

Securing RPL-Based Networks Against Version Number and Rank Attacks

Alaa Eddine Khalfoune ^{1,2} , Rachid Beghdad ^{1,2} 

¹ Département d'Informatique, Faculté des Sciences Exactes, Université de Bejaia, Bejaia, Algeria

² Laboratoire LAMIE, Faculté des Mathématiques et d'Informatique, Université de Batna 2, Batna, Algeria

Corresponding author: Rachid Beghdad (rachid.beghdad@univ-bejaia.dz)

Abstract

The increasing reliance on Low-power and Lossy Networks (LLN) in the Internet of Things (IoT) and their vulnerability to various attacks have made their protection necessary. Most of the proposed approaches to protecting such networks neither support scalability nor are lightweight enough to be incorporated into these constrained networks. In this paper, we present a lightweight approach to protecting LLN networks from rank and version number attacks. The proposal relies on light exchanged messages between the network nodes and the root and network nodes themselves. Successive comparison processes are used to detect the attack while two blocking techniques are introduced against malicious nodes. Simulations demonstrated the effectiveness of the approach, outperforming similar approaches such as Sink-Based Intrusion Detection Systems (SBIDS), Secure Routing Protocol (SRPL) for LLN, the Machine-Learning Technique based on K-Nearest Neighbour (MLTKNN), secure trust-aware Routing Protocol for Low Power and Lossy Networks (SecTrust) and Shield in terms of appropriate parameters, without causing extra charges on the network.

Keywords

Internet of things; LLN; Routing protocol for low power and lossy networks; Security; Blocking techniques.

Citation: Khalfoune, A.E., & Beghdad, R. (2024). Securing RPL-Based Networks Against Version Number and Rank Attacks. *Acta Informatica Pragensia*, 13(3), 340–358. <https://doi.org/10.18267/j.aip.234>

Academic Editor: Zdenek Smutny, Prague University of Economics and Business, Czech Republic

Copyright: © 2024 by the author(s). Licensee Prague University of Economics and Business, Czech Republic.

This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution License (CC BY 4.0).

1 Introduction

The advantages provided by Internet of Things (IoT) networks such as mobility, wirelessness and absence of an energy source have made it in demand in different areas such as smart grids, connected vehicles, healthcare, environmental monitoring and so on, which have in turn increased the numbers of connected IoT devices to cover almost everything around us. IoT devices might suffer from scarce resources, especially processing power, energy and memory; these restrictions have led to the creation of the Low-power and Lossy Networks (LLN), which is intended for constrained networks. The Routing Protocol for Low power and lossy Networks (RPL) is considered a result of applying the IPv6 to LLN networks to allow them communicate with the internet and to provide the needed number of addresses (Rehman et al., 2016). IoT devices are vulnerable to different internal and external attacks, especially at the routing layer. Among the most destructive attacks are version number and rank attacks; however, the constrained nature of RPL makes the implementation of an adequate security system impossible. Thus, designing lightweight protection countermeasures for the RPL protocol becomes more than necessary.

Several approaches have proposed to encounter this attack. Nikravan et al. (2018) presented a signature-based scheme where the root is responsible for generating signatures, whereas other nodes use generated signatures to validate version number increase;; however, the authors did not assess the scheme's performance practically. Airehrour et al. (2019) proposed a trust-based approach to overcome the rank attack in which each node is responsible for computing the trustworthiness of its neighbours based on their behaviour, then forwarding packets via trustworthy neighbours only. However, this solution is not suitable for constrained networks since nodes must run in a promiscuous mode on the clock. Shafique et al. (2018) presented a centralized approach to detect illegitimate rank changes using DAO messages; nevertheless, their approach takes considerable time to detect a distant malicious node from the root.

Glissa et al. (2016) used hash authentication to validate rank changes, but the effectiveness of their solution decreased dramatically as the number of nodes increased. Ioulianou et al. (2018) presented a hybrid approach where monitoring detectors are distributed among the nodes to control the traffic and inform the sink in the case of any suspicious behaviour, which in turn make the decision; however, the authors did not provide any performance assessment. Zaminkar & Fotuhi (2020) presented a comparison mechanism to detect rank manipulation based on the parent's rank and the received rank in DIO messages; they assumed that a malicious node has always zero as a rank value, which is not practicable. A lightweight approach was proposed by Arış et al. (2019); despite being accurate, their approach has a high false negative rate.

Due to the limitations of the previous works dedicated to rank and version number attacks, we propose a combined solution to enhance RPL security by introducing two lightweight mechanisms. In the first mechanism, the root manages a table that contains information about all the nodes of the network. The table is filled by information messages sent by the nodes and the root performs successive checking processes upon receiving information messages to detect any illegitimate rank change. If so, the root warns the nodes using an alert message that contains the malicious node address. The second mechanism is deployed in the nodes, in which each node owns a checking table that contains all neighbour IDs, ranks and version numbers. Upon receiving an incremented version number, the receiver node computes a probability value using the stored information in the checking table and decides whether to update the version number based on the conducted probability value. We present a novel isolation technique to handle malicious nodes.

The proposed approach was implemented and evaluated using the Cooja simulator of Contiki OS, we adapted the RPL stack files to create the needed data structures and comparison functions; the proposed method is executed automatically by the nodes upon running the RPL network. The performance

assessment proved that the proposed approach gives promising results and outperforms similar approaches in terms of power consumption, detection accuracy and PDR.

The rest of the paper is organised as follows: Section 2 covers the RPL background. Section 3 describes related work. The proposed approach is presented in Section 4. Simulations and performance analysis are discussed in Section 5. Finally, Section 6 concludes the paper.

2 Background

2.1 RPL protocol overview

In RPL, the network is represented as a destination-oriented directed acyclic graph (DODAG), which is created and maintained using four main messages. The DODAG root is also a border router for the network nodes; the root is responsible for shaping the network formation by broadcasting DODAG information object (DIO) messages. When the nodes receive a DIO message, they join the network and choose the root as a default parent (Perazzo et al., 2017).

Upon joining the network, the root children forward updated DIO messages to their neighbours containing the rank value; depending on that rank, every node chooses its preferred parent, which will become the default gateway to the root. At the end of the process, all nodes in the network have an upward default path to the root. The destination advertisement object (DAO) message is used to propagate destination information upward along the DODAG; it may optionally be acknowledged by its destination with a DAO-ACK message back to the DAO sender. Finally, a DIS message is broadcast by the unattached nodes that could not obtain a DIO message to solicit potential parents; the receiver of the DIS message answers with a DIO message to help the sender join the network. Figure 1 shows the flow of control messages building and maintaining the DODAG.

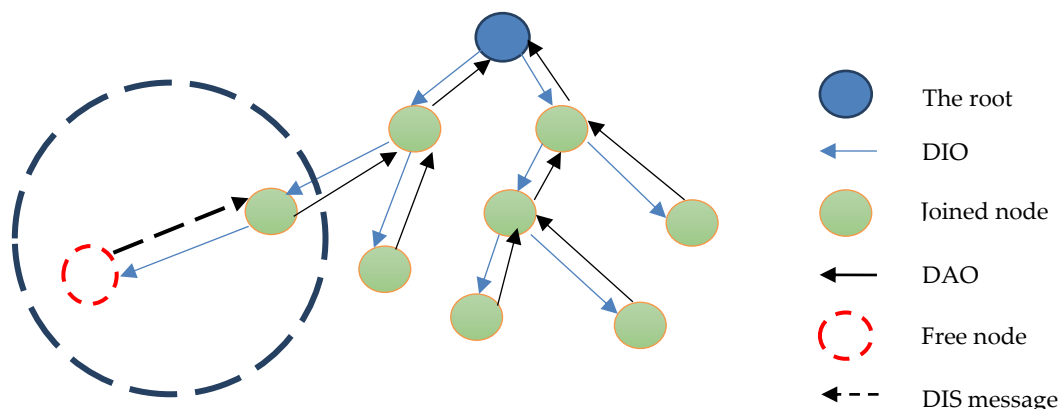


Figure 1. Flow of DIO, DAO and DIS messages in DODAG.

2.2 RPL threats and vulnerabilities

RPL networks are vulnerable to external and internal attacks that manipulate RPL specifications (fields of control messages); the internal ones are considered more dangerous because most of them use a compromised node to perform the attack internally, which hampers the detection process and causes much more damage. RPL has two self-healing mechanisms: local repair and global repair. Local repair is executed by a node in the case of link failure with the preferred parent or in the case of loop detection. It allows the corresponding node to find an alternative path or preferred parent to route its data; the alternative path might not be the best. Global repair is triggered by the root in the event of detecting many inconsistencies in the network. The root updates the version number and broadcasts a DIO with the new version number; thus, all nodes are required to join the new DODAG and the whole network topology is rebuilt.

A rank attack is executed by an illegal change of rank property either by increasing or decreasing as a first step to perform numerous attacks such as selective forwarding attack, black hole attack, sinkhole attack and DAG inconsistency attack. Unfortunately, there is no mechanism implemented in RPL to protect the rank property and every single node in the network can change its own rank freely; therefore, many works in the literature have proposed ways to overcome the rank attack.

On the other hand, the version number property refers to the latest created DODAG; the root is the only node allowed to change this property when it decides to reconstruct the topology (global repair), which occurs in the case of severe damage to the topology; a malicious node might increment the version number illegitimately before sending a DIO message. Unfortunately, there is no mechanism for the receiver node to distinguish whether the version number was incremented by the root or by a malicious node; this leads to unnecessary reconstruction of the topology as well as depletion of resources.

Based on the impact of each attack and the studies of Ariş et al. (2016) and Le et al. (2013), we concluded that rank and version number attacks have the largest destructive capacity on an RPL-based network while being easy to perform by internal nodes.

3 Related Work

The security of RPL-based networks has been widely studied and discussed by researchers recently. They have tried to secure these networks against different attacks; Alzubaidi et al. (2018), Raza et al. (2013) and Zaminkar & Fotohi (2020) have proposed mitigation solutions against sinkhole attacks based on rank property. Alzubaidi et al. (2018) proposed an approach that uses two powerful surveillance nodes; the first covers the circumference of the sink, while the second covers the network border. Both nodes are responsible for gathering data and exchanging query packets to detect sinkhole attacks performed by faraway nodes claiming to be close to the sink. Raza et al. (2013) proposed SVELTE to detect altered information, sinkhole and selective forwarding attacks. It consists of three modules: 6Mapper, which gathers information about all nodes based on request packets; a detection component that analyses the collected data to detect any malicious behaviour; and a mini firewall against external network attacks. Despite promising results in true positive rates, SVELTE might suffer from the single point of failure problem and scalability problems. Zaminkar & Fotohi (2020) presented a defence technique against sinkhole and flooding attacks; they defined a threshold to restrict the number of received DIO messages in a slot of time to avoid the DIO flooding attacks. However, the assumption of a sinkhole node having zero as a rank is not realistic.

Ariş et al. (2019), Ioulianou et al. (2018), Nikravan et al. (2018), Ahmed & Ko (2016), Alsukayti & Singh (2022) and Osman et al. (2021b) have discussed the version number attack. Nikravan et al. (2018) introduced a lightweight approach to mitigate version number and rank attacks, based on an offline–online signature scheme. The complicated tasks such as generating private key and offline signature are done in offline mode, both version number and rank properties are protected and encrypted before being sent using the generated keys and signatures of each node; then, receiver node reverses the process to check the validity of both properties. However, the proposed approach was not assessed in practical realistic simulations. Ioulianou et al. (2018) presented IDS against DoS and version number attacks (DVIDS), where light detectors are deployed among the network nodes and are responsible for monitoring the traffic. If a node's behaviour appears to be a familiar attack, the related packet is forwarded to the sink, which in turn decides whether to block the responsible node by checking the stored signatures of familiar attacks; unfortunately, the proposed approach has not been evaluated.

Ariş et al. (2019) proposed a lightweight technique where they introduced two rules to avoid illegitimate version number updates. In the first one, a version number update cannot be announced by a deeper node to a closer node to the root, while in the second one, a version number update is made by a given node if

and only if the majority of closer neighbours has updated their version numbers. Despite of the efficiency of the proposed approach under different topologies, it has a very high false negative rate (95%). Ahmed & Ko (2016) presented a distributed and cooperative verification mechanism (DVCM); upon receiving an incremented version number via a DIO message, each node starts a cooperative checking process. It sends version number checking queries to two-hop neighbours and stores the received answers with senders' addresses in a temporary table; based on recorded version numbers, the receiver node checks the temporary table to decide whether the version number was updated legitimately. Even with very promising results, the approach seems to be costly in terms of energy due to the many exchanged queries, which is not covered by the authors' performance evaluation.

Alsukayti & Singh (2022) provided a collaborative and distributed security scheme called CDRPL, where the nodes check the legitimacy of an incremented version number before participating in a global repair. The authors created a copy version of a DIO message called an S-DIO message, which has a modified flag and additional information (parent information). The S-DIO message is broadcast by each node in the network that receives a regular DIO message with the updated version number; then, comparison processes are triggered to verify the legitimacy of the received version number using the S-DIO message and the sender information. In the case of an illegal increment, the recipient discards the DIO message and informs the sink using an edited DAO message, which in turn alerts the other DODAG nodes using the aforementioned S-DIO message to blacklist the malicious node. Despite the promising results shown by the performance evaluation, the authors did not provide any authentication methods to secure the edited exchanged S-DIO and S-DAO control messages, which makes its falsifying a simple task for intruders. Osman et al. (2021b) presented a machine learning model to detect VN attacks. The proposed method consists of four phases: the first is data collection, in which data about all exchanged control messages in the network are sniffed and saved in a file; the second is data re-processing, in which the collected data are treated to become more suitable for machine learning to read; the third is feature selection, where only needed and relevant information are kept; the fourth is machine learning, where the developed scheme operates on the data and gives results. The proposed approach was assessed using the Cooja simulator in terms of accuracy, precision, detection rate and F-score. Regardless of its superiority compared to standard RPL and similar approaches, the proposed solution was not tested under realistic conditions such as network size and number of malicious nodes.

The rank attack has been widely studied and Airehrour et al. (2019), Glissa et al. (2016), Nikravan et al. (2018), Shafique et al. (2018), Karmakar et al. (2021), Neerugatti & Reddy (2019) and Osman et al. (2021a) have presented various mitigating approaches to counteract it. Airehrour et al. (2019) proposed an approach named SecTrust against Sybil and rank attacks, where each node computes the trustworthiness of its direct neighbours as well as second-hop neighbours based on the behaviour of each of them and nodes with high trust values are used for routing. However, the proposed approach is not suitable for constrained networks since nodes must run in promiscuous mode and using only few trusty nodes all the time to secure communications depletes their energy in time. Shafique et al. (2018) introduced a sink-based IDS (SBIDS). The authors edited the DAO control messages to carry more information about the network nodes to the sink; the latter performs consecutive verification processes to check the legitimacy of the sender node. The SBIDS achieved high accuracy detecting the rank attacks under both static and mobility cases; nevertheless, it takes considerable time to detect attacks by distant malicious nodes (25 seconds to capture the 8th hop).

Osman et al. (2021a) proposed an artificial neural network called MLRPL to detect the decreased rank attack. In the proposed solution, the authors used a data vector as an input layer; then, the data are treated in three hidden layers by neurons using a rectified linear function, whereas the output layer uses a logistic function as an activation function. The proposed solution was evaluated in terms of accuracy, detection rate, precision and accuracy, and the results showed the superiority of the proposed method in detecting

attacks compared to similar methods. However, the authors did not provide any implementation techniques for the proposed solution in RPL-based networks. Karmakar et al. (2021) proposed a lightweight method to overcome rank attacks called LEADER, which consists of two modules. The first module is distributed in all network nodes and it is responsible for gathering and sending information to the sink using modified DAO control messages; the second module is located in the sink and is responsible for the checking processes and detecting of malicious behaviour using the data collected from the nodes. Moreover, the authors used a lightweight hash function called LOCHA to guarantee the integrity of sent data. The proposed countermeasure was assessed in terms of accuracy rate, energy consumption, latency and PDR, and the results proved the efficiency of the proposed method; nevertheless, the proposed method was compared only to one similar proposed approach.

Glissa et al. (2016) presented SRPL, an authentication approach against sinkhole, black hole and selective forwarding attacks; the root generates three random numbers, hashes them with one-way hashing function, then broadcasts them; each receiver uses the hashed values and the rank as entries to compute its own hashed values that allow each preferred parent to monitor its children. SRPL achieved high performance in terms of PDR, control message overheads and power consumption; unfortunately, it takes a long time to react and its performance decreases dramatically when the number of nodes increases. Neerugatti & Reddy (2019) introduced a machine learning approach named MLTKNN based on K-nearest neighbour. The nodes unicast periodical DAO messages that contain the rank value to the sink waiting for an acknowledgement message; upon receiving the DAO messages, the root uses the sender node aggregating distances of intermediate nodes (if any) to calculate the distance of the sender, which must match the sender's rank; however, the evaluation was done with only one malicious node.

Pongle & Chavan (2015) presented RTIDS, an IDS based on location information and received signal strength indicator (RSSI) to detect wormhole attacks. Each node checks periodically the number of neighbours; if it increases, it sends an information message with neighbours' IDs to the sink. The latter uses the stored information and the RSSI values to check whether each pair of neighbours are inside their ranges; if they are not, the sink sends a victim packet to both nodes containing their addresses and they respond with the RSSI values of each other; finally, the sink uses collected RSSI information to locate the suspicious nodes and the node with higher suspicion times is considered the attacker node.

Ariş & Oktug (2020) discussed the case of multiple version number attackers. They evaluated the network performance under different attacker positions and different attacker numbers. After a set of tests and simulations, the authors concluded that increasing the number of attackers decreases the PDR, slightly increases the power consumption and causes shorter average delay; for attacker position, the authors concluded that when attacker nodes are placed closer to the root, they cause more damage in terms of power consumption and average delay, while central positions are more effective for PDR. The authors evaluated the approach proposed in Ariş et al. (2019) for aforementioned scenarios, concluding that Ariş et al. (2019) performed very well in all cases except attackers closer to the root and three attackers.

3.1 Discussion

Rank and version number attacks have been treated widely in the literature; many authors have proposed different mechanisms to secure the rank property and detect rank attacks. The proposed approaches to secure RPL-based networks against rank attacks are based mainly on promiscuous mode. This mode forces the nodes to stay in a watching state all the time, thus depleting their energy fast and decreasing the network life; moreover, using cryptography and signatures also introduces an extra overhead to the already constrained nodes, especially in terms of memory and processing, which makes applying these solutions impracticable in IoT networks. Additionally, using RPL control messages as a means of detecting malicious nodes in centralized approaches delays the detection time proportionally as the network becomes wider; hence, these approaches prevent network scalability.

On the other hand, few of the proposed approaches have treated the version number attack. These approaches are either difficult to implement realistically on constrained nodes or have recorded high false negative rates as a result of the difficulty of recognizing the difference between legitimate and illegitimate increments in version number. In addition, the proposed works targeting version number attacks did not take into consideration the case of legitimate version number update during the tests.

Given the potential dangers caused by rank and version number attacks and the shortcomings of the proposed approaches, we will propose in the following a solution to mitigate both attacks and overcome the drawbacks cited previously.

4 Proposed Approach

The proposed approach aims to protect an RPL network against rank and version number attacks, this is done by detecting and preventing any illegitimate change of both properties in DIO messages. The proposed mechanism combines both approaches of Ariş et al. (2019) and Shafique et al. (2018) with the appropriate modifications and additions to ameliorate performance. The improved approaches are considered lightweight in terms of resource requirements and they both have great potential of improvement to increase the efficiency. The main contributions of our article are:

- Network protection against both rank attacks and version number attacks.
- Introduction of information and alert communication messages to speed up detection time.
- Introduction of temporary and permanent blocking methods for rank attacks.
- Introduction of a new efficient probability system detecting version number attacks that decreases the false negative rate.
- Improved performance under attack.

Table 1 shows the additional improvements of the proposed approach compared to the combined countermeasures of Shafique et al. (2018) and Ariş et al. (2019) and some recent proposed approaches.

Table 1. Improvements of combined proposed approach.

			Shafique et al. (2018)	Proposed approach
Rank attack	Decreased rank attack detection		✓	✓
	Increased rank attack detection		/	✓
	Malicious node blocking		/	✓
	Performance	PDR	✗	✓
		Detection accuracy	✗	✓
		Detection delay	✗	✓
Version number attack			Ariş et al. (2019)	Proposed approach
	Version number attack detection		✓	✓
	Performance	Power consumption	✗	✓
		Detection accuracy	✗	✓
		PDR	✗	✓
		Control message overheads	✗	✓

4.1 Protocol description

The proposed approach has two kinds of implemented modules. The first is centralized in the root, which is considered an unlimited-resource powerful node that does the heavy tasks. The root maintains a table called a graph table with an entry for each node in the network that contains the node ID, preferred parent ID, rank, altered times, number of suspicions and a timestamp that helps it rebuild and update the network graph (parent/child relations). It uses the received periodic information messages from the network nodes to update the fields of each entry. The graph table is used by the root to mitigate rank attacks.

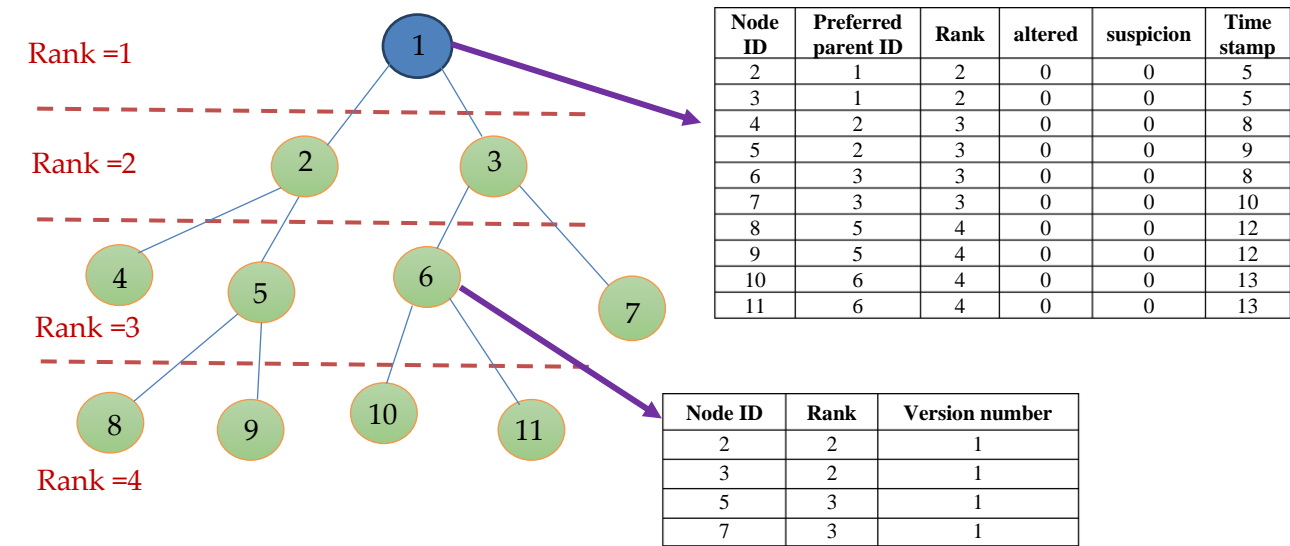


Figure 2. Illustrative examples of graph and checking tables.

The second module is lightweight and distributed along all the nodes. Each node maintains a checking table with rank and version numbers of its neighbours. The table is used to check the legitimacy of version number updates; it is also used by the nodes to detect any abnormal change in rank property; if so, the node sends an information message. Graph and checking tables are illustrated in Figure 2.

4.1.1 Protection against rank attacks

The root uses the graph table to detect and mitigate the rank attack by checking the legitimacy of node ranks periodically when any update occurs (it receives an information message). The proposed technique is based on the approach of Shafique et al. (2018) with additional functions. Shafique et al. (2018) considered each increase in node rank to be an instance of mobility, which is not always the case, since a node can increase its rank illegitimately to perform increased rank attack, which leads to huge battery consumption as mentioned in Sahay (2018). Furthermore, the approach of Shafique et al. (2018) only detects malicious behaviour without any further procedures to eliminate the malicious node, which is not enough to save the network resources.

Information and alert messages

Each node in the network is responsible for sending periodic encrypted information messages to the root, to avoid extra overheads; these information messages are sent occasionally when a node is about to change its preferred parent or when a node receives a very high/low rank value from the preferred parent. The information message contains four fields: the sender rank, address, its preferred parent address and the encryption fields.

Upon receiving the information message, the root decrypts the received message. In the case of any mismatch in encryption fields, the root ignores the message; otherwise, the root extracts the information and updates the graph; then, it starts the checking processes.

When a malicious node is detected, the root performs a broadcasting process, sending the attacker address via an alert message. The receiver nodes broadcast the same message in the network downward, which allows propagation of the message until the leaves of the tree topology. The lightweight format of the alert message is designed to avoid network congestion.

Attacker model

The attacker model follows the next assumptions:

- The attacker can deploy multiple malicious nodes among legitimate nodes.
- The malicious node exchanges control messages as usual legitimate nodes to not be compromised, it does not send any information message to the root.
- The attacker cannot break the encryption system of legitimate nodes.

Figure 3 shows a flowchart of the successive comparison processes of our proposed approach, which is ameliorated and enriched by the approach of Shafique et al. (2018). The framed part in the flowchart indicates the additional processes, including the increased rank attack detection method and blocking mechanisms.

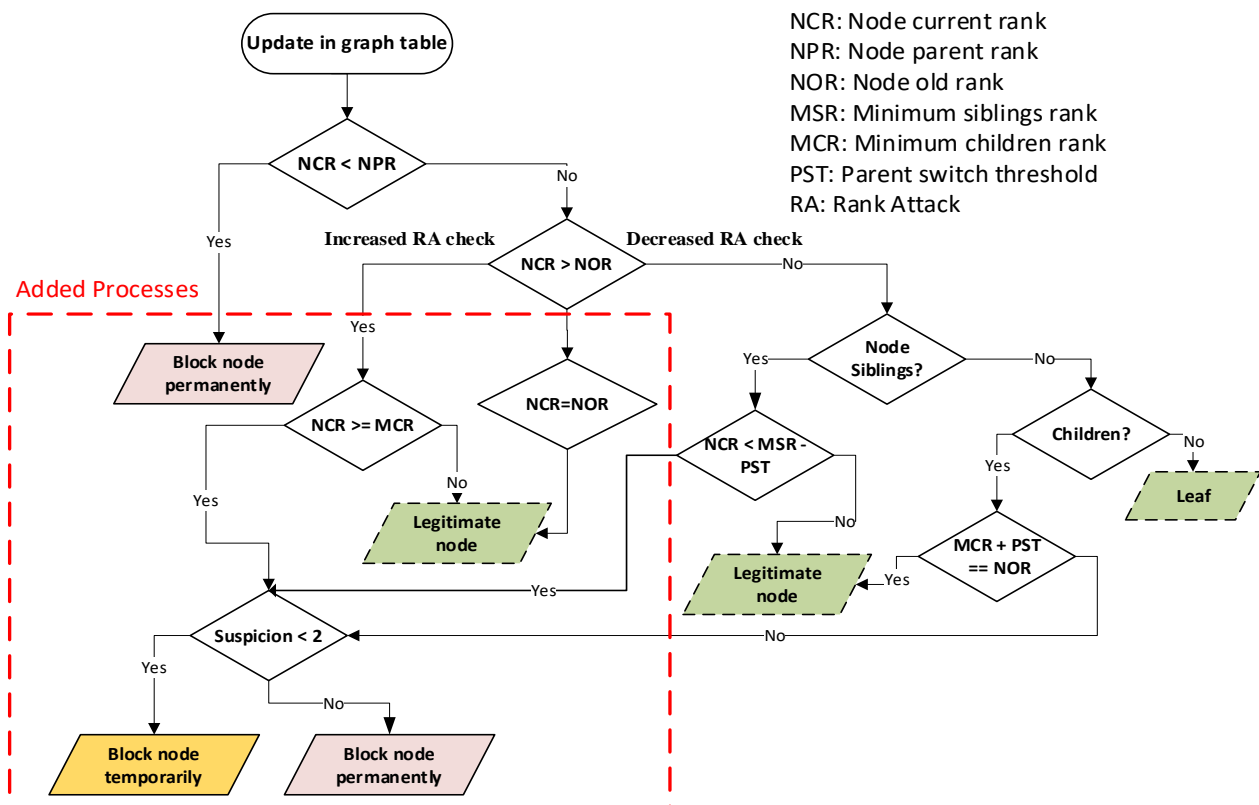


Figure 3. Flowchart of ameliorated rank mitigation technique of SBIDS. Source: (Shafique et al., 2018).

First, each node in the graph table can never have a rank (NCR) lower than its parent's rank (NPR); if so, the node is considered malicious and is blocked permanently. Otherwise, the NCR is compared to the node's old rank (NOR).

If the NCR is less than the NOR, the system checks for decreased rank attack. A node cannot decrease its rank lower than the minimum rank of parent switching threshold of siblings' ranks, if any. Otherwise, the root checks if the corresponding node is a leaf; if so, it is a legitimate node; otherwise, the node's new rank must be equal to the minimum children rank plus the parent switching threshold. If the NCR is greater than the NOR, the system checks whether it is an increased rank attack or a mobility case. Each node in the graph table can never have a rank greater than or equal to its children's minimum rank.

A node can change its preferred parent if and only if the next preferred parent's path cost is better than the current preferred parent's path cost by at least PST. Note also that in each checking process and before rank values become stable, the root uses only fresh information for checking based on the timestamp field. However, if the root finds that the needed information to check is outdated, it can solicit an update by sending a solicitation message.

In the case of a violation of the aforementioned legitimacy rules, altered and suspicion fields are used to deduct the threshold and the appropriate procedure to be executed. If the corresponding node is suspected of performing an increased rank attack, the root increments the altered field. Being suspected of a decreased rank attack increments the altered field by 2 since it is much more dangerous.

If the altered value exceeds 2, the root considers the corresponding node malicious, increments the suspicion field and temporarily blocks it. The root broadcasts an alert message containing the malicious node address, blocking type flag and allowed rank for this malicious node; the children in turn broadcast the same message downward all the way to the leaves. The temporary blocking lasts until the next received DIO message with a legitimate rank from the malicious node. Note that blocking suspicious nodes does not affect the network performance since the children have an alternative preferred parent to route packets through. Figure 4 shows alert message propagation, blocking and unblocking a suspicious node.

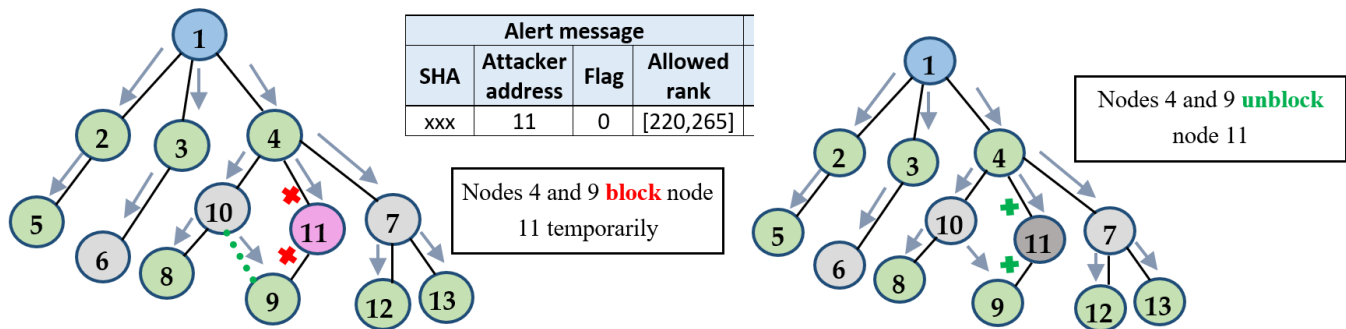


Figure 4. Alert message broadcast by root.

The temporary blocking method decreases the false negative alarms because it allows the network to avoid blocking legitimate nodes permanently since it is possible that a legitimate node's rank gets outside the threshold range once or twice because of disconnecting/connecting processes. However, getting temporarily blocked twice in a small time slot means that the suspicious node is malicious, which causes isolation.

4.1.2 Protection against version number attacks

First, the checking table is initiated without any entries. Upon receiving a DIO message from a neighbour, the receiver node records the neighbour's ID, rank and version number in the checking table. An update process is triggered when it receives a DIO message from an existing neighbour. Algorithm 1 is executed by the nodes when any update occurs in the checking table. We use the same steps as Arış et al. (2019) with additional functionalities.

Algorithm 1: Protection against version number attacks

```

1:  INITIALIZE ( )
2:  when receives a DIO message ADD_ENTRY ( )
3:   $\alpha = 0$   $\beta = 0$            $\omega = 0.75$ 
4:  For each node ND in the checking table
5:    if (ND's version number > node's version number) then
6:      if (ND's rank < node's rank) then

```

```

7:      increment  $\alpha$ 
8:      else
9:      increment  $\beta$ 
10:     end if
11:  end if
12: end for
13: probability  $\gamma = (\alpha + \beta * \omega) / N$ 
    where N is the number of checking table entries
14: Update version number if and only if  $\gamma \geq 0.5$ ;
15: Clear VN fields of checking table

```

Algorithm 1 is executed by all the nodes but the root. The first step is the initialisation step, which allows nodes to create the checking table without any entries. Each entry in this table contains three fields: node ID, rank and the version number, as shown in Figure 2. The table is filled by Procedure 1.

Procedure 1: Fill checking table

```

1: procedure ADD_ENTRY ( )
2:   if (sender exists in the checking table) then
3:     if (sender's rank became much greater than receiver's rank)      then
4:       Delete the sender from the checking table
5:     else
6:       Update rank and version number values of sender
7:     end if
8:   else
9:     if (sender's rank not much greater than receiver's rank) then
10:      Create a new entry for the sender
11:      Add sender's ID, rank and version number
12:    else
13:      Discard the message
14:    end if
15:  end if
16: end procedure

```

Procedure 1 is triggered when the node receives a DIO message; it is responsible for adding new entries as well as updating existing entries in the checking table. If the sender of the DIO message does not exist in the checking table and it has an acceptable rank which is not much greater than the receiver's rank, it is added to the checking table; otherwise, it is ignored.

When an existing node in the graph table sends a new DIO message, the receiver checks whether the sender's rank is still interesting; if so, the receiver maintains the sender's entry and updates its fields in the case of changes. Otherwise, the receiver deletes the sender from the checking table. Procedure 1 is very important since Algorithm 1 relies on the checking table totally to check the validity of version number increment.

Right after the execution of Procedure 1, Algorithm 1 starts checking for any legitimate version number changes (lines 3 to 16 in Algorithm 1). Each node uses its own checking table to avoid illegitimately incremented version numbers. When it receives a DIO message with an incremented version number, it first updates the version number of the sender neighbour in the checking table, then it computes the probability γ and increments the version number if and only if γ is greater than or equal to 0.5 as shown in Algorithm 1 (lines 13 and 14). The probability γ is computed using the number of nodes with lower rank and having already updated their version number (α), the number of nodes with same rank and having already updated their version number (β) and the degradation coefficient ω . We attach more score

to neighbours that have lower rank (closer to the root) than others when computing the probability using the degradation coefficient ω .

The degradation coefficient ω and the probability threshold γ are of great importance in our proposed scheme. After a set of tests and simulations using different values for both variables (ω, γ), we found that using the values 0.75 and 0.5 for ω and γ respectively gives the most accurate results.

This technique was used by Arı̇s et al. (2019), but the authors based it only on neighbours with lower ranks as entries in the checking table; thus, the neighbours that have the same rank or a higher rank were totally ignored. Conversely, our proposed approach adds all neighbours that have lower rank, the same rank or even close rank to the receiver, which we consider the same rank. This is done to reduce the false negative rate by taking into consideration more nodes in the neighbourhood. Additionally, in some cases, surrounding nodes with the same rank update their version numbers before those with higher rank that are in different sub-DODAG.

5 Performance Analysis

The performance evaluation is divided into two parts: first, we evaluated the performance of the rank attack detection mechanism, and then, we evaluated the performance under a version number attack in the second part. Both proposed mechanisms were assessed and compared to similar recently proposed approaches that target the same attacks under the same parameters to deduce realistic results. In this section, we focus more on the results related to the modifications and improvements that were added to the combination.

5.1 Experimental environment

The simulations were performed using the Cooja simulator of Contiki OS. Fifty sky mote nodes were randomly deployed including the root in a 300 x 300 m area, and MRHOF-OF was used as an objective function. Figure 5 shows the random topology.

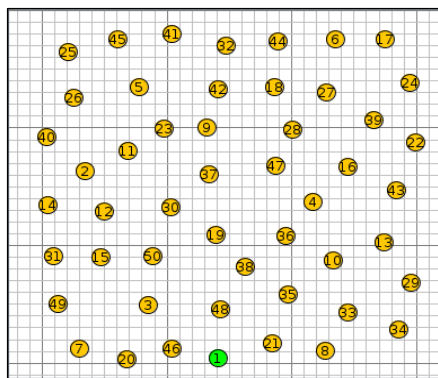


Figure 5. Simulation topology.

5.2 Evaluation of rank attack detection mechanism

First, the performance of the proposed approach was evaluated in terms of different metrics such as power consumption, detection accuracy, PDR and detection latency. For each metric, we conducted ten sets of experiments and recorded the average. The proposed approach performance was compared to that of several other proposed approaches.

The following assumptions were respected during the tests:

- The root is an uncompromised and unconstrained powerful node.
- The attack is performed after a period of network stability.

The reason behind executing the attack after a period of network stability is that the rank decreases sharply before the stable state; moreover, in the discovery state, the nodes do not have a relative rank value to the root yet.

5.2.1 Detection accuracy

The detection accuracy is computed based on the number of correctly detected/undetected malicious nodes to the entire existing malicious nodes in the network. In this experiment, the detection accuracy was assessed with various attack densities (progressively increasing from 0% to 50%). To make the evaluation more realistic, we performed ten experiments and placed the attacking nodes in different positions for each simulation. Figure 6 shows the detection accuracy results compared to SecTrust-RPL, MLTKNN, SRPL and SBIDS.

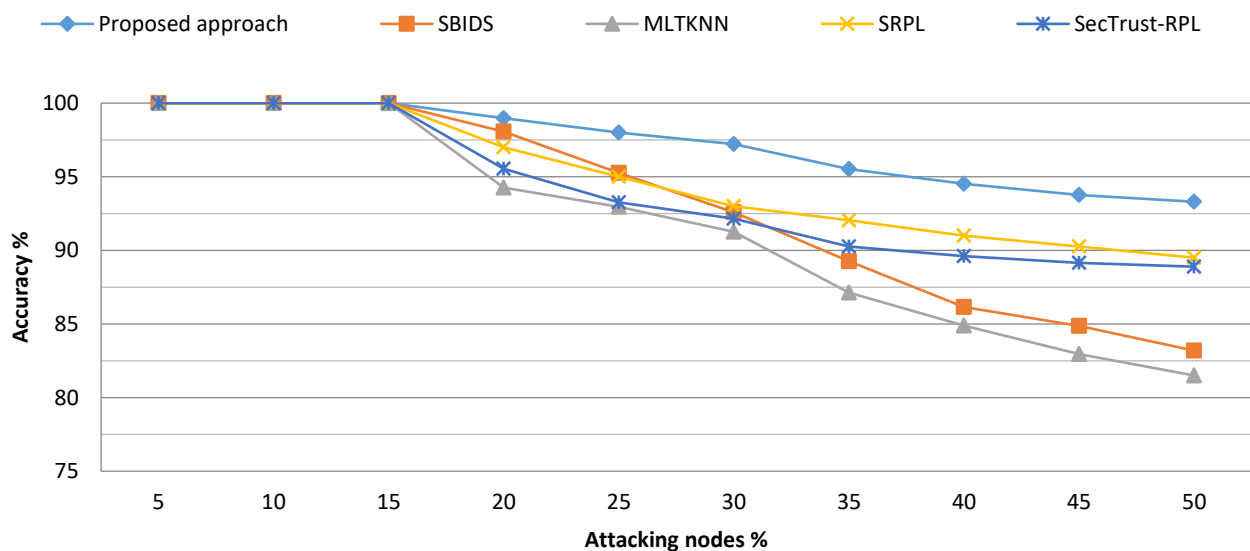


Figure 6. Detection accuracy of proposed approach compared to SecTrust-RPL, SRPL, MLTKNN and SBIDS.

According to Figure 6, all the represented approaches perform well with lower attacking densities since the detection accuracy is stable at 100% for all approaches from 5% to 15% of attacking nodes. However, the detection accuracy decreases proportionally as attack density increases, which is justified by the packet loss due to link failures caused by the attack. For 20% of attacking nodes, the proposed approach excels over the other approaches, reaching 99%, followed by SBIDS, SecTrust and MLTKNN with 98%, 95.5% and 94%, respectively. Nevertheless, as the attacking node percentage increases, the gap between the proposed approach and the other approaches widens gradually.

Starting from 30% of attacking nodes, the detection accuracy of SBIDS and MLTKNN decreases sharply, whereas it starts declining slightly for the proposed approach, SecTrust and SRPL; upon reaching 50% of attacking nodes, the proposed approach achieves the best detection accuracy, outperforming SRPL, SecTrust, SBIDS and MLTKNN by 4.25%, 5%, 12% and 14.5%, respectively.

The superiority of the proposed approach in terms of accuracy has two reasons: the first is integrating the counter and the timestamp technique; secondly, both techniques help reduce notably the false positive rate by giving the suspicious nodes more opportunities when they misbehave unlike the other countermeasures, where the suspicious nodes are detected as malicious.

5.2.2 Detection delay

Detection latency is the time that the system takes to detect malicious behaviour. This time is very important for reducing the damage caused by the attack as much as possible. In the experiment, the

malicious node was placed deep in the eighth hop and we measured the detection latency of the proposed approach compared to similar centralized approaches, namely SBIDS, MLTKNN, RTIDS and DVIDS. The results are shown in Table 2.

Table 2. Comparison of proposed approach and SBIDS, MLTKNN, RTIDS and DVIDS in terms of latency.

	Proposed approach	SBIDS	MLTKNN	RTIDS	DVIDS
Latency (seconds)	9	25	27	14.8	17.9

According to the results in Table 2, the proposed approach has the lowest latency represented by 9 seconds; it outperforms SBIDS, MLTKNN, RTIDS and DVIDS by 16, 18, 5.8 and 8.9 seconds, respectively. The superiority of the proposed approach is justified by the very light tasks assigned to the already constrained nodes, which are responsible only for supplying the root with needed information, whereas they are responsible for some tasks in DVIDS and RTIDS. Moreover, the large size of the messages exchanged between the root and the nodes increases the detection delay much more in SBIDS and MLTKNN. Hence, the proposed approach supports network scalability better than the other centralized approaches.

5.2.3 Packet delivery ratio

The PDR is the ratio of correctly delivered packets to the total packets sent. In the following experiment, we assessed the PDR of the proposed approach and then compared it to similar approaches such as SecTrust, SBIDS, SRPL and MLTKNN. In order to achieve realistic results, the comparison between approaches was made under the same conditions, namely that 10% of the network nodes are malicious and perform rank attacks. Figure 7 shows the plotted results.

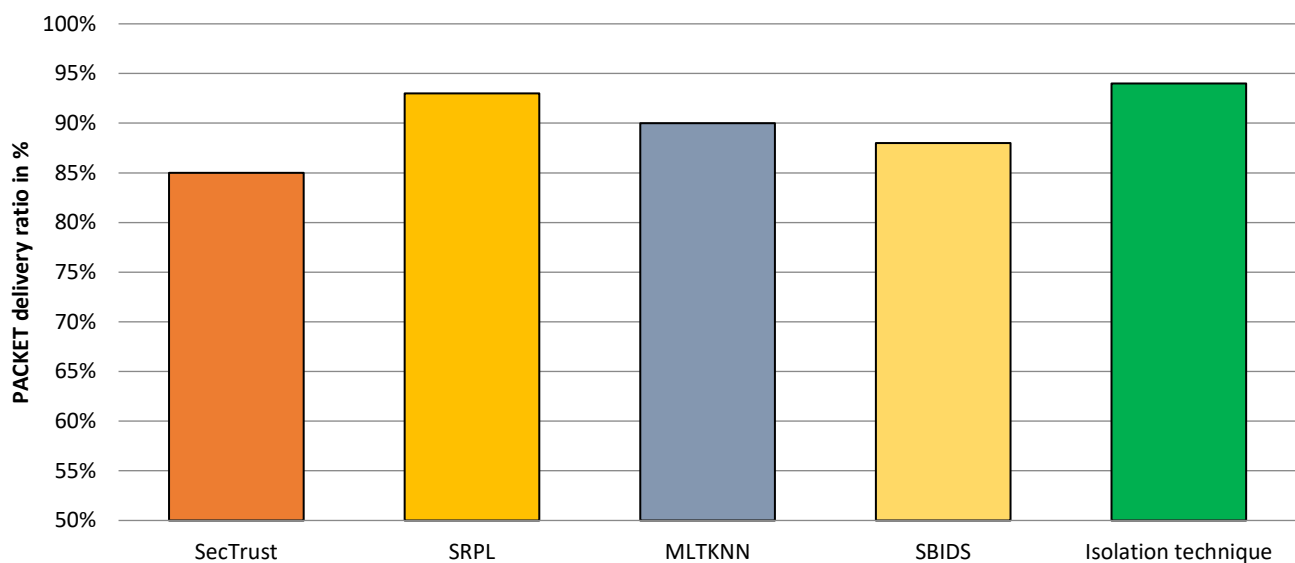


Figure 7. Comparison of proposed isolation technique, SecTrust-RPL, SRPL, MLTKNN and SBIDS in terms of PDR.

According to Figure 7, our proposed isolation technique achieved the highest degree of PDR, surpassing SRPL, MLTKNN, SBIDS and SecTrust, by 1%, 4%, 6% and 9%, respectively. The reason behind the superiority of the proposed approach is the isolation technique. More specifically, when the proposed approach detects a malicious node, an alert message is broadcast from the root; however, if the malicious node's sub-DODAG has no other alternative ways, it cannot receive the alert message so it keeps using the malicious node as a preferred parent to forward the packets. In other approaches, the malicious node is blocked immediately when it is detected; this isolates some nodes and degrades the network's PDR.

5.2.4 Power consumption

Power consumption is a very important factor in the internet of things because of the constrained nature of these networks. In this experiment, we used a powerful tool offered by Cooja called Powertrace to measure the average power consumption for 15 minutes. The average power consumption was measured for both detection and isolation techniques and then compared to the similar approaches as depicted in Figure 8.

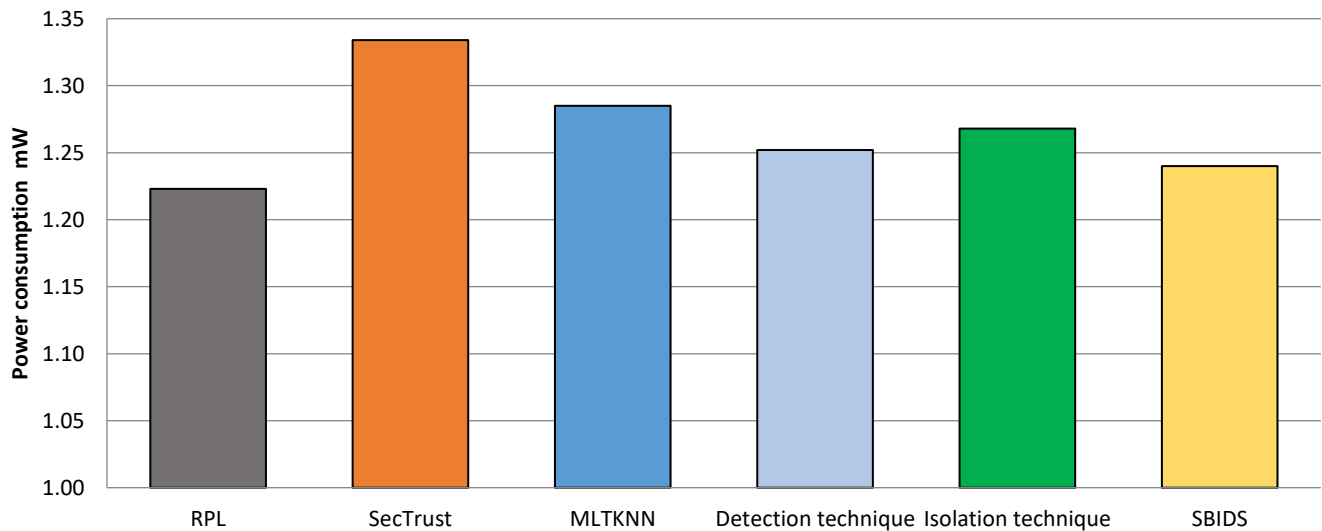


Figure 8. Average power consumption of detection (1st contribution) and isolation (2nd contribution) of proposed approach compared to default RPL, SecTrust, MLTKNN and SBIDS.

According to Figure 8, the average power consumption exceeds 1.30 mW for SecTrust and MLTKNN due to the complicated processes of detecting the malicious nodes, whereas it is approximately the same and ranges between 1.223 mW and 1.268 mW for RPL, the detection technique, the isolation technique and SBIDS. RPL had the lowest average power consumption followed by SBIDS; the latter consumed only 0.017 mW more than normal RPL, which is justified by the increased size of the DAO message, which causes a little more overhead.

The proposed detection technique consumed only 0.012 mW more energy than SBIDS; this can be explained by the information messages exchanged between the root and the nodes. Thus, the nodes spend more energy on creating and sending the information message towards the root; in addition, the intermediate nodes should do additional tasks forwarding the information messages.

The proposed isolation technique consumed a considerable amount of power represented by 1.268 mW, with 0.012 mW more energy than the detection technique and 0.028 mW more energy than SBIDS. The additional energy consumption of the isolation technique is the consequence of extra exchanged alert messages that hold the attacker node address. However, both detection and isolation techniques can be considered lightweight since they cause only 2.37 % and 3.68 % additional energy consumption, respectively, and they consume much less than SecTrust and MLTKNN. Nevertheless, the proposed approach, including the detection and isolation techniques, remains very competitive despite the little additional power consumption that is well-deserved given the achieved efficiency in terms of detection accuracy (Figure 6), detection delay (Table 2) and PDR (Figure 8).

5.3 Evaluation of version number attack detection mechanism

In this section, the proposed approach against a version number attack is assessed and compared to similar approaches in terms of different metrics, such as detection accuracy, PDR, control message overheads and power consumption. The following assumptions are taken into consideration during the tests:

- The attack can be performed by any node in the network except the root.
- The attack is performed only after the network's convergence (completed topology).
- The root performs legitimate version number updates during the tests.

5.3.1 Detection accuracy

The detection accuracy of the proposed approach was assessed and compared with similar approaches. Detection accuracy represents a crucial term for measuring the efficiency of any system distinguishing between legitimate and malicious version number updates.

Figure 9 depicts the false positive and false negative rates as well as the detection accuracy of the proposed approach compared to DCVM (Ahmed & Ko, 2016), DVIDS (Iouliau et al., 2018) and Shield (Ariş et al., 2019).

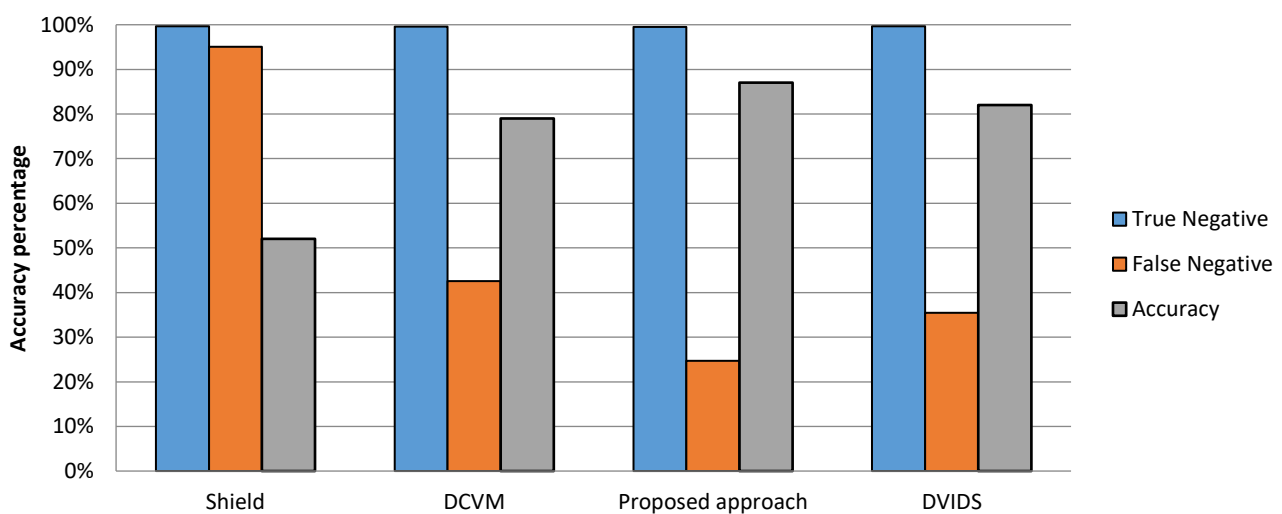


Figure 9. Detection accuracy of proposed approach compared to Shield, DVIDS and DCVM approaches.

In Figure 9, the proposed approach, DVIDS, DCVM and Shield achieved almost the same true negative rate around 100%, which means that illegitimate version number updates by malicious nodes are successfully detected. However, this is not the case for the false negative rate, where the proposed approach achieved the lowest rate (70.40%, 17.88% and 10.78% less than Shield, DCVM and DVIDS, respectively). This can be justified by the efficiency of the proposed approach differentiating between legitimate and illegitimate version number updates. In other words, the proposed approach allows more legitimate updates unlike DCVM, DVIDS and Shield, where several legitimate updates are ignored basically because they were designed focusing only on detecting illegitimate version number updates, without taking into consideration legitimate changes in the version number property, which is a very necessary process for RPL networks. The proposed approach achieved the best accuracy, outperforming DVIDS, DCVM and Shield by 5%, 8% and 35%, respectively; the superiority in the accuracy rate results from superiority in terms of the false negative rate.

5.3.2 Power consumption, control message overheads and PDR

In this experiment, we measured the PDR, power consumption and control message overheads using the UDP (client/server) files offered by Cooja. In the experiments, each client node sends 15 UDP packets to the root via the intermediate nodes during the simulation. Moreover, we used the Powertrace tool to record the power consumption during simulations.

Table 2 shows a comparison of the proposed approach with the Shield approach, standard RPL, attacked RPL and SRPL in terms of PDR, power consumption and control message overheads.

Table 2. Comparison of proposed approach with Shield approach, standard RPL, attacked RPL and SRPL in terms of PDR, power consumption and control message overheads.

	Standard RPL	Attacked RPL	SRPL	Shield	Proposed approach
PDR	92.12%	62.33%	84.04%	85.1%	87.3%
Power consumption [mW]	2.44	3.75	2.59	2.68	2.53
Control message overheads	1366	2959	1700	1776	1599

According to Table 2, the version number attack has a great impact on the network: the attack decreased the PDR by around 30%, increased the power consumption by around 54% and also increased the control message overheads dramatically from 1366 to almost 3000 exchanged DIO/DAO messages; however, the RPL performance improved notably with countermeasures, while the proposed approach outperformed the other approaches.

Applying the SRPL and Shield approaches increased the PDR by around 22% and 24%, respectively, whereas the proposed approach reached the highest value increasing the PDR up to 87.3%. This increase is justified by the technique used to avoid global repairs and their effects by ignoring illegitimate version number increments. The proposed approach achieved better PDR compared to the ameliorated Shield by around 3% due to the use of more neighbouring nodes to distinguish between legitimate and illegitimate version number changes, which in turn led to lower control message overheads and increased the PDR as explained above.

The increase in power consumption under attack can be explained by the extra exchanged control messages, which in turn resulted from the global repairs, compelling each node in the network to remove all neighbours from its table list and start a new discovery process. The proposed approach, SRPL and Shield approaches notably decreased the effects of the attack by 32.5%, 31% and 28%, respectively. This improvement is justified by the success in identifying and preventing illegitimate version number increments. The slight advantage of the proposed approach results from the effectiveness of the new technique used in computing the probability based on more neighbours.

The control message overheads were reduced by Shield from around 2959 messages down to 1776; this considerable improvement can be explained by the filtering technique, which allows only real version number changes to be made in the network. Similarly, SRPL decreased the control message overheads down to 1700 under attack for the same reason.

The proposed approach achieved 1599 messages, which represents the lowest control message overheads under attack with 7% and 10% less than SRPL and Shield, respectively. The superiority of the proposed approach is justified by the increased false negative rate of Shield and SRPL compared to the proposed approach. The incapability of detecting legitimate version number changes leads to more DIO messages sent by the root seeking for version number changes by the network nodes.

6 Conclusion

RPL-based networks are vulnerable to various types of attacks, including internal and external attacks. Rank and version number attacks are considered among the most dangerous attacks that target network consistency. Both attacks use RPL control messages and cause huge disruption to the network resources without any means provided by RPL to overcome both attacks. In this paper, we proposed two lightweight mechanisms to enhance the security of RPL-based networks against both attacks. The mechanism against rank attacks was designed and implemented into two techniques. The first technique allows only detecting the malicious nodes without any extra procedures, whereas the second one aims to isolate the malicious nodes by ordering to limit its damage and restore the network performance. The latter consumes little

more energy with much better results; the mechanism against version number attacks allows filtering version number changes and avoiding unnecessary version number increments caused by malicious nodes, which in turn causes huge overheads in exchanged control messages and power consumption.

The proposed approaches were evaluated via simulation experiments. The achieved results proved the efficiency in restoring optimal performance of RPL, outperforming similar approaches in terms of quantitative metrics, while keeping in mind that both approaches have well-deserved charges in power consumption.

As a future work, we intend to design and implement a system that would secure RPL-based networks against coordinated attacks which can target network nodes simultaneously, such as a version number/rank attack. We believe that coordinated attacks might cause catastrophic damage to the network, especially with the absence of any default protection line in RPL-based networks. Thus, designing a system against coordinated attacks is interesting considering the lack of such systems in the literature.

Additional Information and Declarations

Conflict of Interests: The authors declare no conflict of interest.


Author Contributions: A.E.K.: Conceptualization, Methodology, Software, Validation, Writing – Original draft preparation, Investigation. R.B.: Supervision, Writing – Reviewing and Editing, Project administration.

Data Availability: The data that support the findings of this study are available from the corresponding author.

References

- Ahmed, F., & Ko, Y. B. (2016). A distributed and cooperative verification mechanism to defend against DODAG version number attack in RPL. In *PECCS 2016 – Proceedings of the 6th International Joint Conference on Pervasive and Embedded Computing and Communication Systems*, (pp. 55–62). Scitepress. <https://doi.org/10.5220/0005930000550062>
- Airehrour, D., Gutierrez, J. A., & Ray, S. K. (2019). SecTrust-RPL: A secure trust-aware RPL routing protocol for Internet of Things. *Future Generation Computer Systems*, 93, 860–876. <https://doi.org/10.1016/j.future.2018.03.021>
- Alsukayti, I. S., & Singh, A. (2022). A Lightweight Scheme for Mitigating RPL Version Number Attacks in IoT Networks. *IEEE Access*, 10, 111115–111133. <https://doi.org/10.1109/ACCESS.2022.3215460>
- Alzubaidi, M., Anbar, M., Chong, Y. W., & Al-Sarawi, S. (2018). Hybrid monitoring technique for detecting abnormal behaviour in RPL-based network. *Journal of Communications*, 13(5), 198–208. <https://doi.org/10.12720/jcm.13.5.198-208>
- Ariş, A., & Oktug, S. F. (2020). Analysis of the RPL Version Number Attack with Multiple Attackers. In *2020 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)*. IEEE. <https://doi.org/10.1109/CyberSA49311.2020.9139695>
- Ariş, A., Oktug, S. F., & Örs Yalcın, S. B. (2016). RPL version number attacks: In-depth study. In *Proceedings of the NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, (pp. 776–779). IEEE. <https://doi.org/10.1109/NOMS.2016.7502897>
- Ariş, A., Örs Yalcın, S. B., & Oktug, S. F. (2019). New lightweight mitigation techniques for RPL version number attacks. *Ad Hoc Networks*, 85, 81–91. <https://doi.org/10.1016/j.adhoc.2018.10.022>
- Glissa, G., Rachedi, A., & Meddeb, A. (2016). A secure routing protocol based on RPL for internet of things. In *2016 IEEE Global Communications Conference*. IEEE. <https://doi.org/10.1109/GLOCOM.2016.7841543>
- Iouliauou, P., Vasilakis, V., Moscholios, I., & Logothetis, M. (2018). A Signature-based Intrusion Detection System for the Internet of Things. ePrint. White Rose Research Online. https://eprints.whiterose.ac.uk/133312/1/ictf_2018_loT.pdf
- Karmakar, S., Sengupta, J., & Bit, S.D. (2021). LEADER: Low Overhead Rank Attack Detection for Securing RPL based IoT. In *2021 International Conference on Communication Systems and Networks*, (pp. 429–437). IEEE. <https://doi.org/10.1109/COMSNETS51098.2021.9352937>
- Le, A., Loo, J., Lasebae, A., Vinel, A., Chen, Y., & Chai, M. (2013). The Impact of Rank Attack on Network Topology of Routing Protocol for Low-Power and Lossy Networks. *IEEE Sensors Journal*, 13(10), 3685–3692. <https://doi.org/10.1109/JSEN.2013.2266399>
- Neerugatti, V., & Reddy, A. R. M. (2019). Machine learning based technique for detection of rank attack in RPL based internet of things networks. *International Journal of Innovative Technology and Exploring Engineering*, 8(9 Special Issue 3), 244–248. <https://doi.org/10.35940/ijitee.I3044.0789S319>

- Nikravan, M., Movaghar, A., & Hosseinzadeh, M.** (2018). A Lightweight Defense Approach to Mitigate Version Number and Rank Attacks in Low-Power and Lossy Networks. *Wireless Personal Communications*, 99(2), 1035–1059. <https://doi.org/10.1007/s11277-017-5165-4>
- Osman, M., He, J., Mahioub, F., Mokbal, M., & Zhu, N.** (2021a). Artificial Neural Network Model for Decreased Rank Attack Detection in RPL Based on IoT Networks. *International Journal of Network Security*, 23(3), 496–503. [https://doi.org/10.6633/IJNS.202105_23\(3\).15](https://doi.org/10.6633/IJNS.202105_23(3).15)
- Osman, M., He, J., Mokbal, F. M. M., Zhu, N., & Qureshi, S.** (2021b). ML-LGBM: A Machine Learning Model Based on Light Gradient Boosting Machine for the Detection of Version Number Attacks in RPL-Based Networks. *IEEE Access*, 9, 83654–83665. <https://doi.org/10.1109/ACCESS.2021.3087175>
- Perazzo, P., Vallati, C., Arena, A., Anastasi, G., & Dini, G.** (2017). An Implementation and Evaluation of the Security Features of RPL. In *Ad-hoc, Mobile, and Wireless Networks*, (pp. 63–76). Springer. https://doi.org/10.1007/978-3-319-67910-5_6
- Pongle, P., & Chavan, G.** (2015). Real Time Intrusion and Wormhole Attack Detection in Internet of Things. *International Journal of Computer Applications*, 121(9), 1–9. <https://doi.org/10.5120/21565-4589>
- Raza, S., Wallgren, L., & Voigt, T.** (2013). SVELTE: Real-time intrusion detection in the Internet of Things. *Ad Hoc Networks*, 11(8), 2661–2674. <https://doi.org/10.1016/j.adhoc.2013.04.014>
- Rehman, A., Khan, M. M., Lodhi, M. A., & Hussain, F. B.** (2016). Rank attack using objective function in RPL for low power and lossy networks. In *2016 International Conference on Industrial Informatics and Computer Systems*, (pp. 1–5). IEEE. <https://doi.org/10.1109/ICCSII.2016.7462418>
- Sahay, R., Geethakumari, G., & Modugu, K.** (2018). Attack graph — Based vulnerability assessment of rank property in RPL-6LOWPAN in IoT. In *IEEE 4th World Forum on Internet of Things*, (pp. 313–318). IEEE. <https://doi.org/10.1109/WF-IoT.2018.8355171>
- Shafique, U., Khan, A., Rehman, A., Bashir, F., & Alam, M.** (2018). Detection of rank attack in routing protocol for Low Power and Lossy Networks. *Annals of Telecommunications*, 73(7–8), 429–438. <https://doi.org/10.1007/s12243-018-0645-4>
- Zaminkar, M., & Fotuhi, R.** (2020). SoS-RPL: Securing Internet of Things Against Sinkhole Attack Using RPL Protocol-Based Node Rating and Ranking Mechanism. *Wireless Personal Communications*, 114(2), 1287–1312. <https://doi.org/10.1007/s11277-020-07421-z>

Editorial record: The article has been peer-reviewed. First submission received on 11 February 2024. Revisions received on 1 April 2024 and 29 April 2024. Accepted for publication on 3 May 2024. The editor in charge of coordinating the peer-review of this manuscript and approving it for publication was Zdenek Smutny .

Acta Informatica Pragensia is published by Prague University of Economics and Business, Czech Republic.

ISSN: 1805-4951
