



## **A new lightweight decentralized mitigation solution against Version Number Attacks for IoT Networks**


### **Mehdi Rouissat**

(Univeristy Center Nour Bachir, 32000, El-Bayadh, Algeria  
STIC Laboratory, University Aboubekr Belkaid, Tlemcen, Algeria  
 <https://orcid.org/0000-0002-4444-2754>, mehdi.m.rouissat@gmail.com)


### **Mohammed Belkheir**

(LIMA Laboratory, Univeristy Center Nour Bachir, El-Bayadh, Algeria  
 <https://orcid.org/0000-0003-1258-3656>, belkheir\_m@yahoo.fr )

### **Hicham Sid Ahmed Belkhira**

(LTTNS Labortory, Djillali Liabes University, Sidi Bel Abbes, Algeria  
 <https://orcid.org/0000-0002-8125-789X>, belkhira.hichem@gmail.com)


### **Sofiane Boukli Hacene**

(EEDIS Laboratory, Djillali Liabes University, Sidi Bel Abbes, Algeria  
 <https://orcid.org/0000-0002-9760-3806>, boukli@gmail.com)

### **Pascal Lorenz**

(MIPS Laboratory, University of Haute-Alsace, Mulhouse and Colmar, France)

### **Merahi Bouziani**

(LTTNS Labortory Djillali Liabes University, Sidi Bel Abbes, Algeria  
 <https://orcid.org/0000-0002-9030-4395>, bouzi\_mera@hotmail.com)

**Abstract:** The present work describes a new technique to mitigate the version number attack (VNA), which is classified as one among the known denial of service (DDoS) damaging attacks targeting RPL-based (Routing Protocol for Low Power and Lossy Networks) IoTs networks. Through a VNA, the malicious behavior induces an increase in the control overhead and affects nodes' ressources in terms of processing and memory, thereby the network availability is directly targeted. The lightweight proposed algorithm is run by each node where the main purpose is to halt the spread of a faked version number over the network and to recover victim nodes. The proposed solution has been implemented and simulated using Cooja under Contiki OS. Simulation results obviously show that our proposed technique promises significant improvements in various measured metrics while optimizing the node resources in terms of processing and memory usage. Compared to the network under attack, the control overhead has been shortened by 83% and the energy consumption has been reduced by 74%. In addition, the packet delivery ratio (PDR) has been improved to reach (99,6%), and the latency has been restored to attain the same value as in the normal case.

**Keywords:** RPL, IoT, security, version number, countermeasure

**Categories:** L.4.0

DOI: 10.3897/jucs.85506

## 1 Introduction

In the recent years, smart technologies have overwhelmed our lives, since they are involved in almost every aspect of our day-to-day [Wang et al., 2021]. This has been accomplished by the emergence and the maturation of a new trend of networks denoted IoTs (Internet of Things) [Preet et al, 2020]. The latter are mainly built with small smart wirelessly connected devices called sensors, interacting together in order to fold data from the monitored environment and transfer it to the internet for further processing and decisions [Hajjaji et al, 2021, Javaid and Khan, 2021]. The numerous advantages offered by IoT networks in terms of the easy connectivity, scalability and cost effectiveness allow them to be ubiquitous within various emergent ecosystems such as: healthcare systems, wearable devices, transportation and logistics, military, education, and along with many other fields of application [Selvaraj and Sundaravaradhan, 2020, Kanoun and Derbel, 2021, Chung, 2021, Tun et al., 2021]. This outstanding growth of the IoT networks opens up a gap to several vulnerabilities and security challenges that constitute an evolved research topic [Shresta and Furkan, 2020, Obaidat et al., 2020]. The security challenge regarding IoT networks is essentially inherited from the limited characteristics of sensors and other IoT devices compared to smartphones or other mobile devices [Imteaj et al., 2021]. Their memory capacity size is about 10 of Kbytes, and the processing capability is very limited to run heavy cryptographic solutions [Alfa et al., 2021]. Moreover, IoTs are evolved to a new paradigm baptized M-IoT (Internet of Mobile Things) that introduces a novel era of cloud and fog computing. This overwhelming trend leads to a variety of other challenges concerning the security components and makes the IoT networks vulnerable [Lee and Lee, 2021, Meng et al., 2021, Sharma et al., 2020]. Statistics witness that the cyber security posture is on the rise, where threats are daily targeting the IT networks especially IOTs [Pour et al., 2021]. To this end, various researches and studies have been carried out to embrace the security and privacy tendencies of IoT networks and to ensure reliable and efficient networking scheme [Snehi and Bhandari, 2021, Abiodun et al., 2021, Aversano et al., 2021].

Since the core routing operation is considered as the brain functionality of the networking domain, our work is focused on the network layer of the IoT networks, especially the security concern. Various protocols are implemented by the IoT networks depending on the operational layer (application, network and perception layer) [Khan and Salah, 2018, Salman and Jain, 2019], the well-known protocols in the perception layer are IPV6, 6LowPAN. [Boudouaia et al., 2020, Lamkimel et al., 2018, Verma and Ranga, 2020, Kharrufa et al., 2019]. Besides, the network layer implements one of the most known routing protocols designed for IoT networks named RPL (Routing Protocol for Low Power and Lossy Networks) [Boudouaia et al., 2021, Butun et al., 2020]. The key feature of RPL is to cope with the limited inherited resources of smart devices in terms of processing, memory usage and power consumption. Although RPL has been developed to offer a suitable flexibility, reliability and energy efficiency for constrained networks, it suffers from various threats and security issues that have been widely addressed [Simoglou et al., 2021, Pishdar et al., 2021, Farris et al., 2019]. Despite a panoply of solutions that have been proposed to secure IoT networks against

the known routing threats, they are commonly based on cryptographic or cooperative algorithms generating thereby an extra control overhead and leading to nodes resources exhausting [Mrabet et al., 2020].

Our present work aims to propose a decentralized solution to mitigate one of the well-known and destructive security threat against RPL named version number attack (VNA), whereby a malicious node archly disseminates a falsified version number over the network and continually provokes inconsistencies that increase the control overhead targeting to worsen the overall network routing efficiency and lifetime [Krishna et al., 2021].

The main attracting features offered by our proposed solution are:

- The solution includes an original and lightweight algorithm to mitigate VNA, in order to cope with the limited resources of nodes, less energy consumption, less memory footprint.
- The proposed algorithm is executed by each node. It allows detecting the falsified received version number based on other values of its neighborhood. In addition, the proposed algorithm allows recovering victim nodes from their malicious ancestor node, which is a valuable added value and a very interesting feature.
- Before starting the legitimacy verification, the solution reduces the number of potential victim nodes,
- The solution does not add any new control messages to the RPL standard. In addition, the core networking scheme is kept unchanged.
- The solution shows a similar latency compared to the attack free case, and a very slight increase in energy consumption, which is much better compared to existing solutions,
- All possible positions of the intruder within the topology are studied to analyze the accuracy of the solution,

In order to highlight the insight gained by our new solution, in what follows we firstly describe the core functioning of RPL protocol and its philosophy based on building a direct acyclic graph that organizes the routing scheme within an IoT network. Afterwards, we give some detail about the security challenge that RPL encounters and how a malicious node can perform a VNA and its harmful effect targeting the network resources and topology. The section 5 emphasizes some proposed solutions that address the VNA. In section 6 we detail the performance evaluation of our proposed scheme within different scenarios of a RPL-based network, in a normal case, in case of attack and after involving our new process scheme. At the end of the paper, we give some future outlooks to enhance our proposed method.

## 2 Rpl Routing Protocol Overview

IETF has defined RPL as the main routing protocol for Low power and Lossy networks in the RFC 6550 [Winter et al., 2012]. RPL is based on the IPV6 protocol, where the topology adopted is mesh/tree, forming an acyclic graph named DODAG (Destination Oriented Directed Acyclic Graphs). The latter is built from the root which represents the data sink towards other nodes based on their ranking position. To ensure the core routing operation and managing the DODAG, RPL defines four control messages: DIS, DIO, DAO and DAO-ACK messages [Min et al., 2020], as depicted by figure 1.

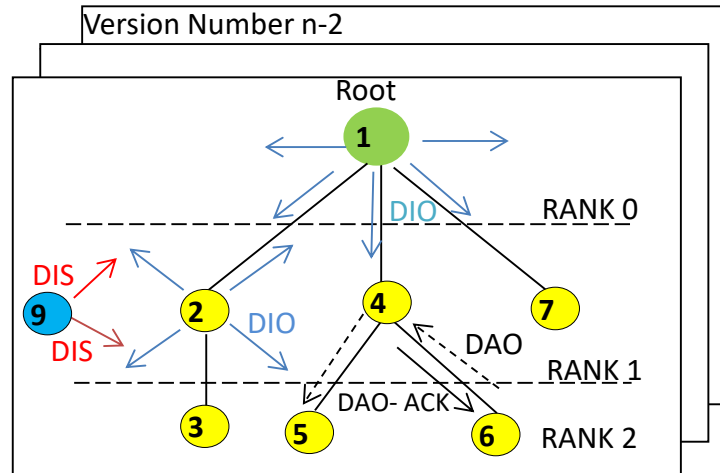


Figure 1: RPL Topology And Control Messages

Every node wishing to join the network should request a set of repositories from other neighboring nodes or the root itself, by sending a DIS message (DODAG Information Solicitation). The DIO message (DODAG Information Object) represents a reply to a DIS message, where it contains a set of topology parameters as: DODAG ID, Version Number, Rank value and MOP (Mode of Operation), these parameters are the responsibility of the root, and allow each node to join the DODAG. DIO messages are triggered based on a “trickle timer” where its value is modified upon a topology changing ensues to update DODAG parameters. If the topology remains stable, the trickle timer interval is increased in order to reduce the flooding of DIO messages and thus optimizing nodes resources. DIO messages participate in the construction of the upward routes from the nodes toward the root, whereas, for building downward routes RPL implements DAO messages (Destination Advertised Object) to share other suitable information like parent-children relationship and other necessary parameters to propagate destination information upward the DODAG.

As depicted by figure 1, the ranking position of each node estimates how long distance between the node and the root. It is calculated using the following equation:

$$Rank_{node} = MHRI * (1 + floor(Rank_{parent} / MHRI)) \quad (1)$$

Where:

- Floor(x): calculates the greatest integer less than or equal to x.

The “MHRI” is defined as “the Minimum Hop Rank Increase” and describes the link cost between the node and its best parent. “MHRI” is calculated using an objective function (OF) implemented by RPL and defines how nodes can choose the best routing metrics for selecting the optimized paths toward the root. The objective function is

defined by IETF in RFC 6551 [Lamaazi and Benamar, 2020]. “MHRI” increases while moving away from the root.

Other important parameter called “version number” that depicts a screenshot of the actual DODAG topology. This parameter remains the same for overall the network and it is updated only by the root. The root may increment the actual version number in order to refresh the network topology or to avoid an eventual inconsistency that occurs.

In addition, RPL implements a “trickle timer” to manage DIO messages being sent [Lamaazi and Benamar, 2020, Violettas et al., 2019]. The trickle timer is the responsibility of the root and it is calculated as follows:

$$I_{doub} = \log_2 \frac{I_{max}}{I_{min}} \quad (2)$$

Where :

- $I_{max}$ : defines the maximum interval of a DIO message
- $I_{min}$ : defines the minimum interval of a DIO message
- $I_{doub}$ : defines the doubling parameter between  $I_{min}$  and  $I_{max}$

The RFC 6550 defines the values of  $I_{min} = 8 \text{ ms}$ , and  $k = 20$ , whereby:

$$I_{max} = I_{min} * 2^k \approx 2,3 \text{ hours} \quad (3)$$

In the RPL Contiki implementation,  $I_{min}$  and  $I_{max}$  are given by 4 s and 17.5 minutes, respectively. These configuration parameters are broadcast to all the nodes via DIO messages. This means, a greater trickle timer value is, a great interval is between the transmission of two respective DIOs and less control messages are generated. Thus, the trickle timer estimates the network convergence time and measure the stability level of a RPL topology.

### 3 Version Number Attack and Inconsistencies in RPL

The tuple (RPL Instance ID, DODAGID, DODAG Version Number, Rank) are the main parameters to establish and maintain the topology of a RPL network. However, security breaches may occur where altered values of rank and version number are disseminated by malicious nodes attempting to destabilize the network topology and resources. These hateful behaviors lead to inconsistencies and institute various kinds of threats and loops that should be addressed to ensure a high level of data security and quality of service. Therefore, a wide range of recent researches have classified the attacks against RPL into three main categories regarding their conducts and their effects (topology, resources, traffic) [Agiollo et al., 2021, Khraisat and Alazab, 2021, Simoglou et al., 2021]. In addition, a malicious node may conduct a hybrid-attack by simultaneously carrying out multiple counterfeit actions. This intends to double the damaging effect by the malicious node and makes very difficult the detection and the mitigation processes [Almusaylim et al., 2020, Mbarek et al., 2021].

To overcome inconsistencies and loops, RPL implements a repair mechanism in order to constantly keep the same DODAG parameters for all the nodes and thereby maintain the topology [Niu, 2021]. For a particular reason, a node wishing to reset its repositories or to improve some metrics may locally trigger a “local repair” mechanism.

In such a way, it should poison its routes by advertising an “INFINITE\_RANK” and thus informs its neighborhood by these changes via DIO messages. In addition, a root observing an irregular set of parameters that are exchanging between nodes, initiates a mechanism named “global repair” to disseminate a new incremented version number via DIO messages, thus aiming to revalidate the network integrity. Therefore, a repetitive occurrence of these processes yields to an extra control overhead that affect nodes resources and the topology strength. As a consequence, a malicious node may take advantage of this gap for disrupting the network extrinsic and intrinsic resources and thus reducing its lifetime, since RPL has no mechanism to prevent such behaviors [Bhatt and Ragiri, 2021].

The version number attack (VNA) is considered as DDoS (Distributed Denial of Service) aiming to exhaust nodes resources [Aris et al., 2016]. A malicious node performing such attack advertises a counterfeit version number value to its neighboring nodes through DIO messages. Upon receiving a new version number, each node performs the following actions:

- Poisons its routes by advertising an “INFINITE\_RANK”
- Leaves its preferred parent
- Chooses a new parent
- Recalculates its rank value
- Resets its trickle timer
- Sends a DIO message with the refresh parameters

In addition, when the root receives the illegitimate version number, it triggers a “global repair” while incrementing the current version number and broadcasts the new topology parameters in the DODAG.

The malicious node via this harmful operation leads to overstress the network, by forcing targeted nodes to instantaneously generate extra DIO messages in order to flood the present version number value and their new rank values, DAO messages are also sent to rebuild the downward routes. Besides the increase of the control overhead, an overload is handled by nodes to process these generated messages, forward messages, and also to calculate their updated parameters. Afterwards, the resources of the nodes will be exhausted, and thereby the network lifetime is targeted.

This recurrent upsetting behavior creates a harmful volatility problem in the network [Aris et al., 2020] and leads to affect the routing process by creating loops, making the established routes unavailable and forcing the unnecessary rebuilt of the DODAG.

## 4 Related Work

Recently, the security aspect has become a very interesting research topic, where various studies have focused on developing algorithms to enhance the RPL core functioning against the aforesaid threats. Regarding the VNA, a crowd of recent works have proposed different solutions to detect and mitigate this attack against RPL networks. These solutions are based on a cooperative scheme or implement cryptographic and trust based algorithms. Likewise, other solutions take advantage of

the cloud or the fog computing to perform the VNA detection. By the following, we relate some of the proposed secure methods addressing the VNA.

Authors in [Almusaylim et al., 2020] proposed a modified RPL version, named “SRPL-RP” for secure RPL, where a timestamp and a threshold are included to detect the legitimacy of a DIO sender. Each node is identified by an ID and the root node ID is encrypted to avoid its imitation. In addition, a monitoring table is established in conjunction with the DODAG building process, contains nodes parameters (ID, Rank) that help the receiving node to detect further changing of nodes behaviors. A blacklist table includes nodes that have been detected as malicious in order to be isolated. Upon considering a node as malicious an alert message should be broadcasted to inform other nodes to add this node in their blacklist. The proposed method adds multiple complicated operations that need an extra processing and memory usage of the node. In addition, more control messages are broadcasted over the network to alert nodes about the detected suspicious behavior. Other work is proposed in [Ahmed and Ko, 2016], where authors implement a cooperative technique upon receiving a DIO message with a new VN. The node performing the verification stores the new received VN as well as the parameters of the sender node, in a table. In the verification process, a node sends a request message to the two hops neighbors (alternate parent, sibling) to get their actual VN. Thereby, the node can detect if the advertised VN is legitimate or falsified. This proposed solution needs also generating an extra control overhead by adding request/Response messages to the 2-hops neighboring nodes, which may cause compatibility and interoperability issues with the default RPL. Furthermore, authors didn’t give details about how the performing node can gather information from its 2-hops neighbors. The proposed solution in [Mayzaud et al., 2017] consists on a distributed monitoring scheme mainly composed by special nodes with high capabilities that play the role of monitors. They are intended to analyze the traffic and to store the necessary information that allows detecting any malicious behavior of nodes. This solution needs to add other dedicated nodes to monitor the network, where their number continually increases according to the monitored area. The authors didn’t show details on how we can consider a VN as falsified and how the malicious node is detected. In [Nikravan et al., 2018], authors proposed a secure scheme based on IBOOS (Identity Based Offline-Online signature) where nodes are identified via IDs. The root node should sign the incremented VN via an algorithm denoted OnSign. Nodes are able to run Unsign algorithm to read the new VN. Nodes should verify the signature before processing the incoming DIO, if the signature matches so the VN is considered as legitimate; otherwise, the receiving node discards the DIO message and considers it as falsified. The work has not been tested regarding the nodes resources, since IoT nodes are constrained devices and heavy algorithms lead to exhaust their resources.

In [Sahay et al., 2019], authors implement a framework for the VNA detection based on the cloud computing, where the network state is constantly captured through PCAP files, then sent to a cloud service to detect if any VNA occurs. Thereafter, a list of malicious nodes is sent by the cloud to an EDGE router for further use. This solution needs a remote computing process held by the cloud and added devices such as the EDGE router. Moreover, authors didn’t give details on how the cloud service can detect the malicious VN node.

Proposed solution	Implementation Scheme	Detection and mitigation strategy	ROM	RAM	Accuracy	Delay	Energy	PDR	Control Overhead
[Almusaylim et al., 2020]	Decentralized	Digital Signature and alert messaging	N/A	N/A	98,17%	N/A	1231,75	98,48	991 Packets/s
[Ahmed and Ko, 2016]	Decentralized	2-neighboring verification	N/A	N/A	N/A	N/A	N/A	99,99	3000
[Mayzaud et al., 2017]	Centralized	Monitoring nodes	N/A	N/A	N/A	N/A	N/A	N/A	N/A
[Nikravan et al., 2018]	Centralized	Identity based signature	N/A	N/A	N/A	N/A	N/A	N/A	N/A
[Sahay et al., 2019]	Centralized	cloud service implementation	N/A	N/A	98%	N/A	N/A	N/A	N/A
[Arosu et al., 2019]	Decentralized	Neighboring VN observation	95484	7962	N/A	87%	63%	86%	71%
[Anitha and Arockiam, 2021]	Decentralized	Neighboring VN observation	N/A	N/A	94,40%	N/A	N/A	N/A	7000
Proposed solution	Decentralized	Comparison to reliable nodes	46.9	5.34	100%	0.39	0.06%	99.6 %	34.4%

Table 1: VNA related works and performance results



In [Arosu et al., 2019], authors presented two complementary methods to mitigate the VNA effect. The first method consists on eliminating any updated VN coming from a leaf node, based on their observations; the farthest the attacker from the root, more damaging the VNA is. The second work held by authors in [Arosu et al., 2019] is named “Shield” where the idea is to not trust any single VN sent by a best ranked neighbor. Thus, the performing node should construct a table which contains the best ranking neighbors IDs and their VN. The VN is updated if the majority of nodes in the table advertise the same updated value. In [Anitha and Arockiam, 2021], authors proposed a similar solution as [Arosu et al., 2019] named VeNaDet, based on the comparison of the received VN with values of the neighbors having a best rank than the node performing the process. If the majority of nodes have the same VN then the latter is considered as legitimate. Otherwise it is considered as a falsified value sent by a malicious node. The two previous solutions are based on a observing the neighborhood changing. However, if a node has a good rank it easily falsifies its whole SUB-DODAG. On another hand, a performing node may have the common neighbors as siblings sharing the same parent, thus the performed table will be filled by mistaken VN that may falsify the made decision.

Table 1 summarizes the results gained by each aforesaid proposed solution. It shows that even though the tested solutions achieve better insights in terms of routing metrics, only authors in [Anitha et al., 2021] have depicted the efficiency of their solution regarding the intrinsic resources of nodes. However, this measure is crucial in order to demonstrate the effectiveness of the proposed technique in terms of resource preservation.

Compared to the aforesaid solutions, our proposed solution offers,

- Decentralized treatment, no need to exchange specific data among nodes, no remote computing process held by the cloud, that requires heavy processing and memory usage,
- No extra control overhead is generated, no additional specific request/response messages are needed,
- No encryption is needed,
- No need to add other dedicated nodes to monitor the network,
- Lightweight algorithm, does not exhaust the resources of the nodes
- No needs to add devices such as EDGE router are monitoring devices

## 5 The proposed Mitigation solution

The aim of our proposed algorithm is to reduce the effect of the version number attack, through two steps. Starting by detecting the attack and stopping the spreading of the wrong version number in the network, the second step is to recover the victim nodes. The attractive features of the solution lies in the fact that there is no need to send specific data among nodes, every node executes the algorithm at its own level, to mitigate centralized control and possible falsified warnings.

The idea behind the proposed algorithm is to consider as new VN only values advertised by the preferred parent of the node, and check the legitimacy of the received VN from other sources, through nodes that do not have the same preferred parent (not siblings) as the node in question, and are also not its children. Thus, as long as a node

has a source "reliable node" to verify the legitimacy of the advertised VN by its parent, then the attack will be detected.

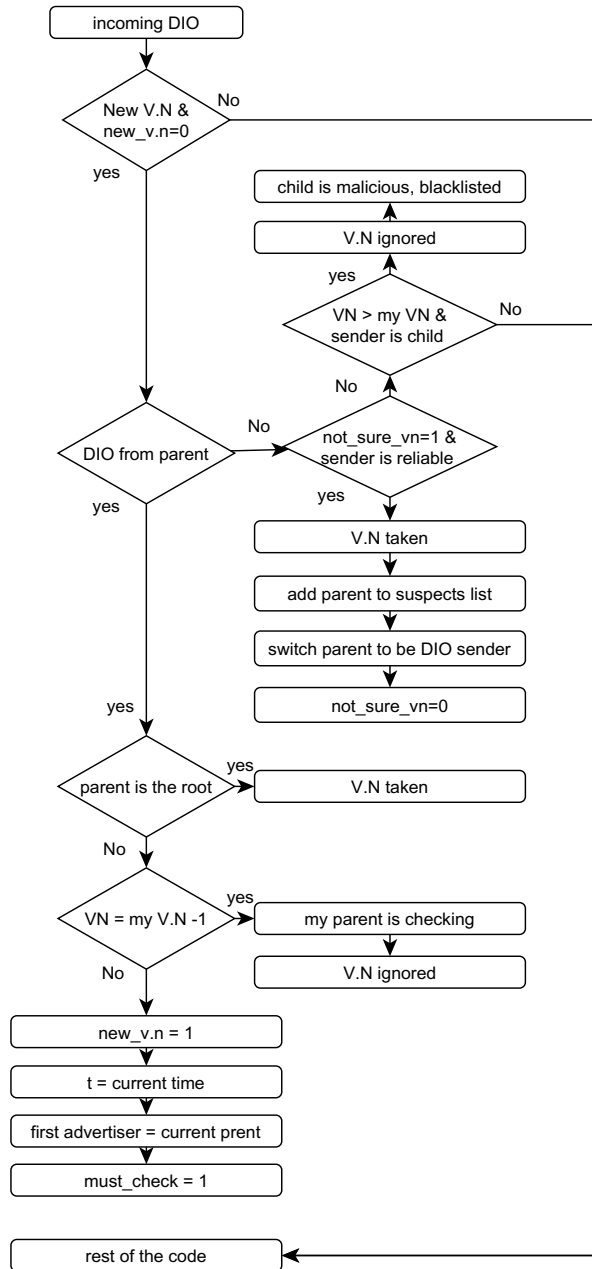


Figure 2: Flow diagram of the first part of the detection and stopping attack algorithm

The proposed strategy reduces the number of the possible victim nodes, where only the direct children of the malicious node are possible victims, which make checking the legitimacy of a node faster and accurate.

Following, we define the terminologies suspicious node, malicious node, reliable node and In doubt node :

- *Suspicious Node*: A child node judges its preferred parent to be suspicious if it advertises a new version number, but it appears to be illegitimate after verification,
- *Malicious node* : A parent node judges its child node to be malicious if it advertises a new incremented version number,
- *Reliable Node* : A node is said to be reliable if it is not the child, the parent nor a sibling (does not have the same parent) to a given node.
- *In doubt node* : is a node that has adopted a new version number but it had no second choice to check its legitimacy, it has the value (`not_sure_vn = 1`),

The figure 2 presents a flow diagram of the first part of the detection algorithm; the figure depicts how a node deals with a new received version number.

This part of the first phase is explained in the following points:

- A New version number event is treated only if the node is not checking an already received new version number (`new_vn = 0`),

Our solution does not stop at detecting the malicious behavior and stopping its spread, but it aims also to recover the victim nodes. If the sender is a reliable source, and the node is not sure about its current version number (`not_sure_vn = 1`), then the node will take and consider the coming VN, add its current preferred parent to the suspects list and switch it to be the DIO sender, otherwise the coming VN is ignored. This part presents the recovering step.

- If the preferred parent is the Root, the new VN will be taken. If the parent is not the root, then we have two possibilities :
  - If the VN is equal to the node's VN minus 1, that means, the parent is in a checking phase, the new coming VN is ignored.
  - Otherwise, the legitimacy of the new coming VN must be checked. The time of the event and the advisor node are temporary stored. The values of "`must_check`" and "`new_vn`" are set equal to 1.

Figure 3 presents a flow diagram of the second part of the detection algorithm. Based on the figure, once the advertised version number is potential and must be verified, the checking node advertises a value of VN equal to its own minus one, during 30 seconds, for the following goals:

- Avoid to be chosen as best parent, by advertising a lower VN value.
- Push neighbor nodes to send DIOs, allowing the node to have other sources to verify (sharing a lower VN provokes neighbors to resets their trickle timers and send DIOs),

- If a reliable node is also checking, the node can figure out that its own parent is not lying, since its neighbor is already checking a new VN coming from other source,

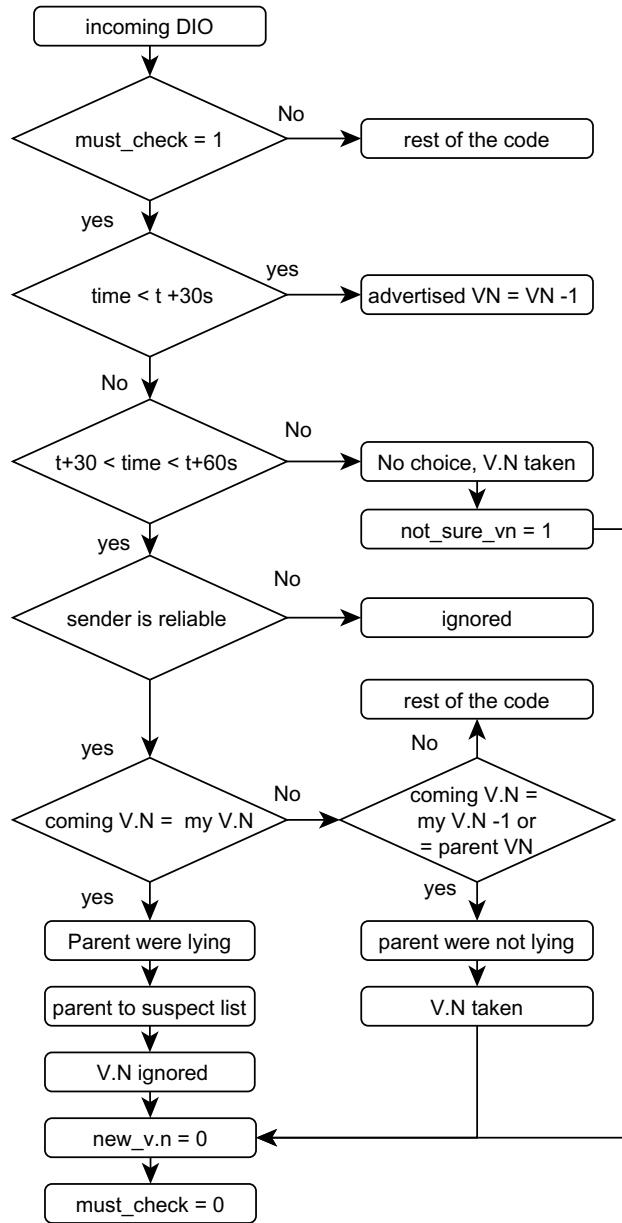


Figure 3: Flow diagram of the second part of the detection and stopping attack algorithm

During this 30 seconds, all the received VNs are ignored, in order to give time to a potential legitimate VN value to be spread and get to the neighbors, avoiding hurry decisions.

- After the 30 seconds, and during another 30 seconds, if the node receives a VN equal to its own value from a reliable source, that means its own parent were lying. Therefore the node nullifies its parent and adds it to the suspects list.
- If the received value is equal to its own minus one, that means the neighbor is also checking its source. Thus, the node can be certain about the credibility of its preferred parent. On the other hand, if the received VN is equal to its own preferred parent's, that also means its parent were not lying, this is the case where the neighbor has already taken the new version number.
- After 60 seconds (30 +30), if the node did not receive any DIO from other reliable source, then it has no choice but to take its parent version number, while the value "not\_sure\_vn" is set to 1.

Usually, in a dense topology a malicious node takes place in the middle of the network in order to have as many victim nodes as possible. Using our proposed solution, a potential victim node is having many reliable neighbor nodes in its neighboring to check the legitimacy of the received VN. Thus, the recovery step is rare in a dense targeted topology.

## 6 Performance Evaluation And Discussion

In this section, we show and explain the destructive effect of the version number attack on the performance of a network. Afterward, we demonstrate how our proposed mechanism can mitigate the version number attack and reduce its effects. The performance evaluation is based on the following indicators and metrics:

- Control packet overhead,
- Consumed Energy,
- Packet Delivery Ratio (PDR),
- Latency,
- Memory footprint.

We make use of RPL protocol implementation in the well-known lightweight open source Contiki operating system. It is a publicly available OS designed for IoT networks, running within an Ubuntu Linux virtual machine, it runs on highly optimized and tested networking stack that includes popular IoT standards like CoAP (application layer), UDP (transport layer), RPL (network layer), 6LoWPAN (adaptation layer), and IEEE 802.15.4 (physical and MAC layer) [Dunkels et al., 2021, Hashemi and Shams Aliee, 2019]. This OS incorporates well-tested standard fundamental mechanisms of RPL protocol which are compatible with different hardware platforms [Fotouhi et al., 2017]. It is used to study the effect of the version number attack on RPL-based IoT network, and to evaluate the performance and the efficiency of our proposed defense mechanism to mitigate such attack.

## 6.1 Simulation environment

To simulate the different scenarios related to our research work, we used Contiki's network simulator, i.e. Cooja, which is a Java based application with a graphical user interface (GUI based on Java's standard Swing toolkit) [Roussel and Zendra., 2016], that allows to simulate different network topologies. Our research implementation is based on Z1 nodes, which are a MSP430 ultralow-power based boards that has a radio IEEE 802.15.4 (for link layer protocol) compliant CC2420 radio transceiver operating at 2.4 GHz [Mathur et al., 2016]. We employed this type of mote for all the nodes, including the malicious node. Table 2 summarizes the main features of the used Z1 nodes.

Based on table 2, the powertrace tool, natively implemented in Contiki, is used to obtain data related to energy consumption in CPU processing, LPM mode, transmission and reception modes. The formula below implemented in a Perl script allows calculating the consumed energy by a given node for a given mode in mJoules:

$$Consumed\_energy = \frac{Energest\_value * current * voltage}{Rtimer\_second} \quad (4)$$

Where :

$Energest\_value$  is the number of ticks during each energy mode,

Parameter	Value
CPU idle current	0.426 mA
Current consumption in TX mode	17.4 mA
Current consumption in RX mode	18.8 mA
CPU power down current	0.020 mA
Maximum RAM size	8 kB
Maximum ROM size	92 kB
RTIMER_SECOND	32768 ticks /s

Table 2: Hardware specification of Z1 node [Zolertia, 2021]

For analyzing and studying the different behaviors of the network during the attack and after implementing the proposed algorithm, we consider a network topology with 12 motionless nodes, as shown in figure 4, where we have positioned the malicious node near to the DAG root. The remaining nodes are positioned in a specific way in the topology in order to cover every possible case. Table 3 shows the different used simulation parameters.

The goal behind choosing the studied particular graph structure is mainly to cover three special cases (worst possible cases):

- Where a node has no source to check the legitimacy of the received V.N (node 7. Fig.4), but it can be recovered by its child that has a second source to check the legitimacy of the received V.N.

- Where a node has only one source to check the legitimacy of the received V.N (node 8 Fig.4),
- Where the node has no children and no source to check the legitimacy of the received V.N (node 12 fig.8).

Studying such different possibilities allows to show the efficiency of the proposed solution in detection and isolating the malicious node, as its shown in the next sections. On the other hand, a random topology may not cover all possible cases studied in the chosen topology.

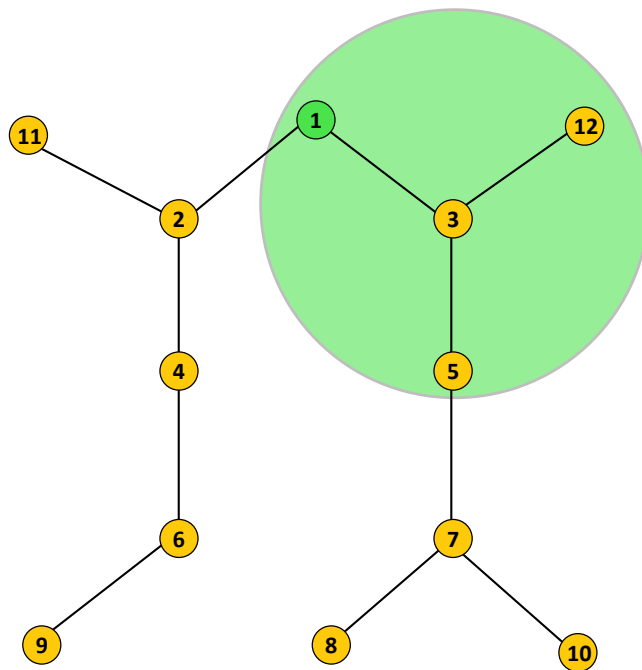


Figure 4: Network topology in the reference network

Initially, we run our simulations in a reference network, where all the nodes are considered static and no malicious nodes are present in the topology. It is used as a baseline to evaluate and compare the different behaviors of the network, under attack and after implementing the proposed mechanism. Each conducted simulation lasts for a lifetime of 15 minutes.

## 6.2 Impact of Version Number attack

In version number attack, a malicious node advertises a false incremented version number in its DIO messages, frequently and without the prior knowledge and authorization of the root, to force its neighbors to conduct a global repair and

reconstruct the topology [Rouissat et al., 2022- Verma and Ranga, 2020]. Depending on the position of the malicious node in the topology, this process goes on, the root node also gets the falsified DIO message with a higher version number than its own. The root considers a modified version number in the network as an inconsistency that requires a global repair, which leads to restarting the topology construction process for a second time by the root node.

To implement this attack in our simulations, we modified the RPL protocol stack of the Contiki OS on the malicious node. More precisely, we modified the "rpl-icmp6.c" file, where the malicious node advertises an incremented version number for each outgoing DIO message:

$$dio.version = dag->version ++ \quad (5)$$

Parameter	Values
Network layer Protocol	RPL
Operating System	Contiki 3.0.
Simulator	Contiki Cooja
Emulated nodes	Z1
MAC layer Protocol	IEEE 802.15.4
Radio model	UDGM
Simulation area	200 m × 200 m
Simulation time	15 minutes
Data transmission	1 Packet / 20s
Objective function	MRHOF
Malicious nodes	1
Legitimate nodes	11
TX range	50 m
Interference range	100 m

Table 3: Simulation Parameters

Similar to other nodes, the malicious node joins the network and actively participates in the creation and maintenance of the DODAG. The main difference between the malicious node and the others is that it runs a tempered code.

### 6.2.1 Impact of the attack on the control overhead

The overhead is evaluated in terms of DIO and DAO messages. A sent DAO message can be either generated by a given node, or generated by n-hops child and being forwarded to parent. On the other hand, this generated DAO message can either be a regular DAO to build upward routes, or a No-Path DAO to advertise the non existence of a route.



	Sent messages				
	Generated			Forwarded	
	Regular DIO	Regular DAO	No-Path DAO	DAO	Total
Reference	170	98	0	156	424
Under Attack	1357	242	592	1187	3378

Table 4: Control overhead in the reference network and after the VNA

The overhead in terms of DIO and DAO messages exchanged in the network during 15 minutes of simulation, in the reference network and in the case of the attack is depicted in table 4. The obtained results demonstrate that the VN attack significantly increased the DIO overhead from 170 to 1357 messages, because of the frequent trickle timer resets that led to more frequent DIO transmissions. The control packet overhead is also increased in terms of regular DAO messages from 98 to 242. The No-Path DAO, which is a decisive indicator of the topology stability jumped from zero to 592. This prominent increase reflects the behavior of the nodes when multiple global repairs are launched, where each node starts a local repair at its level. When a node launches a global repair it does the following:

- Removes all parents (neighbors),
- Removes the preferred parent,
- Removes the default route,
- Sends a No-Path DAO to the removed preferred parent,
- sets preferred parent NULL
- Resets the MRHOF.

Beside the generation of the control overhead, the attack added a serious forwarding task to the nodes, where the forwarded DAO messages shows an immense increase of 660 %, from 165 to 1187 messages.

The nodes frequently reset their trickle timers in the three cases:

- After receiving a new version number (global repair),
- After receiving an old version number,
- Detecting a loop.

Factor	Number of resets
Old version number received	372
Participating in the global repair	545
Loops	7

Table 5: Factors of timer resets

Based on the obtained results, table 5 shows that 372 times an old version number had been detected by the different nodes, while 545 global repairs is launched and 7 loops had been detected, giving a total of 924 trickle timer resets recorded.

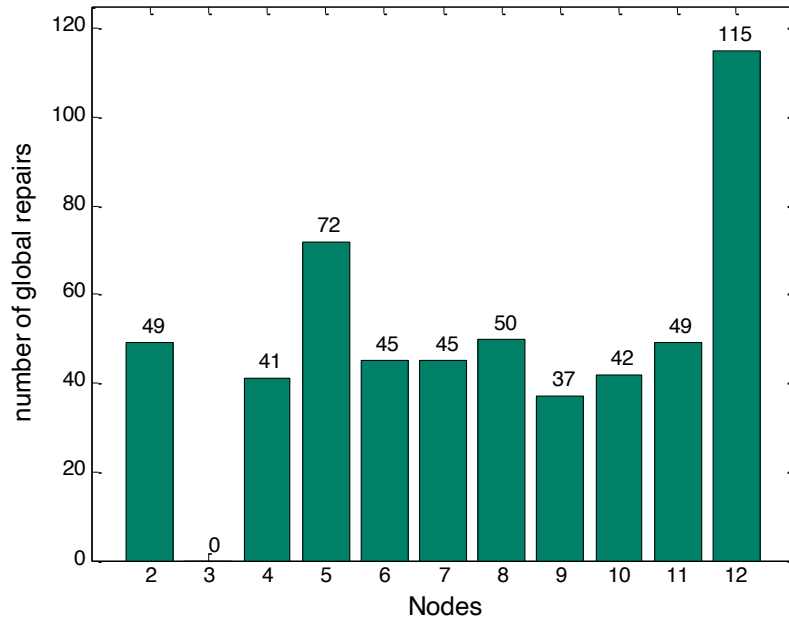


Figure 5: Launched global repairs per node

The number of launched global repairs by the nodes are shown in figure 5. According to the obtained results, the malicious node sent 132 DIO messages, where 73 DIO messages had been received by node "5", a percentage of 63%. This low rate is explained by the effect of the interference and collisions. The node 12 had received 116 DIO messages from the malicious node, a percentage of 88%, this relatively high ratio led the node 12 to record the highest launched global repairs, 115 times. On the other hand, the remaining nodes show different recorded numbers. As such, the version number attack is considered one of the worst attacks than many others, because it impacts the entire network rather than the attacker's neighborhood.

It can be noticed that the malicious node hadn't record any global repair. This remarkable behavior is extremely important to judge how serious the attack is. It is explained by the fact that the malicious node is always a step ahead in incrementing the version number. Moreover, the advertised version numbers by the root are frequently considered as inconsistencies; lower than the values advertised by the malicious node. Thus, it did not receive an increased version number, and as a result it did not launch any global repair.

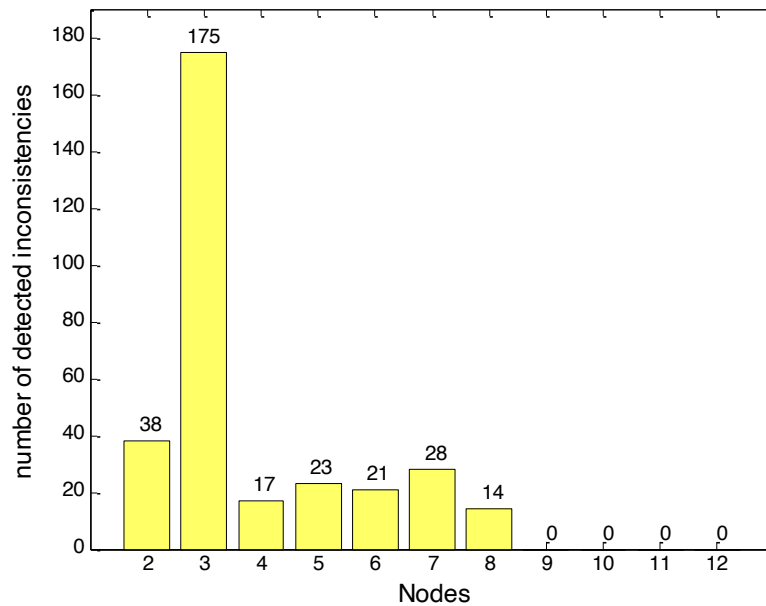


Figure 6: Number of times VN inconsistency detected per node

Figure 6 depicts the number of times a "version number inconsistency" (lower version number) is detected. The total number of inconsistencies is 372, where the node 3 (malicious node) shows the highest rate of detection. This is explained by the fact that it is the one increasing the version number, while the victim nodes are usually sending a lower version number. On the other hand, nodes 9, 10, 11 and 12 did not detect any inconsistency, because of their radio range positions from the malicious node and their unique sources.

The high ratio recorded by node 3 is the main and direct factor that led to the continuous increase in version number, because as discussed earlier the version number is incremented each time a DIO message is sent. Since the malicious node is detecting inconsistencies, it is resetting its trickle timer over and over leading to more frequent sent DIO messages, consequently more version number attacks. This exceedingly interesting result is used in our algorithm to mitigate the version number attack. Note that, we used the default values related to the trickle timer  $I_{min}$  and  $I_{doubling}$ . The attack could be more destructive if the malicious node was working with lower values. The influence of such attack could be more damaging if the nodes advertised fake lower trickle timer parameters.

Moreover, an inconsistency occurs in the case of loops, a loop is detected when a node receives a unicast DAO from a node with a lower rank, seven loops has been detected during the attack.

The immense increase in the control overhead in terms of DIO and forwarded DAO messages leads to an important amount of the consumed energy in the transmission mode and the listening mode of the forwarder nodes along the path to the root. Consequently, it exhausts and causes ruinous effects on the nodes resources by

increasing the average energy consumption, which affects the network lifetime and consequently reduces its performances.

In the well known version number attack, the farthest the malicious node from the sink is, the most damaging attack will be, because all the nodes toward the sink will launch a global repair twice for every VN incrementing, one based on the malicious node false advertisement, and the second is based on the global repair launched by the sink when detecting the inconsistency in the VN. On the other hand, in our proposed algorithm, a new advertised VN from children is not considered, and the effect of the attack depends on the available reliable neighbors. In order to see this latter impact, we performed several simulations while changing the position of the malicious node, where table 6 exhibits the obtained results. For the node "3" (the closest node to the root) 4 victim nodes are recorded, with one non recovered node. For node "7" (the farthest node from the root) only 1 node is victim and it is recovered after detection, whereas the detection of the attack is very fast, 55 seconds. These obtained results show that the harm of the attack is directly related to the unavailability of the reliable neighbors.

Attacker node	Rank	Time to detect the attack (s)	Victim nodes	Non recovered nodes
3	1	327	4	1
5	2	117	2	0
7	3	52	1	0

Table 6: Time to detect the attack function of the malicious node's position

### 6.2.2 Impact of the attack on energy consumption

The figure 7 illustrates the energy consumption in the reference network and after the attack in terms of LPM, CPU TX and RX. The radio transceiver (TX and RX) is the most energy consuming component of an IoT-based microcontroller board, which is shown in the figure. The obtained results also show that the total consumed energy jumped from 15.20 Joules to 54.12 Joules, an increase of 255 %. This increase is dominated by the TX mode, where it jumps from 5.3 Joules to 29.46 Joules, which presents an increase of 455 %. This immense upsurge is justified by the increasing number of sent, forwarded and received control messages in the topology.

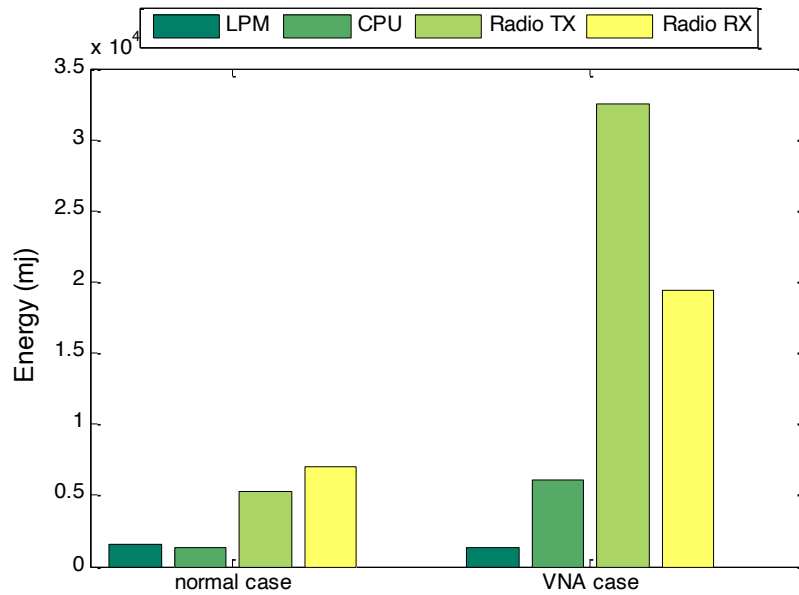


Figure 7: Energy consumption in the reference network and after the attack

These obtained results demonstrate the damaging impact of the attack on the energy, which considerably affect the nodes autonomy and consequently the network lifetime.

### 6.3 Impact of the proposed solution

Our developed algorithm is not proposed only to detect the attack, but also to stop its spreading through the network topology, and recover the victim nodes. The figure 8, shows the network topology after detecting the attack. Nodes 12, 5, 7 and 10 had not a second source to verify the legitimacy of the received version number, firstly advertised by node "3", which did not let any option to those nodes but to take the fake advertised version number value. Node 8 on the other hand, had node 6 (reliable node) as a second source of information. Since node 6 was advertising the same legitimate version number (not incremented) like node 8, this latter had figured out that the new advertised version number by its parent is illegitimate. Consequently, node 8 did not increment its VN, and it switches its parent from node 7 to node 6.

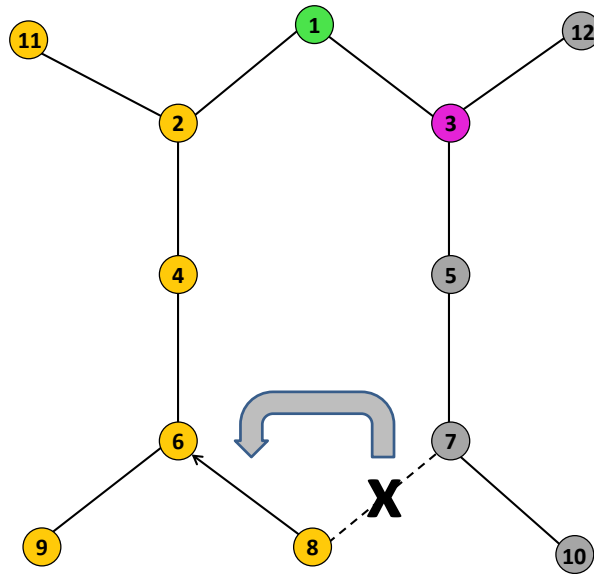


Figure 8: Network Topolgy after detecting the attack

After detecting the malicious behavior in the network, the second phase consists on recovering the victim nodes.

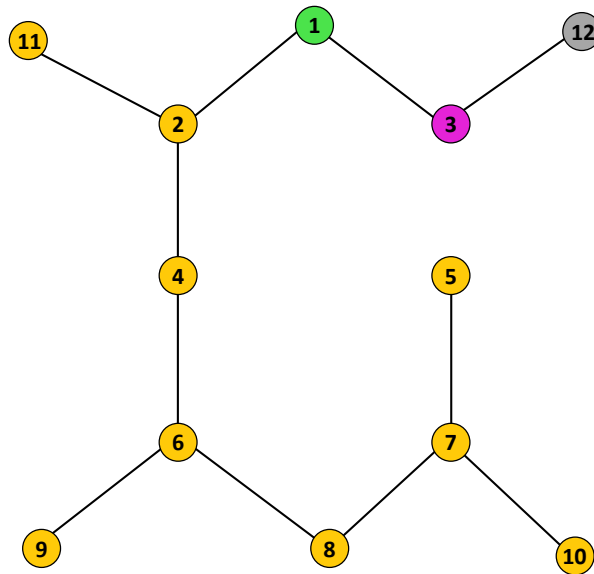


Figure 9: Network topolgy after recovering the victim nodes

The network topology after recovering the victim nodes is shown in figure 9. Node 8 is the one which stopped the spreading of the fake VN, since it had a second source to verify the legitimacy of the advertised VN of its parent. Nodes 12, 5 and 7 are setting

the value "not\_sure\_vn" to "1", since they had no a choice. After the detection of the attack, node 7 has a new reliable source, its ex-child " node 8". Based on the advertised VN by node 8, node 7 can now be certain that its current VN is not legitimate. Thus, node 7 changes its parent to be node 8, while updating its version number. The same logic is applied by node 5. On the other hand, node 12 is isolated and it has only one source, what did not give him any option to check its legitimacy. This is the only case where our proposed algorithm could be limited, which is generally a rare situation, especially in dense topologies, where if a node has no option, then its parent has. In other words, if the isolated node cannot check the legitimacy of a spread VN, it will not be a big issue since its parent has already done it, like the case of node 10.

0				7				15				31			
RPL Instance ID								V. Number				Rank			
G	0	Mop	Prf	DTSN				Flags				not_sure_vn			
DODAG ID															
Options															

Table 7: DIO Message updated by the field "not\_sure\_vn"

In order to give less chance to those nodes that are not certain about the legitimacy of their VN to be chosen a preferred parents, a reserved byte in the DIO message is exploited to advertise this situation (Table 7). This field, called "not\_sure\_vn", is set to 1 if the node is not certain about its VN. The value of this field is used as a metric for rank calculation in the parent selection process. Accordingly, the objective function has been modified to give less chance to those nodes to be chosen as preferred parent. To do so, the file "rpl-mrhof.c" is modified, in such way the calculated rank value is modified from its default value given by:

$$new\_rank = [(base_{rank}) + rank_{increase}] \quad (6)$$

to be :

$$new\_rank = [(base_{rank}) + (not\_sure\_vn * 4 * (base_{rank}))] + rank_{increase} \quad (7)$$

The figure 10 shows a possible situation, where node 14 has two choices; node 12 and node 13. Based on our modifications to the O.F the node 13 is advertising a rank value of 3023, while node 12 is advertising a rank value of 4644. These calculated rank values pouch the node 14 to choose the node 13 as preferred parent.

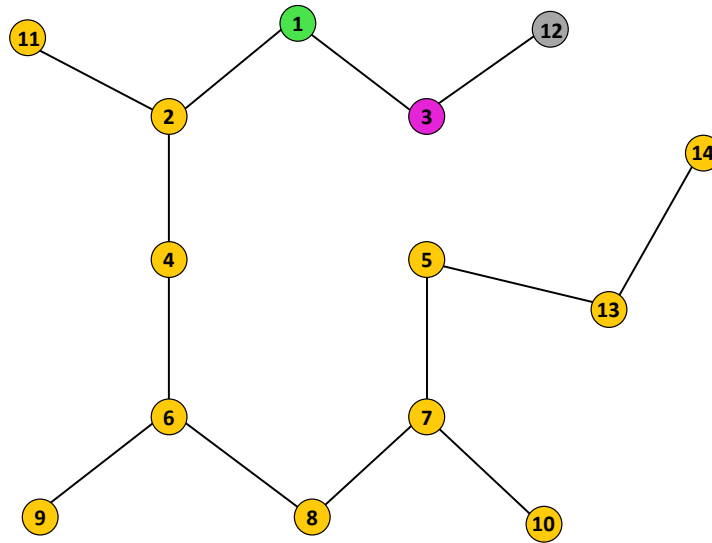


Figure 10: Joining the network after recovery

### 6.3.1 Impact of the proposed solution on the overhead

IoT nodes are resources constrained in terms of computing abilities and energy autonomy, which restricts the use of resource-hungry security mechanisms. Thus, it is primordial to measure the inducted overhead by the proposed solution in the network.

Limiting the affected area and recovering the affected nodes decreases significantly the effect of the attack on the control overhead. Table 8 shows the obtained results regarding the overhead, before and after implementing the proposed algorithm.

	Sent messages				
	Generated			Forwarded	
	Regular DIO	Regular DAO	No-Path DAO	DAO	Total
1	170	98	0	156	424
2	1357	242	592	1187	3378
3	208	113	7	241	569

Table 8: Control overhead in the reference network after the VN attack, and after the solution

Table 8 shows a decrease of 493 % in the total control overhead, from 3378 to 569 messages, where a noticeable improvement is achieved in the different control messages, in particular no-path DAO messages where it drops from 592 to 7 messages, which reflects the stability of the topology. The slight difference compared to the normal case, is due to the limited effect of the attack, in term of space (just few nodes were victims) and time (the danger of the attack is controlled after recovering the



affected nodes). Thus, the proposed developed algorithm efficiently mitigates the effect of the version number attack by significantly bringing down the overhead in the network.

### 6.3.2 Impact of the proposed solution on the energy

Regarding the energy, which is a significant and decisive metric on increasing the network lifetime, figure 10 presents the obtained results from 15 min simulations of the reference network, in the case of the attack and after implementing our solution.

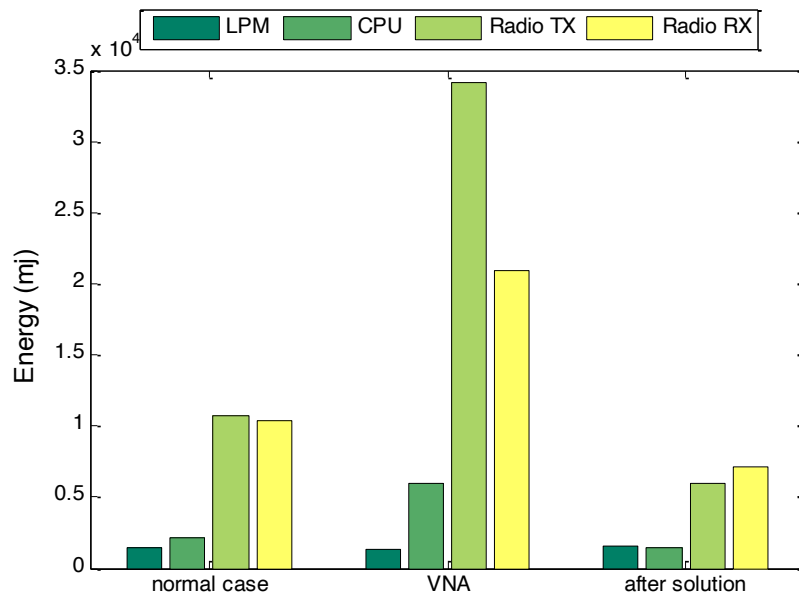


Figure 11: Energy consumption for the normal case, VN attack and after the solution

The implemented algorithm had successfully reduced the consumed energy from 59.39 Joules to 16.1 Joules, a decrease of 269%, as figure 11 depicts. Compared with the normal case, the consumed energy is a bit higher, that is due to the additional consumed energy before detecting the attack. These obtained results prove that the proposed mitigation scheme efficiently mitigates the effect of the attack and confirm its efficiency in reducing the attack negative effects, by significantly bringing down the consumed energy. Moreover, the proposed algorithm proved to be lightweight detection mechanism that does not waste network energy.

### 6.3.3 Impact of the proposed solution on the PDR

The PDR presents the ratio of the total number of received data packets by the root node to the total number of sent data packets by the network's nodes. It is a parameter to evaluate the end to-end reliability of the network.

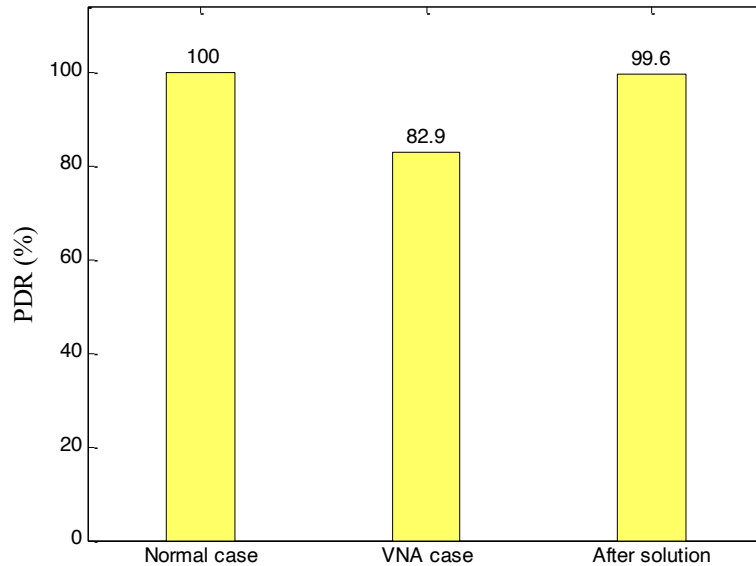


Figure 12: The PDR for the normal case, VN attack and after the solution

Figure 12 shows the obtained PDR for three cases. The figure shows that the recorded PDR suffers under the attack, where a ratio of 82.9 % is recorded, this degradation is mainly due to the packets collision and the congestion incurred due to the high overhead in the network, in particular at the forwarder nodes. The degradation has been overcome by applying our mitigation algorithm, in which we almost restore the same efficiency of the reference network. An improvement, from 82 % in the case of the attack to 99.6 % after implementing our proposed solution is obtained, which shows and confirms the effectiveness of our proposed mitigation algorithm. Note that, the recorded lost in packets in the case of the proposed solution is observed at the beginning, before neutralizing the attack.

Regarding the 30 seconds period of sending data packets, it is irrelevant value to the solution efficiency in terms of detecting the malicious node. A period of 5 seconds for sending data packets is too short to be suitable to standard IoT networks, considered as LLNs.

We conducted additional simulations while changing the period of sending data packets to 5 seconds and to 60 seconds. As table below depicts, even in a case of attack free topology a period of 5 seconds shows a low PDR (94%) compared to higher periods, this is mainly caused by the packets collision and the network congestions.

	5 s	30 s	60 s
Normal case	94 %	100 %	100 %
Under attack case	77 %	82.9 %	91 %
Under attack + solution case	87 %	99.6 %	99.7 %

Table 9: PDR function period of sending data packets

The choice of the used 30 seconds is inspired from many papers in the field that used this value in RPL security countermeasure. It's worth mentioning that the main goal of the proposed solution is to detect and isolate the malicious node, consequently to establish the network in an its attack free state (normal case).

### 6.3.4 Impact of the proposed solution on the latency

The latency is the average time taken for a given number of successful data packet transmission by the network's nodes to be received by the root node. Dropped and lost packets are not considered in this calculation.

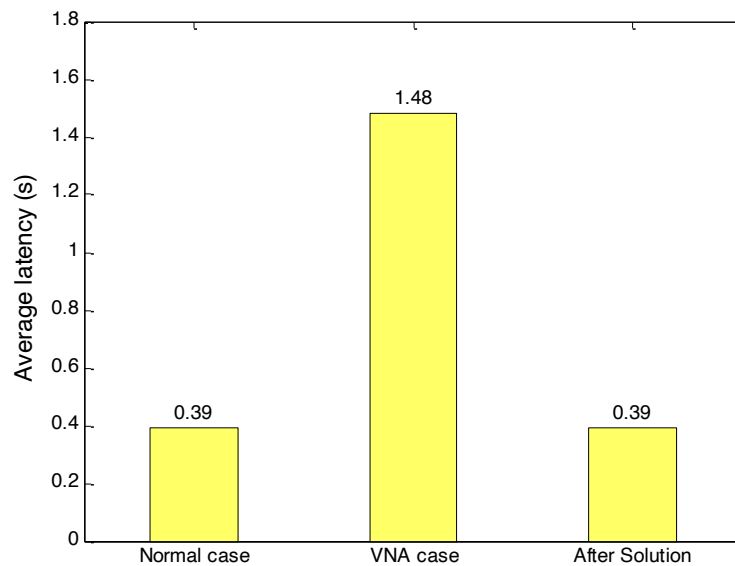


Figure 13: The latency result for the normal case, VN attack and after the solution

Figure 13 shows the recorded latency in the three cases. It is evident from the figure that the latency has been adversely affected by the attack, where it raised from 0.39 s in the normal case to 1.48 s in the case of the attack, which presents an increase of 380 %. This deterioration can be attributed to the high overhead in the topology, especially at the neighbor and forwarder nodes, that leads to inducing a higher and significant congestion. The degradation in the network performance in terms of latency has been drastically reduced when applying our proposed mitigation algorithm, where the

latency dropped to 0.33 s. The simulation results show and confirm the efficiency of the proposed mechanism.

### 6.3.5 Impact of the proposed solution on the memory footprint

Typically, the nodes have inherent constraints in terms of processing capabilities, available memory, and energy consumption. Memory usage usually includes flash memory (ROM) and random access memory (RAM). The nodes often have constrained RAM and ROM in kilobytes (about 20 kilobytes of RAM and 100 kilobytes of ROM). Thus, due to the limited amount of memory available, adopting solutions with small footprint mechanisms is critical.

	RAM (bytes)		ROM (bytes)	Total
	data	bss	Text	
<b>Standard RPL</b>	316	4908	43447	48671
<b>Modified RPL</b>	328	5012	46900	52240
<b>Delta</b>	12	104	3453	3569

Table 10: Memory usage in standard RPL versus modified RPL

Where :

- Text : shows the size of the code section in bytes,
- bss : gathers variables that are not declared with initial values, filled with zeros,
- data : stores initialized variables for copying to RAM;

The memory overhead of the standard RPL against the modified RPL is illustrated in Table 10. The proposed solution has been integrated with about 46.9kB of ROM, 3.4 KB extra when compared to standard RPL, representing only 3.7% of additional footprint. Regarding the RAM usage, it shows a slight increase with 0.11 kB, allocating just 1.16% of additional RAM footprint.

The results show that the proposed algorithm induces a slight and very acceptable increase in memory, mainly in the code (text) segment. The extra usage in RAM memory footprint is used to store additional information, such as "suspect parent", "not\_sure\_vn", "first advertiser" ...etc. Similarly, more ROM is required to store instructions that are used to check the legitimacy of the advertised version numbers.

### 6.4 Efficiency of the proposed solution on random topologies

Usually in a dense random topology, a node has many sources to check the legitimacy of the received VN. Thus, more the network is dense more the algorithms is more efficient. For this purpose, we conducted additional simulations on random topologies having a total number of nodes 20, and 30.

Total nodes	malicious node's neighbors	Potential victim nodes	average time to detect the attack	Recovered nodes
20	7	2	81 s	100%
30	8	3	74 s	100%

Table 11: Results on random topologies

The obtained results shown in table below, illustrate the efficiency of the proposed solution in a dense random topology, where the average time to detect the attack is inversely proportional to the total number of nodes.

## 7 Conclusions

RPL-based IoT networks are currently being used in many applications of our lifecycle. Securing these networks against threats is one of the biggest challenges among the researchers presently. In this paper, we have firstly deeply analyzed the version number attack, in which a malicious advertises an illegitimate incremented version number, pushing other nodes to perform frequently unnecessarily and unauthorized global repairs.

Afterwards, an efficient mitigation algorithm was presented and explained to effectively diminish the version number attack. By the proposed method, only values coming from preferred parent are considered as new version number, then the legitimacy of the received VN is determined by employing a second reliable source. We evaluated the proposed approach against standard RPL and RPL with malicious nodes, where our proposed mitigation algorithm proved its efficiency, where it detected and mitigated the VN attack while significantly improved the routing performance. In the proposed solution, the latency has been restored to attain the same value as in the normal case, the PDR is very close to the normal case, a very slight increase in energy consumption and overhead is recorded. Consequently, the proposed approach also contributes to the network lifetime extension and reduced energy consumption with regard to peer-to-peer connectivity information and the detection of malicious nodes.

In the future, we plan to study the efficiency and verify robustness of the proposed algorithms in the presence of more than one malicious node, across different dense topologies. And we plan also to validate the most adequate thresholds of time during the detection, that could a dynamic values based on the number of available neighbors. Future work will also address the testing and experimentation of the proposed solution to evaluate its performance in real deployment.

## References

[Abiodun et al., 2021] Abiodun, O. I., Abiodun, E. O., Alawida, M., Alkhaldeh, R. S., and Arshad, H. (2021). A review on the security of the internet of things: challenges and solutions. *Wireless Personal Communications*, 119, 2603-2637.

- [Agiollo et al., 2021] Agiollo, A., Conti, M., Kaliyar, P., Lin, T. N., and Pajola, L. (2021). DETONAR: Detection of routing attacks in RPL-based IoT. *IEEE Transactions on Network and Service Management*, 18(2), 1178-1190.
- [Ahmed and Ko, 2016] Ahmed, F., & Ko, Y. B. (2016, July). A Distributed and Cooperative Verification Mechanism to Defend against DODAG Version Number Attack in RPL. In *PECCS* (pp. 55-62).
- [Alfa et al., 2021] Alfa, A. A., Alhassan, J. K., Olaniyi, O. M., and Olalere, M. (2021). Blockchain technology in IoT systems: Current trends, methodology, problems, applications, and future directions. *Journal of Reliable Intelligent Environments*, 7(2), 115-143.
- [Almusaylim et al., 2020] Almusaylim, Z. A., Alhumam, A., Mansoor, W., Chatterjee, P., and Jhanjhi, N. Z. (2020). Detection and mitigation of rpl rank and version number attacks in smart internet of things.
- [Almusaylim et al., 2020] A. Almusaylim, Z., Jhanjhi, N. Z., & Alhumam, A. (2020). Detection and mitigation of RPL rank and version number attacks in the internet of things: SRPL-RP. *Sensors*, 20(21), 5997.
- [Anitha and Arockiam, 2021] Anitha, A. A., & Arockiam, L. (2021). VeNADet: version number attack detection for RPL based Internet of Things. *Solid State Technology*, 64(2), 2225-2237.
- [Aris et al., 2016] Aris, A., Oktug, S. F., and Yalcin, S. B. O. (2016, April). RPL version number attacks: In-depth study. In *NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium* (pp. 776-779). IEEE.
- [Aris et al., 2020] Arıř, A., & Oktuđ, S. F. (2020, June). Analysis of the RPL version number attack with multiple attackers. In *2020 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)* (pp. 1-8). IEEE.
- [Arosu et al., 2019] Arıř, A., Yalçın, S. B. Ö., and Oktuđ, S. F. (2019). New lightweight mitigation techniques for RPL version number attacks. *Ad Hoc Networks*, 85, 81-91.
- [Aversano et al., 2021] Aversano, L., Bernardi, M. L., Cimitile, M., and Pecori, R. (2021). A systematic review on Deep Learning approaches for IoT security. *Computer Science Review*, 40, 100389.
- [Bhatt and Ragiri, 2021] Bhatt, S., and Ragiri, P. R. (2021). Security trends in Internet of Things: A survey. *SN Applied Sciences*, 3, 1-14.
- [Boudouaia et al., 2020] Boudouaia, M. A., Ali-Pacha, A., Abouaissa, A., and Lorenz, P. (2020). Security against rank attack in RPL protocol. *IEEE Network*, 34(4), 133-139.
- [Boudouaia et al., 2021] Boudouaia, M. A., Abouaissa, A., Ali-Pacha, A., Benayache, A., and Lorenz, P. (2021). RPL rank based-attack mitigation scheme in IoT environment. *International Journal of Communication Systems*, 34(13), e4917.

- [Butun et al., 2020] Butun, I., Österberg, P., and Song, H. (2019). Security of the Internet of Things: Vulnerabilities, attacks, and countermeasures. *IEEE Communications Surveys & Tutorials*, 22(1), 616-644.
- [Chung, 2021] Chung, S. H. (2021). Applications of smart technologies in logistics and transport: A review. *Transportation Research Part E: Logistics and Transportation Review*, 153, 102455.
- [Dunkels et al., 2021] Dunkels, A., Schmidt, O., Finne, N., Eriksson, J., Österlind, F., and Durvy, N. T. M. (2011). The contiki os: The operating system for the internet of things. *Online*, at <http://www.contikios.org>, 605.
- [Farris et al., 2019] Farris, I., Taleb, T., Khettab, Y., and Song, J. (2018). A survey on emerging SDN and NFV security mechanisms for IoT systems. *IEEE Communications Surveys & Tutorials*, 21(1), 812-837.
- [Fotouhi et al., 2017] Fotouhi, H., Moreira, D., Alves, M., and Yomsi, P. M. (2017). mRPL+: A mobility management framework in RPL/6LoWPAN. *Computer Communications*, 104, 34-54.
- [Hajjaji et al, 2021] Hajjaji, Y., Boulila, W., Farah, I. R., Romdhani, I., and Hussain, A. (2021). Big data and IoT-based applications in smart environments: A systematic review. *Computer Science Review*, 39, 100318
- [Hashemi and Shams Aliee, 2019] Hashemi, S. Y., and Shams Aliee, F. (2019). Dynamic and comprehensive trust model for IoT and its integration into RPL. *The Journal of Supercomputing*, 75, 3555-3584.
- [Imteaj et al., 2021] Imteaj, A., Thakker, U., Wang, S., Li, J., and Amini, M. H. (2021). A survey on federated learning for resource-constrained IoT devices. *IEEE Internet of Things Journal*, 9(1), 1-24.
- [Javaid and Khan, 2021] Javaid, M., and Khan, I. H. (2021). Internet of Things (IoT) enabled healthcare helps to take the challenges of COVID-19 Pandemic. *Journal of Oral Biology and Craniofacial Research*, 11(2), 209-214.
- [Kanoun and Derbel, 2021] Kanoun, O., and Derbel, N. (Eds.). (2021). *Advanced Systems for Biomedical Applications*. Springer International Publishing.
- [Khan and Salah al., 2018] Khan, M. A., and Salah, K. (2018). IoT security: Review, blockchain solutions, and open challenges. *Future generation computer systems*, 82, 395-411.
- [Kharrufa et al., 2019] Kharrufa, H., Al-Kashoash, H. A., and Kemp, A. H. (2019). RPL-based routing protocols in IoT applications: A review. *IEEE Sensors Journal*, 19(15), 5952-5967.
- [Khraisat and Alazab, 2021] Khraisat, A., and Alazab, A. (2021). A critical review of intrusion detection systems in the internet of things: techniques, deployment strategy, validation strategy, attacks, public datasets and challenges. *Cybersecurity*, 4, 1-27.

- [Krishna et al., 2021] Krishna, R. R., Priyadarshini, A., Jha, A. V., Appasani, B., Srinivasulu, A., and Bizon, N. (2021). State-of-the-art review on IoT threats and attacks: Taxonomy, challenges and solutions. *Sustainability*, 13(16), 9463.
- [Lamaazi and Benamar, 2020] Lamaazi, H., and Benamar, N. (2020). A comprehensive survey on enhancements and limitations of the RPL protocol: A focus on the objective function. *Ad Hoc Networks*, 96, 102001.
- [Lamaazi and Benamar, 2020] Lamaazi, H., and Benamar, N. (2020). RPL enhancement based FL-trickle: A novel flexible trickle algorithm for low power and lossy networks. *Wireless Personal Communications*, 110(3), 1403-1428.
- [Lamkimel et al., 2018] Lamkimel, M., Naja, N., Jamali, A., and Yahyaoui, A. (2018, November). The Internet of Things: Overview of the essential elements and the new enabling technology 6LoWPAN. In *2018 IEEE International Conference on Technology Management, Operations and Decisions (ICTMOD)* (pp. 142-147). IEEE.
- [Lee and Lee, 2021] Lee, J. Y., and Lee, J. (2021). Current research trends in IoT security: a systematic mapping study. *Mobile Information Systems*, 2021, 1-25.
- [Mathur et al., 2016] Mathur, A., Newe, T., and Rao, M. (2016). Defence against black hole and selective forwarding attacks for medical WSNs in the IoT. *Sensors*, 16(1), 118.
- [Mayzaud et al., 2017] Mayzaud, A., Badonnel, R., and Chrisment, I. (2017). A distributed monitoring strategy for detecting version number attacks in RPL-based networks. *IEEE transactions on network and service management*, 14(2), 472-486.
- [Mbarek et al., 2021] Mbarek, B., Ge, M., & Pitner, T. (2021). Proactive trust classification for detection of replication attacks in 6LoWPAN-based IoT. *Internet of Things*, 16, 100442.
- [Meng et al., 2021] Meng, D., Xiao, Y., Guo, Z., Jolfaei, A., Qin, L., Lu, X., and Xiang, Q. (2021). A data-driven intelligent planning model for UAVs routing networks in mobile Internet of Things. *Computer Communications*, 179, 231-241.
- [Min et al., 2020] Min, S. W., Chung, S. H., Lee, H. J., and Ha, Y. V. (2020). Downward traffic retransmission mechanism for improving reliability in RPL environment supporting mobility. *International Journal of Distributed Sensor Networks*, 16(1), 1550147720903605.
- [Mrabet et al., 2020] Mrabet, H., Belguith, S., Alhomoud, A., and Jemai, A. (2020). A survey of IoT security based on a layered architecture of sensing and data analysis. *Sensors*, 20(13), 3625.
- [Nikravan et al., 2018] Nikravan, M., Movaghar, A., and Hosseinzadeh, M. (2018). A lightweight defense approach to mitigate version number and rank attacks in low-power and lossy networks. *Wireless Personal Communications*, 99, 1035-1059.
- [Niu, 2021] Niu, X. (2021). Optimizing DODAG build with RPL protocol. *Mathematical Problems in Engineering*, 2021, 1-8.



- [Obaidat et al., 2020] Obaidat, M. A., Obeidat, S., Holst, J., Al Hayajneh, A., and Brown, J. (2020). A comprehensive and systematic survey on the internet of things: Security and privacy challenges, security frameworks, enabling technologies, threats, vulnerabilities and countermeasures. *Computers*, 9(2), 44.
- [Pishdar et al., 2021] Pishdar, M., Seifi, Y., Nasiri, M., and Bag-Mohammadi, M. (2022). PCC-RPL: An efficient trust-based security extension for RPL. *Information Security Journal: A Global Perspective*, 31(2), 168-178.
- [Pour et al., 2021] Pour, M. S., Watson, D., and Bou-Harb, E. (2021, June). Sanitizing the iot cyber security posture: An operational cti feed backed up by internet measurements. In *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)* (pp. 497-506). IEEE.
- [Preet et al, 2020] Preet, I. K., and Saini, K. S. (2020). A Systematic Evaluation of Literature on Internet of Things (IoT) and Smart Technologies with Multiple Dimensions. *Journal of Technology Management for Growing Economies*, 11(1), 1-10.
- [Rouissat et al.] Rouissat, M., Belkheir, M., & Belkhira, H. S. A. (2022). A potential flooding version number attack against RPL based IOT networks. *Journal of Electrical Engineering*, 73(4), 267-275.
- [Roussel et al., 2016] Roussel, K., and Zendra, O. (2016, February). Using Cooja for WSN simulations: Some new uses and limits. In *EWSN 2016—NextMote workshop* (pp. 319-324). Junction Publishing.
- [Sahay et al., 2019] Sahay, R., Geethakumari, G., Mitra, B., & Sahoo, I. (2020). Efficient framework for detection of version number attack in internet of things. In *Intelligent Systems Design and Applications: 18th International Conference on Intelligent Systems Design and Applications (ISDA 2018) held in Vellore, India, December 6-8, 2018, Volume 2* (pp. 480-492). Springer International Publishing.
- [Salman and Jain, 2019] Salman, T., and Jain, R. (2019). A survey of protocols and standards for internet of things. *arXiv preprint arXiv:1903.11549*.
- [Selvaraj and Sundaravaradhan, 2020] Selvaraj, S., and Sundaravaradhan, S. (2020). Challenges and opportunities in IoT healthcare systems: a systematic review. *SN Applied Sciences*, 2(1), 139.
- [Sharma et al., 2020] Sharma, V., You, I., Andersson, K., Palmieri, F., Rehmani, M. H., and Lim, J. (2020). Security, privacy and trust for smart mobile-Internet of Things (M-IoT): A survey. IEEE access, 8, 167123-167163.
- [Shrestha and Furkan, 2020] Shrestha, S. K., & Furqan, F. (2020, November). IoT for Smart Learning/Education. In *2020 5th International Conference on Innovative Technologies in Intelligent Systems and Industrial Applications (CITISIA)* (pp. 1-7). IEEE.
- [Simoglou et al., 2021] Simoglou, G., Violettas, G., Petridou, S., and Mamas, L. (2021). Intrusion detection systems for RPL security: a comparative analysis. *Computers & Security*, 104, 102219.

- [Simoglou et al., 2021] Simoglou, G., Violettas, G., Petridou, S., & Mamatas, L. (2021). Intrusion detection systems for RPL security: a comparative analysis. *Computers & Security*, 104, 102219.
- [Snehi and Bhandari , 2021] Snehi, M., and Bhandari, A. (2021). Vulnerability retrospection of security solutions for software-defined Cyber-Physical System against DDoS and IoT-DDoS attacks. *Computer Science Review*, 40, 100371.
- [Tun et al., 2021] Tun, S. Y. Y., Madanian, S., and Mirza, F. (2021). Internet of things (IoT) applications for elderly care: a reflective review. *Aging clinical and experimental research*, 33, 855-867.
- [Verma and Ranga, 2020] Verma, A., & Ranga, V. (2020). Mitigation of DIS flooding attacks in RPL-based 6LoWPAN networks. *Transactions on emerging telecommunications technologies*, 31(2), e3802.
- [Verma and Ranga, 2020] Verma, A., and Ranga, V. (2020). Security of RPL based 6LoWPAN Networks in the Internet of Things: A Review. *IEEE Sensors Journal*, 20(11), 5666-5690.
- [Violettas et al., 2019] Violettas, G., Petridou, S., and Mamatas, L. (2019). Evolutionary software defined networking-inspired routing control strategies for the Internet of Things. *IEEE Access*, 7, 132173-132192.
- [Wang et al., 2021] Wang, B., Tao, F., Fang, X., Liu, C., Liu, Y., and Freiheit, T. (2021). Smart manufacturing and intelligent manufacturing: A comparative review. *Engineering*, 7(6), 738-757.
- [Winter et al., 2012] Winter, T., Thubert, P., Brandt, A., Hui, J., Kelsey, R., Levis, P., and Alexander, R. (2012). *RPL: IPv6 routing protocol for low-power and lossy networks* (No. rfc6550).
- [Zolertia, 2021] S.L. Zolertia: Z1 datasheet. (2010). [http://zolertia.sourceforge.net/wiki/images/e/e8/Z1\\_RevC\\_Datasheet.pdf](http://zolertia.sourceforge.net/wiki/images/e/e8/Z1_RevC_Datasheet.pdf).