Represents the training time of the model in seconds. This is the time it takes to train the model on a given training dataset.
- Prediction Time: Represents the model's prediction time in seconds. That is, the time it takes the trained model to predict new data points.
- AUC: Represents the area under the Receiver Operating Characteristic (ROC) curve. This area is used to evaluate the classification performance of the model. The closer to 1, the better the performance of the model.
- Log Loss: Represents the logarithmic loss value. It measures how far the model's predictions are from the true labels. A lower log loss indicates better model performance. The calculation of Log Loss is given in Equation 5.

(y_i): True label (0 or 1)

($\hat{y}_i$): Estimated probability (between 0 and 1)

$$Log\ Loss = -\frac{1}{n}\sum_{i=1}^{n}(y_i\ log(\hat{y}_i) + (1 - y_i)\ log(1 - \hat{y}_i)) \quad (5)$$

Performance evaluation

Table 4 shows the performance of different machine learning algorithms. Performance metrics include Accuracy, Recall, Precision, F1 Score, training time, prediction time, AUC, and Log Loss.

The AdaBoost model provided an Accuracy of 81.53%. Recall and Precision are 72.25% and 74.70% respectively. The F1 score is 73.44%. The training time is 84.56 seconds, while the prediction time is 1.80 seconds. The AUC is 0.7943 and Log Loss is calculated as 6.6574.

KNN model has the highest accuracy rate with 99.80% accuracy rate. Recall is 99.73%, Precision is 99.70% and F1 score is 99.72%. Training time is 3.58 seconds and prediction time is 24.66 seconds. AUC is 0.9979 and Log Loss is 0.0718.

The Random Forest and Decision Tree algorithms have very high accuracy rates (99.98% and 99.99%). Other performance metrics are also very high. Training times increase depending on the number of decision trees.

The Bagging algorithm demonstrates the highest performance among the evaluated machine learning algorithms, achieving an exceptional accuracy of 99.99%. It also exhibits outstanding results in recall (99.99%), precision (99.99%), and F1 score (99.99%). These high values underscore the model's robustness and reliability in detecting and classifying RPL-based routing attacks. However, the Bagging algorithm's training time is significantly longer than the other algorithms, taking 294.46 seconds. This extended training duration is a trade-off for its superior accuracy and robustness. The algorithm's ability to aggregate multiple decision trees enhances its performance but also increases computational complexity and resource requirements. Despite this, the Bagging algorithm remains highly effective choice for scenarios where accuracy and robustness are paramount, and the available computational resources can accommodate the increased training time.

Logistic Regression yields a lower accuracy rate at 72.46%, with Recall at 50.91% and Precision at 53.69%. The model's F1 score stands at 56.67%, and training and prediction times are 1.01 seconds and 0.0078 seconds respectively, making it a fast but less reliable option for this context. AUC is 0.7373, and Log Loss is high at 9.9258.

Table 5. Performance Evaluation of Noisy Data

Algorithm	Accuracy (%)	Recall (%)	Precision (%)	F1 Score (%)	Training Time (s)	Prediction Time (s)	AUC	Log Loss	Accuracy Deviation (%)
AdaBoost	74.4405	68.5732	72.9462	69.5364	371.7664	2.3089	0.7931	0.6121	-7.0892
KNN	80.0297	78.1973	78.1546	78.1755	4.5428	223.8360	0.7820	7.1980	-19.7711
Random Forest	**84.7978**	81.5923	84.5195	82.6986	732.0494	6.2169	0.9182	0.3408	-15.1805
Decision Tree	77.9885	76.0722	75.9234	75.9958	47.5945	0.0777	0.7607	7.9337	-22.0035
Bagging	84.5792	81.5007	84.1192	82.5133	2972.2421	6.9626	0.9151	0.3539	-15.4153
Logistic Regression	72.4199	67.5026	69.8124	68.1653	2.3690	0.0082	0.7373	0.5609	-0.0415
Gaussian NB	71.8279	67.7259	69.011	68.1852	1.2164	0.0479	0.6784	0.8238	+0,1022
Gradient Boosting	78.5013	74.5297	77.0944	75.4244	767.7227	0.3091	0.8552	0.4467	-12.1445
Extra Trees	84.6286	81.3143	84.4100	82.4655	132.4850	9.1348	0.9171	0.3463	-15.3331
XGBoost	78.4861	74.4181	77.1341	75.3477	2.0502	0.0569	0.8531	0.4485	-11.4325
ANN	83.7789	80.4484	83.3294	81.5252	1850.0085	0.11329	0.9128	0.3406	-11.8863
CNN	81.7325	77.4846	81.6391	78.7808	321.4079	7.3378	0.8902	0.3814	-6.2333

Gaussian NB shows an accuracy rate of 71.73%, with Recall at 53.63% and Precision at 61.49%, resulting in an F1 score of 57.30%. The model's training and prediction times are notably short at 0.246 and 0.0546 seconds respectively. AUC is relatively low at 0.6763, and Log Loss is high at 10.1910, indicating limited applicability for this application.

Gradient Boosting achieved an accuracy of 90.65%, Recall at 89.63%, and Precision at 84.14%, giving an F1 score of 87.27%. Training time is considerable at 170.63 seconds, while prediction time is 2.92 seconds. AUC stands at 0.9064, and Log Loss is 3.3715, showing reliable but computationally intense performance.

Extra Trees achieved an accuracy of 99.96%, with Recall at 99.94%, Precision at 99.95%, and F1 score at 99.94%. Training time is 58.36 seconds, and prediction time is 2.97 seconds. AUC is high at 0.9995, and Log Loss is low at 0.0138, making it a strong choice.

XGBoost achieved an accuracy of 99.91%, Recall at 99.03%, and Precision at 99.03%, resulting in an F1 score of 99.03%. The training time is 112.07 seconds, and the prediction time is 2.53 seconds. AUC is 0.9994, and Log Loss is 0.0249, indicating high reliability and confidence in predictions.

The ANN model achieved an accuracy of 95.66%, indicating high classification performance. It has a recall of 93.53%, precision of 94.17%, and an F1 score of 93.84%, which demonstrate its effectiveness in correctly identifying both positive and negative cases. The model's AUC is 0.9518, reinforcing its strong performance in distinguishing between classes. However, the training time for ANN is notably high at 3396.41 seconds, which highlights the model's computational demands. The prediction time is relatively low at 0.1137 seconds, making it efficient during the testing phase. This setup suggests that while ANN requires substantial resources for training, it provides a robust performance and quick predictions, making it suitable for applications where high accuracy is prioritized over training speed.

CNN achieved an accuracy of 87.97%, Recall of 80.08%, and Precision of 85.16%, with an F1 score of 82.46%. While the training time is lengthy at 313.62 seconds, AUC is 0.8618, and Log Loss is 4.3375, making it suitable for tasks involving complex feature extraction at the expense of computational efficiency.

As a result, when model performances are analyzed, it is seen that the Bagging model shows the highest performance. However, ANN has the longest training time among the models. The Decision Tree model stands out with its high accuracy rate and very low training and testing times. It is important to take this into consideration when making a choice in the application context.Given the critical importance of attack detection in IoT networks, where timely and accurate identification of threats is crucial for maintaining network security and performance, selecting the appropriate model is essential. The effectiveness of these models directly impacts the ability to safeguard IoT environments against malicious activities.

Performance evaluation of noisy data

In this study, we also test the performance of machine learning algorithms on noise-added data. In the real world, data is not always clean and error-free. Noise is inevitable in data collection processes. Testing how algorithms deal with noisy data increases the reliability of models in real-world applications.

For noise, a certain level of random normally distributed noise is added to the feature values of the dataset. The noise level is determined by multiplying the standard deviation of the data by 10%. This process is used to simulate real-world data collection errors. The noise addition process involves the following steps:

The noise level is determined. This level determines the magnitude of the added noise. The noise level (σ_{noise}) is a ratio of the standard deviation of the available data. Random noise is generated for each feature. This noise is generated by random numbers drawn from a normal distribution (Gaussian distribution), as shown in Equation 6:

$$noise_i \sim \mathcal{N}(0, \sigma_{noise} \cdot \sigma_i) \quad (6)$$

- ($noise_i$): Random noise for column i

- (σ_i): Standard deviation of column i

- (σ_{noise}): Noise level ($\sigma_{noise} = 0.1$)

The generated random noise (D_{noisy}) is added to a copy of the original data set (D_{noisy_i}), as shown in Equation 7:

$$\boldsymbol{D_{noisy}}[:, \boldsymbol{i}] = \boldsymbol{D}[:, \boldsymbol{i}] + \boldsymbol{noise_i} \quad (7)$$

- $D[:, i]$: i column of the original data set

-$D_{noisy}[:, i]$: i column of the noise-added data set

There are no negative values in the dataset, so Equation 8 is used to take the absolute value of all values in column i. in the matrix D_{noisy}, making negative values positive.

$$\boldsymbol{D_{noisy}}[:, \boldsymbol{i}] = \left|\boldsymbol{D_{noisy}}[:, \boldsymbol{i}]\right| \quad (8)$$

Analysis of noisy data is given in Table 5. The Accuracy Deviation (%) column represents the difference in accuracy between the noisy and original data, expressed as a percentage. According to Table 5, the Random Forest algorithm achieves the highest accuracy on noisy data, with an accuracy of 84.79%, recall of 81.59%, precision of 84.51%, and F1 score of 82.69%. Additionally, it has an AUC of 0.9182 and a Log Loss of 0.3408, which demonstrates the Random Forest algorithm's robustness in handling noisy data effectively.

In contrast, the Decision Tree algorithm experiences a significant decrease in accuracy on noisy data, achieving an accuracy of 77.98%, recall of 76.07%, precision of 75.92%, and F1 score of 75.99%. The AUC for Decision Tree drops to 0.7607, and Log Loss increases to 7.9337, indicating that the Decision Tree algorithm is notably more sensitive to noise. This is likely due to the Decision Tree's reliance on a single tree structure, making it more susceptible to noise or outliers, which can cause larger fluctuations in accuracy.

Both Random Forest and Bagging mitigate this issue by employing multiple decision trees, which collectively reduce the influence of individual tree errors and enhance model generalization. This ensemble approach diminishes the sensitivity to noise, making them more resilient. However, in this study, Random Forest not only outperforms Bagging in terms of accuracy but also achieves faster results due to its methodology of using a random subset of features for each tree. This approach decreases training time and further minimizes the correlation between trees, giving Random Forest an edge over Bagging.

Bagging, on the other hand, trains each decision tree on the full set of features, requiring more computational power and time. This model often involves a larger number of trees to achieve a strong generalization, which extends the training duration. While Bagging provides a highly robust and resilient model, it is more resource-intensive compared to Random Forest. The results in Table 5 demonstrate that Random Forest not only achieves a faster training time but also exhibits higher accuracy and robustness within the context of noisy data, outperforming Bagging.

Other algorithms, such as KNN and Gradient Boosting, also show reasonable resilience to noise, with accuracy rates of 80.30% and 85.50%, respectively. However, Logistic Regression and Gaussian NB exhibit lower performance, with Logistic Regression achieving 72.42% accuracy and Gaussian NB achieving 71.82%, highlighting their limitations in handling noisy data effectively.

Finally, the neural network models ANN and CNN present varying performance levels under noisy conditions. While ANN and CNN demonstrate relatively high accuracy levels of 83.78% and 81.73%, respectively, the training times are considerably longer, particularly for ANN. This suggests that while neural networks can adapt to noisy data, they may require extensive computational resources, which could be a consideration in resource-constrained environments.

In summary, Random Forest stands out as the most effective model for handling noisy data, combining high accuracy, reasonable training time, and resilience against noise, making it a suitable choice in noisy environments.

Discussion

In this study, various machine learning algorithms were evaluated for detecting RPL-based routing attacks using the ROUT-4-2023 dataset. Results show that tree-based models, especially Random Forest, Decision Tree, and Bagging achieved higher accuracy and outperformed deep learning models like ANN and CNN. This can be attributed to the structure of the dataset, which allows tree-based algorithms to effectively separate attack types, resulting in superior performance without the computational complexity of deep learning models.

Due to the novelty of the ROUT-4-2023 dataset, comparisons with prior studies were not possible, as this dataset has not been previously used in the literature. This limitation highlights the need for further studies utilizing ROUT-4-2023 to establish a broader comparative framework.

Overall, the distinct features of the ROUT-4-2023 dataset enabled tree-based methods to provide efficient and accurate classifications. While ANN and CNN showed potential, their longer training times and computational demands did not yield notable benefits over simpler, interpretable tree models in this case. Future studies could explore deep learning approaches further on larger or more complex datasets to reassess their effectiveness in this context.

Conclusion

This study evaluated the effectiveness of various machine learning algorithms for detecting RPL-based routing attacks in IoT networks. Using the ROUT-4-2023 dataset, we demonstrated that tree-based models, particularly Decision Tree, Bagging, and Random Forest, achieved high accuracy, with Decision Tree and Bagging reaching an impressive accuracy of 99.99% on the combined dataset. To represent real-world conditions, a 10% noise level was added to the dataset, where Random Forest performed the best with an accuracy of approximately 84.80%. These high accuracy rates underline the suitability of machine learning-based IDS for classifying attack types in RPL-based networks, enhancing security and resilience in low-power and lossy network environments. On the other hand, due to

the easily separable nature of the dataset, more complex models like ANN and CNN did not achieve similarly high performance, underscoring that simpler models can be highly effective for this specific dataset. Our results highlight the advantages of machine learning in fortifying RPL protocol security, establishing a solid foundation for further research. Future studies may explore more advanced models, such as deep learning, to assess their potential in improving detection capabilities. Additionally, real-world testing across diverse IoT settings would be essential to validate the practical applicability and robustness of these models in live environments.

References

[1] S. Görmüş, H. Aydın, and G. Ulutaş, "Security for the internet of things: a survey of existing mechanisms, protocols and open research issues," Journal of the Faculty of Engineering and Architecture of Gazi University, vol. 33, no. 4, pp. 1247–1272, 2018.

[2] Q. Jing, A. V. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the internet of things: perspectives and challenges," Wireless networks, vol. 20, pp. 2481–2501, 2014.

[3] R. Alexander, A. Brandt, J. Vasseur, J. Hui, K. Pister, P. Thubert, P. Levis, R. Struik, R. Kelsey, and T. Winter, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," RFC 6550, Mar. 2012. [Online]. Available: https://www.rfc-editor.org/info/rfc6550

[4] A. Verma and V. Ranga, "Machine learning based intrusion detection systems for iot applications," Wireless Personal Communications, vol. 111, no. 4, pp. 2287–2310, 2020.

[5] T. Bekar, S. Görmüş, B. Aydın, and H. Aydın, "Q-learning algorithm inspired objective function optimization for ietf 6tisch networks," in 2023 International Conference on Smart Applications, Communications and Networking (SmartNets), 2023, pp. 1–6.

[6] M. EMEÇ and M. H. ÖZCANHAN, "Rout-4-2023: Rpl based routing attack dataset for iot," 2023. [Online]. Available: https://dx.doi.org/10.21227/3mbe-5j70

[7] H. Aydin, S. Goermues, and Y. Jin, "A distributed user authentication mechanism for ietf 6tisch protocol," in 2018 IEEE 87th Vehicular Technology Conference (VTC Spring), 2018, pp. 1–7.

[8] S. Külcü and S. Görmüş, "Improving synchronization time in 6tisch networks with smart antennas," in 2022 30th Signal Processing and Communications Applications Conference (SIU). IEEE, 2022, pp. 1–4.

[9] A. Mayzaud, R. Badonnel, and I. Chrisment, "A taxonomy of attacks in rpl-based internet of things," Int. J. Netw. Secur., vol. 18, pp. 459–473, 2016.

[10] B. Aydın, S. Görmüş, H. Aydın, and S. Kulcu, "A new routing objective function for ietf 6tisch protocol," in 2022 30th Signal Processing and Communications Applications Conference (SIU). IEEE, 2022, pp. 1–4.

[11] A. Verma and V. Ranga, "Elnids: Ensemble learning based network intrusion detection system for rpl based internet of things," in 2019 4th International conference on Internet of Things: Smart innovation and usages (IoT-SIU). IEEE, 2019, pp. 1–6.

[12] E. Aydogan, S. Yilmaz, S. Sen, I. Butun, S. Forsström, and M. Gidlund, "A central intrusion detection system for rpl-based industrial internet of things," in 2019 15th IEEE International Workshop on Factory Communication Systems (WFCS). IEEE, 2019, pp. 1–5.

[13] S. Deshmukh-Bhosale and S. S. Sonavane, "A real-time intrusion detection system for wormhole attack in the rpl based internet of things," Procedia Manufacturing, vol. 32, pp. 840–847, 2019.

[14] B. Farzaneh, M. Koosha, E. Boochanpour, and E. Alizadeh, "A new method for intrusion detection on rpl routing protocol using fuzzy logic," in 2020 6th International Conference on Web Research (ICWR), 2020, pp. 245–250.

[15] A. Agiollo, M. Conti, P. Kaliyar, T.-N. Lin, and L. Pajola, "Detonar: Detection of routing attacks in rpl-based iot," IEEE transactions on network and service management, vol. 18, no. 2, pp. 1178–1190, 2021.

[16] E. Garcia Ribera, B. Martinez Alvarez, C. Samuel, P. P. Ioulianou, and V. G. Vassilakis, "An intrusion detection system for rpl-based iot networks," Electronics, vol. 11, no. 23, 2022.

[17] A. Alazab, A. Khraisat, S. Singh, S. Bevinakoppa, and O. A. Mahdi, "Routing attacks detection in 6lowpan-based internet of things," Electronics, vol. 12, no. 6, 2023.

[18] H. Azzaoui, A. Z. E. Boukhamla, P. Perazzo, M. Alazab, and V. Ravi, "A lightweight cooperative intrusion detection system for rpl-based iot," Wireless Personal Communications, pp. 1–24, 2024.

[19] R. Bokka and T. Sadasivam, "Simulation-based analysis of rpl routing attacks and their impact on iot network performance," Journal of Electronic Testing, pp. 1–15, 2024.

[20] U. Kiran, P. Maurya, and H. Sharma, "Investigating routing protocol attacks on low power and lossy iot networks," SN Computer Science, vol. 5, no. 4, p. 393, 2024.

[21] M. Osman, J. He, K. Zhu, and F. M. M. Mokbal, "An ensemble learning framework for the detection of rpl attacks in iot networks based on the genetic feature selection approach," Ad Hoc Networks, vol. 152, p. 103331, 2024.

[22] A. Verma and V. Ranga, "Rpl-nidds17-a data set for intrusion detection in rpl based 6lowpan networks," Internet of Things, vol. 5, no. 1, pp. 1-20, 2018.

[23] G. De'Ath, "Boosted trees for ecological modeling and prediction," Ecology, vol. 88, no. 1, pp. 243-251, 2007.

[24] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "Rusboost: A hybrid approach to alleviating class imbalance," IEEE transactions on systems, man, and cybernetics-part A: systems and humans, vol. 40, no. 1, pp. 185-197, 2009.

[25] D. Airehrour, J. Gutierrez, and S. K. Ray, "Securing rpl routing protocol from blackhole attacks using a trust-based mechanism," in 2016 26th International Telecommunication Networks and Applications Conference (ITNAC), 2016, pp. 115-120.

[26] A. Mayzaud, R. Badonnel, and I. Chrisment, "A taxonomy of attacks in rpl-based internet of things," International Journal of Network Security, vol. 18, no. 3, pp. 459-473, 2016.

[27] T. NGUYEN, T. NGO, T. NGUYEN, D. TRAN, H. A. TRAN, and T. BUI, "The flooding attack in low power and lossy networks: A case study," in 2018 International Conference on Smart Communications in Network Technologies (SaCoNeT), 2018, pp. 183-187.

[28] A. Mayzaud, R. Badonnel, and I. Chrisment, "Detecting version number attacks in rpl-based networks using a distributed monitoring architecture," in 2016 12th International Conference on Network and Service Management (CNSM), 2016, pp. 127-135.

[29] A. D. Seth, S. Biswas, and A. K. Dhar, "Detection and verification of decreased rank attack using round-trip times in rpl-based 6lowpan networks," in 2020 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), 2020, pp. 1-6.

[30] O. H. Abdulganiyu, T. Ait Tchakoucht, and Y. K. Saheed, "A systematic literature review for network intrusion detection system (ids)," International Journal of Information Security, vol. 22, no. 5, pp. 1125-1162, 2023.

[31] M. Douiba, S. Benkirane, A. Guezzaz, and M. Azrour, "An improved anomaly detection model for iot security using decision tree and gradient boosting," The Journal of Supercomputing, vol. 79, no. 3, pp. 3392-3411, 2023.

[32] J. B. Awotunde, F. E. Ayo, R. Panigrahi, A. Garg, A. K. Bhoi, and P. Barsocchi, "A multi-level random forest model-based intrusion detection using fuzzy inference system for internet of things networks," International Journal of Computational Intelligence Systems, vol. 16, no. 1, p. 31, 2023.

[33] J. B. Awotunde, S. O. Folorunso, A. L. Imoize, J. O. Odunuga, C.-C. Lee, C.-T. Li, and D.-T. Do, "An ensemble tree-based model for intrusion detection in industrial internet of things networks," Applied Sciences, vol. 13, no. 4, p. 2479, 2023.