
Article:
Garcia Ribera, Eric, Martinez Alvarez, Brian, Samuel, Charisma et al. (2022) An Intrusion Detection System for RPL-Based IoT Networks. Electronics. 4041. ISSN: 2079-9292

https://doi.org/10.3390/electronics11234041

---

electronics

Article

An Intrusion Detection System for RPL-Based IoT Networks

Eric Garcia Ribera, Brian Martinez Alvarez, Charisma Samuel, Philokypros P. Ioulianou and Vassilios G. Vassilakis *

Department of Computer Science, University of York, York YO10 5GH, UK
* Correspondence: vv573@york.ac.uk

Abstract: The Internet of Things (IoT) has become very popular during the last decade by providing new solutions to modern industry and to entire societies. At the same time, the rise of the industrial Internet of Things (IIoT) has provided various benefits by linking infrastructure around the world via sensors, machine learning, and data analytics. However, the security of IoT devices has been proven to be a major concern. Almost a decade ago, the IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) was designed to handle routing in IoT and IIoT. Since then, numerous types of attacks on RPL have been published. In this paper, a novel intrusion detection system (IDS) is designed and implemented for RPL-based IoT. The objective is to perform an accurate and efficient detection of various types of routing and denial-of-service (DoS) attacks such as *version number attack*, *blackhole attack*, and *grayhole attack*, and different variations of flooding attacks such as *Hello flood attack*, *DIS attack*, and *DAO insider attack*. To achieve this, different detection strategies are combined, taking advantage of the strengths of each individual strategy. In addition, the proposed IDS is experimentally evaluated by performing a deep analysis of the aforementioned attacks in order to study the impact caused. This evaluation also estimates the accuracy and effectiveness of the IDS performance when confronted with the considered attacks. The obtained results show high detection accuracy. Furthermore, the overhead introduced in terms of CPU usage and power consumption is negligible. In particular, the CPU usage overhead is less than 2% in all cases, whereas the average power consumption increase is no more than 0.5%, which can be considered an insignificant impact on the overall resource utilisation.

Keywords: RPL; industrial IoT; intrusion detection; routing attacks; DoS attacks

Citation: Garcia Ribera, E.; Martinez Alvarez, B.; Samuel, C.; Ioulianou, P.P.; Vassilakis, V.G. An Intrusion Detection System for RPL-Based IoT Networks. *Electronics* **2022**, *11*, 4041. https://doi.org/10.3390/electronics11234041

Academic Editors: Panagiotis Radoglou-Grammatikis, Panagiotis Sarigiannidis, Thomas Lagkas and Vasileios Argyriou

Received: 8 November 2022
Accepted: 30 November 2022
Published: 5 December 2022

1. Introduction

The Internet of Things (IoT) could be defined as a concept that extends the connectivity of standard connected devices (e.g., workstations, laptops, and tablets) to the integration of physical objects into a global network by connecting all of them to the Internet [1]. In general, the main purpose of an IoT device is to process, collect, and analyse data generated in the real world in order to provide certain specific services to an individual, to a company, or to a field of study. For example, a body sensor could be used to provide live data for a medical study, CCTV cameras to monitor an area for security purposes, or a meteorological station to supply information on the current and expected weather. At the same time, the industrial IoT (IIoT) marries physical production and operations with smart digital technology, machine learning, and big data analytics [2]. IIoT is currently an active area of research that encompasses industrial applications, including smart manufacturing, robotics, and software-defined production processes [3].

The IIoT industry is rapidly growing year by year, both economically and in the number of devices used. According to the Statista report, the global market for IIoT was sized at over USD 260 billion in 2021 and is estimated to offer an economic impact of more than USD 1 trillion a year by 2028 [4]. Thus, it is becoming more important as new services and features appear. The demand for such devices increases every year, which

translates to the generation of terabytes of new information every second. The increase in the number of IoT devices can be accompanied by numerous benefits for states, companies, and individuals. Nevertheless, as the demand grows, attackers are also becoming more interested, and as a result, security incidents are on the rise [5–7].

In order to ensure the efficient functioning and operation of IoT networks, in 2012, the Internet Engineering Task Force (IETF) proposed a routing protocol standard for IPv6 over Low-power Wireless Personal Area Network (6LoWPAN), which is named IPv6 Routing Protocol for Low-power and Lossy Networks (RPL) [8]. Moreover, the IETF and academia consider RPL as a potential routing protocol for the industrial low-power and lossy networks of field devices [9]. RPL is expected to improve the productivity and safety of industrial plants while simultaneously providing more timely information for efficient plant operations [10]. Since the standardisation of the RPL, there has been an increase in research activity focused on relevant attacks and countermeasures. More specifically, the published attacks on the RPL-based IoT exploit the power limitations, the wireless connectivity, and the lossy network environment in 6LoWPAN in which the devices operate. Traditionally, one of the popular methods of securing Internet-connected systems is to set up appropriate security mechanisms, such as intrusion detection systems (IDS), to be able to detect attacks that target systems and networks [11,12]. Some more recent solutions rely on deep learning techniques for IoT data analysis [13], cloud-based security mechanisms [14], cross-layer intrusion detection [15], reinforcement learning (RL) [16], and machine learning (ML)-based attack detection [17].

For the purpose of performing real or simulated experiments on RPL-based networks, ContikiOS [18] is the operating system (OS) used in the majority of the published literature. ContikiOS focuses on resource-constrained and low-power wireless IoT devices, and it provides a built-in network simulator called Cooja [19]. Recently, as a fork of the original ContikiOS, a new version called Contiki-NG [20] offers a more efficient routing protocol, RPL-Lite, as well as more continuity on the Contiki project.

The increasing interest in IoT devices has made them more appealing to attackers than before. Hence, new security issues must be countered to ensure the correct functioning of the IoT environment [21]. In such environments, denial-of-service (DoS) and routing attacks are two of the common security threats to cope with [22,23]. DoS attacks take advantage of the resource constraint characteristics of the IoT devices by draining the battery or exhausting the resources of the device. In RPL-based networks, examples of DoS attacks are: *Hello flood attack*, *DIS attack*, and *DAO insider attack* [24–26]. On the other hand, routing attacks focus on distorting the routing information in order to cause a negative impact on the network. Examples of these attacks are *version number attack, blackhole attack, greyhole attack, wormhole attack*, and *rank decrease attack* [24,27–29]. To detect the aforementioned attacks, IDSes are being studied and tested on RPL-based networks. However, as there is a wide variety of attacks, this still remains an open research problem. For these reasons, our work proposes a hybrid intrusion detection approach to detect multiple and diverse attacks on RPL-based IoT. Our hybrid approach refers to the combination of multiple strategies and methodologies to provide a general-purpose solution.

The contributions of this work are as follows. (i) We experimentally evaluate and compare the following RPL attacks: Hello flood attack, DIS attack, DAO insider attack, version number attack, blackhole attack, and greyhole attack. This evaluation consists in the implementation of the aforementioned attacks and the experimentation with their behaviour and operation. To compare and analyse these attacks, the following metrics were used: CPU consumption, transmission rate, and reception rate. (ii) We design a hybrid IDS to detect the previously mentioned attacks by using existing techniques and implementing new ones. Detection methods are based on thresholds, on a matching signature strategy, and on heartbeat messages. Thresholds are used for DIS/DAO/Hello attack detection, a signature matching algorithm is used for version number attacks, whereas the blackhole attack is detected using a lightweight heartbeat protocol. In addition, a UDP-based heartbeat protocol for the detection of greyhole attacks is developed. (iii) We implement the

proposed IDS using Contiki-NG and evaluate it with the Cooja tool in terms of its accuracy in attack detection and its performance regarding CPU consumption, transmission and reception rates, and memory consumption.

The rest of the paper is organised as follows. Section 2 presents the background information on basic RPL concepts. Section 3 introduces the existing studies on RPL attacks and on IDSes for the IoT. Moreover, specific improvements for IDSes are explained. Section 4 describes the proposed IDS design by focusing on the following four categories: placement strategy, detection method, security threats, and validation strategy. In addition, we specify the communication protocol for the different components of the hybrid IDS. Section 5 presents an evaluation of the IDS implementation as well as an analysis of the obtained results, for both the IDS design and each of the implemented RPL attacks. Finally, Section 6 concludes the paper and describes possible future research directions.

2. Background

This section briefly presents the required background information on the IoT and RPL. First, we present the basic characteristics of IoT devices and networks. Next, we present the basic operation of RPL and its associated security attacks.

2.1. IoT Device Characteristics

It is widely accepted that typical IoT devices have the following characteristics:

- *Small size*: In order to provide connectivity to small objects, these should have a computer inside them to be able handle this feature. Therefore, IoT devices are typically small.
- *Small battery*: For the majority of IoT devices, due to the need to be small and independent, the battery size must fit accordingly. Therefore, batteries for IoT devices are far from big, which limits the power usage of the devices.
- *Resource constrained*: IoT devices cannot avoid an efficient usage of power. As a consequence, and related to their small batteries, the resources used by these devices are limited (e.g., CPU, memory, or disk storage).

2.2. IoT Network Characteristics

Based on the required characteristics of IoT devices, the network management needed to enable appropriate communication among a group of smart devices is not trivial. Indeed, to establish and maintain a network link, the following aspects must be taken into account:

- *Wireless*: In order to provide a good link between devices, wired connections are usually the best option. However, in some scenarios where there are a lot of devices to be connected, the wired option is not the most appropriate. In these cases, wireless connections are the most popular and the ones that are used by the majority of IoT devices. Although a wireless connection has some advantages, it may have problems with interference, distance, and weather conditions [30].
- *Limited power*: As stated before, IoT devices are resource-constrained machines with small batteries. Because of this, the transmission range is typically not very large, and it becomes even worse when the battery is running low. Hence, the network must be configured correctly in order to reduce to the minimum the possible problems caused by the limited power.
- *Lossy network*: Taking into account the two previous characteristics, it is clear that, depending on the situation, there will be packet loss and the network may become unstable [31].

As said, IoT networks have relevant characteristics that must be taken into account when designing or using a protocol for a specific purpose. Fortunately, nowadays, there is a protocol standard that can help to cope with the main problems of a 6LoWPAN network: the RPL routing protocol. In addition, since some decades ago, IP global addressing has been a concern because of IPv4 address exhaustion [32], and with the increasing number of devices connected to the Internet, there is a need for the transition to IPv6 in order to have

a sufficient number of addresses that can be used to respond to the growing demand each year. IoT devices are not exempt from this problem. Hence, the standard protocols to be used by IoT devices would typically rely on IPv6.

The 6LoWPAN was developed by IETF in order to fulfil the demand of the network usage by small sensors that need to be interconnected with each other [33]. 6LoWPAN networks offer encapsulation and header compression mechanisms in order to transmit IPv6 packets over the network standard IEEE 802.15.4, where the channel conditions are far from perfect.

2.3. Basics of RPL

RPL is a routing protocol proposed by IETF for low-power and lossy networks (LLNs). The protocol is based on IPv6, and the network devices are interconnected in a tree topology, which is named destination oriented directed acyclic graph (DODAG) [8]. An RPL network can be composed of one or more instances, where each instance has a dedicated DODAG. In addition, a DODAG is formed by one root node that operates as the main maintainer of the entire graph, and by the rest of the nodes that adapt their topology according to a set of exchanged messages among all the nodes inside the transmission and reception range. The purpose of the RPL tree topology is to allow the nodes in the network to find the best path based on a *rank* metric. The rank is set by each node, depending on the distance to the root node, and by the link cost (i.e., the quality of the link) of the nodes in range. This metric allows nodes to provide information about their own position to the rest of the nodes in the network to enable the others to choose the best parent. As the distance from the root node increases and the link quality becomes poorer, the node rank will continue to increase, and vice versa. Hence, the RPL protocol provides ICMPv6 control messages, which are sent and received by the network nodes. The purpose of these messages is, for example, to receive information about the neighbours of a node (i.e., nodes in range) or to be able to select the parent by providing the best path to the root node. The relevant RPL control messages are the following [34]:

- *DIO message*: The DODAG information object (DIO) message is broadcasted by the root node at the beginning in order to advertise a DODAG instance. The message contains information that allows the nodes in range to determine, for example, the RPL instance, the instance version, or the rank of the root, so that they can decide whether to join the network or not. Afterwards, DIO messages are sent among nodes to build and maintain the topology.
- *DIS message*: While waiting for a DIO message, which would allow a node to join a network, nodes periodically send DODAG information solicitation (DIS) messages to inform their neighbours of their presence and increase the possibility of joining a DODAG. Upon hearing a DIS message from a neighbour, the node will send a DIO message back to the sender so that it can join the DODAG with knowledge of the required information about the network.
- *DAO message*: If a node decides to join a DODAG, it will select the best parent based on the rank received by hearing DIO messages and the link cost, and it will later send back a destination advertisement object (DAO) message to register itself to the network. DAO messages are always sent to the root node in order to create the downward route from the root to the sender node. However, if the root is not in range, the sender node will send the message to its parent, and the latter will relay the DAO message to the root.

In addition, there are two main mechanisms that allow an RPL instance to heal itself (self-healing mechanisms):

- *Global repair mechanism*: This mechanism is triggered when the root node detects inconsistency in the network; it results in an increase in the version number of the DODAG instance. This new version is announced in the next DIO message sent to the neighbours and forces the whole DODAG to rebuild itself.

- *Local repair mechanism*: In some scenarios, for example, when a node is turned off and so it is not in the network any more, a global repair is not necessary, and it would be too costly in terms of performance. In those cases, a local repair is the best option. It relies on using a different neighbour node for routing instead of the failing one by switching parents or by routing to a new node [35].

2.4. Attacks in RPL-Based Networks

LLNs have some specific characteristics that must be taken into account by protocol standards defined to work in such environments. In addition to these characteristics, security must play a big role in order to resolve or mitigate some concerns that may arise when using resource-constrained devices. Below, we briefly present the most common attacks in RPL networks.

Hello flood attack: This refers to the launch of a flooding attack on the victim by using Hello messages (i.e., the initial message sent from a node to the neighbours in order to join a network). Two variants of the Hello flood attack can be distinguished. Some authors consider that the initial message in RPL is the DIS control message because it is the first message to be sent from a node that wants to join the network [36]. On the other hand, there are others that consider the Hello message to be the message that a node sends to another to finally join the network (i.e., a DIO message) [37]. In this paper, the Hello flood attack is considered as a DIO message flood sent by the attacker to the victim, and the Hello flood attack based on DIS messages is called a DIS attack (see below). By constantly sending DIO messages to the victims, the attacker creates a huge activity in the network and causes the victims to process the DIO message, thus consuming even more CPU time. The network congestion and the increased CPU consumption reduce battery life significantly. This attack can be considered as a DoS attack as the main focus is to drain the battery of the nodes in the network so that they become unavailable.

DIS attack The idea behind the DIS attack is very similar to that of the Hello flood attack in terms of methodology. However, the message to be sent is a DIS control message instead of a DIO message. Depending on the characteristics of the attacker, the DIS attack may be more or less effective than the Hello flood attack. For example, by sending a unicast DIS message to the victim, it is forced to reply with a new DIO message to the attacker, adding more overhead to the communication and causing the attacker node to consume more CPU as it is receiving as many DIO messages as the DIS messages sent [27]. In this case, if the attacker controls a node that must be online as much time as possible, the DIS attack will not be the best option as the malicious node will need to handle more messages and the battery will not last as expected. On the other hand, if the attacker has compromised a node and the objective is to disturb the network, increase the CPU usage of the neighbours, or drain the battery of the compromised node, the DIS attack is the best option because it also drains the battery of the compromised node.

DAO insider attack: Note that DAO messages are sent by a node that wants to be registered in a network to the root node, when switching parents or when the trickle timer is triggered. If correctly transmitted, this will create a downward and upward route to enable the communication between the root and the DAO sender node. During the DAO insider attack, DAO messages are continuously sent by a child node (i.e., the attacker) to the parent. This forces the parent to handle and retransmit the DAO message to the next parent until it reaches the root node. By handling and retransmitting DAO messages, the intermediate nodes will consume more CPU, and thus the battery will be drained faster [26]. In addition, in the RPL implementation of ContikiOS, a DAO-ACK message is sent back to the DAO message sender node in order to provide information about the DAO message reception by the root node. This attack is more specific than the Hello flood attack as it can choose the target more specifically (i.e., parents, in this case). Moreover, the DAO insider attack allows the attacker to harm or disturb nodes that are not in the transmission and reception range.

Version number attack: The version number in a DODAG is a variable used to maintain the DODAG version, which acts as a counter and can only be incremented by the root node. This variable keeps an order inside the DODAG so that if a node receives a higher version of it from the root node, it must renew its DODAG information. On the other hand, if the root receives a higher version number than the expected one, this will indicate that something is wrong with the DODAG, which will then prompt it to launch a global repair in order to rebuild the entire topology. The version number attack exploits the global repair mechanism in the RPL to disturb the network operation [24]. In particular, an attacker may send a higher version number of the DODAG to the neighbours so that a global repair is forced. By doing this, all the nodes in the network will consume more resources in order to handle the DODAG rebuild. In addition, depending on the topology and the distance between the attacker and the root node, loops can be created, thus exhausting the rest of the nodes even more [24].

Blackhole attack: The main purpose of a blackhole attack is to cause an internal DoS to the child nodes by silently discarding all the messages received from the rest of the nodes [27]. Furthermore, a node performing a blackhole attack does not send any messages, as it is a hole that absorbs everything. This behaviour, if executed at the right moment and position (i.e., when the attacker node is the parent of many children), may cause nodes in the downward route from the attacker node to be isolated from the network. Despite the fact that this attack may be effective in some scenarios, it has been shown that RPL's self-healing mechanisms will modify the topology during an attack so that the victims will become unaffected by the attacker, thus reducing the impact of the attack [38].

Greyhole attack: Greyhole or selective forwarding attack acts similarly to a blackhole attack, but instead of blocking every packet received, a greyhole attack blocks all traffic, except for a selected type of message (e.g., RPL control messages). This attack avoids being detected by simple detection mechanisms and also prevents the self-healing mechanisms from fixing the problem.

3. Related Work

This section explores the work accomplished in terms of IDSes focused on the RPL-based networks. In order to have a wide perspective of the work performed on IDSes, the relevant works that studied the previously mentioned RPL attacks in Section 2.4 are also reviewed.

3.1. Attacks in RPL-Based Networks

A great deal of literature work based on attacks in RPL-based networks is presented below. Raoof et al. in [24] presented a large study of well-known RPL attacks. Among the studied attacks, the authors explored blackhole and greyhole attacks, sinkhole attack, Hello flood attack, rank decrease attack, version number attack, and DIS attack. In addition to an explanation of the attacks, countermeasures were also presented. Another survey of RPL attacks was presented by Kamble et al. in [28]. The authors classified the attacks into three categories: attacks against resources (e.g., Hello flood attack and version number attack), attacks on topology (e.g., sinkhole attack and blackhole attack), and attacks on traffic (e.g., rank decrease attack). Later on, they provided security solutions to mitigate attacks based on the RPL topology. Pongle and Chavan [27] presented a survey focused on attacks on RPL and 6LoWPAN. The relevant attacks mentioned in this survey are the selective forwarding attack, sinkhole attack, Hello flood attack, blackhole attack, version number attack, and DIS attack. Moreover, IDS was mentioned as a countermeasure, explaining its limitations and its benefits. Some studies have also been performed on the DIS attack. Pu in [39] presented the spam DIS attack; its main idea was to send a large number of DIS messages in multicast by using multiple fictitious identities to make the victims send back a DIO message to exhaust the affected nodes and the network. Le et al. in [25] also described the DIS attacks as a powerful weapon against network performance. Ghaleb et al. in [26] introduced the DAO insider attack and showed the big impact caused in network performance, power

consumption, and latency. In addition, the authors proposed a new scheme to reduce the effects of such attacks. In terms of the version number attack, a deep study was carried out by Aris et al. in [40], where the authors analysed the attack from multiple points of view. More recent work has been performed in terms of RPL attacks. Ioulianou et al. in [41] presented a study on the possibility of battery-drain DoS attacks on the RPL-based IoT. The authors used the Cooja simulator to implement both the version number attack and the Hello flood attack, and to demonstrate the impact that these caused.

3.2. IDSes for IoT and Wireless Sensor Networks

A lot of work has been carried out in order to comprehend the complex topic of IDSes in computer systems. For example, Milenkoski et al. [42] surveyed some of the key aspects to take into account when evaluating an IDS. In general, the authors showed the common practices applied in IDS evaluation, discussed the most controversial issues and challenges faced when using IDS technologies, and suggested a set of guidelines to follow when planning an IDS study. However, the work performed by the authors did not focus on IDSes for the IoT.

With regard to the IoT, Zarpelao et al. [22] provided a survey on the relevant questions. Firstly, the authors depicted some of the most relevant terms that need to be understood when addressing this topic. Those terms are intrusion detection and the IoT. In the intrusion detection subsection, the authors illustrated the meaning of what an IDS is and described the most important characteristics to comprehend the multiple types of IDSes in general. Later on, in the IoT subsection, an explanation was given of the IoT concept, the standards followed in the IoT industry, and the possibilities offered by the IoT paradigm. Secondly, once the key concepts had been depicted, the authors showed part of the relevant work performed on surveying intrusion detection methods for technologies similar to the IoT. Anantvalee and Jie [43] presented a survey on intrusion detection for mobile ad hoc networks (MANETs) by describing some of the architectures used for IDSes in MANETs. The authors concluded that the majority of the reviewed IDSes were distributed due to the characteristics of MANETs.

In line with the MANET topic, Kumar and Dutta in [44] provided a survey of the most relevant intrusion detection techniques for MANETs, focusing on the technology layout and detection algorithms. The survey classified the detection techniques into nine different categories: statistical-based, heuristic-based, rule-based, state-based, signature-based, reputation-based, routing information-based, cross-layer-based, and graph theory-based. Abduvaliyev et al. in [45] reviewed recent work on IDSes for wireless sensor networks (WSNs) and introduced the following IDS approaches: anomaly detection, specification-based detection, and misuse detection. The authors found that although the IDS topic for WSN was going in the right direction, some core areas, such as the architecture or accuracy vs. resource usage, should be studied more in depth. Modi et al. [46] introduced the topic by reporting multiple threats that could affect the security triad (confidentiality, integrity, and availability) of cloud computing. Moreover, the authors reviewed and compared IDSes used in cloud computing by classifying them according to the network placement and the detection method, and by discussing the pros and cons of each IDS scheme.

As discussed above, some works have focused on the design and development of new IDSes for the IoT paradigm. Raza et al. [47] proposed SVELTE, an IDS specifically developed for the IoT and implemented in ContikiOS. The main objective of this IDS was to detect network layer and routing attacks occurring in wireless LLNs. In order to detect such attacks, SVELTE was designed to comprise three main modules. The first module is the 6LoWPAN Mapper, which collects information about the network to obtain an accurate picture of the entire network and its components. The second module is based on the intrusion detection, whose main focus is to analyse the previously mapped data to detect possible malicious intrusions. Finally, the third module acts as a firewall. Distributed among all the nodes in the network, this firewall prevents external attackers from tampering with the members of the RPL network. The authors concluded that using

SVELTE is feasible in the context of RPL networks as the results showed good performance in terms of false positives and false negatives. However, when using SVELTE, the energy overhead increased by 30%. Despite this fact, depending on the purpose of the scenario, it may be worth using to secure an IoT network.

Another relevant IDS design for the IoT is the one proposed by Cervantes et al. [48]. Their system is called INTI and aims to carry out intrusion detection with respect to sinkhole attacks on 6LoWPAN. More specifically, INTI uses reputation and trust mechanisms in order to detect suspicious behaviour that could lead to sinkhole attacks. In addition, INTI is capable of isolating the attacker by broadcasting a message to the rest of the nodes once the sinkhole attack is detected. Upon receiving this message, benign nodes will change their parents in order to avoid using the malicious node. The experimental results showed that INTI had a false negative rate of about 8%. Moreover, these results showed a better rate of false positives and false negatives when compared with SVELTE.

Another work around the topic of IDSs for IoT was performed by Ioulianou et al. [49]. The authors presented a signature-based IDS design and analysed two DoS attacks (Hello flood attack and version number attack) by using ContikiOS and the Cooja simulator. Additionally, in order to exclude the attacker from the network, the authors proposed an IDS module installed in the border router that would create a new rule in the firewall to remove the attacker from the network if the IDS detected that it was acting as a malicious node. Ioulianou et al. concluded their work by showing the high impact caused by the Hello flood attack and the version number attack. A future work is expected to implement the IDS and to test it against DoS attacks in the Cooja simulator to analyse the performance of this IDS design.

Recently, Pasihkani et al. [16] proposed an adversarial reinforcement learning (ARL) framework to produce efficient intrusion detection systems for RPL-based IoT environments with evolving data. The authors demonstrated that by detecting a 'concept drift' and using a subsequent adaptation, it was possible to identify changes in the RPL network and improve the detection performance. This was accomplishedh by considering both black-box and grey-box ML-based adversaries.

3.3. IDS Improvements in the Literature

As part of the objective of securing IoT environments, there are some authors that aimed to improve the way an IDS works through the use of new techniques. A relevant work whose objective was to enhance IDSes focused on IoT environments is the one presented by Oh et al. [50], where the authors introduced a lightweight technique to secure systems with the use of an innovative malicious pattern-matching engine. The authors presented the topic by explaining how a specific type of IDS works. In fact, they focused on IDSes using a pattern-matching algorithm to verify predefined sets of signatures in order to detect malicious intentions. The motivation of this work came from the idea that a conventional engine for pattern-matching is not suitable for IoT devices due to their resource-constrained characteristics. Hence, Oh et al. proposed a novel algorithm that would ignore unnecessary matching processes, with the objective of reducing the resources used by the matching operation. First of all, the authors introduced a new method called auxiliary shift value, which would skip to a higher number of characters when matching a text as compared with the fast traditional method called the Wu–Manber algorithm [51]. Later on, they applied a technique to sort prefixes in order to reduce matching operations as well and to allow the final algorithm to be more efficient when making comparisons.

Wallgren et al. in [37] provided a protocol to be used in a 6LoWPAN network using RPL called lightweight heartbeat protocol. The main idea of this protocol was to send heartbeat messages between nodes in order to know if other nodes in the network were available or not. This kind of technique can be included in an IDS environment in order to detect attacks that cannot be identified by using conventional detection techniques. The heartbeat protocol and other existing techniques to design an IDS are used in our work to mitigate several RPL attacks. In addition, two new techniques are also implemented

and evaluated to demonstrate the way they work. The next section explains the proposed IDS design.

4. IDS Design

In this section, we describe the design of our proposed IDS as well as our motivations behind different design choices. There could potentially be multiple IDS configurations that would provide different results, depending on the specific environment in which the IDS is deployed and operates. Depending on the needs of the involved parties and the considered security attacks, one IDS configuration or another could be considered in order to maximise the achieved efficacy based on the current circumstances.

The main aim of this work is to design an IDS capable of detecting multiple and distinguished security attacks in order to demonstrate that a general-purpose IDS for RPL-based IoT is feasible. To accomplish this objective, the design cannot rely on a single detection strategy or be a monolithic entity; it must include multiple cooperative components and employ flexible and adaptive approaches in order to maximise the intrusion detection accuracy. Our IDS design is based on a hybrid configuration of the different attributes, placement strategies, and detection methods. These are detailed in the following subsections.

4.1. Architecture and Components

The high-level architecture of the IDS follows the design proposed in [49] and is briefly described in this subsection. In addition to the typical sensor nodes, we consider two new types of devices: (i) *Central IDS* or *border routers (BR)*, which are centralised detection modules with routing and firewall capabilities, and (ii) *IDS detectors*, which are distributed detection modules that are less powerful, sensor-like devices used to monitor and send suspicious traffic or alerts to the central IDS. In a typical scenario of a small IoT network, there are one BR and several IDS detectors. This means that sensors requiring to communicate with a server will send all the requests through the router. All passing traffic is checked by the BR, which will take the decision as to whether the sending node is malicious or not.

IDS detectors monitor the sensors' traffic to help in detecting malicious nodes. Compromised devices may attempt to internally disrupt the network without having to communicate with the BR or external networks. For such cases, the detectors log network traffic, and if a node's activity resembles a known attack behaviour or triggers a known attack signature, an alert along with any related information is forwarded to the BR for decision-making. The traffic exchanged between the sensors is monitored by the nearest detector within range. Afterwards, a lightweight algorithm is executed to decide whether or not traffic should be forwarded to the BR. The collaboration between the BR and the detectors helps in capturing traffic from both internal and external interfaces. For example, some malicious devices may attempt to establish communication with a remote command & control server in order to obtain further malicious instructions or updates. Other malicious devices may exchange traffic locally. In cases where a wireless channel between the BR and the detectors is unavoidable or preferable, an appropriate and secure wireless communication scheme will be put in place (e.g., [52]).

4.2. Placement Strategies

With respect to the placement strategy of the IDS modules, three options can be distinguished: distributed, centralised, and hybrid placement. A distributed placement strategy consists in installing and configuring an IDS detector in every network node in order to increase the chance of detecting an attack as much as possible. However, this strategy will result in higher power consumption at the nodes. On the other hand, a centralised placement strategy implies using one powerful node as the IDS (e.g., at the BR). This strategy does not involve the rest of the nodes in the network when detecting possible attacks; hence, this consumes less power. Nevertheless, the centralised strategy is only able to detect those attacks that are in range or those that need to send malicious

packets to external networks through the BR. Finally, the hybrid placement combines both strategies. For example, it has one node acting as the central IDS, and multiple detectors (in charge of monitoring some areas of the network) gathering information on the network that the central IDS is not capable of obtaining.

In general, a hybrid-based placement approach seems to be the best and the most flexible choice as it allows the IDS to detect more attacks by increasing the number of nodes involved in the monitoring and detection process, but it does not consume as much power as a distributed strategy. Hence, in our design, two types of nodes are used to perform the IDS tasks: a central IDS (which is typically placed in the DODAG root or BR node) and multiple detectors. Figure 1 represents a possible architecture to exemplify the hybrid design. As can be seen, the root node (aaaa::1), which is the one acting as the central IDS, does not have a large range. In fact, only nodes aaaa::2, aaaa::3, and aaaa::4 are in transmission (TX) and reception (RX) range. Therefore, if node aaaa::11 acts as a malicious node performing a Hello flood attack, it will send a large number of Hello messages to the nodes in range (aaaa::8, aaaa::10, and aaaa::9). Consequently, the central IDS (aaaa::1) will never be able to detect such an attack. On the other hand, by having multiple detectors in the network, the possibilities of detecting an attack increases. In this example, detector node aaaa::8 receives the attack from aaaa::11 and is able to send an alert to the central IDS for further actions.

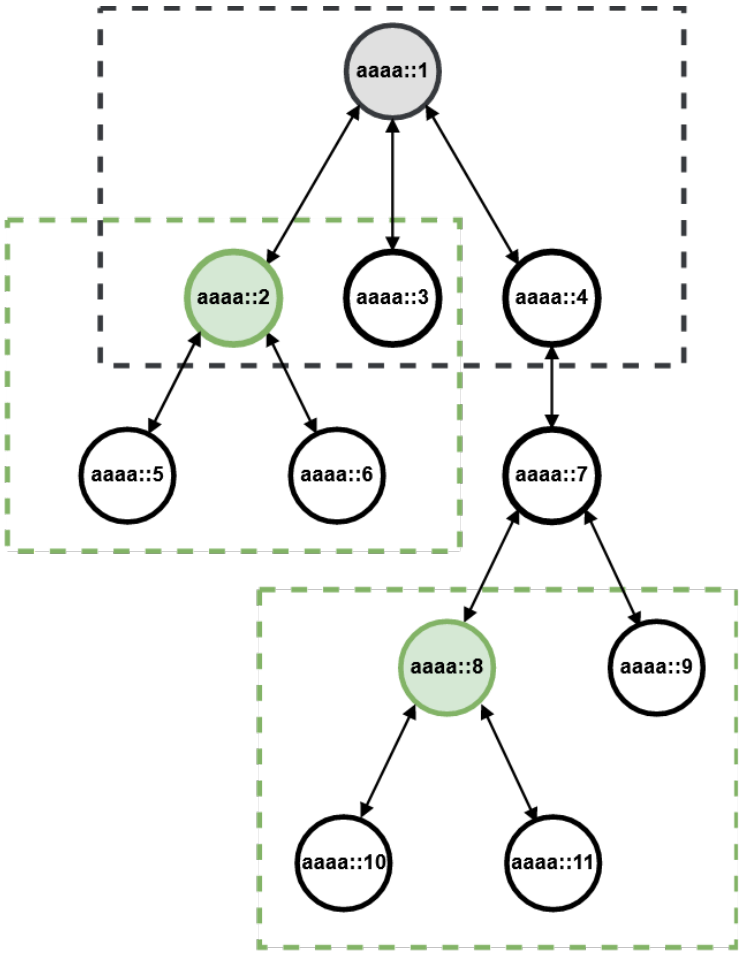

Figure 1. Hybrid IDS design example. The shaded node (aaaa::1) is the root, and it hosts the central IDS. The doted rectangle in black represents the nodes in TX/RX range from the root node. Nodes in green (aaaa::2 and aaaa::8) represent IDS detectors. The doted rectangle in green represents the nodes in TX/RX range from the IDS detector nodes.

4.3. Detection Methods

The following types of intrusion detection methods can be distinguished: signature-based, specification-based, anomaly-based, and hybrid. Signature-based detection aims at matching existing signatures with incoming packets to check for intrusions. Some of the limitations of this approach are the large database needed and the impossibility of detecting 0-day attacks. Specification-based and anomaly-based approaches share some similarities. Specification-based detection modules are typically configured manually, whereas anomaly-based are often supported by machine learning algorithms or by algorithms that allow the IDS to adapt to network changes [53]. Both approaches are based on specific statistics in a network by gathering relevant data from the devices. If the IDS detects an anomalous behaviour (i.e., different from the expected functioning of the network), it will raise an alert. Therefore, for some types of attacks, especially 0-day attacks, the IDS has a higher chance of detection. However, such approaches may produce a high rate of false positives since a small change in the network may be interpreted as an attack. Finally, the hybrid-based detection strategy consists in a combination of the previously mentioned detection methods to maximise the advantages and reduce the encountered issues.

In our design, with the aim of detecting multiple attacks, a hybrid detection strategy is composed of a single module that handles possible attacks by using different types of detection methods. However, due to the placement characteristics of the IDS nodes (see Figure 1) and the nature of the attacks (some of them cannot be detected if not in range), central IDS and IDS detector nodes use a different hybrid detection method. These are detailed in the following subsections. The meanings of the symbols used in Algorithms 1–5 are given in Table 1.

Table 1. Symbols used in Algorithms 1–5.

Symbol	Meaning
P	Packet received
$DIO\_counter$	DIO messages counter
$DIS\_counter$	DIS messages counter
$DAO\_counter$	DAO messages counter
T_{DIO}	Threshold for DIO messages
T_{DIS}	Threshold for DIS messages
T_{DAO}	Threshold for DAO messages
T_i	Threshold calculated at node i
$RankSum$	Total sum of all ranks at node i
$MaxRank$	The maximum rank value at node i
$LastReceivedRank$	The last received rank at node i
K	Predefined constant for balancing the threshold value
N	The number of nodes in the network

4.3.1. Central IDS

Signature-based and specification-based methods are combined into a hybrid approach for the central IDS design in order to detect the following attacks: Hello flood, DIS attack, DAO insider, and version number (Blackhole and greyhole attacks can also be detected from the central IDS; this is explained in Section 4.4). Figure 2 illustrates the detection method used by the central IDS.

Specification-based detection: Hello flood attack, DIS attack, and DAO insider attack have similar behaviour. All of them continuously send specific packets to a target in order to cause an unexpected reception of packets and drain the battery of the victim. As a result of this similar characteristic, the central IDS can use the same strategy to detect them.

In this case, specification-based detection is used. According to the normal behaviour of the network, the root node, which also acts as the central IDS, does not receive a large number of DIS, DIO, and DAO packets per minute. Hence, by analysing the network, appropriate thresholds T_{DIS}, T_{DIO}, and T_{DAO} can be specified to detect a high rate of DIS, DIO, or DAO messages if the threshold is exceeded. Algorithm 1 represents the pseudocode for Hello flood attack, DIS attack, and DAO insider attack detection. As shown, upon receiving a packet, the central IDS stores a counter, depending on the type of packet. If the threshold is exceeded, an alert is raised.

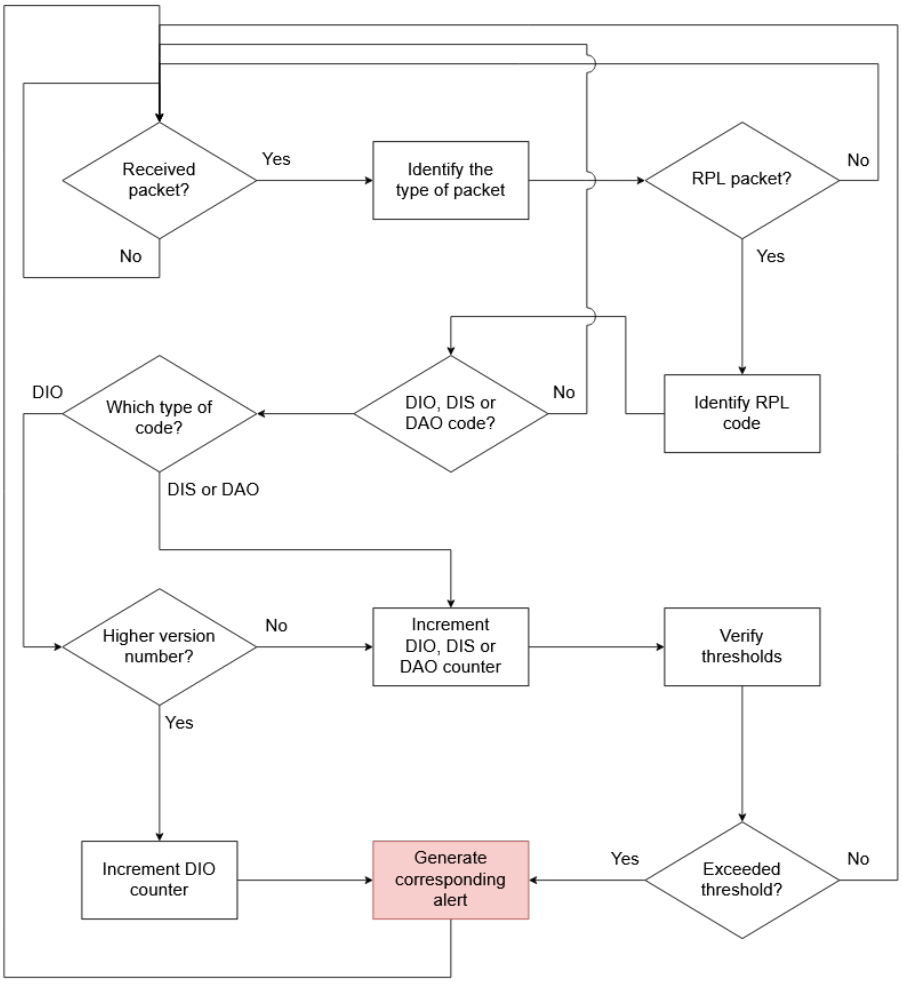

Figure 2. Central IDS detection diagram.

Signature-based detection: Version number attack is a substantially different attack than those described above. Therefore, it requires a different strategy in order to be detected. Briefly explained, an attacker launching the version number attack issues a higher DODAG version number to the rest of the nodes in order to cause disruption in the network. If the IDS is capable of identifying this malformed packet, the attack will be detected as soon as the packet is inspected. In this scenario, a signature-based detection method is required in order to detect the attack. Thus, by taking into account that only the root node issues a higher version of the DODAG version number, if the central IDS (which acts as the root node) receives a DIO message (control message containing the DODAG information) with a version number that is higher than the current one, the IDS will know that an attack is taking place and will raise the corresponding alert. Algorithm 2 represents the detection algorithm for the version number attack.

Algorithm 1: Detection of Hello flood attack, DIS attack, and DAO insider attack.

input: P - Packet received
input: *From* - IP of the sender of the packet
input: T_{DIO}, T_{DIS}, T_{DAO} - Predefined thresholds

```
/* DIO messages counter                                    */
DIO_counter = 0;
/* DIS messages counter                                    */
DIS_counter = 0;
/* DAO messages counter                                    */
DAO_counter = 0;
if isDIOMessage(P) then
    ++DIO_counter;
else if isDISMessage(P) then
    ++DIS_counter;
else if isDAOMessage(P) then
    ++DAO_counter;
else
    print("INFO: Not a control message. Skipping...");
end
if ((DIO_counter > T_DIO)||(DIS_counter > T_DIS)||(DAO_counter > T_DAO))
 then
    raiseAnAlert(From);
    resetCounters();
end
```

Algorithm 2: Detection of version number attack from central IDS.

input: P - Packet received
input: *From* - IP of the sender of the packet

```
if isDIOMessage(P) then
    if getVersionNumber(P) > currentVersionNumber() then
        raiseAnAlert(From);
    end
end
```

4.3.2. IDS Detector

IDS detectors are in charge of the detection of possible intrusions in a specific region of the network, after which they alert the central IDS. This type of node combines the concepts of signature-based, specification-based, and anomaly-based detection methods, as explained below. Figure 3 illustrates the detection method used by the IDS detector.

Specification-based detection: Regarding the Hello flood attack, DIS attack, and DAO insider attack, the detection method required to recognise those is the same as that for the central IDS (Algorithm 1 represents the detection pseudocode). Hence, specification-based detection is used by specifying the required thresholds for this purpose: T_{DIS}, T_{DIO}, and T_{DAO}. However, the case of the DAO Insider attack is slightly different. Recall that the objective of the DAO insider attack is to make the victims (i.e., parent nodes on the path to the root) consume more power. To do so, the attacker sends a DAO message to perform the attack, with the root node as the destination. As a result of this, intermediate IDS detector nodes will simply forward the packet since they are not the packet destinations (In a 6LoWPAN network, an additional header is included in the packet when this needs to be forwarded [54]). Therefore, with the intention of detecting the DAO insider attack, every time an IDS detector forwards a packet, this will need to be deeply inspected for the IDS

detector to realise that the packet is a DAO message. Central IDS does not need to deeply inspect the packet as it already knows that the packet is a DAO message because the central IDS is the destination. That being said, as the central IDS is always the destination of a DAO message, one may think that IDS detector nodes are not required to detect this attack, because even if the central IDS is not in the range of the attacker, the packet will always reach the IDS. However, in a scenario where the IDS is capable of removing malicious nodes from the network, such a restriction will be better applied by an IDS detector node close to the network region where the attacker is located as the blocking rule will be applied faster.

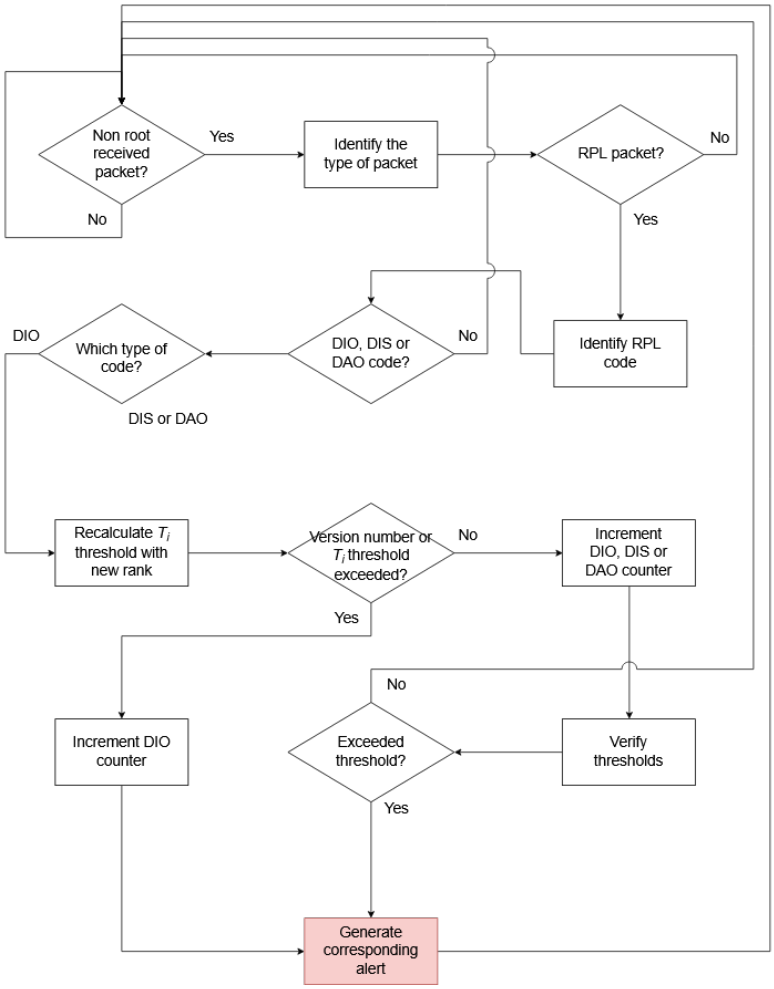

Figure 3. IDS detector functionality.

Signature-based detection: With respect to the version number attack, the detection algorithm used by the central IDS can also be applied here. Upon receiving a DIO message from a source node other than the root, and with a higher version number than the current one, the IDS detector will issue an alert as this is the typical behaviour of an attacker performing a version number attack. Algorithm 3 represents the pseudocode of the version number attack detection performed by the IDS detector.

Anomaly-based detection: The IDS detector also incorporates the anomaly-based detection method to detect a rank decrease attack, which consists of issuing a message with a rank lower than the real one in order to advertise a "fake" better route. If it succeeds, neighbours receiving a lower rank than the one used in their best path will change the preferred parent, which is the attacker node. In order to detect such attacks, a technique based on a scheme proposed by Iuchi et al. [55] to secure the parent node selection process in RPL is used. The main idea behind this scheme is based on the possibility for an attacker to falsely claim a malformed rank in order to legitimate the nodes and thus become the

parent of the victim nodes. With this in mind, an algorithm is executed in every node of the network to avoid malicious nodes and to choose the best parent. The algorithm consists of calculating a threshold T_i, which is the result of the following equation calculated from a node i by taking into account the rank values of the neighbours:

$$T_i = R_{ave,i} - R_{max,i} \times K \tag{1}$$

where $R_{ave,i}$ is the average rank value among the neighbours in range of a node i, $R_{max,i}$ is the maximum rank of the set of neighbour nodes in range of a node i, and $K(0 < K < 1)$ is a constant defined before the execution of the algorithm. The value of this constant is used to balance the value of the threshold. Low threshold values may result in a higher number of false positives, whereas high threshold values may miss some attacks. Once T_i is calculated, if the node i receives a rank value lower than the calculated threshold T_i from a node j, i will consider j as an attacker, and j will not be included during the best parent selection process of i. Therefore, by using the algorithm, an IDS detector will analyse the current behaviour of a network region in order to establish a threshold T_i for the detection of a malicious node performing a rank decrease attack, and to be able to issue the corresponding alert to the central IDS. Algorithm 4 represents the pseudocode executed by the IDS detector in order to measure the value of T_i. Once this value is determined (i.e., every time the rank of the neighbours is updated upon receiving a DIO message), the IDS detector will verify if this newly received rank exceeds the threshold. If it does, an alert specifying the IP address of the attacker and the type of attack will be sent to the central IDS.

Algorithm 3: Detection of version number attack by IDS detector.

```
input: P - Packet received
input: From - IP of the sender of the packet
if isDIOMessage(P) && !isRoot(From) then
    if getVersionNumber(P) > currentVersionNumber() then
        raiseAnAlert(From);
    end
end
```

Algorithm 4: Threshold T_i calculation by node i (i.e., IDS detector) to detect the rank decrease attack.

```
input: Neighbours - Neighbours in range of the IDS detector
input: K - Predefined constant
/* Calculated threshold where i is the node running the algorithm */
T_i = 0;
/* Total sum of all ranks                                         */
RankSum = 0;
/* Max rank value                                                 */
RankMax = 0;
for Neighbour in Neighbours do
    LastReceivedRank = getRank(Neighbour);
    RankSum = RankSum + LastReceivedRank;
    if LastReceivedRank > RankMax then
        RankMax = LastReceivedRank;
    end
end
T_i = (RankSum ÷ getLength(Neighbors)) − RankMax × K;
```

4.4. Blackhole and Grayhole Detection

Recall that the proposed IDS uses a hybrid method to detect some of the attacks mentioned earlier (i.e., Hello flood attack, DIS attack, DAO insider attack, version number attack, and rank decrease attack). However, in order to detect the blackhole or greyhole attacks, the use of traditional detection methods (i.e., signature-based, specification-based, and anomaly-based) may not be sufficient. In the case of a signature-based detection strategy, there are difficulties in detecting the blackhole and greyhole attacks as such attacks do not send packets to the victims. Therefore, an IDS using a signature-based method will not be able to match a malicious signature with a depicted malicious packet because no such packets will be received. On the other hand, specification-based and anomaly-based detection methods, if properly configured, may be able to detect both the blackhole and greyhole attacks [56,57]. Nevertheless, in the case of specification-based detection methods, the previous analysis of the network to specify the correct set of rules for the detection of such attacks will be difficult. In addition, any modification in the network topology or configuration will require a new analysis and new specifications, which will consume time and effort. The use of anomaly-based detection to detect blackhole and greyhole attacks has similar problems. In a network without too many changes in topology and data transmission patterns, an anomaly-based IDS is able to detect malicious behaviours as the network will suffer significant changes caused by the attack. In such circumstances, the anomaly-based detection method may work. However, in dynamic network scenarios and with frequent variations (e.g., in packet transmission rate, packet size, peak hours, etc.), the anomaly-based IDS is unable to establish a normal behaviour for the network. In fact, the normal behaviour is for a network to have too many changes. Hence, a blackhole or greyhole attack might not be efficiently detectable.

Due to the aforementioned problems regarding the use of specification-based and anomaly-based detection methods for blackhole and greyhole attacks, our IDS design also includes a *heartbeat* technique whose objective is to detect such attacks. Note that a blackhole attack does not forward or send any packet. Thus, the lightweight heartbeat protocol (LHP) of [37] will notice this behaviour, and the attack can be detected. On the other hand, with respect to the greyhole attack, the LHP may not work. Depending on the implementation of the attack, the node in charge of performing the malicious action may decide to forward only control messages. In such a case, RPL control messages and ICMPv6 packets will be forwarded, and the IDS will not notice anything suspicious as the LHP will receive the corresponding replies from other nodes in the network. For this reason, in addition to the LHP, the IDS design includes an enhanced heartbeat protocol that is based on UDP messages [58]. The main idea is that instead of using ICMPv6 messages, the IDS and other nodes will exchange UDP messages. The main benefit of this option is that it can detect both blackhole and greyhole attacks by utilising a single protocol. However, there is a major drawback when using a UDP-based heartbeat: other nodes are required to handle these UDP packets and send the corresponding responses to the IDS. Therefore, every network node will require a modification in its code to be able to communicate using the UDP-based heartbeat protocol.

All in all, in order to detect possible intrusions, the IDS has been based on a hybrid placement strategy that combines signature-based, specification-based, and anomaly-based detection methods to detect a large variety of attacks. Central IDS and IDS detectors use a different combination of detection methods due to their different characteristics. In addition, the IDS design includes the heartbeat protocol as a detection method to detect both blackhole and greyhole attacks. As the IDS performing this detection needs full communication with all the nodes, the heartbeat protocol is executed from the central IDS. If the requirements of the network are focused on detecting only blackhole attacks, the LHP is the option used in this design. On the other hand, if greyhole attacks also need to be detected, the UDP-based heartbeat protocol will be used.

Algorithm 5: Proposed method: Detection phase

```
i ← 0
N ← nodes.size
while i < N and nodes[i].used do
    check_counter(nodes[i].IP, nodes[i].count)
    send_UDP_REQUEST(nodes[i].IP)
    nodes[i].heartbeat_sent ← nodes[i].heartbeat_sent + 1
    i ← i + 1
end while
```

4.5. Central IDS and IDS Detector Communication

In order to detect the attacks described above, it is necessary to provide a hybrid placement IDS module throughout the entire network. As explained before, the IDS design is composed of a central IDS and multiple IDS detectors. The central IDS is focused on detecting specific attacks, but one of its core functions is to handle the alerts received by the IDS detectors. If the central IDS is not capable of "hearing" those alerts, some attacks will go unnoticed. For this reason, ensuring appropriate communication between the central IDS and IDS detectors is crucial.

Every T_c (where T_c represents an interval of time in seconds), the central IDS sends a message to all IDS detectors in the network. These messages are sent to ensure that the communication between the two parties is working correctly. Hence, if the central IDS does not receive a reply, the link between this specific IDS detector and the central IDS will be marked as down, and further actions from the network administrators will be required. At this point, the design will have two variants, which were evaluated in Section 5. One of the variants is that, in addition to the security message, the IDS detector will make use of the packet sent to raise an alert, if required, so that the central IDS can be made aware of possible attacks outside its range. The other variant consists of sending an alert as soon as the IDS detector is aware of it. The first variant reduces the messages exchanged inside the network, but the IDS will detect attacks every T_c seconds (e.g., 30 or 60 s). On the other hand, if the IDS detector needs to send the alert as soon as the attack is detected, resource usage is increased as well as the number of exchanged messages when an attack takes place. Figure 4 illustrates the communication protocol for the aforementioned variants. IDS Detector 1 uses the variant of sending only relevant information every time the central IDS asks for this information. On the contrary, IDS Detector 2 sends the attack information as soon as it detects the suspicious behaviour.

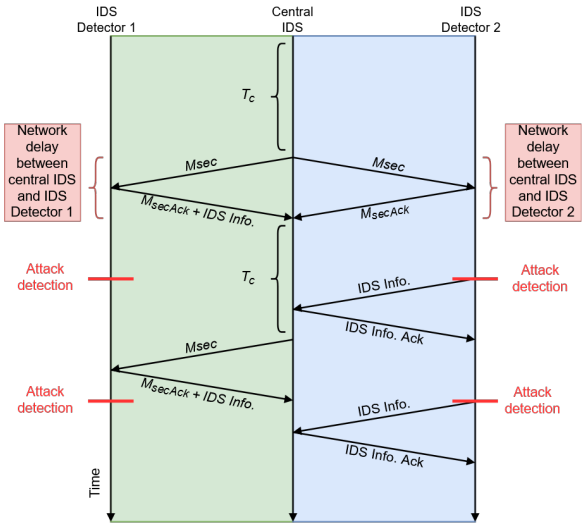

Figure 4. Diagram of central IDS and IDS detector communication variants.

5. Evaluation

This section aims to experimentally evaluate the proposed IDS design and its implementation. This is carried out via the verification of the accuracy of the attack detection, and by measuring the performance of both the central IDS and the IDS detectors in terms of consumed CPU, transmission (TX) and reception (RX) power usage, and memory usage. To this end, we implemented the aforementioned attacks and the IDS components in Contiki-NG.

5.1. Attack Detection Accuracy

Below, we present the experimental results for the detection accuracy of flooding and version number attacks.

5.1.1. Flooding Attacks Detection

An analysis of the accuracy of the proposed IDS in detecting flooding attacks is provided in this subsection. In particular, the IDS design is executed during the Hello flood attack, DIS attack, and DAO insider attack to verify its detection accuracy. First, an evaluation is carried out to verify the detection accuracy on an attack-by-attack basis (i.e., run a simulation with only one attack at a time). Later on, a simulation is run with all flooding attacks at the same time.

The topology used in order to test the proposed IDS against a Hello flood attack is shown in Figure 5. In this scenario, node 1 is the central IDS, node 5 is the attacker, node 9 is an IDS detector, and the rest of the nodes are benign (dummy nodes). As can be seen, the attacker node 5 is performing a Hello flood attack against the nodes in range. In this scenario, both the central IDS (ID 1) and the IDS detector (ID 9) are receiving the attack packets. Figure 6 illustrates the logs, which show the time when the attack started (i.e., at 02:01.055), the detector IDS and the central IDS detecting a DIO flood (at 02:04.266 and 02:04.834, respectively) that corresponds to a Hello flood attack, and the moment when the IDS detector sent the DIO alarm (at 02:04.305) to the central IDS and its arrival (at 02:04.540). The IP address fe80::212:7405:5:505 corresponds to node 5. In addition to this detection, after minute 3, the central IDS asks the IDS detector for information, and the latter again sends the DIO alarm corresponding to a Hello flood attack. After this, the IDS detector detects the attack again and sends an additional alert message that is received by the central IDS.

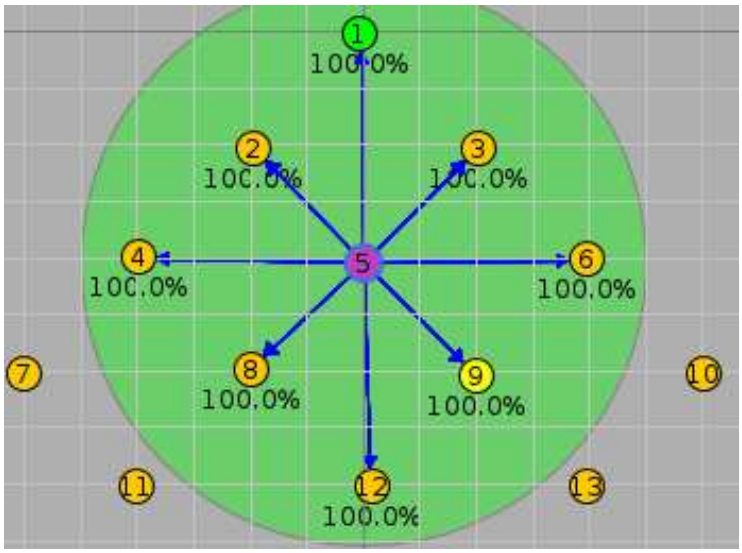

Figure 5. Hello flood attack network topology under IDS detection. Node 1 is the central IDS. Node 5 is the Hello flood attacker. Node 9 is an IDS detector. The rest of the nodes are benign.

```
02:01.055  ID:5   HELLO FLOOD Attack started
02:04.266  ID:9   ALARM DIO: '51'
02:04.305  ID:9   Alarm sent
02:04.540  ID:1   Received DIO alarm from: fe80::212:7405:5:505
02:04.834  ID:1   ALARM DIO: '51'
```

Figure 6. Hello flood attack log showing detection.

The detection of the DIS attack occurs in a similar way to the one performed during the Hello flood attack simulation. Figure 7 represents the DIS attack detection by both the central IDS (ID 1) and the IDS detector (ID 9). The DIS attack starts at 02:01:055. After that, both the central IDS and the IDS detector detect the attack (at 02:02.517 and 02:02.292, respectively). As soon as the IDS detector detects the attack, an alarm is sent to the central IDS (at 02:02.331), which receives the alert at 02:02.589. As before, the IP address fe80::212:7405:5:505 corresponds to node 5, the DIS attacker. Moreover, after minute 3, the central IDS asks the IDS detector for information, and the detector loops over all the neighbours to detect a possible attack. In fact, it did detect an attack at 03:01.167 and sent an alarm to the central IDS, which received the alert at 03:01.523.

In the case of the DAO insider attack, a different topology for the simulation was used. This topology is shown in Figure 8. As mentioned before, a DAO message is retransmitted through all the parents of the sender until the message reaches the root node. A DAO insider attack is always detected by the central IDS as the destination of the DAO message is the root. However, in order for the IDS detectors to detect such an attack, they must be the parent of the attacker as the message is sent through them. For this simulation, the parent of the attacker node is the IDS detector. More specifically, the IDS detector (node 9 with IP address fd00::212:7409:9:909) is the parent of the attacker node 12 (IP address fd00::212:740c:c:c0c). Once the simulation started, our logs showed that the attack was being detected. The DAO insider attack starts at 02:00.915, and after 21 s, an alarm is raised by the IDS detector and the alert is sent to the central IDS. The attack is detected after 21 s because the threshold for DAO messages (T_{DAO}) is set to 20, and for this specific attack, the attacker sends a DAO message every one second. After this, the central IDS indicates in the log that it received an attack from node 12 at 02:21.501 (alert sent from the IDS detector), and that it also detected the attack at 02:22.526. In addition, when the central IDS sends a message to retrieve information from the IDS detectors at minute 3, an alert is sent back to central IDS, indicating a DAO insider attack at 03:01.001.

```
02:01.055  ID:5   DIS Attack started
02:02.292  ID:9   ALARM DIS: '21'
02:02.331  ID:9   Alarm sent
02:02.517  ID:1   ALARM DIS: '21'
02:02.589  ID:1   Received DIS alarm from: fe80::212:7405:5:505
```

Figure 7. DIS attack log showing detection.

In addition to the aforementioned detection simulations against flooding attacks, more simulations have been performed by using different topologies and changing the number of nodes in the simulation. All the simulations have succeed in detecting the attacks as the alert is only raised if the thresholds (T_{DIO}, T_{DIS}, and T_{DAO}) are exceeded, which clearly indicates an attack.

To conclude the flooding attack detection testing, one last experiment was performed. It consisted in a simulation with one central IDS, two IDS detectors, one Hello flood attacker, one DIS attacker, one DAO insider attacker, and seven dummy nodes. The topology is shown in Figure 9. Our logs show that the detection was performed at around minute 2. It can be observed that at minute 2, three attacks were detected. More precisely, the Hello flood attack was detected from IDS detector node 9 at 02:04.621, while the DIS attack was detected from IDS detector node 7 at 02:01.812. Some seconds later, the central IDS node

1 detected the DAO insider attack at 02:22.129. With this simulation, one can see that the IDS design was working perfectly with more than one IDS detector. In fact, those are the scenarios where the hybrid placement is completely useful, and thanks to the distributed topology in this case, all the attacks were correctly detected.

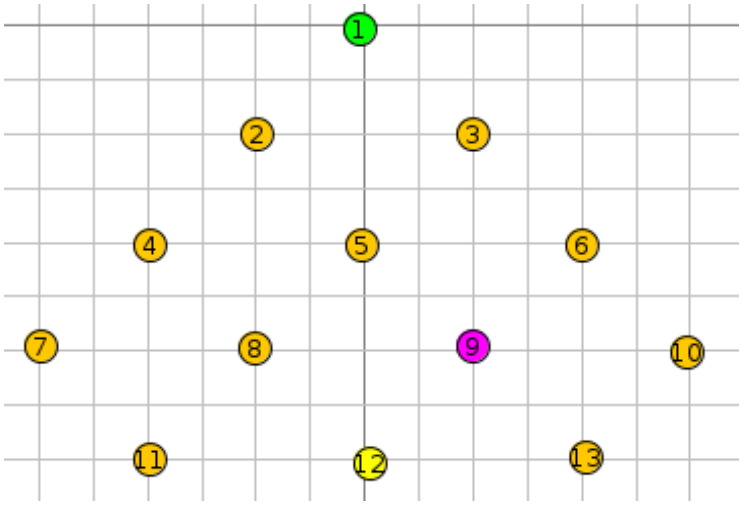

Figure 8. Network topology of DAO insider attack detection. Node 1 is the central IDS. Node 9 is an IDS detector. Node 12 is the DAO insider attacker. The rest of the nodes are dummy nodes.

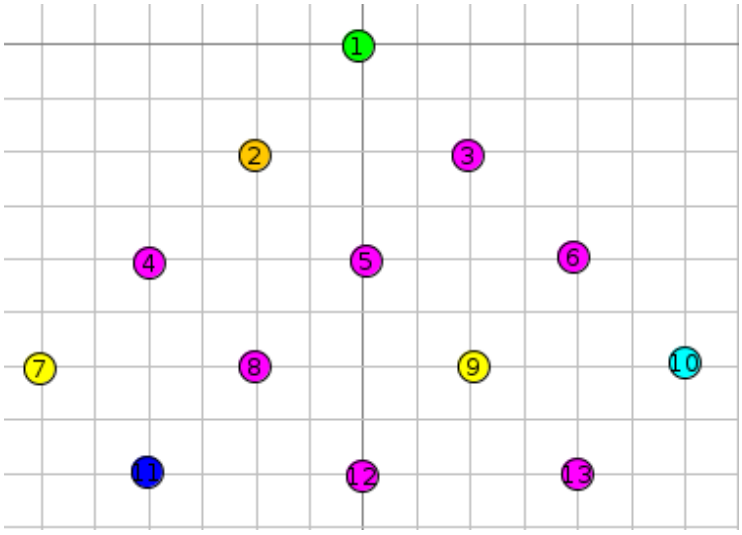

Figure 9. Full flooding attack detection, simulation topology. Node 1 is the central IDS. Nodes 7 and 9 are the IDS detectors. Node 2 is the DIS attacker. Node 10 is the Hello flood attacker. Node 11 is the DAO insider attacker. The rest of the nodes are dummy nodes.

5.1.2. Version Number Attack Detection

As explained in Section 4.3, the version number attack can be detected by both the central IDS and the IDS detectors, and it consists of verifying that the version number advertised by the nodes is not higher than the expected value (i.e., the current version of the DODAG issued by the root). The detection in this case relies on the proximity of the IDS node to the attacker. If the IDS detector or the central IDS is not in range of the attacker, they will not receive the malformed version number from the malicious node, and therefore the detection will not be accurate. In order to evaluate this behaviour and the effectiveness of the IDS against a version number attack, the simulation used the topology shown in Figure 10. During the simulation, the log of the attacker showed that it launched the attack at around minute 2. After this, any DIO message sent from the attacker would have had to contain a higher version number to disrupt the network. Indeed, at minute 02:48.080,

node 12 (the attacker with IP: fe80::212:740c:c:c0c) sent a DIO message in multicast to the nodes in range, indicating a version number of 241 instead of 240, which is the expected value (this is shown in Figure 11). Notice the information given under the ICMPv6 section; the message was a DIO message, and the advertised version was 241. At minute 02:48.108, the IDS detector (ID 9) reported an alert, indicating that node 12 was performing a version number attack. After this, the IDS detector sent an alert to the central IDS, and the latter received the alert at minute 02:48.237.

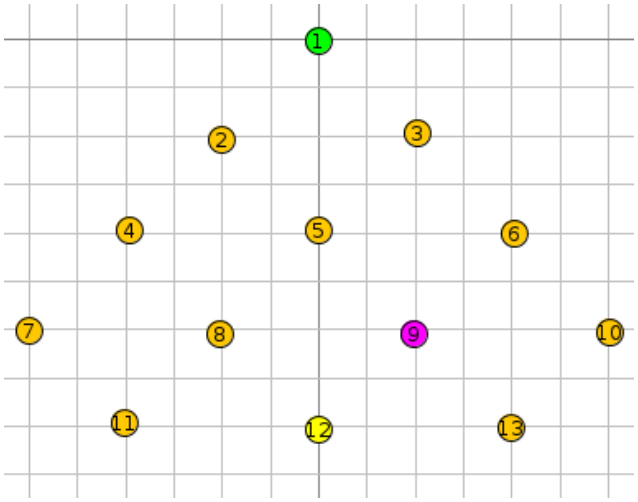

Figure 10. Version number attack detection simulation topology. Node 1 is the central IDS. Node 9 is an IDS detector. Node 12 is the version number attacker. The rest of the nodes are dummy nodes.

As demonstrated earlier, the attack had been identified and the attacker had been spotted. However, some minutes later, the network was completely disrupted due to the version number attack. First of all, the topology in terms of parent nodes and child nodes was changing constantly. The log files show the parents of the network at minute 2, just before the attack. On the other hand, the parents of the network were at minute 4 after the attack. It has been observed that the parents are completely different. More precisely, the IDS detector node 9 (IP: fd00::212:7409:9:909) has node 8 (IP: fd00::212:7408:8:808) as a parent, node 8 has node 12 (IP: fd00::212:740c:c:c0c) as a parent, and node 12 has node 9 as a parent. Therefore, a loop was created during the version number attack. After all of this, the IDS detector continued to issue alarms, but the log showed a different behaviour. As reflected, the IDS detector (ID 9) sent multiple DAO alarms to the central IDS. These alerts are generated due to the high number of times that a node changes the parent as a result of the version number attack. Every time a node changes its parent, a DAO message is issued. Hence, the IDS detects this as an attack. In addition to this, at minute 04:01.119, the IDS detector sent an alert about a new version number attack generated by node 8. In theory, this is wrong as the only attacker node is node 12. However, as the version number was propagated throughout the network, the rest of the nodes also advertised a wrong version number and acted as attackers. Another remarkable issue is that the alerts sent by the IDS detector to the central IDS were not received after minute 4. This was caused by the loop created in the network, where the IDS detector was no longer able to reach the central ID. In conclusion, the developed IDS works properly in the detection of version number attacks. However, after some time, the network becomes unstable, and the IDS starts sending incorrect alerts about the attack. Therefore, the first alert received by the central IDS is correct; it could correctly identify the attacker and the type of attack. After that, it is important to isolate the attacker from the network in order not to cause a chaotic situation and to avoid false positives. If the attacker is not removed, the network behaviour becomes unpredictable.

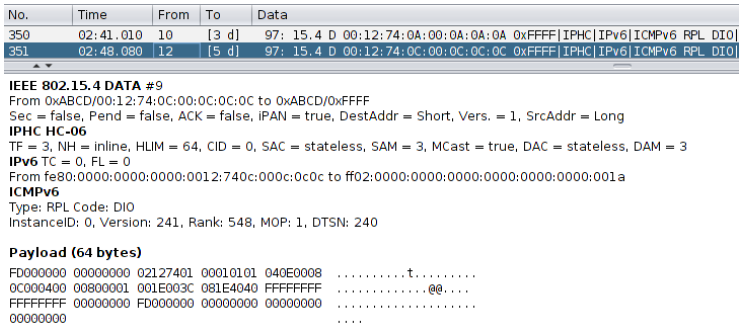

Figure 11. Version number attack simulation. DIO message information.

5.2. IDS Performance

Once the accuracy of the proposed IDS design on the detection of RPL attacks has been evaluated, its performance needs to be analysed as well. First of all, a comparison of the resources consumed by the central IDS, the IDS detector, a benign (dummy) node, and a root node was performed. The results of the obtained metrics can be seen in Table 2. Note that in order to obtain the CPU usage, TX rate, and RX rate metrics for the node types Dummy Node and Dummy Node Heartbeat UDP, an average of 12 nodes for the different simulations was used to obtain the final value. On the other hand, the value obtained for the rest of the node types was extracted from only one simulation as the results are the same when using the same topology. This is the case because in all the performed simulations, there was more than one dummy node and only one node of other node types.

In Table 2, it can be seen that the Dummy Node and Root Node are the ones that used less resources as compared with the rest. This is not unexpected as these nodes only contain the required code for joining and maintaining an RPL network. Although these nodes use less resources than the rest, it can be noticed that the ROM and RAM values are very high. Therefore, it can be confirmed that the memory available for these devices is very limited. Regarding the nodes using more resources, they can be divided into two groups: the ones using more CPU, TX, and RX, and the ones that use more memory. As shown in the table, node types Central IDS Lightweight Heartbeat and Central IDS Heartbeat UDP use 1.94% of the CPU and have high TX and RX rates as compared with the original nodes. Central IDS node types should be compared with the Root Node node type as the Central IDS code is an extension of the Root Node. Therefore, it is confirmed that implementing the IDS solution adds an expected overhead. The overhead in terms of CPU, TX, and RX is slightly high, but not too much. On the other hand, in terms of ROM and RAM, the overhead is more significant. Finally, the IDS Detector node type is the one consuming more ROM as it requires a lot of code in order to perform as expected, but the rest of the values are acceptable and do not differ too much if compared with the Dummy Node type. Figure 12 summarises the comparison between different node types in terms of RAM and ROM usage.

Table 2. Node type metrics table. TX: transmission. RX:reception.

Node Type	CPU	TX	RX	ROM	RAM
Dummy Node	1.22%	0.01%	0.04%	87.93%	70.68%
Dummy N. Heartbeat UDP	1.41%	0.02%	0.10%	89.96%	73.54%
Root Node	1.11%	0.01%	0.03%	87.95%	70.70%
Central IDS	1.39%	0.01%	0.03%	94.11%	79.20%
Central IDS L. Heartbeat	1.94%	0.05%	0.11%	96.59%	79.82%
Central IDS Heartbeat UDP	1.94%	0.04%	0.10%	98.21%	82.60%
IDS Detector	1.56%	0.01%	0.06%	99.83%	75.16%

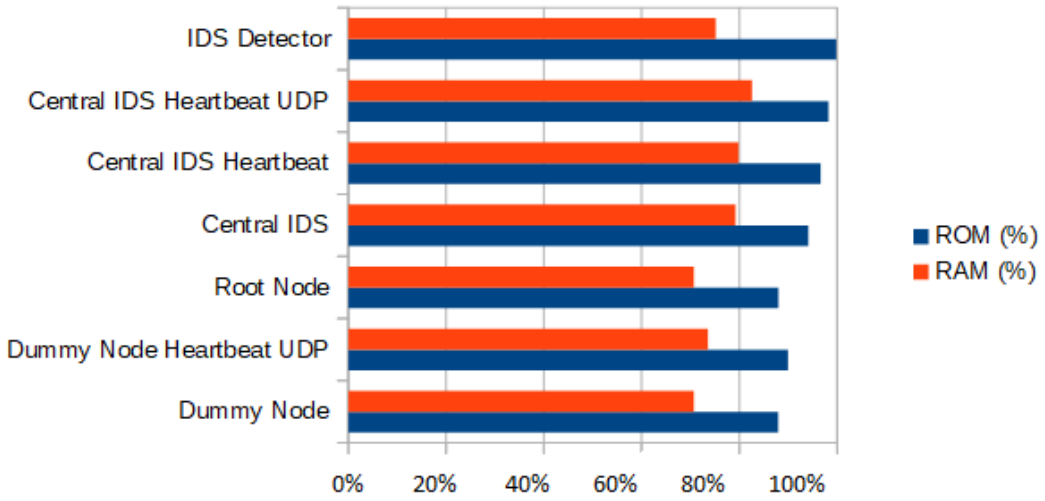

Figure 12. Comparison between node types in terms of RAM and ROM usage.

The next experiment focused on checking how the proposed heartbeat implementation affects network performance. The results were obtained after executing 30 min of the simulation; no differentiation was made between the time before the attack and the time under the DIO flood attack. To execute this experiment, flooders, selectors, and the IDS detector were replaced by dummies but placed in the same positions.

Upon analysis of the results obtained in Table 3, we can observe that the impact is negligible. Firstly, the Radio On rate shows almost equal results. Secondly, the TX rate shows a slight change, which on average is just a difference of 0.03%. Notice how the root obtained a TX rate that was two times or four times more as compared to the dummies. The root node has to send *N-1* requests, where *N* is the number of nodes that the network has. However, dummy nodes only send one message as a reply. Thirdly, the RX rate also shows a small difference. On average, the difference in rate when using the heartbeat protocol is 1.4%, which can be considered insignificant for the overall resource utilisation impact.

Table 3. Evaluation of the heartbeat protocol with UDP.

	Heartbeat Enabled				Heartbeat Disabled			
ID	**Radio ON (%)**	**Transmission Tx (%)**	**Reception Rx (%)**	**CPU Usage (%)**	**Radio ON (%)**	**Transmission Tx (%)**	**Reception Rx (%)**	**CPU Usage (%)**
1	99.75	0.08	0.13	1.22	99.75	0	0.02	0.61
2	97.85	0.1	0.2	1.67	97.85	0.01	0.04	0.83
3	98.62	0.02	0.22	0.89	98.62	0.01	0.05	0.78
4	97.8	0.02	0.23	1.00	97.8	0.01	0.05	0.83
5	95.48	0.02	0.22	0.94	98.48	0.01	0.05	0.78
6	97.44	0.04	0.26	1.17	97.44	0.01	0.04	0.78
7	98.25	0.01	0.1	0.83	98.25	0.01	0.04	0.72
8	98.07	0.04	0.21	1.22	98.07	0.01	0.05	0.83
9	99.07	0.01	0.08	0.72	99.07	0.01	0.02	0.67
Avg.	98.37	0.04	0.18	1.07	98.93	0.01	0.04	0.76

6. Conclusions

In this work, we experimentally evaluated and compared the following RPL attacks: Hello flood attack, DIS attack, DAO insider attack, blackhole attack, and greyhole attack. Firstly, we implemented these attacks and then analysed the behaviour and results obtained from the simulated experiments. The performance metrics used were the CPU usage and the TX/RX rates. In addition to the evaluation of the attacks, a hybrid IDS was designed and implemented using Contiki-NG. After the implementation, the IDS was evaluated in

terms of the accuracy of the attack detection and its performance regarding CPU usage, TX/RX rate, and memory usage. To detect the aforementioned attacks, existing techniques from the available literature were used (i.e., threshold usage to detect flooding attacks, and signature matching to detect the version number attack). Moreover, new methods for detection were developed, such as the UDP-based heartbeat protocol for blackhole and greyhole attacks detection.

A study on specific RPL attacks was provided during the experimental evaluation. The main conclusion to be taken from this evaluation is that the studied RPL attacks can have a big impact inside an RPL network by exhausting the nodes or by isolating them from the network. Therefore, it is mandatory to include security measures in order to mitigate potential attacks and reduce the damage caused by the attackers. In order to mitigate those RPL attacks, a new IDS design and implementation was proposed, and according to the obtained results, the proposed IDS was able to detect flooding attacks, blackhole attacks, greyhole attacks, and version number attacks with high accuracy and very quickly.

Regarding performance, the developed solution was shown to be appropriate as the overhead caused by the IDS implementation was negligible in terms of CPU usage and TX/RX rates. On the other hand, as the devices under consideration had very limited RAM and ROM, the memory overhead caused by the IDS implementation was close to the maximum limit. Therefore, the IDS design is suitable for execution on resource-constrained devices, but the memory of these devices must be taken into account. Regarding the version number attack, the evaluation showed that the problem occurred after the attack was propagated through the network. At that moment, the network became unstable and loops were created. To mitigate this, a new blocking mechanism can be included in the IDS design to remove the attacker node from the network before it is too late. The IDS will provide information on the alert and the target, and the blocking system must block the attacker as soon as possible in order to reduce the impact of the version number attack. To summarise, the CPU usage overhead is less than 2% in all cases, whereas the average power consumption increase is no more than 0.5%, which can be considered an insignificant impact on the overall resource utilisation.

Possible future research directions could include the detection of other attacks, such as a rank decrease attack. For example, an improved parent node selection strategy could be used for rank decrease attack detection. A deeper study is needed in order to develop the appropriate detection mechanism and algorithm. Possible future improvements could also rely on deep and heterogeneous group convolutional neural networks (CNN) [59,60], which can be used to automatically mine accurate information about the intrusions based on their observed properties.

Author Contributions: Conceptualization, E.G.R., B.M.A., and V.G.V.; methodology, E.G.R., B.M.A., P.P.I., and V.G.V.; software, E.G.R. and B.M.A.; validation, E.G.R., B.M.A., and C.S.; formal analysis, E.G.R., B.M.A., C.S., and P.P.I.; writing—original draft preparation, E.G.R., B.M.A., and C.S.; writing—review and editing, P.P.I. and V.G.V.; supervision, V.G.V.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

6WLoPAN	IPv6 over Low-power WPAN
ARL	Adversarial Reinforcement Learning
BR	Border Router
CNN	Convolutional Neural Network

CPU	Central Processing Unit
DAO	Destination Advertisement Object
DIO	DODAG Information Object
DIS	DODAG Information Solicitation
DODAG	Destination Oriented Directed Acyclic Graph
DoS	Denial of Service
IDS	Intrusion Detection System
IETF	Internet Engineering Task Force
IIoT	Industrial IoT
IoT	Internet of Things
IP	Internet Protocol
LHP	Lightweight Heartbeat Protocol
LLN	Low-power and Lossy Network
MANET	Mobile Ad Hoc Network
ML	Machine Learning
OS	Operating System
RL	Reinforcement Learning
RPL	IPv6 Routing Protocol for LLNs
RX	Reception
TX	Transmission
UDP	User Datagram Protocol
WPAN	Wireless Personal Area Network
WSN	Wireless Sensor Network

References

1. Atzori, L.; Iera, A.; Morabito, G. The internet of things: A survey. *Comput. Netw.* **2010**, *54*, 2787–2805. [CrossRef]
2. Malik, P.K.; Sharma, R.; Singh, R.; Gehlot, A.; Satapathy, S.C.; Alnumay, W.S.; Pelusi, D.; Ghosh, U.; Nayak, J. Industrial Internet of Things and its applications in industry 4.0: State of the art. *Comput. Commun.* **2021**, *166*, 125–139. [CrossRef]
3. Pliatsios, D.; Sarigiannidis, P.; Lagkas, T.; Sarigiannidis, A.G. A survey on SCADA systems: Secure protocols, incidents, threats and tactics. *IEEE Commun. Surv. Tutorials* **2020**, *22*, 1942–1976. [CrossRef]
4. Statista. Industrial Internet of Things (IIoT) Market Size Worldwide from 2020 to 2028. Available online: https://www.statista.com/statistics/611004/global-industrial-internet-of-things-market-size/ (accessed on 28 November 2022).
5. Dhirani, L.L.; Armstrong, E.; Newe, T. Industrial IoT, cyber threats, and standards landscape: Evaluation and roadmap. *Sensors* **2021**, *21*, 3901. [CrossRef] [PubMed]
6. Grammatikis, P.I.R.; Sarigiannidis, P.G.; Moscholios, I.D. Securing the Internet of Things: Challenges, threats and solutions. *Internet Things* **2019**, *5*, 41–70. [CrossRef]
7. McNulty, L.; Vassilakis, V.G. IoT Botnets: Characteristics, Exploits, Attack Capabilities, and Targets. In Proceedings of the 2022 13th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP), Porto, Portugal, 20–22 July 2022; pp. 350–355.
8. Winter, T. RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks; *Internet-Draft, ROLL, IETF* 2012. Available online: https://www.rfc-editor.org/rfc/rfc6550 (accessed on 28 November 2022).
9. Phinney, T.; Thubert, P.; Assimiti, R. RPL applicability in industrial networks: Draft-phinney-roll-rpl-industrial-applicability-00. *Internet-Draft* **2011**. Available online: https://datatracker.ietf.org/doc/html/draft-ietf-roll-rpl-industrial-applicability-00 (accessed on 28 November 2022).
10. Kharrufa, H.; Al-Kashoash, H.A.; Kemp, A.H. RPL-based routing protocols in IoT applications: A review. *IEEE Sens. J.* **2019**, *19*, 5952–5967. [CrossRef]
11. Kelli, V.; Argyriou, V.; Lagkas, T.; Fragulis, G.; Grigoriou, E.; Sarigiannidis, P. IDS for industrial applications: A federated learning approach with active personalization. *Sensors* **2021**, *21*, 6743. [CrossRef] [PubMed]
12. Zhang, Y.; Yang, C.; Huang, K.; Li, Y. Intrusion detection of industrial internet-of-things based on reconstructed graph neural networks. *IEEE Trans. Netw. Sci. Eng.* **2022**. [CrossRef]
13. Lakshmanna, K.; Kaluri, R.; Gundluru, N.; Alzamil, Z.S.; Rajput, D.S.; Khan, A.A.; Haq, M.A.; Alhussen, A. A Review on Deep Learning Techniques for IoT Data. *Electronics* **2022**, *11*, 1604. [CrossRef]
14. Vashishtha, M.; Chouksey, P.; Rajput, D.S.; Reddy, S.R.; Reddy, M.P.K.; Reddy, G.T.; Patel, H. Security and detection mechanism in IoT-based cloud computing using hybrid approach. *Int. J. Internet Technol. Secur. Trans.* **2021**, *11*, 436–451. [CrossRef]
15. Canbalaban, E.; Sen, S. A cross-layer intrusion detection system for RPL-based internet of things. In *Proceedings of the International Conference on Ad-Hoc Networks and Wireless*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 214–227.
16. Pasikhani, A.M.; Clark, J.A.; Gope, P. Adversarial RL-Based IDS for Evolving Data Environment in 6LoWPAN. *IEEE Trans. Inf. Forensics Secur.* **2022**, *17*, 3831–3846. [CrossRef]

17. Ioulianou, P.P.; Vassilakis, V.G.; Shahandashti, S.F. ML-based Detection of Rank and Blackhole Attacks in RPL Networks. In Proceedings of the 2022 13th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP), Porto, Portugal, 20–22 July 2022; pp. 338–343.
18. Contiki, O.S. Available online: http://www.contiki-os.org/ (accessed on 28 November 2022).
19. Cooja Simulator. Available online: https://anrg.usc.edu/contiki/index.php/Cooja_Simulator (accessed on 28 November 2022).
20. Contiki-NG. Available online: https://github.com/contiki-ng/contiki-ng/wiki (accessed on 28 November 2022).
21. Mahbub, M. Progressive researches on IoT security: An exhaustive analysis from the perspective of protocols, vulnerabilities, and preemptive architectonics. *J. Netw. Comput. Appl.* **2020**, *168*, 102761. [CrossRef]
22. Zarpelao, B.B.; Miani, R.S.; Kawakani, C.T.; de Alvarenga, S.C. A survey of intrusion detection in Internet of Things. *J. Netw. Comput. Appl.* **2017**, *84*, 25–37. [CrossRef]
23. Ioulianou, P.P.; Vassilakis, V.G. Denial-of-service attacks and countermeasures in the RPL-based Internet of Things. In *Computer Security*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 374–390.
24. Raoof, A.; Matrawy, A.; Lung, C.H. Routing Attacks and Mitigation Methods for RPL-Based Internet of Things. *IEEE Commun. Surv. Tutor.* **2018**, *21*, 1582–1606. [CrossRef]
25. Le, A.; Loo, J.; Chai, K.; Aiash, M. A specification-based IDS for detecting attacks on RPL-based network topology. *Information* **2016**, *7*, 25. [CrossRef]
26. Ghaleb, B.; Al-Dubai, A.; Ekonomou, E.; Qasem, M.; Romdhani, I.; Mackenzie, L. Addressing the DAO insider attack in RPL's Internet of Things networks. *IEEE Commun. Lett.* **2018**, *23*, 68–71. [CrossRef]
27. Pongle, P.; Chavan, G. A survey: Attacks on RPL and 6LoWPAN in IoT. In Proceedings of the 2015 International Conference on Pervasive Computing (ICPC), Pune, India, 8–10 January 2015; pp. 1–6.
28. Kamble, A.; Malemath, V.S.; Patil, D. Security attacks and secure routing protocols in RPL-based Internet of Things: Survey. In Proceedings of the 2017 International Conference on Emerging Trends & Innovation in ICT (ICEI), Pune, India, 3–5 February 2017; pp. 33–39.
29. Samuel, C.; Alvarez, B.M.; Ribera, E.G.; Ioulianou, P.P.; Vassilakis, V.G. Performance evaluation of a wormhole detection method using round-trip times and hop counts in RPL-based 6LoWPAN networks. In Proceedings of the 2020 12th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP), Bari, Italy, 19–23 July 2020; pp. 1–6.
30. Datsika, E.; Vardakas, J.; Kalfas, G.; Vagionas, C.; Mesodiakaki, A.; Verikoukis, C. End-to-end delay performance of analog fiber wireless architecture for 5G NR fronthaul. In Proceedings of the 22nd International Conference on Transparent Optical Networks (ICTON), Porto, Portugal, 20–22 July 2020; pp. 1–4.
31. Vasseur, J.; Agarwal, N.; Hui, J.; Shelby, Z.; Bertrand, P.; Chauvenet, C. *RPL: The IP Routing Protocol Designed for Low Power and Lossy Networks*; Internet Protocol for Smart Objects (IPSO) Alliance: 2011; Volume 36. Available online: http://www.cse.chalmers.se/edu/year/2016/course/DAT285B/PAPERS/rpl.pdf (accessed on 28 November 2022).
32. Huston, G. The changing foundation of the internet: Confronting IPv4 address exhaustion. *Internet Protoc. J.* **2008**, *11*, 19–36.
33. Mulligan, G. The 6LoWPAN architecture. In Proceedings of the 4th Workshop on Embedded Networked Sensors, ACM, Cork, Ireland, 25–26 June 2007; pp. 78–82.
34. Gaddour, O.; Koubâa, A. RPL in a nutshell: A survey. *Comput. Netw.* **2012**, *56*, 3163–3178. [CrossRef]
35. Korte, K.D.; Sehgal, A.; Schönwälder, J. A study of the RPL repair process using ContikiRPL. In Proceedings of the IFIP International Conference on Autonomous Infrastructure, Management and Security, Munich, Germany, 4–5 June 212; Springer: Berlin/Heidelberg, Germany, 2012; pp. 50–61.
36. Mayzaud, A.; Badonnel, R.; Chrisment, I. A taxonomy of attacks in RPL-based Internet of Things. *Int. J. Netw. Secur.* **2016**, *18*, 459–473.
37. Wallgren, L.; Raza, S.; Voigt, T. Routing attacks and countermeasures in the RPL-based internet of things. *Int. J. Distrib. Sens. Netw.* **2013**, *9*, 794326. [CrossRef]
38. Kumar, A.; Matam, R.; Shukla, S. Impact of packet dropping attacks on RPL. In Proceedings of the 4th International Conference on Parallel, Distributed and Grid Computing (PDGC), Waknaghat, India, 22–24 December 2016; pp. 694–698.
39. Pu, C. Spam DIS Attack Against Routing Protocol in the Internet of Things. In Proceedings of the 2019 International Conference on Computing, Networking and Communications (ICNC), Istanbul, Turkey, 25–29 April 2019; pp. 73–77.
40. Aris, A.; Oktug, S.F.; Yalcin, S.B.O. RPL version number attacks: In-depth study. In Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS), Honolulu, HI, USA, 18–21 February 2016; pp. 776–779.
41. Ioulianou, P.P.; Vassilakis, V.G.; Logothetis, M.D. Battery Drain Denial-of-Service Attacks and Defenses in the Internet of Things. *J. Telecommun. Inf. Technol.* **2019**, 37–45. [CrossRef]
42. Milenkoski, A.; Vieira, M.; Kounev, S.; Avritzer, A.; Payne, B.D. Evaluating computer intrusion detection systems: A survey of common practices. *ACM Comput. Surv. (CSUR)* **2015**, *48*, 12. [CrossRef]
43. Anantvalee, T.; Wu, J. A survey on intrusion detection in mobile ad hoc networks. In *Wireless Network Security*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 159–180.
44. Kumar, S.; Dutta, K. Intrusion detection in mobile ad hoc networks: Techniques, systems, and future challenges. *Secur. Commun. Netw.* **2016**, *9*, 2484–2556. [CrossRef]
45. Abduvaliyev, A.; Pathan, A.S.K.; Zhou, J.; Roman, R.; Wong, W.C. On the vital areas of intrusion detection systems in wireless sensor networks. *IEEE Commun. Surv. Tutor.* **2013**, *15*, 1223–1237. [CrossRef]

46. Modi, C.; Patel, D.; Borisaniya, B.; Patel, H.; Patel, A.; Rajarajan, M. A survey of intrusion detection techniques in cloud. *J. Netw. Comput. Appl.* **2013**, *36*, 42–57. [CrossRef]
47. Raza, S.; Wallgren, L.; Voigt, T. SVELTE: Real-time intrusion detection in the Internet of Things. *Ad Hoc Netw.* **2013**, *11*, 2661–2674. [CrossRef]
48. Cervantes, C.; Poplade, D.; Nogueira, M.; Santos, A. Detection of sinkhole attacks for supporting secure routing on 6LoWPAN for Internet of Things. In Proceedings of the 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM), Ottawa, ON, Canada, 11–15 May 2015; pp. 606–611.
49. Ioulianou, P.; Vassilakis, V.; Moscholios, I.; Logothetis, M. A Signature-based Intrusion Detection System for the Internet of Things. In Proceedings of the Information and Communication Technology Forum, Graz, Austria, 11–13 July 2018.
50. Oh, D.; Kim, D.; Ro, W. A malicious pattern detection engine for embedded security systems in the Internet of Things. *Sensors* **2014**, *14*, 24188–24211. [CrossRef] [PubMed]
51. Wu, S.; Manber, U. *A Fast Algorithm for Multi-Pattern Searching*; University of Arizona, Department of Computer Science: Tucson, AZ, USA, 1994.
52. Alohali, B.A.; Vassilakis, V.G.; Moscholios, I.D.; Logothetis, M.D. A secure scheme for group communication of wireless IoT devices. In Proceedings of the 2018 11th International Symposium on Communication Systems, Networks & Digital Signal Processing (CSNDSP), Budapest, Hungary, 18–20 July 2018; pp. 1–6.
53. Tahsien, S.M.; Karimipour, H.; Spachos, P. Machine learning based solutions for security of Internet of Things (IoT): A survey. *J. Netw. Comput. Appl.* **2020**, *161*, 102630. [CrossRef]
54. Olsson, J. 6LoWPAN demystified. *Tex. Instruments* **2014**, *13*, 1–13.
55. Iuchi, K.; Matsunaga, T.; Toyoda, K.; Sasase, I. Secure parent node selection scheme in route construction to exclude attacking nodes from RPL network. In Proceedings of the 2015 21st Asia-Pacific Conference on Communications (APCC), Kyoto, Japan, 14–16 October 2015; pp. 299–303.
56. Tiwari, M.; Arya, K.V.; Choudhari, R.; Choudhary, K.S. Designing intrusion detection to detect black hole and selective forwarding attack in WSN based on local information. In Proceedings of the 2009 Fourth International Conference on Computer Sciences and Convergence Information Technology, Seoul, Republic of Korea, 24–26 November 2009; pp. 824–828.
57. Kurosawa, S.; Nakayama, H.; Kato, N.; Jamalipour, A.; Nemoto, Y. Detecting blackhole attack on AODV-based mobile ad hoc networks by dynamic learning method. *IJ Netw. Secur.* **2007**, *5*, 338–346.
58. Ribera, E.G.; Alvarez, B.M.; Samuel, C.; Ioulianou, P.P.; Vassilakis, V.G. Heartbeat-based detection of blackhole and greyhole attacks in RPL networks. In Proceedings of the 2020 12th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP), Porto, Portugal, 20–22 July 2020; pp. 1–6.
59. Tian, C.; Zhang, Y.; Zuo, W.; Lin, C.W.; Zhang, D.; Yuan, Y. A heterogeneous group CNN for image super-resolution. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**. [CrossRef] [PubMed]
60. Tian, C.; Zheng, M.; Zuo, W.; Zhang, B.; Zhang, Y.; Zhang, D. Multi-stage image denoising with the wavelet transform. *Pattern Recognit.* **2023**, *134*, 109050. [CrossRef]