



MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
UNIVERSITY OF ABDELHAMID IBN BADIS - MOSTAGANEM



Faculty of Exact Sciences and Computer Science
Department of Mathematics and Computer Science

Department: Informatique

FINAL YEAR PROJECT
FOR THE AWARD OF MASTER'S DEGREE IN COMPUTER SCIENCE
Option: **Computer Networking and Systems**

TITLE:
SECURING CONNECTED OBJECTS BY TRUST

PRESENTED BY:
NICHOLUS BHAIRA

Mr Mechaoui Moulay Dris	MCA	University of Mostaganem	President
Mr. Miroud Mohamed	MAB	University of Mostaganem	Examiner
Mdme. Abid Meriem	MCB	University of Mostaganem	Advisor

Academic year 2022-2023

Acknowledgements

As I stand here, reflecting on the arduous yet rewarding path that has led me to this moment, I am filled with profound gratitude to the Almighty God, the steadfast source of my strength and inspiration. Through His unwavering love, presence and guidance, I have found purpose, resilience, and faith, which have propelled me forward in my journey.

To my beloved mother, father, and cherished siblings, Tinotenda, Webster, Tatenda, and Molene, I extend my deepest appreciation for your unwavering support and unyielding belief in me. Your love, encouragement, and sacrifices have been the bedrock of my journey. Each of you has played an integral role in shaping the person I have become, and I am eternally grateful for the bond we share.

A special thank you goes to my exceptional advisor, Madam M. Abid. Your constant availability, patience, and belief in my abilities have been instrumental in bringing this work to fruition. Your invaluable guidance, expertise, and support have pushed me beyond my limits, helping me discover new depths within myself.

I would also like to express my gratitude to the members of the Mostaganem cell group and my dear friends. Your prayers, and encouragement have been a source of comfort and strength during moments of uncertainty. Your belief in my story and support have exemplified the power of genuine friendship and the strength of a supportive community.

To all those whose paths have intersected with mine, leaving an indelible mark on my heart, I extend my heartfelt appreciation. Your presence, kind words, and belief in me have fueled and propelled me forward. The impact you have made, no matter how small, has been instrumental in shaping my journey.

Resumé

Au cours des dernières années les objets connectés, ou Internet des objets (IoT), sont de plus en plus utilisés dans de nombreux domaines. Certains de ces objets principalement des capteurs et des actionneurs forment des réseaux à faible consommation d'énergie et à pertes (LLN), qui se caractérisent par les ressources limitées et les connexions instables. Le groupe RoLL (Routing over Low-power and Lossy networks) a développé le protocole RPL pour permettre aux objets connectés de communiquer entre eux, mais il est vulnérable à diverses attaques de sécurité. Dans ce projet, nous explorons le fonctionnement du protocole RPL et analysons la sécurité des objets connectés dans le contexte de ce protocole. Nous examinons les solutions existantes, proposons et évaluons notre propre approche novatrice pour contrer les attaques qui ciblent ce protocole.

Mots clés : IoT, LLN, RPL, Sécurité, Attaque, Apprentissage machine, Confiance.

Abstract

In recent years, connected devices, or the Internet of Things (IoT), have been increasingly used in many fields. Some of these objects mainly sensors and actuators form low-power and lossy networks (LLN), which are characterized by limited resources and unstable connections. The RoLL (Routing Over Low-power and Lossy networks) group has developed the RPL protocol to enable connected objects to communicate with each other, but it is vulnerable to various security attacks. In this project report, we explore the operation of the RPL protocol and analyzes the security of connected objects in the context of this protocol. We assess existing solutions , propose and evaluate our own novel approach to mitigate security attacks that target this protocol.

Keywords : IoT, LLN, RPL, Security , Attack, Machine Learning ,Trust.

List of Figures

1.1	RPL Instances [30]	2
1.2	Traffic patterns [13]	4
1.3	DODAG Construction [17]	6
2.1	Classification of attacks [17]	9
2.2	Proposed Trust Mechanism	18
3.1	Contiki-NG Directory Structure	23
3.2	DIS Flooding Attack	26
3.3	Proposed Solution	32
4.1	Simulation and Data extraction	34
4.2	Python script for feature extraction and data processing	35
4.3	WEKA	35
4.4	Loading Dataset on WEKA	36
4.5	Choosing ML algorithm	37
4.6	Training Model	38
4.7	Testing Systems	41

List of Abbreviations

DAO	Destination Advertisement Object
DAO-ACK	Destination Advertisement Object ACKnowledgement
DAG	Directed Acyclic Graph
DIO	DODAG Information Object
DIS	DODAG Information Solicitation
DODAG	Destination Oriented DAG
ETX	Expected Transmission Count
IETF	Internet Engineering Task Force
IEEE	Institute of Electrical and Electronics Engineers
IPv6	Internet Protocol version 6
IoT	Internet of Things
LLN	Low power and Lossy Network
MRHOF	Minimum Rank with Hysteresis Objective Function
MP2P	Multipoint-to-point
OF0	Objective Function Zero
P2MP	Point-to-multipoint
P2P	Point-to-point
RPL	Routing Protocol for Low power and lossy networks
RoLL	Routing Over Low power and Lossy networks

Contents

1 RPL and its operation	1
1.1 Introduction	1
1.2 Routing Protocol for Low Power and Lossy Networks	1
1.2.1 Objective Function Zero	2
1.2.2 Minimum Rank with Hysteresis Objective Function	2
1.3 Metrics	3
1.4 Traffic patterns	3
1.5 The construction of the DODAG	4
1.5.1 Control messages in RPL	4
1.5.2 Building procedure of the DODAG	5
1.6 Trickle timer	6
1.6.1 Trickle algorithm description	6
1.7 Downward routes and mode of Operation	7
1.8 Conclusion	7
2 RPL security analysis and trust-based solutions	8
2.1 Introduction	8
2.2 Analysis of attacks	9
2.2.1 Direct attacks	9
2.2.2 Indirect attacks	10
2.2.3 Eavesdropping	11
2.2.4 Misappropriation	12
2.2.5 Sub-optimization	12
2.2.6 Isolation	14
2.3 Trust-based solutions	14
2.3.1 Existing trust-based solutions	15
2.4 Conclusion	20
3 Implementation of RPL Attacks and Proposition of Trust-based Solution	21
3.1 Introduction	21
3.2 Working Environment	21
3.2.1 Contiki-NG	21
3.2.2 Python	23
3.2.3 Weka	24
3.3 Implementation of attacks	24
3.3.1 DIS Flooding Attack	25
3.3.2 Selective Forwarding Attack	26
3.3.3 Sinkhole Attack	26
3.3.4 Version Number Attack	27
3.3.5 Sybil Attack	27
3.4 Machine learning	28

3.4.1	Supervised Learning	28
3.4.2	Unsupervised Learning	30
3.4.3	Reinforcement Learning	31
3.5	Proposed Trust-Based System	31
3.6	Conclusion	32
4	Evaluating our Trust-Based Solution	33
4.1	Introduction	33
4.2	Evaluation of the proposed system	33
4.3	Evaluating Model Performance	38
4.4	Testing system	40
4.5	Conclusion	42

Introduction

In recent years, the deployment of connected objects has surged, leading to the exponential growth of the Internet of Things (IoT) and transforming the way people interact with technology. However, this proliferation of IoT devices also introduces security vulnerabilities due to the transmission and storage of sensitive data, making them attractive targets for attackers. As a result, ensuring the integrity and confidentiality of IoT data has become a critical concern, demanding the development of robust protocols and mechanisms.

This report explores the operation of the Routing Protocol for Low-Power and Lossy Networks (RPL) in the context of IoT security. RPL is a routing protocol specifically designed for low-power and lossy networks (LLNs), characterized by limited resources and unstable connections. The aim is to analyze the security challenges associated with RPL, particularly focusing on secure authentication and key management for mitigating internal attacks and enhancing the overall security of the IoT.

The organization of this work is as follows: Chapter 1 provides an introduction to the IoT and its security concerns, emphasizing the need for protocols and mechanisms to protect sensitive data. It also offers an overview of the RPL protocol, highlighting its role in routing data within LLNs and its significance in improving network reliability and efficiency. Chapter 2 examines the security implications of RPL, analyzing existing challenges and vulnerabilities. It explores various security mechanisms that can be employed to enhance the protocol's robustness. Chapter 3 proposes a novel approach to securing RPL using a trust-based model, outlining how trust can be leveraged to strengthen the security of RPL in IoT environments. Chapter 4, titled "Evaluating our Trust-Based Solution," introduces its evaluation process. Finally, we conclude the report by summarizing the key findings, discussing their implications, and providing recommendations for further research and development in the field of IoT security.

By investigating RPL's operation, analyzing its security aspects, and proposing trust-based solutions, this study aims to contribute to the advancement of secure IoT deployments in LLNs.

Chapter 1

RPL and its operation

1.1 Introduction

The **Internet-of-Things (IoT)** which consists of interconnected objects is gaining importance among information and communication technologies. In IoT, objects are interacting with each other in order to reach common goals in many application domains including healthcare, energy management, military, agriculture, supply chain and smart cities, transportation and logistics. Some of these objects are very constrained and form **Low-power and Lossy Networks (LLN)** which are networks that are characterized by low energy consumption in order to ensure these objects have a long life. With these constraints (scarcity of resources and unreliable links) in mind, the IETF RoLL working group has proposed a new protocol called RPL (Routing Protocol for Low power and Lossy Networks) based on IPv6 and specifically designed for Internet of Things networks.

1.2 Routing Protocol for Low Power and Lossy Networks

The RPL protocol is a proactive, distance-vector routing protocol based on IPv6. In RPL, devices are interconnected in a tree topology using **Directed Acyclic Graphs (DAGs)**. A DAG is a graph structure in which the edges are directed and the nodes do not form any loops. RPL uses a specific type of DAG called a Destination-Oriented Directed Acyclic Graph (DODAG) to form the tree topology [17]. The DODAG is rooted at a special node called the DODAG root or sink, which is typically the device with the lowest rank, more resources and fixed location, which acts as a border router or gateway in the network.

RPL differentiates upward and downward traffic based on message direction. In RPL, downward signifies communication from the root to the nodes, while upward indicates information flow from the nodes to the root.

Each **RPL instance** (which is a set of one or more DODAGs that share a RPLInstanceID) is associated to an objective function which is responsible for calculating a rank that enables selecting the best parent or path to reach the destination, depending on a set of metrics and/or constraints such as the shortest path or the quality of the links.

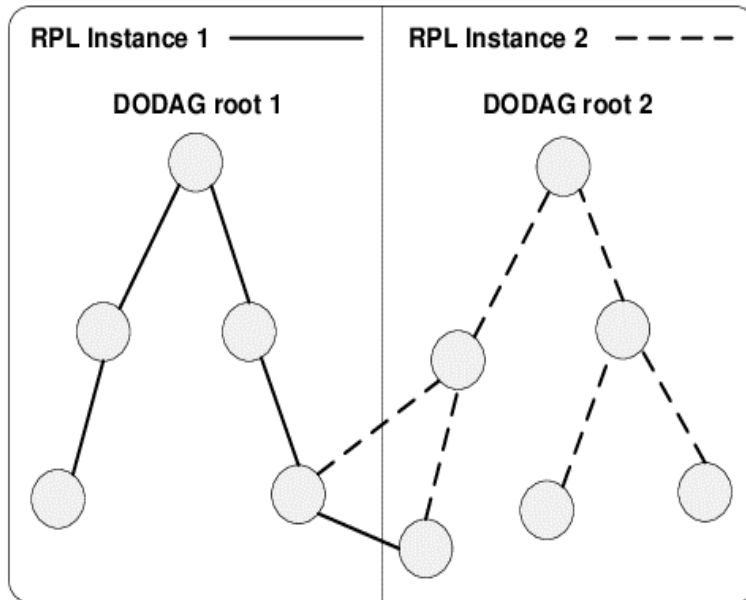


Figure 1.1: RPL Instances [30]

Minimum Rank with Hysteresis Objective Function (MRHOF) [12] and Objective Function Zero (OF0) [31] are the two standardized objective functions in RPL. The OF has as the main function to select and specify the best parent or the optimal path to reach the destination. To ensure path diversity, the rank of each node should increase in a downward direction from the root to the candidate nodes.

1.2.1 Objective Function Zero

OF0 is the default objective function used in RPL that considers hop count as a routing metric to determine the best parents from among the candidate neighbors. When constructing the DODAG, nodes should select the shortest path in terms of hop count to the DODAG root and set their rank to be close to the root. This ensures that the devices with the shortest paths to the root become the preferred parents. [31]

1.2.2 Minimum Rank with Hysteresis Objective Function

OF0 uses a static metric to calculate the node's rank and determine the best parent based on administrative cost, which may select a parent with poor connectivity but fewer hops to the root. To address this issue, the IETF proposed MRHOF, which uses a dynamic link metric (ETX) to achieve topology stability. The ETX metric represents the maximum number of retransmissions required for a packet to be received successfully at its destination. A lower ETX value

indicates a better path. MRHOF selects the lowest-cost path while preventing churn overflow in the network. It does this through two mechanisms: choosing the path with the lowest rank and using hysteresis to ensure that a preferred parent is only selected if its path cost is sufficiently lower than the current path according to a determined threshold.

1.3 Metrics

Two types of routing metrics have been defined by the RPL protocol for calculating paths: i. **node metrics** that considers the attributes relative to a node; ii. and the **link metrics** which takes into account the attributes of links.

1. Node metrics:

- (a) **The number of hops** which is equivalent to the number of nodes crossed to reach the root. The lower the number of hops, the closer the node is to the root;
- (b) **States or attributes** that provide information about the characteristics of the node such as CPU usage, memory consumed;
- (c) **The energy** which represents the energy source of the node (battery, mains, etc.)

2. Link metrics:

- (a) **Latency** which expresses the duration of the transmission of a packet from the sender to the receiver;
- (b) **Link color** which can be a representation of different types of links by abstract values. It is used to avoid or attract specific links for specific types of traffic.
- (c) **Throughput**, which represents the amount of data passing through a link in a unit of time;
- (d) **Link reliability** which is an abstract representation to express the quality of the link, either the ETX (expected number of retransmissions). ETX is the number of transmissions that a node expects to perform to a destination in order to successfully deliver a pack. The more packets there are, the more the value of the ETX increases and vice versa the more the successful delivery rate increases, the more the value of EXT decreases.

1.4 Traffic patterns

There are three traffic patterns in which RPL packets can be forwarded as shown in the diagram below :

- (a) **multipoint-to-point traffic (MP2P)** if the communication is from leaves to the root via upward routes;
- (b) **point-to-multipoint traffic (P2MP)** if the communication is from the root to leaves using downward routes; and
- (c) **point-to-point traffic (P2P)** if the communication is between two nodes illustrated by black arrows using both upward and downward routes [18].

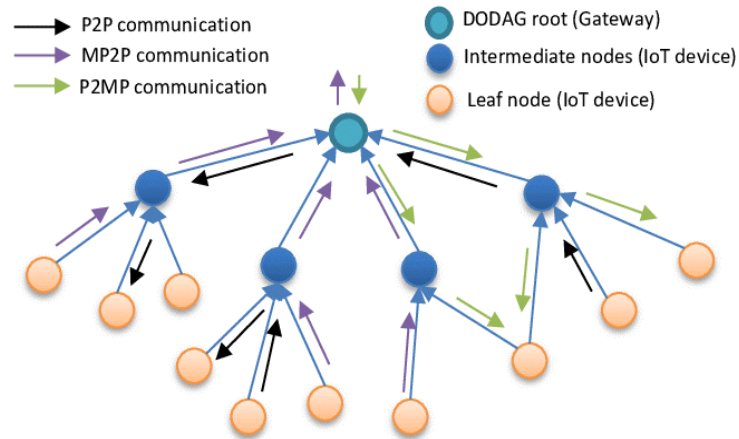


Figure 1.2: Traffic patterns [13]

1.5 The construction of the DODAG

The graph (DODAG) is built gradually from a root node using an objective function (OF). The RPL protocol uses a specific technique to create routing tables (tree or DODAG), and this technique uses four types of messages which are used for the discovery of neighbors, the construction of routes where the packets will circulate and the integration of a new node into the network.

1.5.1 Control messages in RPL

The four types of control messages that are broadcasted in a network using the RPL protocol are:

- (a) **DIO (DODAG Information Object):** the DIO message is periodically broadcasted in multicast initially by the sink node (root) in order to create the DODAG.
- (b) **DAO (DODAG Object Announcement):** these messages are used to build downward routes. After DIO messages have been broadcasted, DAO messages are sent from the leaf nodes to the root, thus

confirming the creation of the DODAG with the routes chosen according to the parameters of the specified metric.

- (c) **DAO-ACK (Destination Advertisement Object Acknowledgement)**: the DAO message is sent by parent nodes to confirm receipt of a DAO message.
- (d) **DIS (Destination Advertisement Object)**: the DIS message is broadcast by a node wanting to join the network. The node that is close to the node wanting to join the DODAG will send it a DIO message, in this way it can send to in turn a DAO message to build a down route (in the storing mode).

1.5.2 Building procedure of the DODAG

With the RPL protocol, the DODAG is built according to an objective function, the metric and the rank (the cost of the link between two network nodes). DODAG construction starts from the root by sending DIO messages to its neighbors. The DIO contains the metric/constraint used by the **objective function** and the rules to join a DODAG (e.g., DIO sending interval). Nodes will receive and process DIO messages potentially from multiple nodes and make a decision to join the graph or not according to the objective function and local policies (if existing). Once a node joins a graph, it automatically has a route towards the root through its parent node. The node then computes its rank within the graph, which indicates its position within the DODAG. If configured to act as a router node, it starts advertising the graph information with the new information to its own neighboring nodes. If the node is a leaf node, it simply joins the graph and does not send any DIO message. The neighboring nodes will repeat this process and perform parent selection, route addition and graph information advertisement using DIO messages. At the end of this process, only upward routes (i.e., to the root) are built. To establish downward routes, a node must send a DAO to its parent containing prefix information of the nodes in its sub-DODAG, when the DAO message arrives to the root, the prefixes are aggregated and the downward routes are then built and made available to the parents, and so on. RPL nodes can also send DIS messages to solicit DIO messages from neighbors. RPL uses the **trickle algorithm** to reduce the DIO messages rate. For example, if the number of DIO messages sent within an interval is not consistent with the network state, RPL resets the trickle timer to a minimum value. Otherwise, if the number of DIO messages is bigger than a certain threshold, the trickle interval (DIO message rate sending) is doubled up to a maximum value [23].

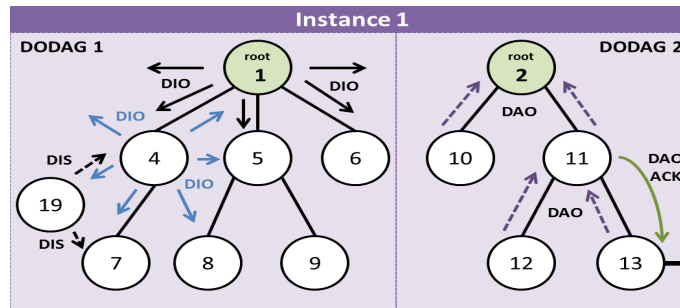


Figure 1.3: DODAG Construction [17]

1.6 Trickle timer

RPL uses the trickle algorithm to reduce the DIO messages rate. For example, if the number of DIO messages sent within an interval is not consistent with the network state, RPL resets the trickle timer to a minimum value. Otherwise, if the number of DIO messages is bigger than a certain threshold, the trickle interval (DIO message rate sending) is doubled up to a maximum value [9]. The Trickle algorithm allows nodes in a lossy shared medium (e.g., low-power and lossy networks) to exchange information in a highly robust, energy efficient, simple, and scalable manner.

1.6.1 Trickle algorithm description

1. Initialization of the interval size I [I_{min} , I_{max}] – that is, greater than or equal to I_{min} and less than or equal to I_{max} .
2. Initialization of variable c (counter) to c to 0 and sets t to a random point in the interval, taken from the range $[I/2, I)$, that is, values greater than or equal to $I/2$ and less than I . The interval ends at I .
3. Whenever Trickle hears a transmission that is "consistent", it increments the counter c .
4. At time t , Trickle transmits if and only if the counter c is less than the redundancy constant k .
5. When the interval I expires, Trickle doubles the interval length. If this new interval length would be longer than the time specified by I_{max} , Trickle sets the interval length I to be the time specified by I_{max} .
6. If Trickle hears a transmission that is "inconsistent" and I is greater than I_{min} , it resets the Trickle timer. To reset the timer, Trickle sets I to I_{min} and starts a new interval as in step 2. If I is equal to I_{min} when Trickle hears an "inconsistent" transmission, Trickle does nothing. Trickle can also reset its timer in response to external "events".

1.7 Downward routes and mode of Operation

The Routing Protocol for Low-Power and Lossy Networks (RPL) supports two modes of operation: the **storing mode** and the **non-storing mode**. The primary difference between the two modes lies in how routing information is stored and maintained.

1. In the **storing mode**, intermediate nodes in the network store routing information for use in forwarding packets. Each node in the DODAG maintains a routing table containing information about its parent and children nodes in the DODAG. In this mode, a DAO message is sent in unicast by the child to the selected parent, which is able to store DAO messages received by its children before sending the new DAO message with aggregate reachability information to its parent. The storing mode can enable or disable multicast mode [11]. This mode is useful in networks with higher traffic volumes, as it reduces the amount of control messages needed to maintain the DODAG structure. However, this mode requires more memory and processing power from intermediate nodes, which can be a problem in resource-constrained networks.
2. In the **non-storing mode**, intermediate nodes do not maintain routing information. Instead, the routing information is stored only at the root node and the destination nodes in the network. Each packet carries information about its destination node, and intermediate nodes forward packets based on this information. In this mode, the DAO message is sent in unicast to the DODAG root, thus, intermediate parents do not store DAO messages, but only insert their own addresses to the reverse route stack in the received DAO message, then forwards it to its parent [11]. This mode is useful in networks with lower traffic volumes, as it reduces the memory and processing requirements for intermediate nodes. However, this mode may require more control messages to maintain the DODAG structure, which can be a problem in networks with a high density of nodes.

1.8 Conclusion

In this chapter we first introduced LLN Networks and their dedicated routing protocol RPL. Through our analysis, we have demonstrated how RPL enables efficient routing in LLN, making it a popular choice for a wide range of IoT applications. In the next chapter delve deeper into the attacks that threatens RPL and some proposed approach that tackle some of these attacks.

Chapter 2

RPL security analysis and trust-based solutions

2.1 Introduction

The characteristics of LLN networks such as resource constraints, lack of infrastructure, limited physical security, dynamic topology and unreliable links makes them vulnerable. The RPL protocol is exposed to a large variety of security attacks.

In this chapter, we will analyze attacks against the RPL protocol paying attention to the element of the RPL network which is impacted and the goals of the attacks.

The security attacks represented in Fig 1. are classified into three categories according to Anth ea Mayzaud [18]. The first category corresponds to attacks targeting the exhaustion of network/node resources (processing, memory and energy) which shorten the lifetime of the devices and hence the lifetime of the RPL network. The second category corresponds to attacks against the network traffic which includes eavesdropping attacks or misappropriation attacks. The third category is comprised of attacks targeting the RPL network topology. They disturb the normal operation of the network.

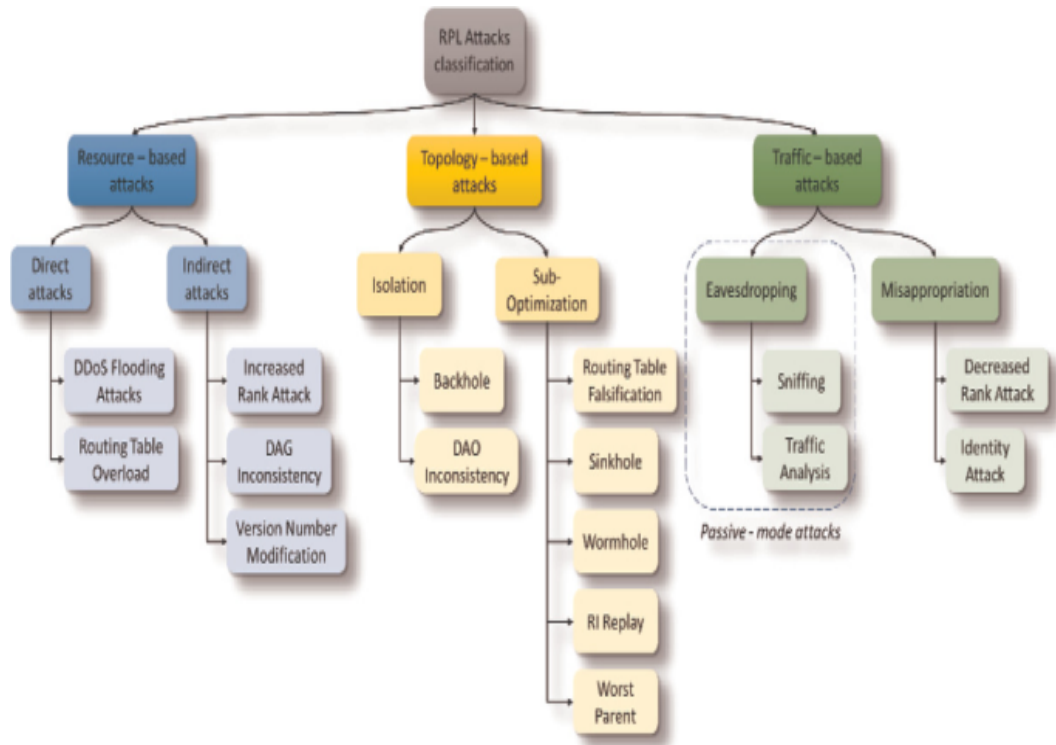


Figure 2.1: Classification of attacks [17]

2.2 Analysis of attacks

Attacks against Resources

The objective of attacks against resources is to drain the energy, memory, or processing power of nodes by compelling them to perform needless tasks, resulting in a depletion of resources. This causes congestion on available links with irrelevant traffic and can impact the network’s availability, reducing the network’s lifespan. There are two subtypes of attacks against resources, which are direct and indirect attacks.

2.2.1 Direct attacks

Direct attacks involve a malicious node deliberately causing an overload in order to weaken the network, and the attacker is solely responsible for the depletion of resources. This can be accomplished through flooding attacks or overloading attacks, both of which are carried out by the attacker [18].

2.2.1.1 Routing Table Overflow

The RPL protocol operates proactively, with RPL router nodes constructing and sustaining routing tables when the storing mode is enabled for those nodes. However, it is feasible to launch direct attacks on resources by overwhelming the RPL routing tables. The approach behind routing table overload involves broadcasting counterfeit routes through DAO messages, which flood the routing table of the intended node. As a result, the creation of new legitimate routes is hindered, adversely affecting network operations, and potentially leading to a memory overflow.

2.2.1.2 Flooding Attacks

A flooding attack involves the creation of an enormous amount of network traffic that causes nodes and links to become unavailable. These types of attacks can be launched by either internal or external attackers. An attacker may choose to send DIS messages to its nearby nodes, which will force them to reset their trickle timer. Alternatively, the attacker may send DIS messages to a specific node, which will prompt the node to respond with DIO messages. In either case, this attack results in network congestion and overloads the RPL nodes, causing saturation [17].

2.2.2 Indirect attacks

In the case of indirect attacks, the attacker will make other nodes generate a large amount of traffic. For instance, such an attack can be performed by building loops in the RPL network so that other nodes produce traffic overhead.

2.2.2.1 Increased Rank Attack

The tactic known as the "increased rank attack" involves intentionally raising the rank value of a RPL node to create loops in the network. In a RPL network, each node is assigned a rank value, indicating its position in the graph structure relative to the root node. The node's rank value always increases downward, maintaining the network's acyclic structure. When a node determines its rank value, it must be greater than the rank values of its parent nodes. To change its rank value, a node must first update its list of parents by removing nodes with a higher rank than its new value. Once a node has established its parent set in the DODAG (Destination-Oriented Directed Acyclic Graph), it selects a preferred parent from the list to optimize routing costs when transmitting packets to the root node. A malicious node advertises a higher rank value than it actually has, and loops are formed when the node's new preferred parent was in its previous sub-DODAG, but provided that the attacker refrains from utilizing loop avoidance mechanisms.

2.2.2.2 DAG Inconsistency

A DODAG inconsistency attack involves a malicious node manipulating the RPL IPv6 header options that monitor DODAG inconsistencies. The goal is to force the target to discard packets, causing denial-of-service and increasing control overhead. This can have a significant impact on the limited energy reserves of constrained devices. A malicious node can use this type of attack to modify all the packets it forwards so that the next-hop node always drops them, which creates a black-hole [19].

2.2.2.3 Version Number Modification

Within RPL, the version number parameter serves as an indicator for global repair operations, and only the root of the DODAG can modify it to reconstruct the network topology. If a malicious node alters the version number, it forces all nodes to begin exchanging control messages. The attacker can exploit this to deplete the limited resources of the nodes and significantly degrade the network's performance [6].

Attacks on Traffic

This second category concerns the attacks targeting the RPL network traffic. It mainly includes eavesdropping attacks on the one hand, and misappropriation attacks on the other hand.

2.2.3 Eavesdropping

The pervasive nature of RPL networks may facilitate the deployment of malicious nodes performing eavesdropping activities such as sniffing and analyzing the traffic of the network.

2.2.3.1 Sniffing

A malicious node listens to the network traffic in order to gain information about the local network topology. The information obtained from the sniffed packets may include partial topology, routing information and data content. In RPL networks, if an attacker sniffs control messages, it can access information regarding the DODAG configuration such as DODAG ID, version number, ranks of the nodes located in the neighborhood. By sniffing data packets, the attacks can not only discover packet content but also have a local view of the topology in the eavesdropped area by looking at source/destination addresses [18].

2.2.3.2 Traffic Analysis

Traffic Analysis provides an indirect means of breaching confidentiality and gaining access to routing information. The objective is, like sniffing attacks, to gather information about the RPL network such as a partial view of the

topology by identifying parent/children relationships. This information allows an attacker to determine traffic flow, node positions, etc. in the network which is used to mount other attacks.

2.2.4 Misappropriation

In misappropriation attacks, the identity of a legitimate node or its performance are overclaimed.

2.2.4.1 Decreased Rank Attack

In case of Decreased Rank Attack, the malicious node falsifies its DIO message by decreasing its rank value and broadcasts the false message. It hence, over claims its proximity to the sink node. As a result neighboring nodes selects the malicious node as their preferred parent, which may increase their Expected Transmission Count. Thus, the malicious node successfully instigates a Decreased Rank Attack [8].

2.2.4.2 Identity Attack

Information gained through sniffed packets can be used by the malicious node to instigate an Identity attack. The malicious node then pretends to be a legitimate node of the network. If the malicious node can clone the root node, it gains control over the entire network topology thus compromising traffic.

2.2.4.3 Sybil attack

In Sybil attack, the malicious node multicasts an excessive number of DODAG Information Solicitation (DIS) messages with different fictitious identities to cause the legitimate nodes to restart the Trickle algorithm frequently and broadcast a large number of DODAG Information Object (DIO) messages. In RPL, DIS and DIO are control messages necessary to build the routing topology. As a result, immoderate receiving and broadcasting control messages drain the limited energy resource of legitimate nodes, and finally cause the legitimate nodes to be unable to communicate and suffer from denial of service [28].

Attacks on Topology

These are attacks that damages the optimal network topology by breaking protocol operations. We distinguish two main categories among these attacks: sub-optimization and isolation.

2.2.5 Sub-optimization

In sub-optimization attacks, the performance of the network may be poor because of the non convergence to optimal paths.

2.2.5.1 Routing Table Falsification

In a routing protocol, it is possible to forge or modify routing information to advertise falsified routes to other nodes. This attack can be performed in the RPL network by modifying or forging DAO control messages in order to build fake downward routes. This can only be done when the storing mode is enabled. For instance, a malicious node advertises routes toward nodes that are not in its sub-DODAG. Targeted nodes have then wrong routes in their routing table causing network sub-optimization. As a result, the path can be longer inducing delay, packet drops or network congestion.

2.2.5.2 Sinkhole

A sinkhole is a compromised node which attempts to capture traffic with the intent to drop messages, thus degrading the end-to-end delivery performance, that is, reducing the number of messages successfully delivered to their destination. The mechanism by which the sinkhole captures traffic occurs when a compromised node performs two malicious acts: First, it attracts legitimate traffic by advertising a favorable route, e.g. through manipulation of the rank field in a Destination Information Object (DIO) message. Second, the sinkhole drops any legitimate data traffic routing through it, degrading the performance of the network [33].

2.2.5.3 Wormhole

Wormhole is an attack in which a malicious actor establishes and controls an out-of-band channel between two distant nodes of the network. Due to its convenience, RPL is induced to use such a channel to forward the traffic. As a result, the malicious actor can control a potentially large amount of traffic and can eavesdrop or discard it. [25].

2.2.5.4 Routing Information Replay

In RPL, the replay attack is mainly performed by replaying control messages rather than data messages. A RPL node can record valid control messages from other nodes and forwards them later in the network. In case of dynamic networks, this attack is quite damaging because the topology and the routing paths are often changed. Replay attacks cause nodes to update their routing tables with outdated data resulting in a false topology and degraded routing performance [17].

2.2.5.5 Worst Parent

This "Rank attack" consists in choosing systematically the worst preferred parent according to the objective function. The outcome is that the resulting path is not optimized inducing poor performance.

2.2.6 Isolation

In case of isolation attacks, the nodes in the network may become isolated and there is a possibility that the nodes are unable to communicate with their parents or with the root.

2.2.6.1 Blackhole

Blackhole attacks perform malicious activities like causing high packet drops and high route and control packet overhead, which depletes the limited resources of the nodes. When malicious nodes propagate blackhole attacks, network latency increases and the ranks of the nodes are altered, which causes a disruption to the RPL network topology and to its stability. Additionally, the rank alteration causes the nodes to recompute their ranks. The rank alteration triggers a local repair—a self-healing mechanism that RPL uses to eliminate local routing loops. However, with the increase in blackhole attacks, the local repair eventually becomes inefficient prompting a global repair by the DODAG root. A continuous initiation of these repair messages causes inefficiencies and disruption to the RPL network [2].

2.2.6.2 DAO Inconsistency

RPL uses some flags which are carried out in IPv6 hop-by-hop option to manage important topological mechanisms. Down ‘O’ flag represents the expected direction of packet, Rank-Error ‘R’ flag indicates rank error in topology, and Forwarding-Error ‘F’ flag represents that the node is not capable of forwarding packet to the set destination. DAO inconsistency is reported by a node when its child node is unable to forward the data to a specified destination, due to unavailability of a route that is learned from fake DAO message (DAO with fake routing information) during topology creation. The attacker exploits this mechanism to perform an attack by setting ‘F’ flag to 1 in the packets and sending it back to its parent. This forces the parent node to discard legitimate available downward routes. DAO inconsistency attack leads to an increase in end-to-end delay, unoptimized topology, and isolation of nodes [32].

2.3 Trust-based solutions

In the existing literature, researchers have proposed various methods to address the security of IoT routing, such as intrusion detection systems, machine learning, and trust-based techniques. Traditional cryptographic methods and complex algorithms that require additional hardware are not feasible for defense against routing attacks due to the energy and processing constraints of IoT devices. Trust-based solutions have emerged as a viable option for securing RPL-based IoT networks, as they offer easy implementation and integration. Trust is a qualitative or quantitative property of a trustee, evaluated by a trustor as a measurable belief, in a subjective or objective manner, for a given task, in

a specific context, for a specific time period [16]. In a trust relationship, the trustor must have confidence in the trustee regarding belief, benevolence, and honesty. This notion of trust is being used in various domains, including sociology and computer science, particularly, communications, networks, and the IoT [21].

In the domain of computer science, the integrity, behavior, and reliability of a sensor node are termed as trust in sensor networks. In the network and communications security, trust is a relation between participating entities. Trust relationship relies on past experiences and current circumstances of entities in the network and determines the efficiency, reconfigurability, and scalability. Certain metrics are used for a node's cumulative trust value estimation, which reflects its legitimacy in the network. The node's indirect or direct neighbours use this trust value to engage this node in network topology and route creation. Trust evolves over time with the changing trust metrics.

2.3.1 Existing trust-based solutions

IoT security has been worked upon continuously with the advent of smart devices. Numerous security solutions and mechanisms, including machine learning-based/deep learning-based [9], Intrusion Detection Systems, cryptographic approaches and trust-based are proposed for IoT routing and network security. For addressing RPL attacks, existing mitigation techniques are typically based on either combining procedures in RPL or modifications in current RPL, for example, revising Objective Function (OF).

In the existing research works, a number of trust models have been proposed for secure routing in IoT. They lack some features, such as consideration of IoT node mobility, heterogeneity in IoT environments, adaptability to IoT networks and routing, and consideration of RPL specific attacks. Furthermore, trust dynamics and network performance are not taken into account in some of the presented solutions. However, some of the papers focus solely on network performance and routing behavior, neglecting to address routing attacks and security concerns. Moreover, critical security attacks, particularly routing attacks in RPL, are not evaluated in some trust-based network security solutions.

2.3.1.1 Trust-based Defence Scheme for Mitigating Blackhole and Selective Forwarding Attacks

In their work [1], they proposed a Trust-based secure RPL routing protocol against Blackhole attacks. They later improved this work to address Selective Forwarding attacks [2]. In both studies, simulation results proved their secure Trust-based system for RPL protocol to be a promising solution to protect RPL from routing attacks. They embedded their Trust-based system in RPL protocol, deployed it and tested their secure protocol against the standard RPL implementation. Based on the test experiments, they provided a proof-of-concept of the validity of their claim that their Trust-based RPL protocol

provides a comprehensive defence (simulation and testbed) against Blackhole and Selective Forwarding attacks [3].

2.3.1.2 Trust aware security mechanism to detect sinkhole attack in RPL-based IoT environment using random forest – RFTRUST

The proposed RFTrust model provides a trust-based lightweight solution for ensuring security in the IoT network. It is primarily designed to address the sinkhole attack in Routing Protocol for Low power and Lossy networks (RPL) based IoT environments. It enhances the trusted routing in the IoT environment by finding and removing sinkhole nodes in the network. The proposed model uses Random Forest (RF) and Subjective Logic (SL) to improve the network performance by identifying sinkhole attack using trust metrics such as delivery ratio, delay, energy consumption, and honesty. The mathematical analysis shows the applicability of the proposed model. The RFTrust model is implemented using Cooja, the Contiki network simulator. The merits of the proposed work are highlighted by comparing performance with the existing similar protocols in terms of delivery ratio, throughput, average delay, energy consumption, false-positive rate, false-negative rate, and detection accuracy [27].

2.3.1.3 PCC-RPL: An efficient trust-based security extension for RPL

The proposed method, which is called PCC-RPL (Parental Change Control RPL), prevents unsolicited parent changes by utilizing the trust concept. In PCC-RPL, all parents monitor their children behavior continuously. When a malicious activity is detected by the parent, it decreases the child's trust level and informs the root by sending a suspicion message. Their simulation results indicate that PCC-RPL can detect almost all common RPL attacks with an acceptable accuracy compared to a well-known method. Low control overhead, low energy consumption, short attack detection delay, and high precision are the main features of the proposed scheme [26]. The low computational overhead and low power consumption features of PCC-RPL combined with high scalability make it an appropriate and viable deployment choice in IoT environments. PCC-RPL has high accuracy, low detection latency in detecting malicious behavior and able to detect worst parent attack, rank promotion and reduction attacks, and wormhole attack.

2.3.1.4 Novel Authentication and Secure Trust based RPL Routing in Mobile sink supported Internet of Things

At first the Novel Authentication and Secure Trust-based RPL Routing in Mobile sink-supported Internet of Things (SecRPL-MS) performs a registration process where all IoT nodes in the network register themselves in the security entity. In their work, the frequent death of IoT nodes is alleviated through deploying mobile sink in the network. If any grid member (GM) node wants

to transmit their data to the grid head (GH) node, then it must undergo authentication process. Secure routing is adopted in RPL by utilising the sail fish optimisation algorithm. Each GM node encrypts its sensed data using the prince algorithm before transmitting it to the GH node. The moving points are selected for the mobile sink using the Quantum Inspired Neural Network (QINN) algorithm. This proposed SecRPL-MS performance is evaluated using the Network Simulator 3 (NS3) in terms of the Packet Delivery Ratio (%), Delay (ms), Energy Consumption (mJ), Key Generation Time (ms) and Malicious Node Detection Accuracy. The proposed SecRPL-MS outperforms 23% of malicious node detection accuracy when compared to existing systems, which represent the proposed SecRPL-MS system providing high security by mitigating the following attacks such as rank attack, Sybil attack, blackhole attack and man in the middle attack [29].

2.3.1.5 SMTrust: Trust-Based Model for Secure Routing against RPL Attacks in Internet of Things

The proposed model considers the mobility metrics for trust computation. It's a trust-based model proposed to improve security in RPL-based IoT , against Rank and Blackhole. SMTrust is evaluated considering static nodes, mobile sender nodes together with a mobile sink node. SMTrust routing algorithms are embedded into RPL, and the protocol is assessed in terms of network performance, including topology stability, throughput, packet loss rate, and power consumption. The proposed protocol trust engine utilizes essential trust metrics to compute the trustworthiness of the nodes. Hence, it attempts to mitigate the effect of routing attacks by detecting and isolating the misbehaving nodes based on their trust rating. The performance of the proposed SMTrust model is evaluated via simulation using ContikiOS/COOJA simulator [22].

2.3.1.6 Trust based mechanism for Securing IoT Routing Protocol RPL against Wormhole and Grayhole Attacks

This proposed method seeks to address sub-optimisation attack called Wormhole and isolation attack called grayhole attack as these attacks can endanger the stability of IoT networks. The traditional cryptographic methods are inoperable for defense against various routing attack as they require high computational, memory and battery power. Hence a Lightweight Trust mechanism is proposed for securing RPL against wormhole and grayhole attacks. The proposed mechanism is energy affable and does not compel undue overhead on network traffic.

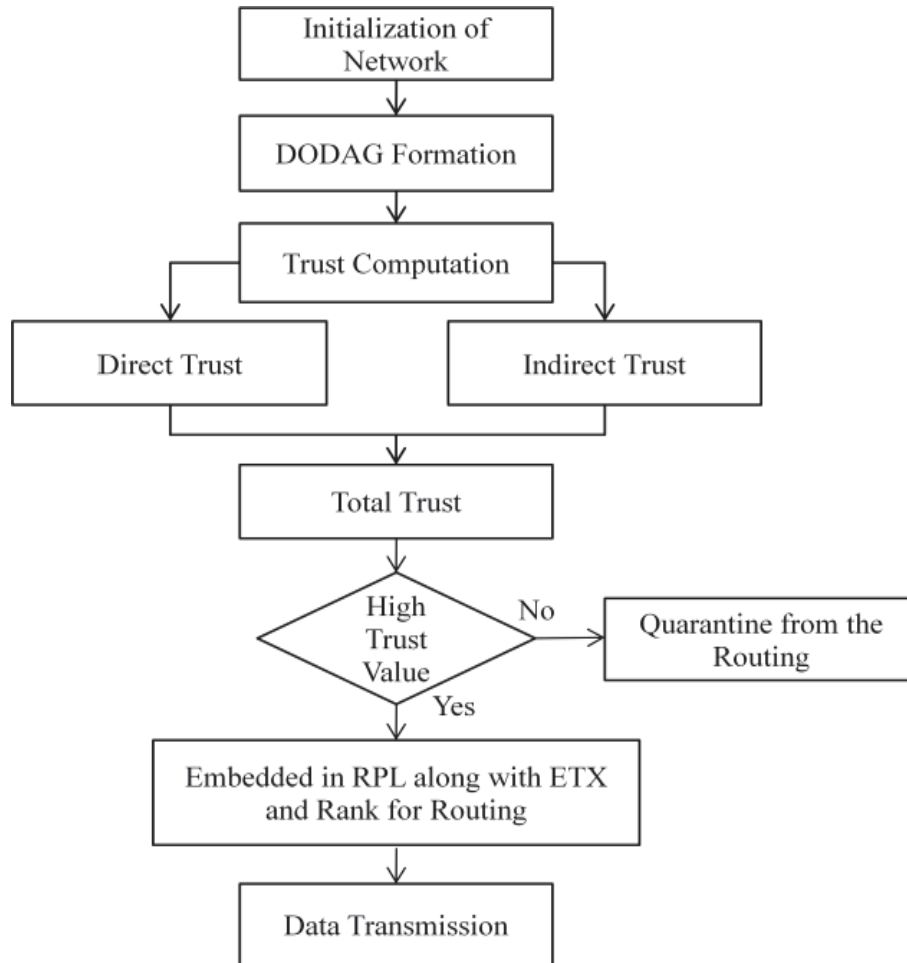


Figure 2.2: Proposed Trust Mechanism

In this mechanism, every node in RPL network monitors its neighboring nodes to check whether they follow the norms of RPL protocol or they deviates from it. Trust computation is based on trust metrics called as Direct Trust (DT) and Indirect Trust (IT). Direct Trust is the estimate of direct trust comes under direct whilst Indirect trust values are collected by node i as other nodes' opinions or recommendations on node j . observation of neighbours Total Trust (TT) is computed by adding values of direct and indirect trust .The computed trust values are then arranged in descending order and are embedded in RPL objective function along with Rank and ETX to route the packets only through trusted nodes and thus malicious nodes are quarantine from the network [20].

Table 2.1: Trust-based Mechanisms for RPL

Mechanism	Techniques	Attacks Addressed	Weaknesses	Validation	Year of Publication
Secure parent node selection scheme in route construction [15]	Trust-based threshold mechanism for node evaluation	Rank attack	Susceptible to attacks like Sybil and Blackhole attacks	Simulation	2015
Trust-based RPL for the Internet of Things [10]	Use of trustworthiness among RPL nodes	Internal and External RPL attacks	No specific attacks are addressed and no simulation or testbed experiment was performed to verify the effectiveness of the Trust-based RPL protocol. Design proposed with no validation performed	Design proposed with no validation performed	2015
RFTRUST: A trust-aware security mechanism to detect sinkhole [27]	Trust establishment and key exchange mechanism among RPL nodes	Sinkhole attacks	Accuracy and reliability heavily rely on the quality and representativeness of the training data used	Simulation	2018
Trust-based mechanism for Securing IoT Routing Protocol RPL [20]	Uses direct trust computed based on node properties and indirect trust	Wormhole, Grayhole attacks	Considers only a single trust metric to evaluate the trust neighbor	Simulation	2018
SecTrust-RPL: A secure trust-aware RPL routing protocol for IoT [4]	SecTrust embedded into RPL routing protocol, detect and isolate attacks	Rank and Sybil attacks	Not assessed for mobile environments and does not consider energy consumption and E2E delays during testing	Simulation, Testbed	2019
SMTrust Trust-Based Secure Routing Protocol [22]	Evaluates critical trust metrics, allows only trustworthy nodes	Selective forwarding and rank attacks	No simulation results to show the effectiveness of the model. Design proposed with no validation performed	Not specified	2020
PCC-RPL: An efficient trust-based security extension for RPL [26]	Monitors children's behavior, decreases trust level on malicious activity	Worst parent, ¹⁹ rank attacks, sinkhole, blackhole, wormhole	System susceptible to internal node compromises; not suitable for battery-powered RPL networks	Simulation	2021

2.4 Conclusion

Studies show that trust-based security for IoT is feasible due to its simple integration and resource-constrained nature of smart devices. Existing trust-based solutions have insufficient consideration of node mobility and recommendation uncertainty. In the following chapter, we propose our own innovative trust-based solution taking into consideration these gaps.

Chapter 3

Implementation of RPL Attacks and Proposition of Trust-based Solution

3.1 Introduction

In this chapter, we focus on the implementation of RPL attacks using Contiki-NG and Cooja, as well as the proposition of our trust-based solution empowered by machine learning algorithms. Our aim was to gain a deeper understanding of the vulnerabilities present in RPL-based networks and explore potential security risks that connected objects face. By implementing these attacks, we aimed to highlight the importance of securing such networks and lay the groundwork for the development of effective countermeasures. This chapter outlines the implementation approach, tools utilized, and the results and analysis of the conducted attacks. Additionally, we introduce a novel trust-based solution that leverages machine learning algorithms to enhance the security of RPL networks.

3.2 Working Environment

3.2.1 Contiki-NG

Contiki-NG(Next Generation) is an open source, cross-platform operating system for severely constrained wireless embedded devices. Contiki-NG was developed as an upgraded version of the original Contiki OS, addressing limitations related to outdated platforms and non-standard protocols. It focuses on dependable (reliable and secure) low-power communications and standardised protocols, such as IEEE 802.15.4 , TSCH, 6LoWPAN, 6TiSCH, RPL, CoAP, MQTT, and LWM2M.

Contiki-NG's primary aims are to :

- facilitate rapid prototyping and evaluation of Internet of Things research ideas,
- reduce time-to-market for Internet of Things applications, and

- provide an easy-to-use platform for teaching embedded systems-related courses in higher education. Contiki-NG started as a fork of the Contiki OS and retains many of its original features. [24]

By utilizing Contiki-NG for our implementation, we were able to take advantage of its rich feature set, including its support for RPL-based networks. This facilitated the simulation and testing of RPL attacks, enabling us to gain insights into the vulnerabilities present in RPL-based networks and the potential security risks faced by connected objects. The flexibility and extensibility of Contiki-NG further empowered us to customize and extend its functionalities to suit our attack scenarios, making it an invaluable tool for our research endeavors.

3.2.1.1 Contiki-NG Directory Structure

Contiki-NG comprises several components that contribute to its functionality and versatility:

- **Operating System (os):** The core of Contiki-NG provides a robust framework for developing applications for resource-constrained IoT devices. It contains the actual Contiki-NG code, including system primitives such as processes, timers, network stack, libraries, and services.
- **Architecture (arch):** Contiki-NG’s architecture defines the system’s structure and interfaces, facilitating seamless integration with different hardware platforms. It contains hardware-dependent code, such as processor, device, and platform drivers. A list of supported platforms can be found under arch/platforms and its subdirectories.
- **examples:** Contiki-NG includes a collection of examples that demonstrate the implementation of various IoT applications. Contains ready-to-use project examples that demonstrate how to use networking, libraries, storage services, etc.
- **tools:** Contiki-NG offers a range of utilities and development tools to assist in creating, debugging, and testing IoT applications. These tools include compilers, debuggers, code editors, and simulation environments. It contains tools that are not included in a Contiki-NG firmware but are intended to run on a computer. These tools include flashing tools, the Cooja simulator (as a submodule), Docker scripts, Vagrant, etc.
- **tests:** It contains all the continuous integration tests. Contiki-NG provides a dedicated testing component that allows developers to verify the functionality, reliability, and performance of their IoT applications.

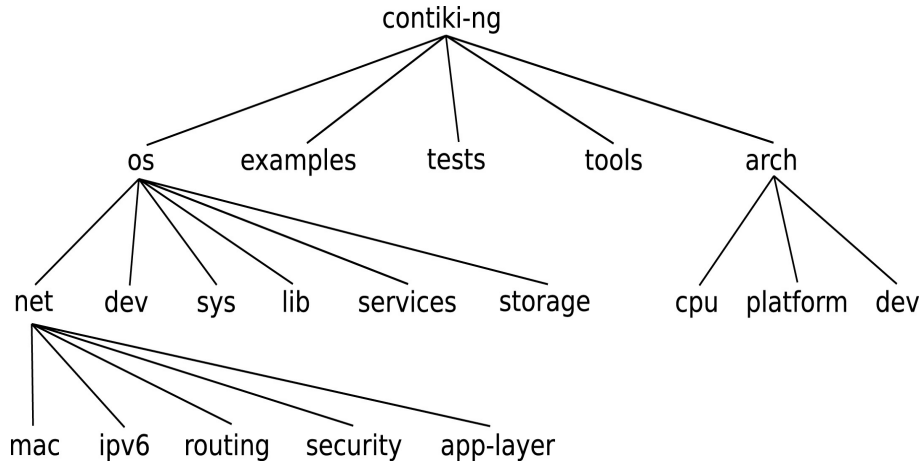


Figure 3.1: Contiki-NG Directory Structure

3.2.1.2 Cooja Simulator

Cooja, as the simulator used in conjunction with Contiki-NG, played a crucial role in our implementation process. It’s an application based on java that provided a graphical interface for configuring and visualizing the simulated RPL-based networks, allowing us to observe the network behavior and assess the effectiveness of our attacks. Cooja’s ability to simulate large-scale networks with multiple nodes and its support for real-time visualization greatly facilitated our research on RPL attacks.

Additionally, Cooja allowed us to monitor and capture network traffic, enabling detailed analysis of the attack effects and their impact on the network performance. This feature proved instrumental in evaluating the success of our attacks and understanding their implications for securing connected objects.

3.2.2 Python

It’s a high-level programming language known for its simplicity and readability. It offers a wide range of libraries and frameworks that make it suitable for various applications, including the implementation of RPL attacks. Python provides extensive support for networking, data manipulation, and machine learning, making it an ideal choice for analyzing and manipulating data in the context of RPL attacks. With its intuitive syntax and vast ecosystem, Python simplifies the implementation process and allows for efficient development and experimentation.

Python played a crucial role in analyzing the collected data. With its versatility, ease of use, and extensive libraries, Python was an ideal choice for developing attack scripts, processing captured network data, and conducting statistical analyses. We utilized Python to create a script for data processing and

feature extraction, leveraging libraries such as pandas and NumPy for efficient data manipulation.

3.2.3 Weka

The WEKA tool is a collection of machine learning algorithms and data pre-processing. It is designed to quickly test existing methods on new datasets in a flexible manner. It provides extensive support for the entire process of experimental data exploration, including data input preparation, statistical evaluation of learning patterns, and visualization of input data and learning results. This diverse and comprehensive toolkit is accessible through a common interface, allowing users to compare different approaches and determine the most suitable method for the problem at hand. [7]

3.3 Implementation of attacks

In this section, we discuss the RPL attacks that have been implemented as part of this research. These attacks aim to demonstrate the vulnerabilities and potential security risks associated with the RPL protocol in IoT environments. By implementing and analyzing these attacks, we gain valuable insights into the weaknesses of RPL and highlight the importance of developing robust security mechanisms for connected objects.

To implement RPL attacks, we utilized the reference implementation provided by Oikonomou, Algahtani, and Tryfonas (2021) in their paper '**A Reference Implementation for RPL Attacks Using Contiki-NG and COOJA**' [5]. They presented a comprehensive framework that combines Contiki-NG and COOJA to simulate RPL attacks on wireless sensor networks. Their work contributes to the understanding and analysis of RPL attacks, providing insights into the vulnerabilities and potential countermeasures.

In their work, they highlight the approach they took to embed code for attacks in Contiki-NG's core modules. Instead of making multiple clones of Contiki-NG's directory for each attack, which would require more storage space, they utilize C preprocessor directives. When we used this approach it allowed us to control which attack code is included during compilation. Additionally, it employs a boolean-like variable to determine when to activate an attack during a simulation. The `script` (javascript code) includes functions for controlling attack timing, accessing memory, and activating/deactivating attacks. These techniques enable efficient customization and facilitate the integration of various attack scenarios into the Contiki-NG framework.

3.3.1 DIS Flooding Attack

Flooding attacks in RPL-based IoT networks aim to exhaust network resources. By utilizing DIS (DODAG Information Solicitation) messages, which are intended to gather DODAG information from neighboring nodes, a compromised node can initiate a flooding attack. This attack triggers neighboring nodes to respond with DIO (DODAG Information Object) messages by resetting their DIO Trickle timers. The malicious node can launch this flooding attack regardless of whether it has already joined the DODAG or not. To carry out the attack, the compromised node reduces the transmission period of its DIS messages, increasing the frequency of solicitation and overwhelming the network with unnecessary traffic.

Algorithm 1: DIS Flooding Attack

Input : Attacker node: A, List of neighbors: L, Current packet: P
Output: Multicast DIS messages to trigger flooding
if $P = DIS$ **then**
 if $Attack_type = unicast$ **then**
 | A.unicast(DIS to N)
 end
 else if $Attack_type = multicast$ **then**
 | A.multicast(DIS to L)
 end
end

To demonstrate the potential for targeted attacks within IoT networks using RPL, we implemented a randomized target selection mechanism within the framework. The attack object's target node was determined using a random number generator, excluding nodes 1 (root) and 0 (undefined) as valid targets. The attack commenced after 5 minutes and lasted for 10 minutes.

During the simulation, it was observed that node 2 was randomly selected as the target node. Node 2 then launched a flood attack by flooding its neighboring nodes with multicast DIS (DODAG Information Solicitation) messages. As a consequence, neighboring nodes, including nodes 7, 6, and 3, were forced to respond by broadcasting DIO (DODAG Information Object) messages in an attempt to maintain network connectivity and stability.

This aggressive behavior by node 2 resulted in a significant increase in multicast traffic within the IoT network, potentially leading to congestion and resource exhaustion.

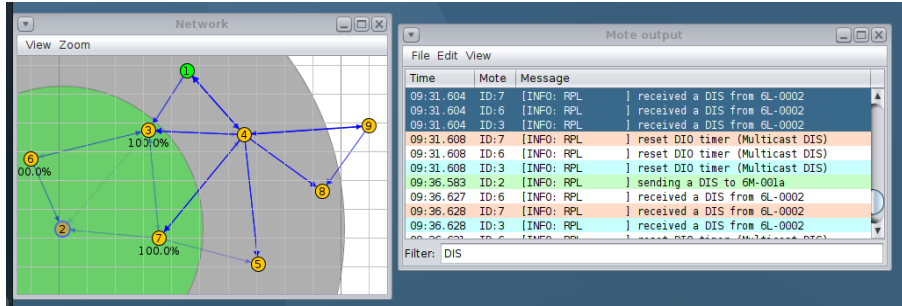


Figure 3.2: DIS Flooding Attack

3.3.2 Selective Forwarding Attack

In this specific attack, a compromised node selectively discards all packets it is supposed to forward, with the exception of control packets. Control packets, such as DAO (Destination Advertisement Object) messages in RPL non-storing mode, are allowed to pass through the compromised node. By strategically dropping critical packets or forwarding them selectively, we highlight the vulnerability of RPL to communication disruption and compromised network performance.

Algorithm 2: Selective Forwarding Attack

Input : Attacker node: A, List of neighbors: L, Current packet: P
Output: Drop or forward packets selectively
if $P.type = DATA$ **then**
 if $A.shouldForwardPacket(P)$ **then**
 | $A.forwardPacket(P)$
 end
end

3.3.3 Sinkhole Attack

In the context of a Sinkhole attack, a malicious node manipulates the routing path by advertising a falsified, more optimal route to its neighboring nodes, with the intention of attracting and intercepting more network traffic. This manipulation is carried out in RPL through the modification of the root's rank value in DIO (DODAG Information Object) messages.

Algorithm 3: Sinkhole Attack

Input : Attacker node: A, List of neighbors: L, Current packet: P
Output: Advertise a better routing path
if $P = DIO$ **then**
 if $P.sender \in L$ **then**
 | A.advertiseBetterPath(P)
 end
end

3.3.4 Version Number Attack

Within RPL, DIO (DODAG Information Object) messages utilize a version number to signify the current version of a DODAG (Destination-Oriented Directed Acyclic Graph). The root node, upon detecting inconsistencies or anomalies, increments the version number to initiate a global repair process. This action involves resetting Trickle timers, clearing routing tables, and initiating the rejoining process of the DODAG. However, in the case of a compromised node, it can maliciously increment the version number without any means for other nodes to verify the integrity of this increase.

Algorithm 4: Version Number Attack

Input : Attacker node: A, List of neighbors: L, Current packet: P,
 Malicious version number: VM
Output: Updated version number VM and multicast DIO messages
if $P = DIO$ and $P.destination = A$ **then**
 | A.DIO[version] \leftarrow VM
 | A.multicast(DIO to L)
 | VM \leftarrow VM + 1
end
else if $P.DIO[version] > VM$ **then**
 | VM \leftarrow P.DIO[version] + 1
 | A.DIO[version] \leftarrow VM
 | A.multicast(DIO to L)
end

3.3.5 Sybil Attack

Sybil Attack involves a malicious node that creates multiple fraudulent identities. These bogus identities can be used to deceive the network by registering fake routes, impersonating legitimate nodes, or circumventing identity-based security measures. As a result, Sybil attacks render many security countermeasures ineffective and pose significant challenges for prevention and detection.

Algorithm 5: Sybil Attack

Input : Attacker node: A, List of neighbors: L, Current packet: P
Output: Perform Sybil attack
if $P.type = ROUTING$ **then**
 if $A.isSybilIdentity(P.source)$ **then**
 A.modifyRoutingInfo(P)
 A.multicastModifiedPacket(P, L)
 end
end

3.4 Machine learning

To realize the potential of our trust-based solution, it is essential to comprehend the fundamentals of machine learning. This section provides an in-depth overview of machine learning techniques, including supervised, unsupervised, and reinforcement learning. By understanding these approaches, we can effectively employ them in our system to detect and mitigate security threats. [14]

Machine learning (ML) plays a pivotal role in our proposed trust-based solution for securing connected objects in IoT environments. It is a powerful technology that empowers computers to learn from data, recognize patterns, and make informed decisions without explicit programming. At its core, machine learning revolves around the **analysis of data to uncover meaningful insights and enable intelligent decision-making**. ML algorithms are trained on labeled data, allowing them to recognize patterns, correlations, and anomalies within vast amounts of information. These algorithms learn from the data, continually refining their models to improve performance and accuracy. In machine learning, we have three main families of models: **supervised learning, unsupervised learning, and reinforcement learning**. These three approaches have distinct characteristics and are used for different purposes in solving various machine learning problems.

3.4.1 Supervised Learning

In supervised learning, the model is trained using labeled data, where the input features are associated with corresponding target labels. The objective is to learn a mapping function that can predict the correct label for new, unseen data points. Examples of supervised learning algorithms include linear regression, logistic regression, support vector machines (SVM), decision trees, and neural networks. Supervised learning is commonly used for tasks such as classification and regression.

3.4.1.1 Decision Tree (DT)

Decision Tree is a supervised learning algorithm that constructs a flowchart-like structure to make decisions based on the input features. It recursively splits the data based on feature conditions, creating a tree structure where each internal node represents a feature test, each branch represents an outcome of the test, and each leaf node represents a class label.

Within the trust-based solution, Decision Tree can be utilized to classify connected objects based on their behavior patterns. By analyzing the attributes and their values, the Decision Tree algorithm can make predictions about the trustworthiness of the objects. The tree's structure allows for interpretability, as each decision path can be traced and understood. Decision Tree models are particularly useful when explainability is crucial, as they provide insights into the decision-making process.

3.4.1.2 Random Forest (RF)

Random Forest is an ensemble learning method that combines multiple decision trees to make predictions. It constructs a set of decision trees using random subsets of the training data and features, and the final prediction is determined by aggregating the predictions of individual trees. Random Forest offers improved accuracy and robustness compared to a single decision tree. It can handle high-dimensional data and mitigate the risk of overfitting. Random Forest can enhance the trust-based solution by providing more accurate and reliable classifications.

3.4.1.3 Naïve Bayes (NB)

Naïve Bayes is a probabilistic classifier based on Bayes' theorem. It assumes that the features are conditionally independent given the class label, which simplifies the modeling process. Naïve Bayes classifiers are computationally efficient and work well even with limited training data. They are particularly useful for text classification tasks but can also be applied to other types of data. In the trust-based solution, Naïve Bayes can be employed to quickly assess the trustworthiness of connected objects based on their behavior patterns.

3.4.1.4 K-Nearest Neighbors (K-NN)

K-Nearest Neighbors is a simple yet effective algorithm that classifies new instances based on the class labels of its k nearest neighbors in the training data. It measures the distance between instances using metrics such as Euclidean distance or cosine similarity. K-NN is a non-parametric algorithm, meaning it does not make assumptions about the underlying data distribution. It is versatile and can handle both classification and regression tasks. In the trust-based solution, K-NN can be used to classify connected objects based on the similarity of their behavior to other known trustworthy or malicious instances.

3.4.1.5 Support Vector Machines (SVM)

Support Vector Machines are powerful classifiers that aim to find an optimal hyperplane that separates the data into different classes while maximizing the margin between them. SVMs can handle high-dimensional data, nonlinear boundaries, and outliers effectively. They are capable of capturing complex relationships in the data and have good generalization performance. SVMs can contribute to the trust-based solution by accurately classifying the behavior of connected objects and detecting potential security threats.

3.4.1.6 Logistic Regression

Logistic Regression is a popular supervised learning algorithm for binary classification tasks. It models the relationship between the input features and the probability of belonging to a particular class. By applying a logistic function to the linear combination of the input features, Logistic Regression estimates the probability of an instance belonging to a specific class. It is suitable for scenarios where the outcome variable is categorical and has two possible outcomes. In the context of our trust-based solution, Logistic Regression can be employed to classify connected objects as trustworthy or malicious based on their behavior patterns and input features.

3.4.1.7 Linear Regression

Linear Regression is a widely used supervised learning algorithm for regression tasks. It models the relationship between the input features and a continuous numerical target variable. By fitting a linear equation to the data, Linear Regression predicts the target variable based on the values of the input features. It is particularly useful for understanding the linear relationship between variables and making predictions based on that relationship. In the context of our trust-based solution, Linear Regression can be utilized to analyze and predict the behavior of connected objects based on various input features.

3.4.2 Unsupervised Learning

Unsupervised learning involves training models on unlabeled data, where the objective is to discover meaningful patterns, structures, or relationships within the data. Unlike supervised learning, there are no predefined labels or targets. Common unsupervised learning algorithms include clustering algorithms like k-means clustering and hierarchical clustering, dimensionality reduction techniques like principal component analysis (PCA) and t-SNE, and association rule mining algorithms. Unsupervised learning is useful for tasks such as clustering, anomaly detection, and data exploration.

3.4.3 Reinforcement Learning

Reinforcement learning focuses on training an agent to interact with an environment and learn optimal actions through trial and error. The agent receives feedback in the form of rewards or penalties based on its actions, and the goal is to maximize cumulative rewards over time. Reinforcement learning algorithms employ techniques like Markov Decision Processes (MDP), Q-learning, and policy gradients.

3.5 Proposed Trust-Based System

In order to detect attacks and enhance the security of our system, we have developed a trust-based solution that leverages machine learning techniques. Our approach aims to identify and mitigate various attacks by implementing a series of steps. These steps involve simulating RPL attacks on Cooja, extracting results using tools like 6LoWPAN Analyzer, preparing the dataset, processing and analyzing the data, and creating a machine learning model for attack detection. By integrating this countermeasure into our system, we can effectively identify and respond to potential threats.

The proposed solution consists of the following steps:

1. **Simulation of RPL attacks on Cooja:** Launch the simulation using Cooja, a network simulator for constrained devices, and simulate RPL attacks by configuring the network nodes accordingly.
2. **Getting results from Cooja using 6LoWPAN Analyzer and other tools:** After the simulation, retrieve the results from Cooja using 6LoWPAN Analyzer. This tool allows you to capture and analyze network traffic in the 6LoWPAN protocol.
3. **Preparation of the dataset:** Convert the obtained results, which may include pcap files and log files, to CSV files. Adapt these CSV files for further analysis using Wireshark, a popular network protocol analyzer.
4. **Data processing and analysis:** Process the CSV files containing network traffic data. Perform feature extraction to extract relevant information from the network traffic, such as packet headers, flow statistics, or any other relevant attributes that can help in detecting RPL attacks.
5. **Creating a model using machine learning algorithms:** Utilize machine learning algorithms to create a model for RPL attack detection. This involves training the model using the processed dataset and selecting appropriate algorithms, such as decision trees, support vector machines, or neural networks. Split the dataset into a training set and a testing set to evaluate the model's performance.

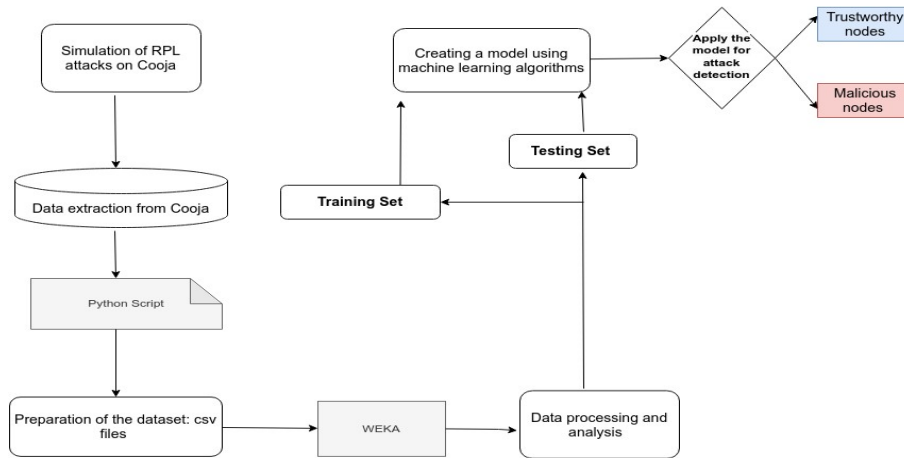


Figure 3.3: Propoposed Solution

3.6 Conclusion

In conclusion, this chapter focused on the implementation of RPL attacks using Contiki-NG and Cooja, as well as the proposition of a trust-based solution empowered by machine learning algorithms. Through the implementation of various RPL attacks, we gained valuable insights into the vulnerabilities present in RPL-based networks and the potential security risks faced by connected objects. The utilization of Contiki-NG provided a robust and flexible framework for simulating and testing these attacks, enabling us to analyze the effects of the attacks on network performance and evaluate their impact.

Furthermore, we introduced a novel trust-based solution that leverages machine learning algorithms to enhance the security of RPL networks. By utilizing Python and the WEKA tool, we were able to process captured network data, extract relevant features, and apply machine learning techniques for anomaly detection and trust evaluation. The combination of machine learning and RPL provides a promising approach for detecting and mitigating attacks in IoT environments. Overall, this chapter serves as a foundation for further research in securing RPL-based networks and developing effective countermeasures against potential threats.

Chapter 4

Evaluating our Trust-Based Solution

4.1 Introduction

The existing literature offers various solutions to address the security challenges in IoT deployments. However, our proposed trust-based solution stands out as a unique approach that harnesses the capabilities of **machine learning** for attack detection.

By utilizing machine learning algorithms, we empower our trust-based system to analyze and classify the behavior of connected objects. This enables us to identify **trustworthy and malicious activities** accurately. Through this innovative approach, we aim to overcome the limitations of existing solutions and provide enhanced security for IoT networks.

Furthermore, our solution not only focuses on detection but also emphasizes the evaluation of its performance. We recognize the importance of concrete results to assess the effectiveness of our system and draw meaningful conclusions. Therefore, rigorous evaluation procedures are an integral part of our approach, allowing us to quantify the impact and capabilities of our trust-based solution accurately.

4.2 Evaluation of the proposed system

The evaluation of the system begins with simulating the network environment using Cooja, an emulator for wireless sensor networks. The simulation allows us to mimic real-world scenarios and study the system's behavior under various conditions. During the simulation, we collect raw data from the network using tools such as 6LoWPAN analyzers and log files obtained from mote output. These tools provide valuable insights into the network traffic, packet durations, source-destination relationships, and other relevant parameters.

The duration of the simulation depends on the specific attack being implemented, as indicated in the JavaScript script provided in the figure. For instance, in the case of a DIS Flooding attack, the attack is initiated after 5 minutes and ceases after 10 minutes. This duration allows us to observe the network's response and analyze the impact of the attack on the system. In this evaluation, Z1 motes have been utilized as the type of motes in the network.

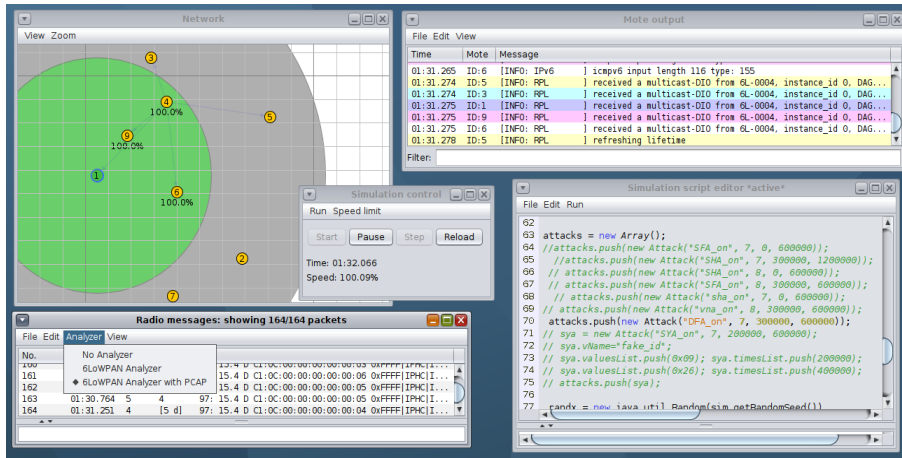
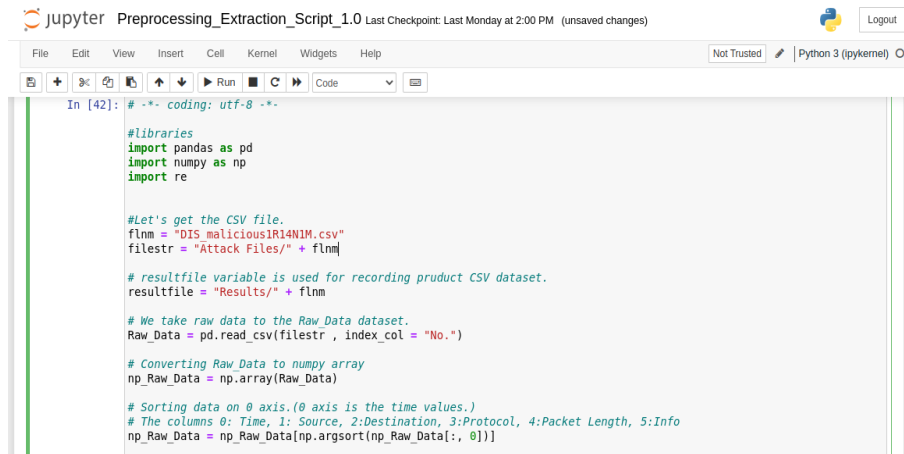


Figure 4.1: Simulation and Data extraction

After obtaining the **log files** and **pcap files** from the simulation, we proceed with further data processing and feature extraction. Using Wireshark, we convert the pcap file to CSV format, which enables easier data manipulation and analysis. Next, we employ a Python script specifically designed for feature extraction and data processing. This script allows us to extract relevant features from the CSV files. The extracted features capture important characteristics of the network traffic and node behavior, which are crucial for training and evaluating machine learning models.

The generated dataset from the Python script includes attributes such as **ID**, **DIO_Count**, **DIS_Count**, **DAO_Count**, **Packets_Sent**, and **Label**. Together, these attributes form a comprehensive dataset that enables feature extraction, analysis, and training of machine learning models.



```
In [42]: # -*- coding: utf-8 -*-

#Libraries
import pandas as pd
import numpy as np
import re

#Let's get the CSV file.
flnm = "DIS_maliciousIR14NIM.csv"
filestr = "Attack Files/" + flnm

# resultfile variable is used for recording product CSV dataset.
resultfile = "Results/" + flnm

# We take raw data to the Raw_Data dataset.
Raw_Data = pd.read_csv(filestr , index_col = "No.")

# Converting Raw_Data to numpy array
np_Raw_Data = np.array(Raw_Data)

# Sorting data on 0 axis.(0 axis is the time values.)
# The columns 0: Time, 1: Source, 2:Destination, 3:Protocol, 4:Packet Length, 5:Info
np_Raw_Data = np_Raw_Data[np.argsort(np_Raw_Data[:, 0])]
```

Figure 4.2: Python script for feature extraction and data processing

Once the data has been processed and the features have been extracted, we create a dataset that can be loaded into **WEKA (Waikato Environment for Knowledge Analysis)**. WEKA is a popular data mining and machine learning software tool that provides a comprehensive environment for data analysis and model training. Within WEKA, we can apply various analysis techniques and train machine learning models using the processed dataset.

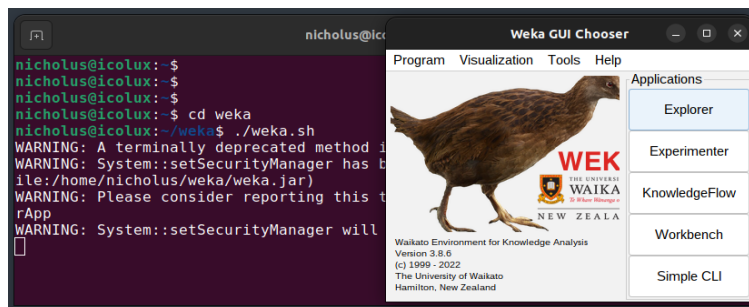


Figure 4.3: WEKA

To load the dataset into WEKA, we follow a simple process. By clicking on "Open File" in WEKA, we can select the desired dataset to load. However, it's important to note that WEKA recognizes the default file format as .arff. Therefore, in order for it to recognize and process our dataset in CSV format, we need to ensure that the file extension is changed from .csv to .arff. This allows it to seamlessly handle and analyze the dataset, providing a smooth workflow for further analysis and training of machine learning models.

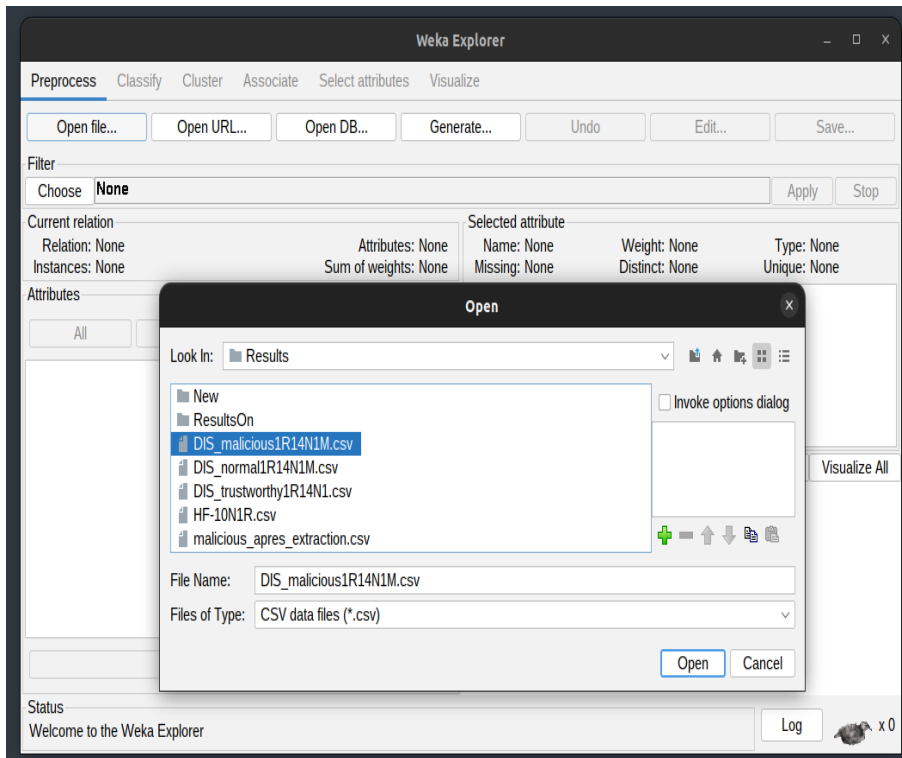


Figure 4.4: Loading Dataset on WEKA

In WEKA's "Classify" tab, we are presented with a variety of algorithms to choose from for our analysis. However, before proceeding, we need to select the target attribute, such as "DIS_Count" or the desired label, to indicate the attribute we want to predict. Once the target attribute is selected, we can proceed to choose the specific algorithm that suits our analysis requirements and specific parameter settings. After selecting the algorithm, we can further configure the settings and choose the desired testing options. Finally, by clicking on the "Start" button, WEKA begins the classification process and generates the output of the classifier, providing valuable insights and predictions based on the selected algorithm and the provided dataset.

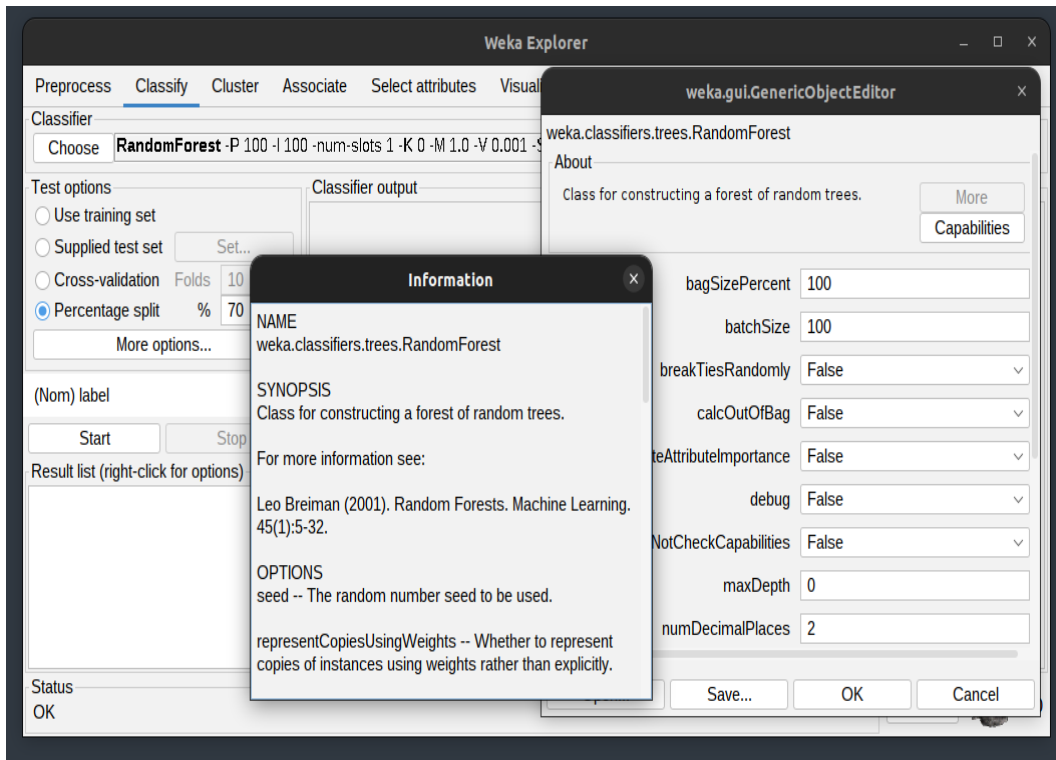


Figure 4.5: Choosing ML algorithm

By following this pipeline, starting from simulation and obtaining log and pcap files, converting to CSV format, performing feature extraction, and utilizing WEKA for analysis and training, we can gain valuable insights into the system's behavior and performance in the presence of different attacks.

4.3 Evaluating Model Performance

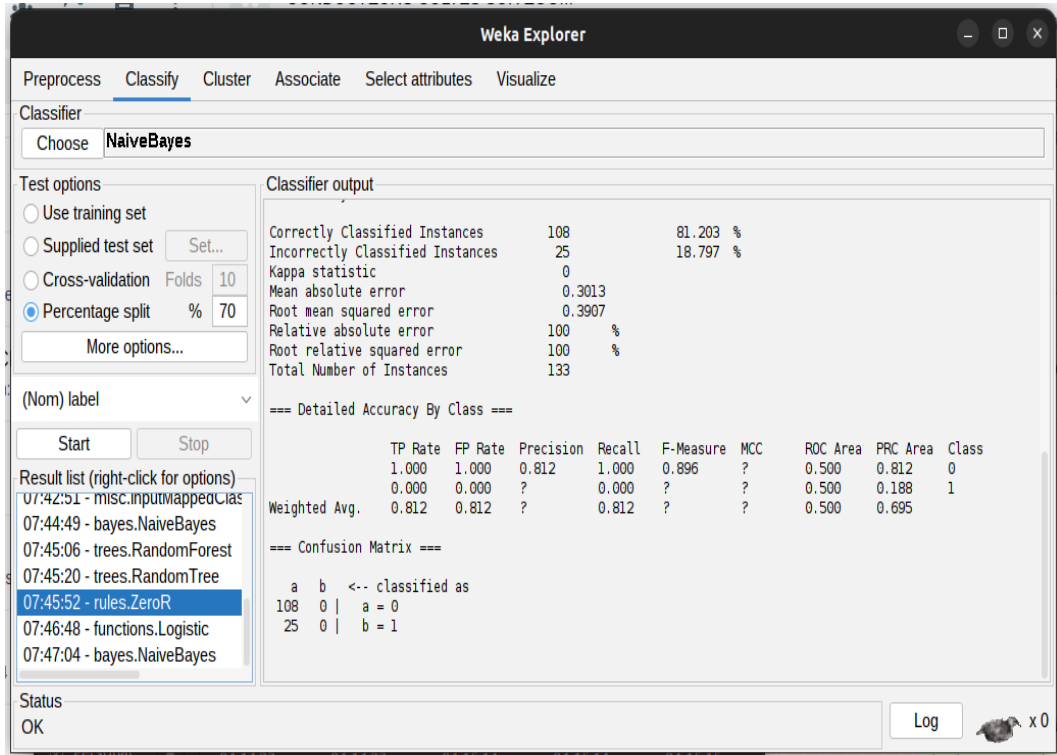


Figure 4.6: Training Model

The output from running the **ZeroR classifier** on the dataset DIS_M2.613.csv is as follows:

- There are 444 instances with 7 attributes including src, dst, packets_sent, DioCount, DisCount, DaoCount, and label.
- The test mode is set to **split 70.0% train, remainder test**.
- The classifier predicts the class value as 0 for all instances and the model was built in 0 seconds.
- During evaluation on the test split, it took 0 seconds to test the model. Out of the total 133 instances in the test split, 119 instances (89.4737%) were **correctly classified**, while 14 instances (10.5263%) were incorrectly classified.

- The mean absolute error is 0.2717, and the root mean squared error is 0.3246.
- The detailed accuracy by class shows that for class 0, the true positive rate, false positive rate, precision, recall, and F-measure are all 1.000, while for class 1, they are all 0.000.
- The weighted average true positive rate and false positive rate are 0.895, indicating relatively higher accuracy for class 0. The confusion matrix shows that all instances belonging to class 0 (119 instances) were classified correctly, but all instances belonging to class 1 (14 instances) were misclassified as class 0.

To calculate the global success rate or accuracy of a classification model, you can use the following formula:

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

In this formula, TP represents the number of instances correctly classified as the target class, TN represents the number of instances correctly classified as not the target class, FP represents the number of instances incorrectly classified as the target class, and FN represents the number of instances incorrectly classified as not the target class.

By summing up the true positives (TP) and true negatives (TN) and dividing it by the total number of instances, including true positives, true negatives, false positives (FP), and false negatives (FN), we can obtain the global success rate or accuracy of the classification model. This measure provides an overall assessment of how well the model performs in terms of correctly classifying instances.

From the confusion matrix in the output, we have:

- TP = 119
- TN = 0
- FP = 0
- FN = 14

Substituting these values into the formula:

$$\text{Success_global} = (119 + 0) / (119 + 0 + 0 + 14) = 119 / 133 \approx 0.8947$$

Therefore, the global success rate or accuracy for the classification model is approximately **0.8947** or **89.47%**.

4.4 Testing system

To save a model in Weka after evaluating its performance, we can follow these steps:

1. In the Weka Explorer interface, we go to the "Classify" tab and select the desired classifier from the list on the left-hand side.
2. We click on the "Save model" button and choose a location and provide a name for the model file.
3. Lastly we click "Save" to save the model. By saving the model, we can reuse it later for predictions on new or unseen datasets without retraining.

To test a dataset using the saved model, we can :

1. Open Weka Explorer and go to the "Classify" tab.
2. Click on the "Load model" button and navigate to the location of the saved model file. Select the model file.
3. Go to the "Preprocess" tab to load your dataset and click on the "Open file" button and select your dataset file.
4. In the "Classify" tab, under the "Test options" section, select "Supplied test set" and click on "Set".
5. Choose the dataset you want to use for testing. Click on the "Start" button to apply the saved model to the dataset and obtain predictions.
6. The predictions can be viewed in the "Classify" tab under the "Test mode" section.

By following these steps, we can test a dataset using the saved model in Weka and observe the predictions made by the model.

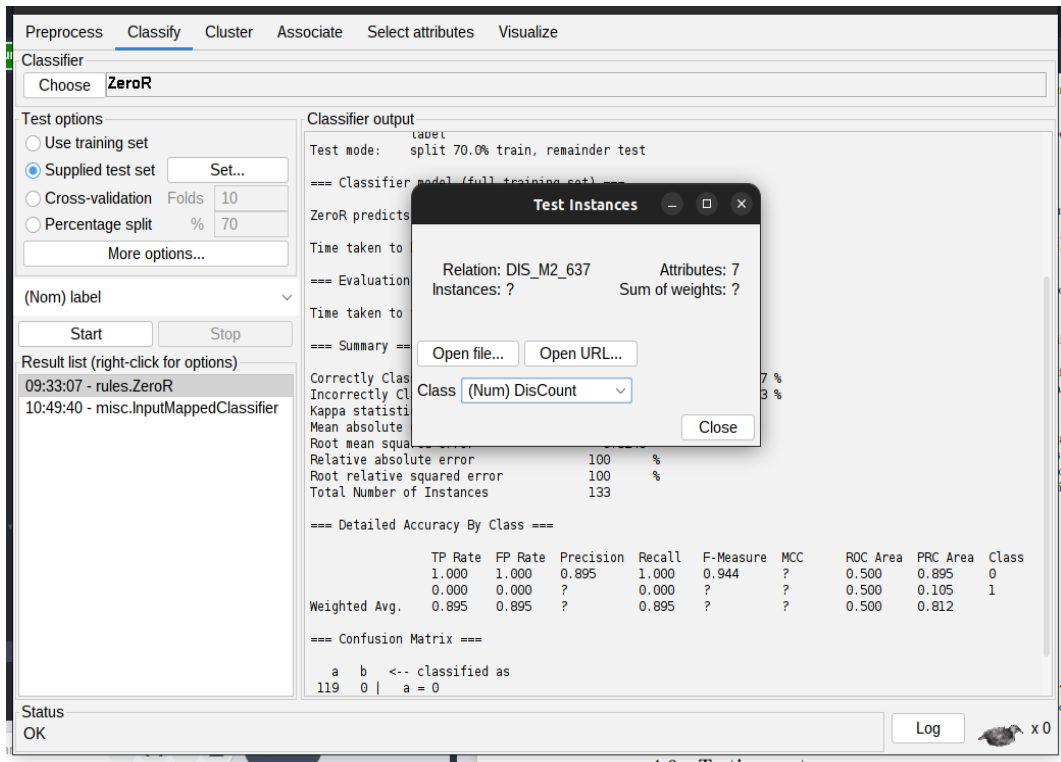


Figure 4.7: Testing Systems

4.5 Conclusion

In conclusion, the evaluation of our trust-based solution has demonstrated its potential to address security challenges in IoT deployments. By leveraging machine learning algorithms, our system effectively analyzes and classifies the behavior of connected objects, enabling accurate identification of trustworthy and malicious activities. Through rigorous evaluation procedures, we have obtained valuable insights into the system's performance.

The evaluation process involved simulating real-world scenarios using Cooja and collecting raw data for analysis. By processing the data, extracting relevant features, and utilizing WEKA for training machine learning models, we achieved a global success rate or accuracy of approximately 89.47%. These results affirm the effectiveness of our trust-based solution in accurately classifying instances and detecting malicious behavior.

Overall, the evaluation has provided strong evidence of the capabilities and reliability of our trust-based solution. The outcomes obtained serve as a foundation for further development and improvement, reinforcing the potential of our approach to enhance the security of IoT networks and mitigate the risks associated with malicious activities.

Conclusion

In conclusion, this work addresses the urgent security challenges faced by IoT networks and proposes a trust-based system empowered by machine learning techniques. The context of the work revolves around the current state of IoT networks and the alarming security issues they encounter, including the need to safeguard networks and protect users' data privacy and integrity.

The main issue addressed in this research is the development of an effective solution to enhance IoT network security. By leveraging machine learning algorithms, the proposed trust-based system enables IoT networks to adapt, learn from network traffic data, and detect anomalies and potential threats in real-time. Through simulated network environments and extensive data analysis, we assessed the system's performance in detecting security threats, such as DIS Flooding, achieving promising accuracy, precision, and recall rates. The evaluation study resulted in a global success rate or accuracy of approximately 89.47%, highlighting the effectiveness of our trust-based system in accurately classifying instances and safeguarding IoT networks..

In future work, we aim to further improve our solution to enhance its capabilities. Specifically, our focus will be on expanding the range of detected attacks and enabling nodes to assign low trust values or scores to untrustworthy nodes. Additionally, we plan to update neighboring nodes to avoid communication with nodes that exhibit malicious behavior.

Overall, this research provides a comprehensive approach to tackle the security concerns associated with IoT networks. The integration of machine learning algorithms and the adoption of a trust-based system offer significant potential to enhance anomaly detection and threat response. Our future efforts will be dedicated to advancing the solution to provide even stronger security measures for IoT networks, ensuring the privacy and integrity of connected objects and users' data.

Bibliography

- [1] David Airehrour, Jairo Gutierrez, and Sayan Kumar Ray. Securing rpl routing protocol from blackhole attacks using a trust-based mechanism. In *2016 26th International Telecommunication Networks and Applications Conference (ITNAC)*, pages 115–120. IEEE, 2016.
- [2] David Airehrour, Jairo Gutierrez, and Sayan Kumar Ray. A trust-aware rpl routing protocol to detect blackhole and selective forwarding attacks. *Journal of Telecommunications and the Digital Economy*, 5(1):50–69, 2017.
- [3] David Airehrour, Jairo Gutierrez, and Sayan Kumar Ray. A trust-based defence scheme for mitigating blackhole and selective forwarding attacks in the rpl routing protocol. *Journal of Telecommunications and the Digital Economy*, 6(1):41–49, 2018.
- [4] David Airehrour, Jairo A. Gutierrez, and Sayan Kumar Ray. Sectrust-rpl: A secure trust-aware rpl routing protocol for internet of things. *Future Generation Computer Systems*, 93:860–876, 2019.
- [5] Faya Algahtani, Theo Tryfonas, and George Oikonomou. A reference implementation for rpl attacks using contiki-ng and cooja. In *2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 280–286, 2021.
- [6] Ahmet Aris, Sema F Oktug, and S Berna Ors Yalcin. Rpl version number attacks: In-depth study. In *NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium*, pages 776–779. IEEE, 2016.
- [7] Francisco Azuaje, Ian Witten, and Frank E. Witten ih, frank e: Data mining: Practical machine learning tools and techniques. *Biomedical Engineering Online - BIOMED ENG ONLINE*, 5:1–2, 01 2006.
- [8] Mohammed Amine Boudouaia, Adda Ali-Pacha, Abdelhafid Abouaissa, and Pascal Lorenz. Security against rank attack in rpl protocol. *IEEE Network*, 34(4):133–139, 2020.
- [9] Semih Cakir, Sinan Toklu, and Nesibe Yalcin. Rpl attack detection and prevention in the internet of things networks using a gru based deep learning. *IEEE Access*, 8:183678–183689, 2020.
- [10] Nabil Djedjig, Djamel Tandjaoui, and Faiza Medjek. Trust-based rpl for the internet of things. In *2015 IEEE Symposium on Computers and Communication (ISCC)*, pages 962–967, 2015.
- [11] Olfa Gaddour and Anis Koubaa. Rpl in a nutshell: A survey. *Computer Networks*, 56(14):3163–3178, 2012.

- [12] Omprakash Gnawali and Philip Levis. The minimum rank with hysteresis objective function. Technical report, 2012.
- [13] Mohammad Aminul Hoque, Mahmud Hossain, and Ragib Hasan. Igaas: An iot gateway-as-a-service for on-demand provisioning of iot gateways. In *2020 IEEE 6th World Forum on Internet of Things (WF-IoT)*, pages 1–6. IEEE, 2020.
- [14] Fatima Hussain, Rasheed Hussain, Syed Ali Hassan, and Ekram Hossain. Machine learning in iot security: Current solutions and future challenges. *IEEE Communications Surveys Tutorials*, 22(3):1686–1721, 2020.
- [15] Kenji Iuchi, Takumi Matsunaga, Kentaroh Toyoda, and Iwao Sasase. Secure parent node selection scheme in route construction to exclude attacking nodes from rpl network. In *2015 21st Asia-Pacific Conference on Communications (APCC)*, pages 299–303, 2015.
- [16] Upul Jayasinghe, Gyu Myoung Lee, Tai-Won Um, and Qi Shi. Machine learning based trust computational model for iot services. *IEEE Transactions on Sustainable Computing*, 4(1):39–52, 2018.
- [17] Anthéa Mayzaud. *Monitoring and Security for the RPL-based Internet of Things*. PhD thesis, Université de Lorraine, 2016.
- [18] Anthea Mayzaud, Remi Badonnel, and Isabelle Chrisment. A taxonomy of attacks in rpl-based internet of things. *International Journal of Network Security*, 18(3):459–473, 2016.
- [19] Anthéa Mayzaud, Anuj Sehgal, Rémi Badonnel, Isabelle Chrisment, and Jürgen Schönwälder. Mitigation of topological inconsistency attacks in rpl-based low-power lossy networks. *International Journal of Network Management*, 25(5):320–339, 2015.
- [20] Ruchi Mehta and MM Parmar. Trust based mechanism for securing iot routing protocol rpl against wormhole & grayhole attacks. In *2018 3rd International Conference for Convergence in Technology (I2CT)*, pages 1–6. IEEE, 2018.
- [21] Mohammad Momani and Subhash Challa. Survey of trust models in different network domains, 2010.
- [22] Syeda Mariam Muzammal, Raja Kumar Murugesan, Noor Zaman Jhanjhi, Mamoona Humayun, Ashraf Osman Ibrahim, and Abdelzahir Abdelmaboud. A trust-based model for secure routing against rpl attacks in internet of things. *Sensors*, 22(18):7052, 2022.
- [23] Jad Nassar, Matthieu Berthomé, Jérémy Dubrulle, Nicolas Gouvy, Nathalie Mitton, and Bruno Quoitin. Multiple instances qos routing in rpl: Application to smart grids. *Sensors*, 18(8):2472, 2018.

- [24] George Oikonomou, Simon Duquennoy, Atis Elsts, Joakim Eriksson, Yasuyuki Tanaka, and Nicolas Tsiftes. The contiki-ng open source operating system for next generation iot devices. *SoftwareX*, 18:101089, 2022.
- [25] Pericle Perazzo, Carlo Vallati, Dario Varano, Giuseppe Anastasi, and Gianluca Dini. Implementation of a wormhole attack against a rpl network: Challenges and effects. In *2018 14th Annual Conference on Wireless On-demand Network Systems and Services (WONS)*, pages 95–102, 2018.
- [26] Mohammad Pishdar, Younes Seifi, Mohammad Nasiri, and Mozafar Bag-Mohammadi. Pcc-rpl: An efficient trust-based security extension for rpl. *Information Security Journal: A Global Perspective*, 31(2):168–178, 2022.
- [27] K. Prathapchandran and T. Janani. A trust aware security mechanism to detect sinkhole attack in rpl-based iot environment using random forest – rfrtrust. *Computer Networks*, 198:108413, 2021.
- [28] Cong Pu. Sybil attack in rpl-based internet of things: analysis and defenses. *IEEE Internet of Things Journal*, 7(6):4937–4949, 2020.
- [29] Bandarupalli Rakesh. Novel authentication and secure trust based rpl routing in mobile sink supported internet of things. *Cyber-Physical Systems*, 9(1):43–76, 2023.
- [30] Vu Chien Thang, Pham Viet Binh, and Nguyen Chan Hung. Ipv6 routing protocol for wireless sensor networks: Fundamental concepts and the state of the art.
- [31] Pascal Thubert. Objective function zero for the routing protocol for low-power and lossy networks (rpl). Technical report, 2012.
- [32] Abhishek Verma and Virender Ranga. Security of rpl based 6lowpan networks in the internet of things: A review. *IEEE Sensors Journal*, 20(11):5666–5690, 2020.
- [33] Kevin Weekly and Kristofer Pister. Evaluating sinkhole defense techniques in rpl networks. In *2012 20th IEEE International Conference on Network Protocols (ICNP)*, pages 1–6. IEEE, 2012.