International Journal of Safety and Security Engineering
Vol. 15, No. 6, June, 2025, pp.1089-1101
Journal homepage: http://iieta.org/journals/ijsse

A New Intrusion Detection System for RPL Based on Machine Learning

Jihad Dazine*, Abderrahim Maizate, Larbi Hassouni

RITM-ESTC / CED-ENSEM, University Hassan II Km 7, Casablanca B.P.8012 Oasis, Morocco

Corresponding Author Email: jihadazine@gmail.com

https://doi.org/10.18280/ijsse.150601

Received: 7 April 2025
Revised: 13 May 2025
Accepted: 24 June 2025
Available online: 30 June 2025

Keywords:
IoT, RPL, IDS, machine learning

ABSTRACT

IoT devices are used increasingly in many fields. The Routing Protocol for Low Power and Lossy Networks (RPL) is a protocol designed for IoT networks. RPL is vulnerable to many routing-based attacks that threaten the confidentiality, the integrity and the availability of IoT devices. However, due to the specific behavior and characteristics and IoT systems' constrained nature in terms of power, processing, memory and network capabilities, the traditional intrusion detection solutions remain ineffective in the IoT environment. Thus, an efficient mechanism for RPL attacks detection and prediction is needed. This paper proposes a framework for RPL intrusion detection based on Machine learning. Our model detects four RPL routing attacks which are: Hello Flood, Increased version, Decreased rank and blackhole attacks. We developed a new IoT specialized dataset based on the simulation of the four real RPL attacks. We extracted the most significant features, and we used preprocessing and data balancing to improve our proposed Intrusion Detection System's performance. We evaluated our system and we compared it to some other existing works, and the results show that the accuracy, the F1-score, the precision and the recall rates of our proposed model are promising.

1. INTRODUCTION

1.1 Background

The Internet of Things (IoT) represents a network of connected devices that exchange data and communicate via the internet. IoT's use is growing because it improves the life's quality by automating tasks efficiently and with less energy. To connect IoT devices and make them part of the internet, the Internet Engineering Task Force (IETF) developed the Routing Protocol for Low Power and Lossy Networks (RPL) to perform routing for the Low Power and Lossy Networks. RPL is vulnerable to several attacks disrupting the IoT network's operation. Indeed, due to IoT devices' constrained nature, and to the use of a non-restricted wireless transmission channel, they are exposed to several attacks, especially routing based ones [1]. Many studies [2-5] analyzed RPL attacks and their effect on the performance of the network. Hence, it is crucial to monitor and analyze intrusion behaviors for IoT systems in order to take precautions and minimize losses. Due to its special characteristics, standard and conventional security solutions are not applicable in the IoT environment. In the literature, some security solutions for RPL intrusion detection were proposed [6-10]. However, most of these solutions are not suitable for resource constraint networks, especially since a large amount of data resulting from attacks should be processed by the Intrusion Detection System (IDS).

In this context, machine learning is an efficient method that can deal with a large amount of sensing data for intrusion detection. Indeed, it can be used in the analysis and monitoring of the IoT systems by finding patterns that allow to identify the malicious traffic. Machine learning uses different learning algorithms to improve the intrusion detection rate. Furthermore, machine learning is effective in the IoT environment because machine learning algorithms respect the limited resources and processing capabilities of IoT devices [1]. The are some studies that propose an IDS for RPL based on machine learning [11-15], but they mostly focus on the detection of particular attack types. Furthermore, there are no publicly available datasets to be used for designing a machine learning intrusion detection mechanism for RPL. Thus, a novel dataset is needed to train a machine learning intrusion detection framework for RPL attacks.

1.2 Objectives and methodology

In this paper, we propose a novel centralized RPL Intrusion Detection System based on machine learning. Our model focuses on the detection of four major RPL routing attacks and uses the binary and the multiclass classification. We create a novel dataset based on the simulation of these attacks using Cooja simulator. We evaluated the performance of our IDS and we compared it to some other existing works. The main contributions are the following:

•To develop a novel dataset based on four RPL attack behaviors and selected IoT features.

•To Design a machine learning based IDS using machine learning algorithms, and to train the proposed model using the developed dataset.

•To maximize the detection rate of our IDS by using preprocessing and data balancing.

The remaining of this paper is structured as follows: Section 2 provides an overview of the literature. Section 3 presents the related works. Section 4 describes the developed dataset and explains the approach followed in the simulation of the RPL attacks and in the selection and the conception of the features. Section 5 presents the evaluation of our model and presents its detection rates, and compares our framework with some other existent works. Section 6 puts forward a discussion of the results. Section 7 is the conclusion of this paper.

2. IDS & RPL ATTACKS

2.1 Intrusion Detection Systems

An Intrusion Detection System (IDS) analyzes and monitors activity and traffic in a network or a device to detect attacks. When an IDS detects a malicious activity, it warns about the attack. IDSs are subject to false negatives by not raising an alarm when an attack is present, and false positives by raising an alarm when attacks are absent [6]. IDS can be classified according to three main criteria: “detection technique”, “detection source”, and “detection architecture”.

Regarding the “detection techniques” criteria, there are mainly four IDS types: anomaly-based, signature-based, specification-based and hybrid-based [16]. Signature based IDS compares the activities in a device or in a network against a predefined attack pattern or signature from a database [16]. This mode cannot detect new attacks unless the database contains their signatures. Anomaly based IDS defines the normal behavior of a device or a network and raises an alarm when the current situation deviates from the baseline. Specification based IDS detects deviation from the normal behavior that requires expert manual assistance to define its specification such as protocol specification [17]. For example, the system raises an alarm if a node does not follow the protocol’s specifications. Hybrid based IDS combines the other detection strategies.

According to the source of detection criteria, IDS can be a “Network-based IDS”, a “Host-based IDS”, or a “Hybrid IDS”. The Network-based IDS (NIDS) monitors the flow of the network traffic and analyses it based on the detection method. The Host-based IDS (HIDS) monitors the traffic transmitted and received by the host machine, and analyses the internal events that occurs within the host machine to identify malicious activities [17]. The Hybrid IDS in terms of source detection combines the security mechanisms of both NIDS and HIDS to analyze network events and node-based information.

The architecture of IDS can be centralized or decentralized (distributed). In Centralized IDS, a central unit performs the monitoring of the network traffic and the analysis of the collected packets, and the other nodes monitor, capture, store and transfer data to the central unit which aggregates and analyses the received data. Monitoring nodes may be deployed to listen passively to the network communications and collect and send data to the root node for analysis. In decentralized IDS, distributed nodes collect and transport packets and are also responsible for data aggregation and decision making. Monitoring nodes may be included in the network, and sniffers can be placed to collect data from the neighboring nodes and send it to the monitoring nodes. Afterwards, these ones can aggregate data and forward it to the root node for further analysis [17].

2.2 Machine learning in Intrusion Detection Systems

Machine learning (ML) is a branch of Artificial Intelligence (AI) that aims to make intelligent decisions and automatically recognize complex patterns [18]. Machine learning can be used for intrusion detection by learning malicious patterns from the network traffic. In supervised classification, an ML model comprises training and testing. A dataset of labeled normal and malicious network packets is used during training to extract patterns and adjust parameters. The test is then carried out to measure the performance of unseen traffic classification on unlabeled packets [19]. The most common methodologies used in IoT applications fall into this category, such as Naive Bayes (NB) and Support Vector Machine (SVM) [20]. SVM technique uses nonlinear mapping to transform data into a higher dimensional space, and aims to find a decision frontier by searching for the best hyperplane that optimally separates classes. SVMs are characterized by a high accuracy but a slow learning time [18]. NB technique uses Bayes' rule for classification, where the distribution of feature classes is independently modelized. It demonstrates time efficiency, but it operates under the “native” supposition that each pair of input features is independent. This algorithm has been used in many IoT security solutions [19]. Tree-based machine learning (ML) models have a tree structure made up of nodes and branches. Nodes are created using features to divide the learning instances, building a tree. End nodes represent classes and branches represent test results [18]. The model traverses the tree via different branches from the root node to predict the classification of an instance [20]. Random Forest (RF) is a tree-based machine learning (ML) model used in IoT intrusion detection with good results. It consists of a set of multiple decision trees built by randomly selecting data points [20]. Unsupervised learning is in general equivalent to clustering, where the learning classes are not labeled. Clustering is therefore used to identify classes [18]. Semi-supervised learning uses both unlabeled and labeled instances for learning, where classes are learned using labeled instances, and classes boundaries are corrected using unlabeled instances [18]. However, detection accuracy is not as good as that of supervised learning [21].

Deep Learning (DL) is a subset of ML that takes inspiration from the biological neural system where data signals are transmitted by a network of interconnected nodes, and is capable of detecting complex data patterns [19]. Deep learning techniques are used to address the other ML techniques’ limitations and to offer high classification accuracy. Several deep learning algorithms, like Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs) and Long-term Memories (LSTMs), can be used in Intrusion Detection Systems [21]. Multilayer Perceptron (MLP) is a variant of neural networks founded on supervised learning, where the information flows unidirectionally without loops [22]. It has three layers: an input layer collecting the input features that should be processed, an output layer performing classification, and a single or multiple hidden layers working as the computational unit of the MLP [22]. It is used in solutions for attack detection. Convolution Neural Network (CNN) extends the multilayer perceptron architecture by incorporating an input layer, an output layer and many connected hidden layers which are down-sampled, non-linear and made by convolution [20].

2.3 RPL protocol

Low Power and Lossy networks comprise constrained devices characterized by restricted memory, energy capacity, and processing capabilities. RPL operates as a Distance Vector routing protocol tailored to the requirements of Low Power and Lossy networks. The IETF Routing Over Low power and Lossy network (ROLL) task force standardized RPL in 2012, and it was specified and described in Request for Comment (RFC) 6550 [23]. RPL combines tree and mesh topologies to interconnect the IoT devices building a Destination-Oriented Directed Acyclic Graph (DODAG). The DODAG is a directed and acyclic graph oriented towards a sink (root or gateway) node that allows access to Internet and orchestrates the tree construction. The DODAG includes a sequence of parent nodes that link the root node to the other nodes [23]. A network can contain many DODAGs that form an RPL instance [24]. Figure 1 describes the main RPL components and structure.

RPL has three forms of communication: multipoint-to-point (MP2P) which supports the transfer of messages from multiple nodes to a central server, point-to-multipoint (P2MP) where messages are transferred from a central node to multiple nodes, and point-to-point (P2P) which is characterized by the communication between individual nodes [17]. The rank refers to the node's position in the DODAG, it depends on the Objective Function (OF) and on the distance between the node and the root [23]. The objective function determines the rank of nodes respecting the routing metrics, and it considers optimization objectives and routing constraints [24]. Two recognized objective functions are: Objective Function zero (OF0) and Minimum Rank with Hysteresis Objective Function (MRHOF), they consider respectively the minimum hop count and the Expected Transmission Count (ETX) as a default routing metric [25]. RPL protocol relies on ICMP-v6 based control messages to maintain the communication routes and the DODAG structure. The structure of RPL control message is described in Figure 2 [26].

The ICMPv6 header consists of the type, the code, and the checksum fields. A base message and other options constitute the message's body. For RPL, the type field is assigned the value 155 defining the ICMPv6 control message's type. The Code field specifies the kind of RPL control message [18]. There are four different RPL control messages labeled: DODAG Information Solicitation (DIS), DODAG Information Object (DIO), Destination Advertisement Object (DAO), and Destination Advertisement Object Acknowledgment (DAO-ACK).

DIO is a downward control message that the root node uses to maintain the DODAG structure and to initiate a DODAG [27]. The other nodes use DIO messages for the discovery of RPL instances, for the selection of a preferred parent and for the maintenance of the rank information [23]. DIO is mapped to 0x01. Figure 3 shows the DIO base object fields.

RPL instance ID, an 8-bit data, defines the RPL instance ID of the DODAG. Version number is the version number of the DODAG, which is initialized by the root node, and changed when the network information is updated and the DODAG is reconstructed. The rank (16-bit) is a field indicating the DIO message sender' rank. DTSN (8-bit) is a flag utilized to keep downward paths open. G flag indicates if the application's intent is satisfied by the current DODAG. The mode of operation of the RPL instance is indicated by the (MOP) field. The Prf field, 3-bit in size, varies between 0x00 (default) and 0x07 (high priority), it identifies the precedence of DODAG root over other DODAG roots. The DODAGID is an IPv6 address of 128-bit that identifies uniquely the DODAG. Furthermore, the DIO base object can include an options field [18]. DIO messages are transmitted following a time set managed by the trickle algorithm, so that the frequency of the control messages' transmission is maximized [18].

DIS: When a node is not receiving any DIO messages, it sends a DIS message to look for the presence of any DODAGs and to ask for neighborhood information [28, 29]. DIS is mapped to the code 0x00. DIS control message contains flags and other fields for potential use [18].

DAO is mapped to 0x02, it is an upward control message propagated along the DODAG to advertise reverse routes information and to announce destination information [27, 29]. DAO base object fields are presented in Figure 4.

RPLInstanceID (8-bit) identifies the RPL instance ID. K flag designates that DAO-ACK is expected to be sent by the recipient. The 'D' flag defines the presence of the DODAGID field. DAOSequence is a counter incremented for every DAO message from the node. DODAG root defines DODAGID, which is a 28 bits field that serves as a unique identifier of the DODAG. The '*' signifies that the DODAGID field is not always present in the DAO base object.

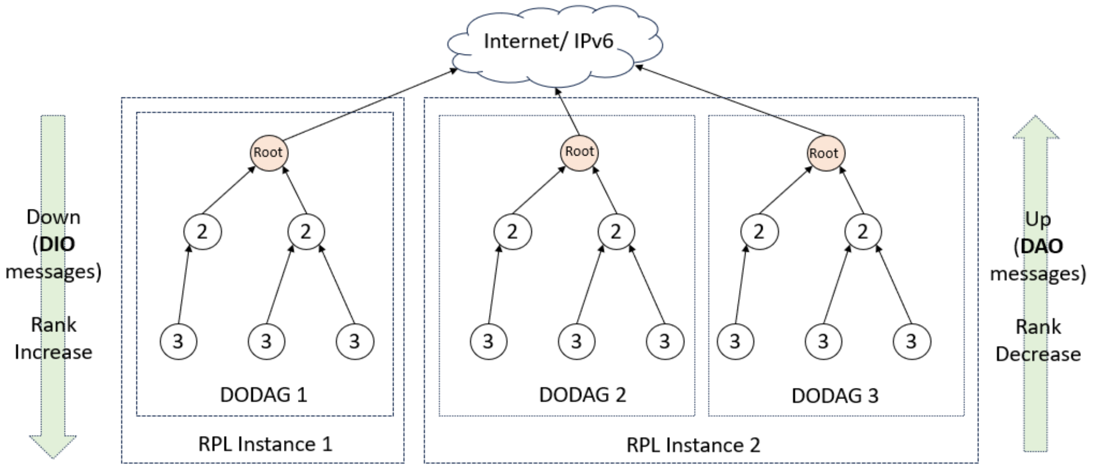


Figure 1. RPL Components and structure

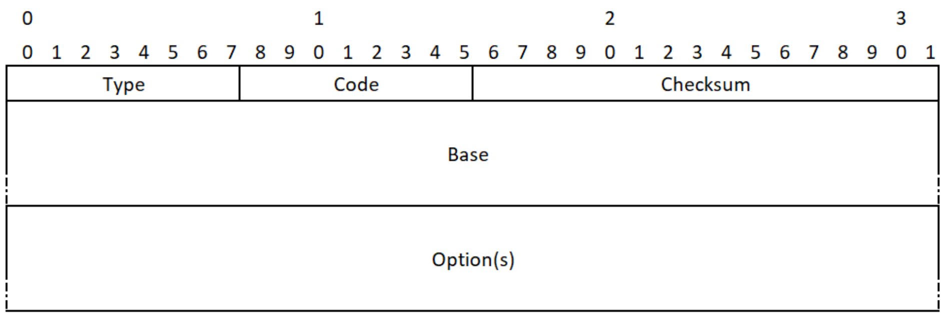


Figure 2. RPL control message [26]

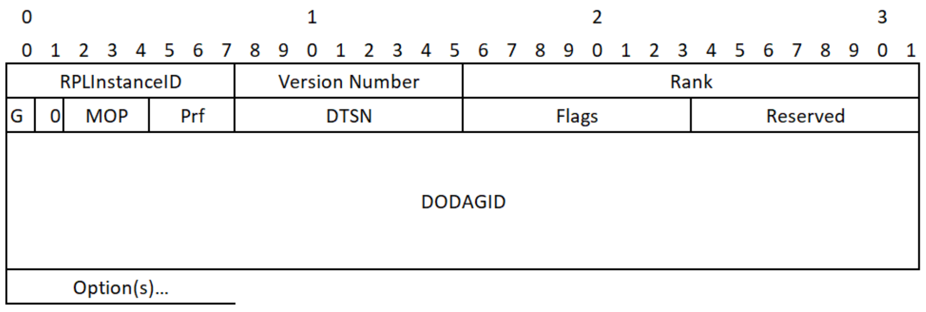


Figure 3. DIO base object [26]

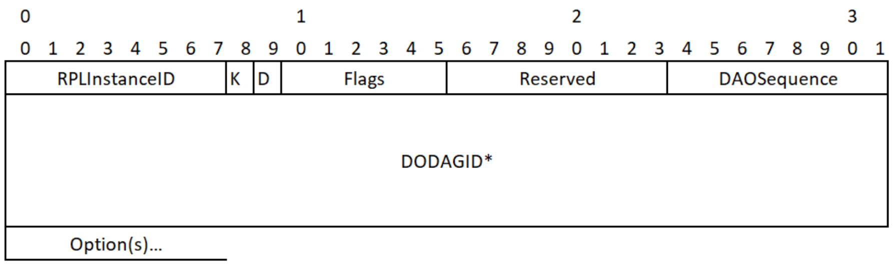


Figure 4. DAO base object [26]

DAO-ACK serves as a reply to a DAO in order to indicate the rejection or the acceptance of the DODAG's connecting call [30].

Figure 5 describes the exchange of RPL control messages.

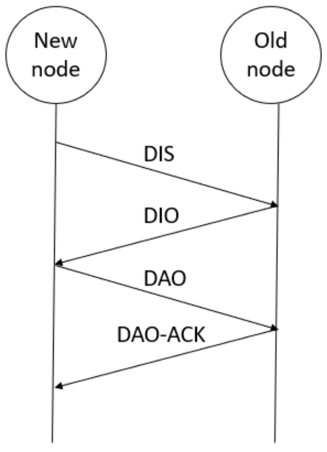


Figure 5. RPL control messages' exchange

2.4 RPL attacks

RPL attacks can target the exhaustion of network resources, modify the network topology, or attack the network traffic by eavesdropping or misappropriation [31]. In this paper, we focus on 4 known RPL attacks.

2.4.1 Hello Flood attack

Hello Flood attack attempts to raise the network traffic so as to make the network's resources unavailable. Many DIS messages are sent by the malicious node to several nodes in the network by reducing the time interval between two sequential DIS messages, which raises the network traffic. When neighbor nodes' trickle timers are reset, they respond with DIO messages, which causes an additional increase in the traffic and a significant waste of energy [30, 32].

2.4.2 Increased version attack

In a DODAG, every DIO packet is associated with a version number set by the root node and changed exclusively during the DODAG's reconstruction, which is known as a global repair. In this process, the root node sends to the children a DIO message incorporating an increased version number value. Upon its receival, the child nodes reset their trickle timer and update their preferred parent, links and routing state [24]. The same process is applied in the increased version attack, in which the malicious node increments the DIO control packet's version number, then forwards it to the neighbors. When the malicious DIO message containing a falsified version number reaches the root, it resets its trickle timer and resends a new DIO. Normal nodes use the received DIO to update their new parent [33]. As a result, the DODAG is reconstructed. This repeated DODAG reconstruction leads to a rise in the network traffic [30, 32].

2.4.3 Decreased rank attack

In a DODAG, a rank is assigned to every node. The rank values decrease along the path toward the root, and a child node chooses as its parent a node that holds the lowest rank. The malicious node forges a DIO packet, decreases its rank value and forwards it to the neighbor nodes. Thus, the malicious node becomes their selected preferred parent, which causes a traffic misappropriation [34] and a graph structure's imbalance [18].

2.4.4 Blackhole attack

It aims to drop the messages received by the child nodes in order to create an internal DoS. The malicious node advertises a better malicious rank with DIO packets in order to attracts the neighboring nodes, then it drops all the received information acting as a hole that absorbs everything [17]. This attack can make the other nodes on the downward path isolated from the network if carried out in the appropriate position, when for example the attacker has many children nodes [35].

3. RELATED WORKS

Foley et al. [36] developed a dataset integrating IoT's network and power features in order to detect multi-vector IoT attacks of Rank and Blackhole, Rank and Version, Decreased Path Metric, and Rank and Sybil against the objective functions MRHOF and OF0. However, the instances of the dataset represent the nodes' status instead of the packets' exchange. In this context, the process and the effort of the dataset's creation in terms of the number of the simulations performed weren't provided. Moreover, the proposed dataset is small sized since it contains 418 instances, and Hello Flood attack isn't considered by the detection system. The suggested intrusion detection model is based on machine learning. The experiments show that preprocessing, normalization, feature selection, sampling and dataset balancing impact the performance of the IDS. The best detection's performance was obtained with the classifier ensemble voting technique, employing RF and MLP models, with an accuracy of 87.08%. However, the values of precision, F1-score and recall were not explicitly stated.

Momand et al. [1] proposed an RPL based Intrusion Detection System named MLRP, that aims to detect version number, rank, and DoS attacks. However, Hello Flood and blackhole attacks are not considered by the proposed detection system. A dataset is created through simulation, but the features were not provided. Principal Component Analysis (PCA) is employed for features reduction and preprocessing, which reduces the energy consumption improves the routing efficiency. Nevertheless, the dataset balancing is not performed before the model's training and testing. To train the dataset, SVM classifier is used. The accuracy, the F1-score, the precision and the recall values were not specified precisely but within a range, with an accuracy between the range of 0.90 to 0.92 and a recall between the range of 0.96 to 0.98. The MLRP performance and efficiency is evaluated, it has a PDR value of 76.8%, it generates 1474 control packets and it utilizes 8.76 joules.

Osman et al. [37] present a machine learning model that aims to trigger decreased rank attacks. IRAD dataset is used and tested in multi and binary detection scenarios. However, Hello Flood and blackhole attacks are not in the scope of the proposed IDS. Data preprocessing is performed, then features are extracted using random forest classifier. 9 network features are selected, but they don't include the nodes' rank or version information to recognize the decreased rank attack. An artificial neural network model is utilized for the intrusion detection. According to the experiments' results, the precision, the accuracy, the detection probabilities, the false-positive rate, the false-negative rate, and the area under the ROC curve (AUC) scores are respectively 97.03%, 97.01%, 97.01%, 4.6%, 1.6% and 98%.

Ghaleb et al. [38] proposed a machine learning model to detect Decreased Rank attack, and investigate the impact of the attacker's position on the detection. A dataset is created by simulating three distinct scenarios where the attacker's position relative to the DODOAG root changes. The details concerning the number of nodes used to generate the dataset and the simulation time are not provided. The proposed dataset is based on 7 network features, but it does not consider the nodes' rank to detect the decreased Rank attack. The findings show that the attack's detection is more accurate when the attacker is three hops away from the network root, with an accuracy of 0.98. However, the performance of the ML models is not good when the attacker is placed at most within two hops of the DODAG root. F1-score values were not specified.

Sawafi et al. [27] proposed a hybrid deep learning based IDS that merges the supervised DANN and the semi-supervised DAE in order to classify the known and the novel RPL malicious behaviors. A dataset named IoTR-DS was generated through the simulation of the Rank attack, the DIS attack, and the Wormhole attack. However, blackhole and increased version attacks are not considered in the proposed model. The proposed IDS is evaluated, and the experiments achieved a mean accuracy of 98% for pretrained multi-class attacks (known traffic), and a mean accuracy of 95% for two attack patterns when unknown attacks are predicted.

Our proposed dataset is based on 4 RPL known attacks: Hello Flood, Blackhole, Increased Version and Decreased Rank, it is composed of network features related to the exchanged traffic, and it adds other specific features that were not included in the other datasets. Our model is explained in details in the following section.

4. PROPOSED MODEL AND DATASET

There are limited datasets used for intrusion detection such as DARPA, UNSW-NB15, NSL-KDD, and KDD99, and they are not dedicated to the IoT environment. Our proposed dataset relies on IoT features, and it is focused on RPL attacks. We created our dataset by simulating real RPL attacks using Cooja simulator.

4.1 Proposed model

4.1.1 Proposed framework

Our proposed model is presented in Figure 6, it is a centralized and anomaly-based IDS where data is routed to the root node for attacks detection. Because public IoT attack datasets are not available, data was generated by performing real equivalent RPL attacks simulations using the open source Cooja simulator.

4.1.2 RPL attacks scenarios

In order to simulate the RPL attacks scenarios, we used Cooja simulator. It is a Java-based network simulator designed to simulate sensor networks. Using Cooja, programs can be loaded and unloaded to simulated sensors, which allows the developers to test their code. It allows to write sensor nodes code in the C language [39]. Used with Contiki operating system, Cooja is a flexible cross-layer simulator that enables to simulate nodes with various software and hardware levels [40], and it interacts with the operating system, the application, and the machine code layers [41]. We used Cooja simulator to simulate the following four well known RPL security attacks: Hello Flooding attack, Decreased Rank attack, Blackhole attack and Increased version attack. For each attack, we simulated the malicious scenario and the normal scenario for the used network topology. In the normal scenarios, the DODAG is composed of eleven normal nodes and one root node (ID:1). In the malicious one, the DODAG contains and one root node (ID:1), one malicious node (ID:12) and ten normal nodes. Table 1 presents the simulation parameters.

In the literature, different time intervals were chosen to conduct the simulations. Indeed, some studies have chosen 5min [1, 42], others 20 minutes [11, 16], and others 1hour or more [43, 44]. In our case, following the nodes' number used in the simulation, we have chosen a simulation time of 10 min because we realized that it was sufficient to create the DODAGs and to restabilize the network following attacks that can slow down the construction of the DODAGs. Each scenario was simulated in 10 minutes for data collection.

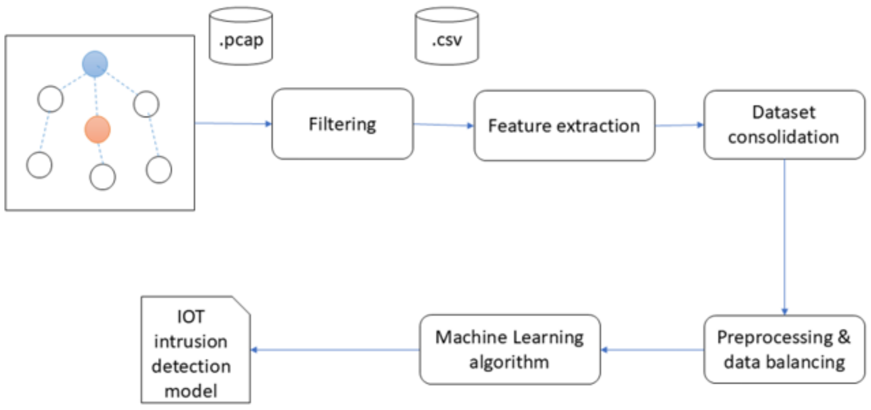

.pcap
.csv
Filtering
Feature extraction
Dataset consolidation
Preprocessing & data balancing
Machine Learning algorithm
IOT intrusion detection model

Figure 6. The architecture of the proposed RPL intrusion detection framework

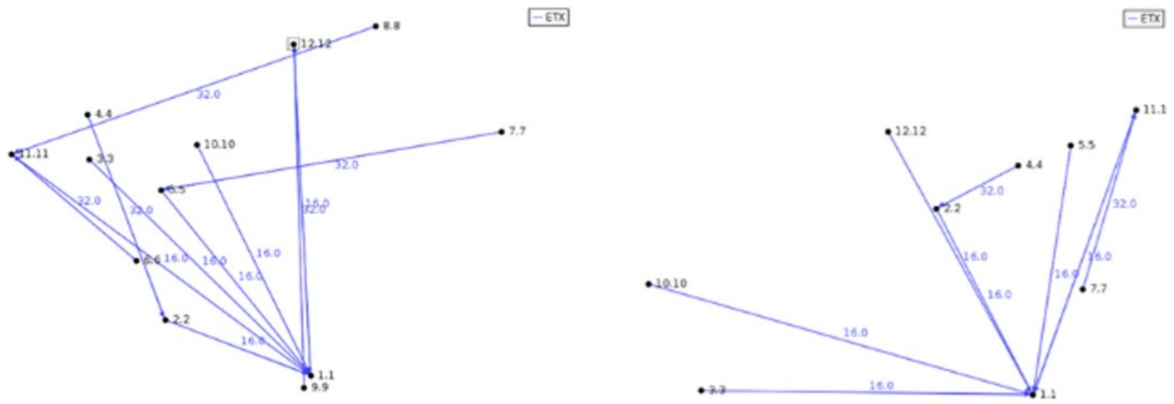


Figure 7. Comparison of DODAG graphs: normal scenario (left) and blackhole attack (right) [45]

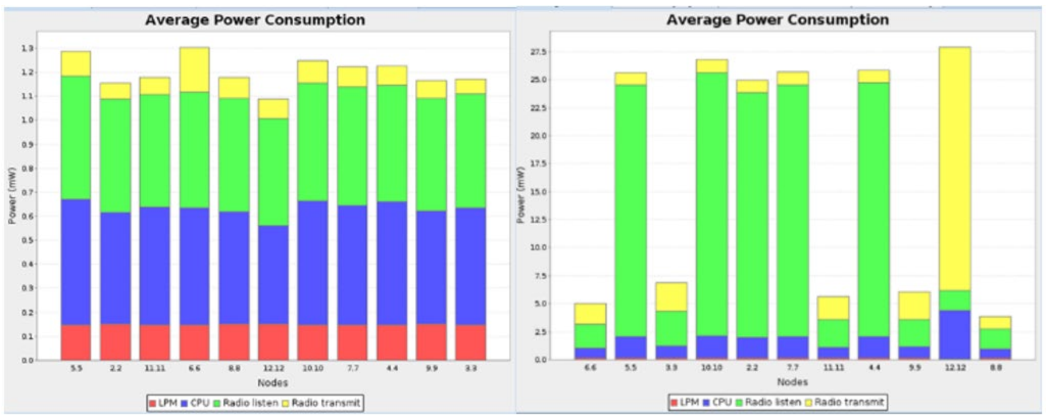


Figure 8. Comparison of average power consumption: Normal scenario (left) and Hello Flood attack (right) [45]

Raw packet capture (PCAP) files are generated and transformed into Comma Separated Values (CSV) files which are fed into the feature extraction module. Thereafter, a binary and a multiclassification datasets are created and consolidated. Moreover, datasets for each scenario are created by merging the normal and malicious data for each attack. Afterwards, preprocessing and data balancing are applied to improve our system's performance. Finally, the datasets are supplied to the machine learning algorithms to perform the system's training and testing, which allows the creation of IoT intrusion detection models.

4.1.2 RPL Attacks scenarios

In order to simulate the RPL attacks scenarios, we used Cooja simulator. It is a Java-based network simulator designed to simulate sensor networks. Using Cooja, programs can be loaded and unloaded to simulated sensors, which allows the developers to test their code. It allows to write sensor nodes code in the C language [39]. Used with Contiki operating system, Cooja is a flexible cross-layer simulator that enables to simulate nodes with various software and hardware levels [40], and it interacts with the operating system, the application, and the machine code layers [41]. We used Cooja simulator to simulate the following four well known RPL security attacks: Hello Flooding attack, Decreased Rank attack, Blackhole attack and Increased version attack. For each attack, we simulated the malicious scenario and the normal scenario for the used network topology. In the normal scenarios, the DODAG is composed of eleven normal nodes and one root node (ID:1). In the malicious one, the DODAG contains and one root node (ID:1), one malicious node (ID:12) and ten normal nodes. Table 1 presents the simulation parameters.

Table 1. RPL attacks simulation parameters

Parameter	Value
Radio Medium	Unit Disk Graph Medium: distance loss
Number of nodes	12
Sink node	1
Routing protocol	RPL
MAC protocol	CSMA + ContikiMAC
Network size	50 x 100 meters
Mote type	Sky Mote
Simulation time	600 s
Seed type	Random Seed
Positioning	Random Positioning

In the literature, different time intervals were chosen to conduct the simulations. Indeed, some studies have chosen 5min [1, 42], others 20 minutes [11, 16], and others 1hour or more [43, 44]. In our case, following the nodes' number used in the simulation, we have chosen a simulation time of 10 min because we realized that it was sufficient to create the DODAGs and to restabilize the network following attacks that can slow down the construction of the DODAGs. Each scenario was simulated in 10 minutes for data collection.

We generated the PCAP files, the DODAG graphs, and the power tracking for each simulation. After analyzing the DODAG graphs, we found that they are different in the normal and malicious scenarios, particularly in the blackhole attack DODAG which doesn't display some nodes in the malicious scenario as illustrated in Figure 7 [45].

Furthermore, power consumption tracking is also different in the malicious scenario, particularly in Hello Flood attack, as illustrated in Figure 8. Hence, power tracking information and DODAG graphs can be considered as indicators to trigger RPL attacks.

In the following section, we will explain our approach to develop a new dataset for machine learning RPL intrusion detection.

4.2 Dataset generation

In this section, we explain the approach followed to create our dataset based on the simulation of RPL attacks, and we describe the proposed dataset's features.

4.2.1 Traffic capture

For each scenario, we captured the network traffic of the IoT network as a PCAP file using the "radio messages" tool of Cooja simulator. We analyzed all the PCAP files using "Wireshark". Every file contains information of the exchanged packets in the network. A sample of the initial raw dataset is presented in Table 2. However, we needed to add more significant features for the intrusion detection of RPL attacks. For this reason, we used "tshark" tool to extract more information from the initial raw dataset. Afterwards, we converted the extracted information to the ".csv" format. In this context, we added 2 features related to the rank and the version of the nodes, which play an important role in the detection of the increased version and decreased rank attacks. Furthermore, we replaced the "Info" feature with the "ICMPv6 code" feature which equals 0 for DIS packets, 1 for DIO packets and 2 for DAO packets. Thus, a sample of our personalized raw dataset is described in Table 3.

Table 2. A sample of the initial raw dataset

No.	Time	Source	Destination	Protocol	Length	Info
212	2.846	fe80::212:7402:2:202	ff02::1a	ICMPv6	97	RPL Control (DODAG Information Object)
213	2.848	fe80::212:7402:2:202	ff02::1a	ICMPv6	97	RPL Control (DODAG Information Object)
214	2.869	fe80::212:740c:c:c0c	fe80::212:7401:1:101	ICMPv6	76	RPL Control (Destination Advertisement Object)

Table 3. A sample of our personalized raw dataset

No.	Time	Source	Destination	Protocol	Length	Code	Rank	Version
339	4.144	fe80::212:7403:3:303	ff02::1a	wpan:6lowpan:ipv6:icmpv6	97	1	768	240
340	4.164	fe80::212:7403:3:303	ff02::1a	wpan:6lowpan:ipv6:icmpv6	97	1	768	240
341	4.181	fe80::212:7407:7:707	ff02::1a	wpan:6lowpan:ipv6:icmpv6	97	1	512	240

4.2.2 New features generation

In order to extract more significant features from the obtained "csv" files, we developed a python script using Pandas and Numpy. Moreover, using this script, we divided each simulation into a window time of 1000ms. Thus, the output displayed the captured packets in each second grouped by source and destination. In this stage, we extracted the following features: Packet Count, Total Packet Duration, Total Packet Length, DIS count, DAO count, DIO count, UDP count, Min version, Max version, Min Rank and Max Rank for each exchange. All the features are explained in the Table 4. "Total Packet Duration" and "Total Packet Length" provide information about the transmission of packets in a time window. DIO, DAO and DIS and UDP counts reflect the networks stability and can change due to an intrusion. "Min version", "Max version" are indicators of the DODAG repair mechanism that happens due to the increased version attack as previously explained. Min and Max Rank can trigger a decreased Rank attack since the attacker node falsely advertises its rank value, that the children nodes rely on to calculate their ranks.

Table 4. The proposed dataset's features and description [45]

No.	Feature Name	Description
1	Source	Source mote (unique number for each mote).
2	Destination	Destination (unique number, same as source).
3	Packet Count	The packet count in a time window.
4	Total Packet Duration	Total packets transmission's duration in a time window.
5	Total Packet Length	All packet lengths' sum in a time window.
6	DIO Message Count	DIO messages count in a time window.
7	DIS Message Count	DIS messages count in a time window.
8	DAO Message Count	DAO messages count in a time window.
9	UDP Message Count	UDP messages count in a time window.
10	MaxRank	Maximum Rank value in a time window.
11	MinRank	Minimum Rank value in a time window.
12	MaxVersion	Maximum version number value in a time window.
13	MinVersion	Minimum version number value in a time window.
14	Label	0 if normal, 1 if malicious (binary model) N, IV, BH, DR, HF (multi classes model)

The pseudocode of the features extraction algorithm is presented in Algorithm 1.

Algorithm 1. Extraction of significant features from the raw dataset

```
function
    array ← RawDataset.csv
    Array Sorted            Sorting by time
    Feature conversion
Feature Extraction:
    Window Size ← 1000ms
    Calculate Feature values within window size
    Label the dataset
    End of the Feature Extraction
End the function
```

For the binary classification model, we added the feature "Label" which equals 1 in the malicious scenarios and 0 in the normal ones. Indeed, we considered the traffic that includes malicious node as malicious since malicious nodes activity affect all the network [41]. For multiclassification model, the "Label" feature can have 5 different values which are: "N" for the normal traffic, "IV" for the increased version attack, "BH" for the blackhole attack, "DR" for the decreased rank attack or "HF" for the Hello Flood attack.

4.2.3 Dataset description

Our dataset contains 16814 samples as shown in Table 5, and it consists of 14 features as described in Table 4.

Table 5. The number of records in the proposed dataset [45]

Normal/Attack	Category	Records Number
Attack	Hello Flooding	5183
	Black Hole	461
	Version Number	3264
	Decreased Rank	6424
Normal	Hello Flooding	286
	Black Hole	375
	Version Number	446
	Decreased Rank	375

4.3 Preprocessing

We converted the sources and destinations IPV6 addresses to the corresponding nodes IDs. For example:

fe80::212:740c:c:c0c --> 12

When the destination address is ff02::1a, it means that broadcast packets are sent by the source node. We choose to convert this value to 99.

ff02::1a --> 99

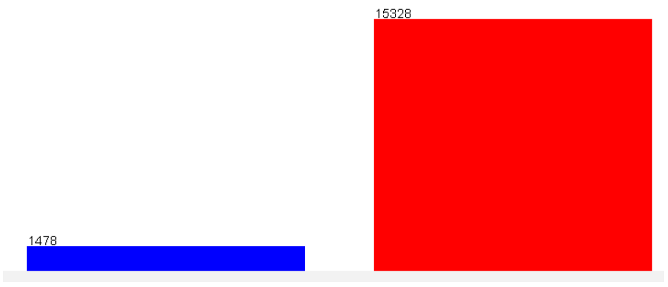
4.3.1 The missing values

Our dataset contained some missing values related to the rank and version information, which were missing in DAO and DIS packets. To remediate to this issue, we developed a python script to fill in the missing values with the immediate previous values of the previous exchange of the same source. For the missing values of the first lines where there are no previous exchanges of the same sources, we choose to replace them by the next values. This script gives as an output the csv files with no missing data. The missing values replacement algorithm's pseudocode is provided in Algorithm 2.

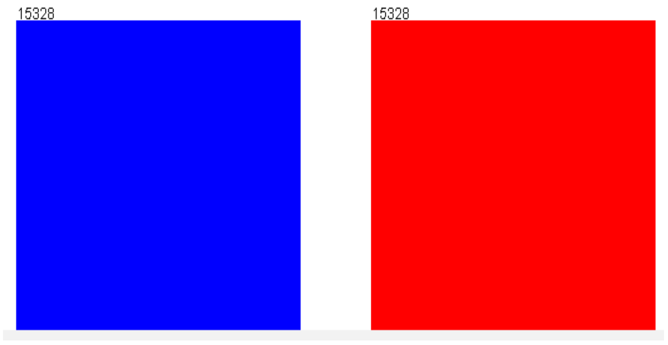
For the following steps, we used "weka" tool for the preprocessing and the classification of our dataset. In this perspective, we modified the csv files and put them in the ".arff" file structure so that they could be imported in Weka.

Algorithm 2. Filling the missing values

```
function
    array ← Dataset.csv
Missing values replacement:
    For every src:
            Filter dataset by src
            Replace missing values with the last values
            Replace the remaining missing values with the next
            values
    End For
    End of Missing values replacement
End the function
```

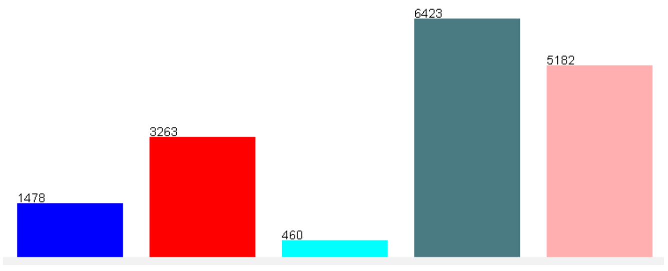


(a) Before dataset balancing

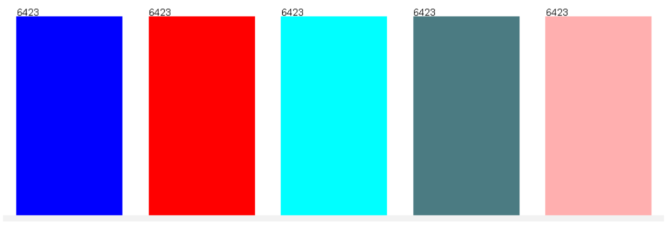


(b) After dataset balancing

Figure 9. The binary classification dataset before and after balancing



(a) Before dataset balancing



(b) After dataset balancing

Figure 10. The multi-class classification dataset before and after balancing

4.3.2 Dataset balancing

In our dataset, the classes are not distributed proportionally since there are 15328 attack samples and 1478 normal samples. Dataset balancing improves the effectiveness of the model. Indeed, the learning algorithms have generally a reduced sensitivity to detecting minority class [36]. Thus, a model trained with an imbalanced dataset will be biased. Class imbalance can be addressed by undersampling the majority class or by oversampling the minority class. Undersampling can cause the removal of significant records if a minority class is very small, which reduces the volume of data allocated for training, testing, and for cross-validation. Oversampling large datasets introduces the possibility to degrade the performance and can cause overfitting if data are not randomized [36].

In this study, we performed minority classes' oversampling utilizing the synthetic minority oversampling technique (SMOTE). Figures 9 and 10 show respectively the binary classification and the multi-class classification datasets before and after balancing.

5. RESULTS AND CLASSIFICATION

5.1 Parameters

A system predicts a True Positive (TP) if it accurately detects an attack which is present, a True Negative (TN) prediction is when there is no attack and the system identifies correctly its absence. A False Positive (FP) takes place when a normal data is predicted as an attack, and a False Negative (FN) is when an attack data is falsely identified as normal.

We carried out an evaluation of our model's performance using different metrics which are: accuracy, recall, precision and F1 score.

$$\text{Accuracy} = (TP+TN)/(TP+TN+FP+FN)$$

$$\text{Recall} = TP/(TP + FN)$$

$$\text{Precision} = (TP)/(TP+FP)$$

$$F1 = 2*(\text{precision}*\text{recall})/(\text{precision}+\text{recall})$$

The performance evaluation parameters are derived from the confusion matrix presented in Table 6.

Table 6. The confusion matrix

	Predicted Positive	**Predicted Negative**
Real Positive	True Positive	False Negative
Real Negative	False Positive	True Negative

5.2 Performance evaluation

We used and tested four classification algorithms: Random Forest, SVM, Naive Bayes, and Multilayer Perceptron. We used 10 folds cross-validation to evaluate the proposed dataset. The model was trained and tested for binary and multi classification.

5.2.1 Binary classification

In the binary classification, we trained firstly the model with 2 classes: the benign class (0) and the malicious class (1). The benign class contains the network traffic from all the benign scenarios, and the malicious class contains the network traffic from the 4 malicious scenarios related to the RPL attacks. We used 10 folds cross-validation to evaluate this model using different algorithms for the classification. Table 7 presents the evaluation results of the binary classification obtained using different algorithms. The optimal accuracy was attained through the use of Random Forest algorithm, with a value of 98.25%. The precision, recall and F1-score results are summarized in Table 8.

Table 9 shows the confusion matrix for the binary classification with Random Forest algorithm.

In order to examine the influence of the Max and Min Rank and Version features, we compared our model when trained with and without these features. The evaluation outcomes are shown in Table 10.

Table 7. Evaluation of the binary classification IDS's performance with various algorithms

Classifier	Benign			Malicious			Accuracy
	Precision	Recall	F1-Score	Precision	Recall	F1-Score	
Random Forest	0.969	0.997	0.983	0.997	0.968	0.982	98.2516%
Naïve Baies	0.876	0.985	0.927	0.983	0.861	0.918	92.2984%
SVM	0.974	0.658	0.785	0.742	0.982	0.845	81.9872%
MP	0.935	0.986	0.960	0.985	0.958	0.958	95.8736%

Table 8. The binary classification performance using Random Forest algorithm

Class	Precision	Recall	F1-Score
0	0.969	0.997	0.983
1	0.997	0.968	0.982
Avg.	0.983	0.983	0.983

Table 9. The binary classification's confusion matrix using Random Forest algorithm

	Predicted Positive	Predicted Negative
Real Positive	15277	51
Real Negative	485	14843

Table 10. Performance of the binary classification IDS with and without Rank and Version features

	Accuracy	Precision	Recall	F1-Score
With Rank & Version Features	98.25%	0.983	0.983	0.983
Without Rank & Version Features	96.663%	0.967	0.967	0.967

Table 11. Performance of IV, BH, DR & HF binary classification IDS with and without Rank and Version features

	Accuracy	Precision (Avg.)	Recall (Avg.)	F1-Score (Avg.)
IV with R & V	98.6515%	0.987	0.987	0.987
IV without R & V	93.9626%	0.940	0.940	0.940
BH with R & V	73.4783%	0.736	0.735	0.734
BH without R & V	62.5%	0.626	0.625	0.624
DR with R & V	99.6886%	0.997	0.997	0.997
DR without R & V	98.0305%	0.980	0.980	0.980
HF with R & V	99.3246%	0.993	0.993	0.993
HF without R & V	99.1509%	0.992	0.992	0.992

Thus, the results show that the use of the Rank and the version features improves the detection's rates of the binary global dataset.

Afterwards, we trained the model with smaller datasets corresponding to each individual RPL attack separately, where the benign and malicious classes correspond to the traffic generated through the simulation of the benign and the malicious scenarios respectively of each attack. We used 10 folds cross-validation for the evaluation. We evaluated each dataset using Random Forest algorithm. Afterwards, we evaluated each attack dataset without the rank and the version features. The results are presented in the Table 11, they show that the use of the Rank & the Version features improves the detection rate of each attack using each attack's dataset.

5.2.2 Multiclass classification

We evaluated the dataset in case of multiclassification using different algorithms. We used 10 folds cross validation for the evaluation. The results are provided in Table 12. The detailed results using Random Forest algorithm are reported in Table 13.

Table 12. Evaluation of the multiclass classification IDS's performance with various algorithms

Classifier	Accuracy	Precision	Recall	F1-Score
Random Forest	96.0704%	0.961	0.961	0.961
Naïve Baies	65.418%	0.724	0.654	0.631
SVM	53.8315%	0.800	0.538	0.554
MP	81.4324%	0.825	0.814	0.818

Table 13. Performance of the IDS with multi-class classification using Random Forest algorithm

Class	Precision	Recall	F1-Score
N	0.896	0.939	0.917
IV	0.992	0.983	0.988
BH	0.936	0.923	0.929
DR	0.992	0.986	0.989
HF	0.990	0.973	0.981
Avg.	0.961	0.961	0.961

5.3 Comparison with other works

We compared our work with the studies discussed in the related works to evaluate its performance. The findings are presented in Table 14.

The findings indicate that our proposed binary classification model using Random Forest algorithm is better in terms of the performance evaluation metrics, with an accuracy of 98.2516%, a precision of 0.983, an F1-score of 0.983 and a recall of 0.983. Several factors may impact the systems' detection performance such as the dataset employed to train the IDS, and the algorithm used for the detection. Foley et al. [36] used a dataset based on network and power features, but it is small sized since it contains 418 instances, which explains the proposed system's performance. In the work proposed by

Momand et al. [1], the dataset balancing was not performed before training and testing, which decreased the detection effectiveness of the proposed system. Furthermore, the features of the datasets used to train the systems proposed by Foley et al. [36], Osman et al. [37], and Ghaleb et al. [38] do not include information about the nodes' rank or version to detect the Decreased Rank attack. Following our results presented previously, the use of the rank and version features contributed in the improvement of our proposed model's detection performance.

Table 14. Comparison of the proposed IDS (binary) with other works

Ref.	Accuracy	Precision	F1-Score	Recall
Foley et al. [36]	87.08%	-	-	-
Momand et al. [1]	90% to 92%	0.96 to 0.98	0.96 to 0.98	0.96 to 0.98
Osman et al. [37]	97.01%	0.9714	0.9536	0.9788
Ghaleb et al. [38]	98%	0.981	-	0.981
Al Sawafi et al. [27]	98%	0.92	0.92	0.92
Our work (binary)	98.2516%	0.983	0.983	0.983

6. DISCUSSION

In this paper, we propose an RPL Intrusion Detection System based on machine learning. We examine the ability of the proposed features to detect four RPL attacks: increased version, blackhole, decreased rank, and Hello Flood. Our proposed model performs binary and multiclassification on these attacks. We used Random Forest classification algorithm because it gives better detection rates in comparison to other tested algorithms. We used 10 folds cross-validation to evaluate the proposed dataset. The routing attacks are successfully detected by our proposed IDS. We compared the proposed model with other existing works. Our proposed binary model has an accuracy of 98.2516 %, a precision of 0.983, an F1-score of 0.983 and a recall of 0.983, which outperforms the other works presented in the related works regarding of the performance evaluation metrics.

We examined the effect of the rank and version features in the improvement of the detection rates of our model. We evaluated the detection rates of the binary classification model when trained with and without the rank and the version features. Furthermore, we trained the model with smaller datasets corresponding to each individual RPL attack separately, and we evaluated the detection's performance of each attack dataset without the rank and the version features. The results show that the use of the Rank and the version features improves the detection's rates of the binary classification model using the binary global dataset, as well as the detection rate of each attack using each attack's dataset.

7. CONCLUSION

In this study, we proposed a machine learning based Intrusion Detection System for RPL protocol. To train our model, we created a dataset based on four real RPL attacks simulated using Cooja, and we performed the preprocessing and the dataset balancing to improve the performance of our proposed Intrusion Detection System. Our model detects successfully four known RPL attacks: Hello Flood, Increased version, Decreased rank and blackhole attacks. The proposed model performs binary and multiclassification of the RPL attacks based on Random Forest algorithm. We evaluated the detection parameters of our model and compared it with some other recent works. The results show that it has an accuracy of 98.25%, a precision of 0.98, an F1-score of 0.98 and a recall of 0.98, which outperforms the other works presented in the related works in terms of the performance evaluation metrics.

Moreover, we compared the performance of our model when trained with and without the rank and the version features, and we concluded that their use improves the detection's rates of the binary classification model using the binary global dataset, as well as the detection rate of each attack using each attack's dataset.

The main issue that we encountered was the lack of datasets for RPL attacks with a good data quality, so we developed a dataset based on real RPL attacks scenarios using Cooja simulator in order to train our IDS.

As a future work, we plan to simulate and collect other RPL attacks types and include them in our model. We plan also to perform the attacks simulation with a bigger number of nodes and more diverse scenarios with multiple attacker nodes and combined attacks. We plan to increase the performance and the accuracy of our model by using ensemble classification and deep learning models, and to attempt to add more features to our model to enable it to detect more RPL attacks.

REFERENCES

[1] Momand, M.D., Khan Mohsin, M., Ihsanulhaq. (2021). Machine learning-based multiple attack detection in RPL over IoT. In 2021 International Conference on Computer Communication and Informatics, ICCCI 2021, Coimbatore, India, pp. 1-8. https://doi.org/10.1109/ICCCI50826.2021.9402388

[2] Perazzo, P., Vallati, C., Arena, A., Anastasi, G., Dini, G. (2017). An implementation and evaluation of the security features of RPL. In 16th International Conference on Ad Hoc Networks and Wireless, ADHOC-NOW 2017, Messina, Italy, pp. 231-238. https://doi.org/10.1007/978-3-319-67910-5

[3] Preda, M., Patriciu, V.V. (2020). Simulating RPL attacks in 6LOWPAN for detection purposes. In 2020 13th International Conference on Communications (COMM), Bucharest, Romania, pp. 239-245. https://doi.org/10.1109/COMM48946.2020.9142026

[4] Medjek, F., Tandjaoui, D., Djedjig, N., Romdhani, I. (2021). Multicast DIS attack mitigation in RPL-based IoT-LLNs. Journal of Information Security and Applications, 61: 102939. https://doi.org/10.1016/j.jisa.2021.102939

[5] Rouissat, M., Belkheir, M., Belkhira, H.S.A., Hacene, S.B., Lorenz, P., Bouziani, M. (2023). A new lightweight decentralized mitigation solution against Version Number Attacks for IoT Networks. Journal of Universal

Computer Science, 29(2): 118-151. https://doi.org/10.3897/jucs.85506

[6] Wallgren, L., Raza, S., Voigt, T. (2013). Routing attacks and countermeasures in the RPL-based Internet of Things. International Journal of Distributed Sensor Networks, 9(8): 794326. https://doi.org/10.1155/2013/794326

[7] Bhalaji, N., Hariharasudan, K.S., Aashika, K. (2020). A trust based mechanism to combat blackhole attack in RPL protocol. In ICICCT 2019–System Reliability, Quality Control, Safety, Maintenance and Management, pp. 457-464. https://doi.org/10.1007/978-981-13-8461-5_51

[8] Gothawal, D.B., Nagaraj, S.V. (2019). Intrusion detection for enhancing RPL security. Procedia Computer Science, 165: 565-572. https://doi.org/10.1016/j.procs.2020.01.051

[9] Violettas, G., Simoglou, G., Petridou, S., Mamatas, L. (2021). A Softwarized Intrusion Detection System for the RPL-based Internet of Things networks. Future Generation Computer Systems, 125: 698-714. https://doi.org/10.1016/j.future.2021.07.013

[10] Belavagi, M.C., Muniyal, B. (2020). Multiple intrusion detection in RPL based networks. International Journal of Electrical and Computer Engineering, 10(1): 467-476. https://doi.org/10.11591/ijece.v10i1.pp467-476

[11] Aydogan, E., Yilmaz, S., Sen, S., Butun, I., Forsstrom, S., Gidlund, M. (2019). A central Intrusion Detection System for RPL-based industrial Internet of Things. In IEEE International Workshop on Factory Communication Systems - Proceedings, WFCS, Sundsvall, Sweden, pp. 1-5. https://doi.org/10.1109/WFCS.2019.8758024

[12] Zahra, F., Jhanjhi, N.Z., Brohi, S.N., Khan, N.A., Masud, M., AlZain, M.A. (2022). Rank and wormhole attack detection model for RPL-based Internet of Things using machine learning. Sensors, 22(18): 6765. https://doi.org/10.3390/s22186765

[13] Morales-Molina, C.D., Hernandez-Suarez, A., Sanchez-Perez, G., Toscano-Medina, L.K., Perez-Meana, H., Olivares-Mercado, J., Portillo-Portillo, J., Sanchez, V., Garcia-Villalba, L.J. (2021). A dense neural network approach for detecting clone ID attacks on the RPL protocol of the IoT. Sensors, 21(9): 3173. https://doi.org/10.3390/s21093173

[14] Choukri, W., Lamaazi, H., Benamar, N. (2020). RPL rank attack detection using Deep Learning. In 2020 International Conference on Innovation and Intelligence for Informatics, Computing and Technologies, 3ICT, Sakheer, Bahrain, pp. 1-6. https://doi.org/10.1109/3ICT51146.2020.9311983

[15] Cakir, S., Toklu, S., Yalcin, N. (2020). RPL attack detection and prevention in the Internet of Things networks using a GRU based deep learning. IEEE Access, 8: 183678-183689. https://doi.org/10.1109/ACCESS.2020.3029191

[16] Napiah, M.N., Bin Idris, M.Y.I., Ramli, R., Ahmedy, I. (2018). Compression header analyzer Intrusion Detection System (CHA - IDS) for 6LoWPAN communication protocol. IEEE Access, 6: 16623-16638. https://doi.org/10.1109/ACCESS.2018.2798626

[17] Pasikhani, A.M., Clark, J.A., Gope, P., Alshahrani, A. (2021). Intrusion Detection Systems in RPL-based 6LoWPAN: A systematic literature review. IEEE Sensors Journal, 21(11): 12940-12968. https://doi.org/10.1109/JSEN.2021.3068240

[18] Seyfollahi, A., Ghaffari, A. (2021). A review of Intrusion Detection Systems in RPL routing protocol based on machine learning for Internet of Things applications. Wireless Communications and Mobile Computing, 2021(1): 8414503. https://doi.org/10.1155/2021/8414503

[19] Sarhan, M., Layeghy, S., Moustafa, N., Gallagher, M., Portmann, M. (2024). Feature extraction for machine learning-based intrusion detection in IoT networks. Digital Communications and Networks, 10(1): 205-216. https://doi.org/10.1016/j.dcan.2022.08.012

[20] Kaur, B., Dadkhah, S., Shoeleh, F., Neto, E.C.P., Xiong, P., Iqbal, S., Lamontagne, P., Ray, S., Ghorbani, A.A. (2023). Internet of Things (IoT) security dataset evolution: Challenges and future directions. Internet of Things, 22: 100780. https://doi.org/10.1016/j.iot.2023.100780

[21] Harbi, Y., Aliouat, Z., Refoufi, A., Harous, S. (2021). Recent security trends in Internet of Things: A comprehensive survey. IEEE Access, 9: 113292-113314. https://doi.org/10.1109/ACCESS.2021.3103725

[22] Naskath, J., Sivakamasundari, G., Begum, A.A.S. (2023). A study on different Deep Learning algorithms used in deep neural nets: MLP SOM and DBN. Wireless Personal Communications, 128(4): 2913-2936. https://doi.org/10.1007/s11277-022-10079-4

[23] Al-Amiedy, T.A., Anbar, M., Belaton, B., Kabla, A.H.H., Hasbullah, I.H., Alashhab, Z.R. (2022). A systematic literature review on machine and deep learning approaches for detecting attacks in RPL-based 6LoWPAN of Internet of Things. Sensors, 22(9): 3400. https://doi.org/10.3390/s22093400

[24] Verma, A., Ranga, V. (2020b). Security of RPL based 6LoWPAN networks in the Internet of Things: A review. IEEE Sensors Journal, 20(11): 5666-5690. https://doi.org/10.1109/JSEN.2020.2973677

[25] Simha, S.N.V., Sahoo, S., Mathew, R., Biradar, R.C. (2020). A review of RPL protocol using contiki operating system. In 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI) (48184), Tirunelveli, India, pp. 259-264. https://doi.org/10.1109/ICOEI48184.2020.9142903

[26] Brandt, A., Hui, J.W., Pister, K.S.J., Vasseur, J., Alexander, R., Levis, P. (2012). RPL: IPv6 routing protocol for low-power and lossy networks. Internet Requests for Comment, RFC Editor, Fremont, CA, USA. http://www.rfc-editor.org/info/rfc6550.

[27] Al Sawafi, Y., Touzene, A., Hedjam, R. (2023). Hybrid Deep Learning-based Intrusion Detection System for RPL IoT networks. Journal of Sensor and Actuator Networks, 12(2): 21. https://doi.org/10.3390/jsan12020021

[28] Kfoury, E., Saab, J., Younes, P., Achkar, R. (2019). A self organizing map Intrusion Detection System for RPL protocol attacks. International Journal of Interdisciplinary Telecommunications and Networking (IJITN), 11(1): 30-43. https://doi.org/10.4018/IJITN.2019010103

[29] Agiollo, A., Conti, M., Kaliyar, P., Lin, T. N., Pajola, L. (2021). DETONAR: Detection of routing attacks in RPL-based IoT. IEEE Transactions on Network and Service Management, 18(2): 1178-1190.

https://doi.org/10.1109/TNSM.2021.3075496

[30] Sharma, S., Verma, V.K. (2021). AIEMLA: Artificial intelligence enabled machine learning approach for routing attacks on Internet of Things. Journal of Supercomputing, 77(12): 13757-13787. https://doi.org/10.1007/s11227-021-03833-1

[31] Mayzaud, A., Badonnel, R., Chrisment, I., Grand Est - Nancy, I. (2016). A taxonomy of attacks in RPL-based Internet of Things. International Journal of Network Security, 18(3): 459-473. https://doi.org/10.6633/IJNS.201605.18(3).07ï

[32] Sharma, S., Verma, V.K. (2021b). Security explorations for routing attacks in low power networks on Internet of Things. Journal of Supercomputing, 77(5): 4778-4812. https://doi.org/10.1007/s11227-020-03471-z

[33] D'Hondt, A., Bahmad, H., Vanhee, J. (2016). RPL Attacks Framework. Mobile and Embedded Computing LINGI2146.

[34] Sahay, R., Geethakumari, G., Modugu, K. (2018). Attack graph - Based vulnerability assessment of rank property in RPL-6LOWPAN in IoT. In 2018 IEEE 4th World Forum on Internet of Things (WF-IoT), Singapore, pp. 308-313. https://doi.org/10.1109/WF-IoT.2018.8355171

[35] Ribera, E.G., Martinez Alvarez, B., Samuel, C., Ioulianou, P.P., Vassilakis, V.G. (2020). Heartbeat-based detection of blackhole and greyhole attacks in RPL networks. In 2020 12th International Symposium on Communication Systems, Networks and Digital Signal Processing, CSNDSP 2020, Porto, Portugal, pp. 1-6. https://doi.org/10.1109/CSNDSP49049.2020.9249519

[36] Foley, J., Moradpoor, N., Ochenyi, H. (2020). Employing a machine learning approach to detect combined Internet of Things attacks against two objective functions using a novel dataset. Security and Communication Networks, 2020(1): 2804291. https://doi.org/10.1155/2020/2804291

[37] Osman, M., Mokbal, F.M.M., He, J., Mahiuob, F., Mokbal, M., Zhu, N. (2021). Artificial neural network model for decreased rank attack detection in RPL based on IoT networks. Article in International Journal of Network Security, 23(3): 496-503. https://doi.org/10.6633/IJNS.202105

[38] Ghaleb, B., Al-Dubai, A., Romdhani, I., Ahmad, J., Aldhaheri, T., Kulkarni, S. (2024). ML-driven attack detection in RPL networks: Exploring attacker position's significance. In 2024 International Conference on Information Networking (ICOIN), Ho Chi Minh City, Vietnam, pp. 478-483. https://doi.org/10.1109/ICOIN59985.2024.10572153

[39] Farea, A.H., Küçük, K. (2022). Detections of IoT attacks via machine learning-based approaches with Cooja. EAI Endorsed Transactions on Internet of Things, 7(28): 1-12. https://doi.org/10.4108/eetiot.v7i28.324

[40] Sharma, M., Elmiligi, H., Gebali, F., Verma, A. (2019). Simulating attacks for RPL and generating multi-class dataset for supervised machine learning. In 2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON): 17th-19th October 2019, University of British Columbia, Canada, pp. 0020-0026. https://doi.org/10.1109/IEMCON.2019.8936142

[41] Yusuf Yavuz, F., Ünal, D., Gül, E. (2018). Deep learning for detection of routing attacks in the Internet of Things. International Journal of Computational Intelligence Systems, 12(1): 39-58. https://doi.org/10.2991/ijcis.2018.25905181

[42] Verma, A., Ranga, V. (2020a). Mitigation of DIS flooding attacks in RPL-based 6LoWPAN networks. Transactions on Emerging Telecommunications Technologies, 31(2): e3802. https://doi.org/10.1002/ett.3802

[43] Krari, A., Hajami, A., Jarmouni, E. (2023). Detecting the RPL version number attack in IoT networks using deep learning models. IJACSA) International Journal of Advanced Computer Science and Applications, 14(10): 614-623. https://doi.org/10.14569/IJACSA.2023.0141065

[44] Essop, I., Ribeiro, J.C., Papaioannou, M., Rodriguez, J., Zachos, G., Mantas, G. (2021). Generating datasets for anomaly-based Intrusion Detection Systems in IoT and industrial IoT networks. Sensors, 21(4): 1-31. https://doi.org/10.3390/s21041528

[45] Dazine, J., Maizate, A., Hassouni, L. (2024). RPL attacks simulation and intrusion detection based on machine learning. In International Conference on Connected Objects and Artificial Intelligence, pp. 417-423. https://doi.org/10.1007/978-3-031-70411-6_63