Enhancement of accuracy of the Rank Inconsistency Detection Algorithm

Ahmad Mujtaba[1], Salal Amjad[1], AmmarRafiq[1*], Asim Mubarik[1], Muhammad Usman Younus[2,3]

1. *Department of Computer Science, NFC Institute of Engineering & Fertilizer Research, Faisalabad, Punjab*
2. *Ecole Math´ematiques, Informatique, T´el´ecommunications de Toulouse, Universit´e de Toulouse, France*
3. *Department of Electrical and Computer Engineering, COMSATS University Islamabad, Sahiwal Campus, 57000, Pakistan*

∗ **Corresponding Author:** Email: ammar.rafiq@hotmail.com

Abstract

Devices used for Internet of Things (IoT) are lacking in resources, and they are susceptible to many attacks that the internet provides a pathway. These devices are connected to the internet using the protocol known as IPv6 ad the network specifically designed for them 6LoWPAN. Authentication and encryption are used in all the layers of the standard IoT protocol stack; however, due to the resource constraints present in IoT devices, the standard security measures used in conventional networks cannot be applied robustly and efficiently to IoT. Hence they are susceptible to both internal and external attacks. That is why a special IDS that suits the requirements of IoT is necessary for the widespread use of IoT.

In this paper, we propose and implement a modified version of the SVELTE Intrusion Detection System (IDS). We have specifically improved the rank inconsistency detection algorithm of the SVELTE IDS. In our implementation, we have tested the system against sinkhole attack; however, the approach can be extended to cater to other emergent attacks of the rank attack. The algorithm has been implemented in Contiki OS, and the simulations for the experiments have been performed in Cooja.Our results show that the modified algorithm performs significantly better in the simulated scenarios than the original algorithm. The results also show that accuracy is not 100%, and there are some false alarms. According to our results, the accuracy for an eight-node network is 89%, the accuracy for sixteen node network is 90.2%, the accuracy for a thirty-two-node network is 87%, and the accuracy for a sixty four-node network is 73%. In terms of delay, a modified version of SVELTE is also giving better performance. Moreover, energy and average power consumption are a little bit better than the modified version of SVELTE.

Key Words: Detectio, IDS, IoT, Network, RPL, SVELTE,Security

1. Introduction

6LowPAN stands for IPv6 over Low-power Wireless Personal Area Network. 6LoWPAN is used to connect resource-constrained devices or things to the internet by using compressed IPv6. It functions as an adaptation layer that assigns IPv6 addresses to IoT devices and enables them to connect to the internet by using 6LowPAN border router. This type of network is also known as a 6LowPAN network. 6LowPAN networks rely on IEEE 802.15.4 as a link layer protocol as well as a physical layer protocol [1]. This IP-connected WSN is susceptible to both internal attacks from 6LowPAN networks and external attacks from the global internet. Some applications of IoT are smart home, smart grid , smart city, self driving vehicles and smart hospitals.

Many researchers have proposed methods to provide security of data packets for IoT using lightweight Datagram Transport Layer Security (DTLS) [2], Internet Protocol Security (IPsec) [3], and link-layer security. These type of IoT networks are still vulnearable to many attacks that can damage the network. The development of IDS for IoT that meet the requirements of IoT networks is necessary to detect such type of attacks. Many IDS have been built for IoT based on conventional networks approach but it does not meet the requirements of IoT networks. The SVELTE IDS is one of the first IDS that is specifically built for IoT networks.

1.1 Background

Protocol for Low power and Lossy Networks (RPL) [4] stands for routing protocol for low power and lossy networks. It was standardized by the IETF in 2012 as the de facto routing protocol to be used in IoT networks. The theoretical background of RPL is discussed briefly in this section.

1.1.1 RPL Operation

RPL creates the network in the guise of a directed acyclic graph (DAG). This DAG is then broken into different Destination Oriented Directed Acyclic Graph (DODAGs) (destination oriented DAG). Each DODAG is distinguished from one another using four parameters:

1. DODAG version number
2. RPL instance ID
3. Rank
4. DODAG ID

It is possible for only one DODAG to exist in the whole network.

Node Rank is a special property that exists in RPL networks which denotes the quality of the path provided by that particular node towards the destination_which is the sink node/border router. The rank of a node is lesser the closer it is to the sink/border router. The lower rank a node has the better its rank is.

Each node has to calculate its rank using the information provided by its parent node. The rank of the parent plays a significant role in calculating the rank of the child node. RPL uses the rank rule that no node can have rank equal to or less than its parent to prevent loop formation [4].

1.1.2 RPL control messages

RPL networks have three major messages that are primarily used for topology construction and one message that is usually not mentioned. All the messages are discussed below:

DIO message: This DODAG information object (DIO) is broadcast by the border router at first, the nodes that are already part of a DODAG also send it to their children and neighbours. DIO message performs a similar function as an advertisement message in a conventional network . This message informs other nodes of the presence of a DODAG. Whenever a change in the DODAG occurs DIO is used to inform all nodes of the change. The change here refers to changes in a node's rank or the change in version number of a DODAG etc. DIO flow from top to bottom (parent to child)[4]. Fig. 1 shows the DIO message structure.

DIS message: The DODAG information solicitation object is sent by a node that is not part of any DODAG to ask for topology information. It is sent by a node if that node is a floating (not part of a DODAG) node and is not connected to any DODAG. This message is sent when the node has not received any DIO messages for some time [4]. This message elicits a DIO in response.

DAO message: DODAG advertisement object is the message that sends destination information upwards during the routing updating process. The DAO message is sent in response to a DIO message and it contains information about the node which is sent towards the sink node. It also asks for permission to join the network [4]. Fig. 2 shows the DAO message structure.

DAO-ACK message: Apart from these three messages there is another message used in RPL networks, it is called DODAG advertisement object acknowledgement or DAO-ACK. This message is sent in response to a DAO message and it contains the information about whether the node that sent the DAO is allowed to join the DODAG or not [4]. There are two ways a node can join a DODAG;

1. As a router
2. As a leaf node

The DAO-ACK also contains information that informs the joining node about how it should connect to the network i.e. as router or as a leaf.

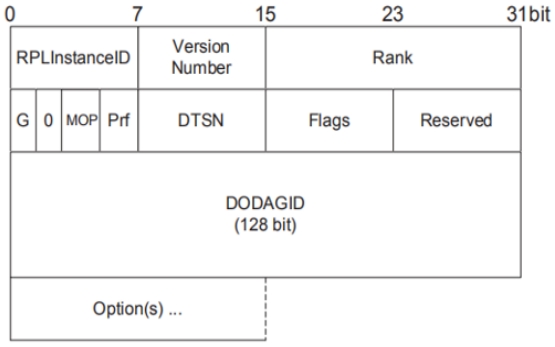

Fig. 1: DIO message structure [4]

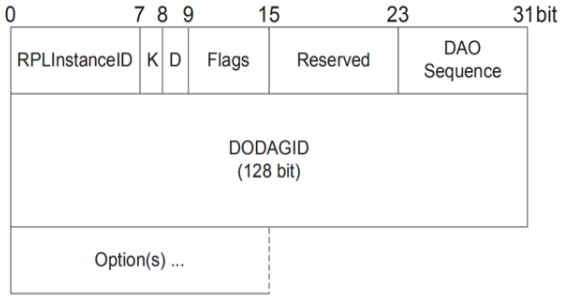

Fig. 2: DAO messzage structure [4]

1.2 RPL operation

Each DODAG has nodes that collect data from sensors and send that data towards a common destination. The other type of node is the sink node which collects the data sent by the other nodes.

Route in a DODAG is selected based on the metric that is being optimized such as distance, energy or estimated transmission count.

To create a topology each node has to choose from a set of parent nodes. Parent nodes are neighbouring nodes that have a much better rank than the node in question.

Parent offering best route i.e having the lowest rank is set as "preferred parent"[4].

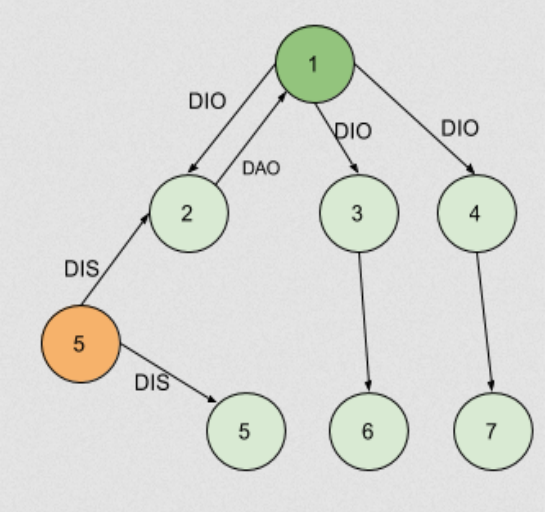

Fig. 3: Simple rpl DODAG

The sink node or border router broadcasts a DIO that advertises to all the nodes in its transmission range that it is making a DODAG and they are welcome to join it. All the nodes that are interested in joining it send back a DAO and a DAO-ACK (acknowledgement) is transmitted to them from the sink that informs them whether or not they have been accepted as part of the DODAG. These nodes then broadcast another series of DIO to other nodes that are in their transmission range but not in the range of the sink and inform them about the DODAG. A DIO has the rank, version number and instance ID of the node that sent it. Any recipient will receive multiple DIOs from different nodes and it will compare the rank value and select the lowest rank node as its 'preferred parent'.

All communication is done through the preferred parent by default. DIS is sent by a node that is trying to locate a DODAG so it can join it. DIS are sent in the case when no DIO messages are received by the node. Fig. 3 shows a simple DODAG and the direction of the control messages sent in a DODAG [4].

1.2.1 Security Challenges

The rank of a node performs an important role which is interconnected to most operations that are performed in RPL. The major tasks it performs are to prevent the creation of any loops in the network, optimise the various overheads that are incurred in a network and construct an ideal topology. Because so many tasks are related to this property that makes it a very lucrative target to leverage in order to disrupt a network. To meet the resource constrained requirements RPL assumes that all nodes in the network are reliable and not compromised. Thus it has no methods to determine whether a node is sabotaged or not [4]. The rank attack is an internal attack that can be carried out against an RPL network due to this assumption.

In this paper, an algorithm is proposed that tries to lower the value of node rank in order to obtain a better position in the network that will trigger the condition. Previously, only one condition was evaluated to select the parent node. However, in this paper second condition is also introduced. The idea behind introducing the second condition was to stabilize the network and discourage frequent switching of parents, and it would be difficult for a valid inconsistency to trigger both conditions at once. A real compromised node will trigger both conditions because for it to gain any significant advantage, the rank value needs to be falsified significantly. A significant change in the rank will trigger both conditions consistently. The minimum rank refers to the minimum rank of the neighboring nodes. The parent switching threshold is a value that attunes the metric that the objective function uses to stabilize the network and discourage frequent switching of parents, as this is detrimental to a network's efficiency and lifetime in the long term

The next section discusses some related works that have been done by other researchers in this field. Section 3 describes the working of SVELTE and the algorithm we have modified to improve the accuracy. Section 4 discusses the implementation details while section 5 outlines the experimental arrangement. Section 6 presents the results and the related discussion. Section 7 and 8 present some future enhancement and conclusion.

2. Related Work

Authors in [5] surveyed the various existing IoT frameworks and compared them with respect to their security features. The frameworks in their study were as follows: Brillo/Weave from Google, AWS IoT from Amazon, Calvin from Ericsson, Kura from Eclipse, ARM Bed from ARM, SmartThings from Samsung. Azure IoT Suite from Microsoft and HomeKit from Apple. They

highlighted the security issues of these systems.

In [6], an intrusion detection mechanism (SVELTE) is developed for IoT networks. The author target to prevent the routing path information (i.e., selective forwarder information, altered routing information, etc.)for the attacker. For the simulation work, Contiki OS was used, which shows it also has false alarms. However the authors in [7] identified four types of rank attacks that didn't use false rank and version but instead compromised the rank rule to select best parent and analyzed how they adversely affected the network performance, they didn't offer any solutions to detect the attacks.

Authors in [8] showed two attacks that allowed an attacker to compromise a node and make it change its rank illegally and send fake DIOs that changed the version number and made other nodes think that they were in the wrong version. They also proposed an algorithm called version number and rank authentication that based on encryption authenticates a nodes rank and version value change. Their analyses of the results showed a tolerable increase in resource consumption and a significant improvement in security.

Authors in [9] proposed an IDS that had two modules and were placed in different parts of the network. Their IDS was tested against sink hole and selective forwarding attacks. They tested the system in both lossy and lossless environment. Their system had good performance in lossless environment but suffered with high false positives in lossy environment.

Authors in [10] reviewed and classified various RPl attack methods and their mitigation methods, such as neighbor attacks, rank attacks etc. Their survey concluded that to weaken many RPL attacks simultaneously the approach that utilized combination based IDS and specification based IDS were the most effective.

Authors in [11] proposed an SBIDS (sink based intrusion detection system) that detects whether a rank attack is happening or not. Their algorithm only uses rank as a metric and its limitation is that the accuracy of this algorithm drops when mobility is present in the network.

Authors in [12] implemented an attack against the objective function zero and compared the destructive effects with MRHOF function and showed through results that the rank attack on MRHOF function is more destructive due to the attackers ability to manipulate multiple metrics.

Authors in [13] proposed a validation and verification method to prevent rank attacks and were very focused on the energy aspect of the nodes. Their E2V system was sink based as it was placed in the sink node and they showed favorable results to prevent an energy metric based rank attack.

An intrusion detection framework, COLlaborative Intrusion DEtection (COLIDE) is proposed for the Industrial Internet of Things (IIoT) [14]. The basic principle of the framework is to collaborate between the edge router and individual sensor nodes. The proposed framework reduces the thread and also energy efficient due to minimum communication overheads.

Authors in [15] surveyed various types of systems that seek to protect against rank attacks. They mainly discussed the specification based, anomaly based and signature based methods along with a few IDS.

As the importance of RPL is increasing day by day due to IoT. It is also facing security-related issues such as two major threads called Wormhole and Sybil attacks, and many researchers are trying to give a solution for the detection of attacks. In [16], two solutions are provided that exploit the concept of Highest Rank Common Ancestor (HRCA) to find the ancestor with the highest rank from all ancestors. This paper's basic focus is the mitigation of adversarial scenarios when multiple or single malicious nodes try to make multiple fake IDs of authorized nodes.

Authors in [17] implemented a trust based defence scheme for mitigating black hole and selective forwarding attacks in their research. Their main method of results validation was to use a test bed in which they implemented their system. Their system showed a marked increase in performance when under attack by the mentioned attacks.

Authors in [18] analyzed the security issues of RPL and proposed a new protocol based on RPL called the M-RPL. Their protocol used hierarchical clustering topology and an intelligent device to create backup routes that were used when the network was compromised. Their results showed that the M-RPL protocol can adequately defend against routing attacks.

Authors in [19] surveyed the works done on RPL. Their main focus was on the research that prioritised topology optimization, security and mobility in RPL. They also provided some suggestions about the future enhancements that could be done in the works that they surveyed. Their aim was to provide a good guideline for

anyone who wanted to do research in this field.

Authors in [20] proposed a new RPL based protocol which they dubbed 'Split'. Their protocol used a remote attestation method to determine the validity of a network node. To reduce the network overhead, they modified the RPL protocol to carry the attestation messages along with the standard RPL control messages. Their results showed that 'Split' is a reliable and scalable protocol.

In [21], the author proposed a scheme for decreasing the DODAG information Solicitation (DIS) flooding attack that increases the control packet overhead. The proposed scheme named Secure-RPL reduced the DIS flooding attack in RPL based 6LoWPAN networks.

Authors in [22] proposed a trust based mechanism to detect attacks and then remove them. Their system called 'secTrust' is embedded in the RPL protocol. They tested their system against rank and sybil attacks. They also provided results from both simulations and test beds to show the superior performance of their system.

Authors in [23] proposed a new metric and a new objective function. Their primary objective was to address the issues related to packet loss and energy depletion in a standard RPL network. They introduced a new objective function called "context aware objective function (CAOF)" and a new metric called "context aware routing metric (CARF)". Using these two new additions that took the context of the nodes into consideration they also addressed the 'thundering herd' problem. Their results showed that the network lifespan was increased compared to standard RPL.

Authors in [24] proposed a new method to detect sink hole attack in RPL network called "DEEM". Their primary aim was to propose a method that had low energy overhead while having good accuracy. "DEEM" works in two phases, an "information gathering" phase and a "detection" phase. Their results proved that their detection method had low energy overhead while also being fairly scalable.

Authors in [25] analyzed the security issues of RPL and proposed a framework to simulate attacks against RPL. They ran four types of attacks against RPL and then collected data from the simulations. The data was used in a machine learning algorithm to create a data set that was used to detect the attacks run by the framework developed by the authors.

All the mentioned papers have addressed the security issues of IoT networks with respect to RPL and rank attack and its emergent attacks. They have used a variety of approaches such as using artificial intelligence, creating new metrics or leveraging existing metrics to mitigate the threats. All of them considered security of IoT from the software point of view. In,[26] an intrusion detection architecture base on a deep neural network (DNN-KNN) is proposed for fog computing. A specific type of attack and non-attacks are classified through two steps, and the rate of information gain is used for selecting the attributes. The proposed approach is based on a k-Nearest Neighbor algorithm and Deep Neural Networks, and it is evaluated with the help of two public databases CICIDS2017 and NSL-KDD.

3. Proposed Method

SVELTE [9] is an opensource IDS developed by linus Wallgren, ShahidRaza and Theimo Voigt. We have used the system provided by the authors and modified one of the algorithms_the algorithm for the detection of rank attack (rank falsification). Understanding SVELTE IDS is fundamental to understanding the core of our work and our contribution in the whole program. Since we have used the existing mechanics of SVELTE IDS, our contribution can be understood much clearly if a basic concept of the said open source IDS is acquired.

Raza et al designed SVELTE to be used alongside 6loWPAN network that utilizes the DTLS protocol [2] and IPsec [3] to ensure the integrity of end-to-end messages. Keeping the resource constrained nature of an IoT network in mind they decided to opt for a hybrid approach when it came to the placement of their system. They placed IDS modules in the border router as well as the constrained nodes that form most of the network. The modules are discussed in this section. There is also a mini firewall in the border router but that is not relevant to this paper.

3.1 6LoWPAN Mapper

This component can be said to be the mostimportant part of the SVELTE IDS as the wholesystem relies on the ability of this module. The6LoWPAN mapper or "6Mapper" for short recreates the whole RPL DODAG in the border router and it also stores additional information about each node in the DODAG. The additional information that it stores is each node's own ID and rank, the neighbours ID and their rank as well the node's parent's ID and rank.\\ To construct theDODAG in the border router the 6LoWPAN Mapper sends out a 'mapping' request to all theparticipating nodes present in the 6LoWPAN network at regular intervals. The specialised

packet also called "request packet" sent back by the sensor nodes contains information that is required to recognize an RPL DODAG such as RPL Instance ID, the DODAG ID etc. The request packet also contains a timestamp (\textit{Ts}) that is used to determine the recentness of the network information used in mapping.

In the mapping response each node in the network sends back a specific packet which has the following information added to it ;

- node ID,
- node rank,
- node parent ID,
- node parent rank,
- ID of all the neighbour nodes and

This information is necessary because the modules placed in the border router use it to compare information for each node and detect inconsistencies in the rank.

3.1.1 Valid Inconsistencies

In the SVELTE IDS 6 mapper module there exists a way that can cause a inconsistent mapping responses to occur without any malicious activity present. This results in false positives and brings down the accuracy of the IDS. Normally an inconsistency occurs when an attacker deliberately changes some information that is being sent or when a node sends outdated information.

A valid inconsistency can occur when the 6 mapper requests information from a node say node A, node A sends its information with a particular rank, however soon after it recalculates its rank and the new rank is a better rank than what it reported to the mapper. The updated rank is sent to neighbours and children using DIO message. The neighbours report the new rank to the mapper in the mapping response. The mapper detects an inconsistency for the rank value of Node A. This is known as a valid inconsistency. To reduce the affect of this phenomenon the authors added a condition that only flags a node as malicious if it is detected to have inconsistent information twice in a row. However it is not out of the question for valid inconsistency to occur twice in a row and thus the problem of high false positive rate persists.

3.1.2 Intrusion detection in SVELTE

A compromised node can falsify its rank and use it to carry out a rank attack (sinkhole attack). In a sinkhole attack a malicious node attracts its neighbours towards itself by advertising false information that labels it as a good quality path provider. It can be paired with more attacks such as wormhole or selective forwarding attack to damage a network. A false rank will cause inconsistencies to occur however it is possible that those inconsistencies occurred due to the valid inconsistency described in the previous section. To draw a line between accidental and malicious inconsistencies two things are considered:

1 number of inconsistent ranks,
2 deviation between reported ranks

The authors used a \textit{Fault threshold} and if the difference between ranks surpasses a certain value (discussed with our algorithm) the \textit{node fault} would be increased. Then if the the \textit{node fault} surpasses the fault threshold then the node is flagged as malicious by the intrusion detection system.

3.2 Enhanced SVELTE algorithm for detection of malicious node

Algorithm 1: Enhanced SVELTE algorithm for detection of malicious node

Input:
require a list of nodes N

Output:
Detection of malicious node
Initialization:
Initialization

```
1   for node in N
2     If (Node.rank + MinHopRankIncrease)
      < (node.parent.rank) & (Node.rank)
      < (min.rank - parent switching threshold)
      do
3           node.fault = node.fault+1;
4   else
5         Do nothing
6   end
7   for node in N
8           If (node.fault) > FaultThreshold
            do
9             raise alarm
10          Else
11            Do nothing
12          End
13  end
```

In this algorithm we have introduced a second condition apart from the first one.

A deliberate effort to lower the value of node rank in order to obtain a better position in the

network will trigger the condition. The idea behind introducing the second condition was that it would be difficult for a valid inconsistency to trigger both conditions at once. A real compromised node will trigger both conditions because for it to gain any significant advantage the rank value needs to be falsified significantly. A significant change in the rank will trigger both conditions consistently.The minimum rank refers to the minimum rank of the neighbouring nodes.

The parent switching threshold is a value that attunes the metric that the objective function uses to stabilise the network and discourage frequent switching of parents as this is detrimental to a networks efficiency and lifetime in the long term.

The time complexity of the algorithm is $O(n) $. SVELTE IDS has a two minute timer after which the algorithm runs. This is to prevent excessive control traffic from clogging the network. The modified algorithm runs within the given time.

The results of using this algorithm compared to the original algorithm are presented and evaluated in section 6.

4. Experimental Setup & Implementation

SVELTE was enacted in the Contiki OS [27]. Contiki is a popular OS for the IoT. Contiki's own implementation of RPL called ContikiRPL is quite extensively tested and has been used in the implementation of SVELTE. This implementation of RPL was used by [9] to develop the 6LoWPAN Mapper and the IDS modules. A firewall was also implemented but it is not relevant to this paper. Raza et al also used the *mu* IP stack which is a contiki IP stack used for the IP communication in 6LoWPAN.They also used the SICSLoWPAN which is Contiki's version of the 6LoWPAN header compression. Since we are using the same system as provided by the authors our implementation is the same. We have only modified an algorithm as shown in section 3. We have used the implementation of the sinkhole attack as provided by the authors of [9] for testing and evaluating the altered detection algorithm.

We implemented our proposed scheme through simulation and tested the system with five configurations of nodes. Contiki's network simulator called Cooja [28] was used to run the experiments. This simulator has been proven to provide realistic results [29]. The code that is run in cooja can also be deployed to real devices that have been emulated in cooja. We have used the Tmote sky [30-32] motes to simulate the network in our experiments. It is vital to note that all our tests and simulations were run in the Lossy network instead of lossless because practically IoT networks are lossy networks.

We can assume that the border router is not a constrained node and that it can be personal computer or laptop device, however in our simulations it is run natively, meaning it is run on linux as that is the platform that contiki is running on. We have run each simulation for the five node configurations five times and from those we have calculated the average values of the following measurements:

1. False positive rate
2. False negative rate
3. True positive rate
4. True negative rate

From these 4 measurements we measured the accuracy of the algorithm. Each simulation was run for ten minutes.

4.1 Node Configurations

The node configurations and the associated details will be discussed in this section. We have run the experiments using five configurations of nodes. The configurations are as follows:

- 8 node configuration
- 16 node configuration
- 32 node configuration
- 48 node configuration
- 64 node configuration

4.1.1 8 node configuration

In Fig. 4 we can see the configuration of an RPL network of 8 nodes, we have 6 nodes that are normal sky motes characterised by their yellow color.\\1 node that is a malicious node characterised by its purple color.\\ 1 node that is the border router and is characterised by its color green.

4.1.2 16 node configuration

In Fig. 5 we can see the configuration of an RPL network of 16 nodes; we have 13 normal nodes (yellow color), 2 malicious nodes (purple color),1 border router (green color).

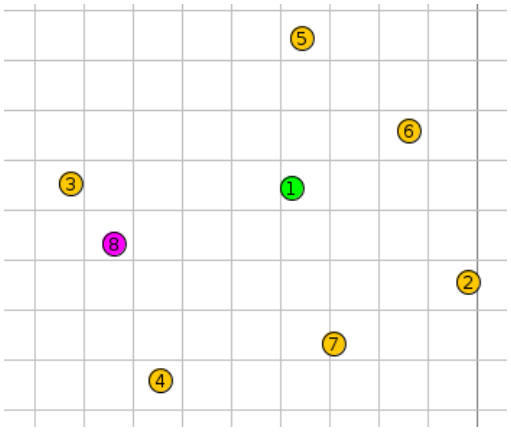

Fig. 4: 8 node configuration

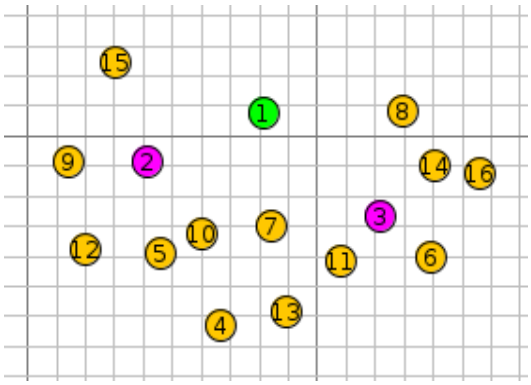

Fig. 5: 16 node configuration

4.1.3 32 node configuration

In Fig. 6 we can see the configuration of an RPL network of 32 nodes; we have 26 normal nodes (yellow color), 4 malicious nodes (purple color),1 border router (green color).

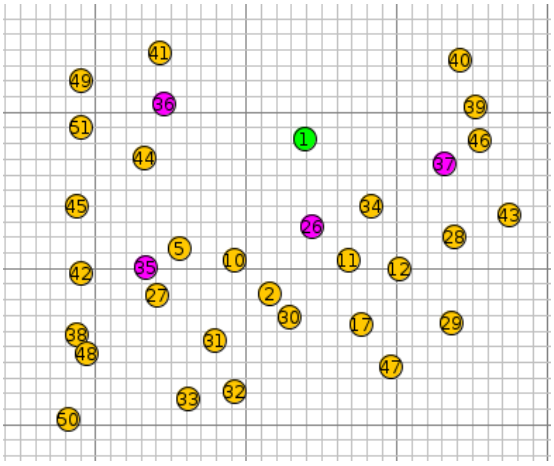

Fig. 6: 32 node configuration

4.1.4 48 node configuration

In Fig. 7 we can see the configuration of an RPL network of 48 nodes, we have:

41 normal nodes (yellow color),
6 malicious nodes (purple color),
1 border router (green color).

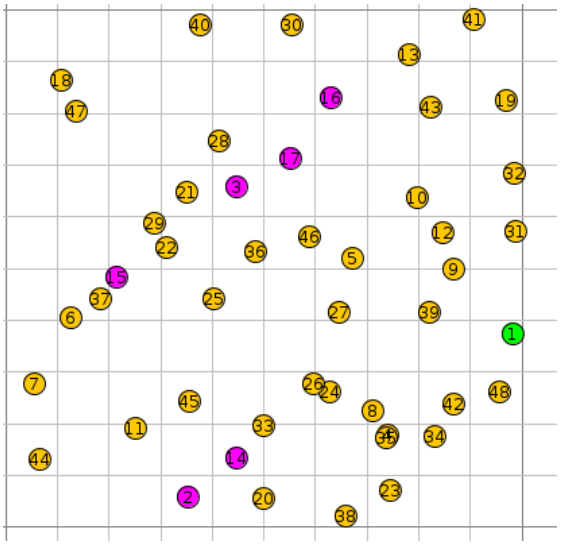

Fig. 7: 48 node configuration

4.1.5 64 node configuration

In Fig. 8 we can see the configuration of an RPL network of 64 nodes, we have:

55 normal nodes (yellow color),
8 malicious nodes (purple color),
1 border router (green color).

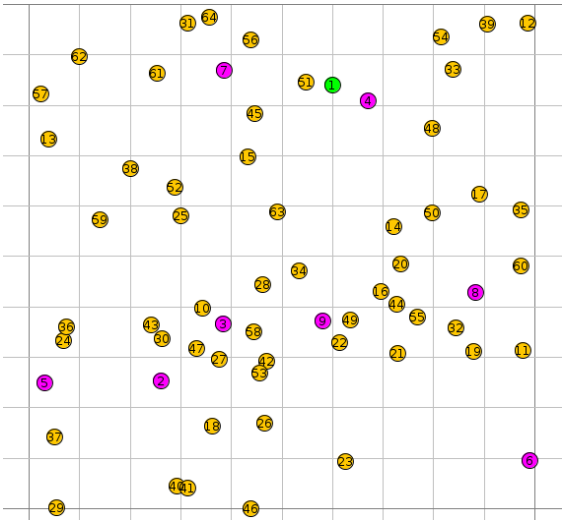

Fig. 8: 64 node configuration

5. Results \& Discussion

5.1 False positive rate

Fig. 9 shows the graph for the calculated average false positive rates for each configuration of the nodes discussed in section 6.

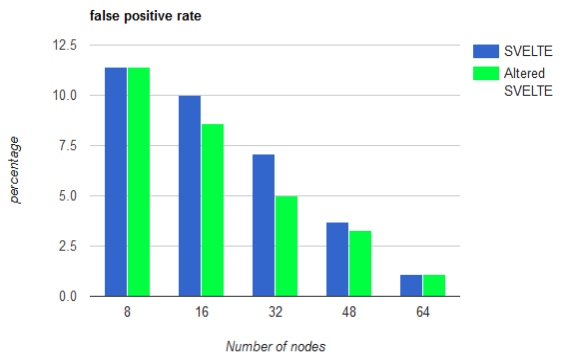

Fig. 9: False positive rate

5.2 False Negative rate

Fig. 10 shows the graph of the average false negative rates measured and calculated for each configuration from section 6.As the parent switching threshold is introduced to attunes the metric that the objective function uses to stabilize the network and discourage frequent switching of parents. It enhances the network's efficiency and lifetime in the long term.

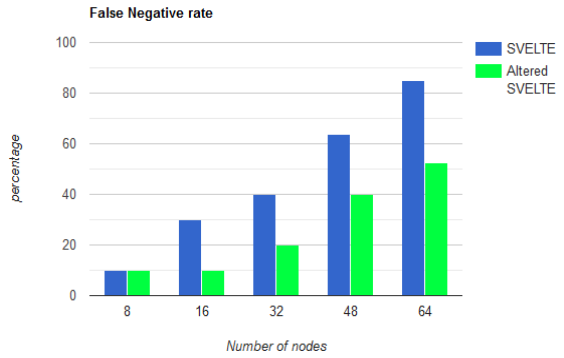

Fig. 10: False negative rate

5.3 True Positive rate

Fig. 11 shows the graph for the average true positive rate measured and calculated for each configuration from section 6.

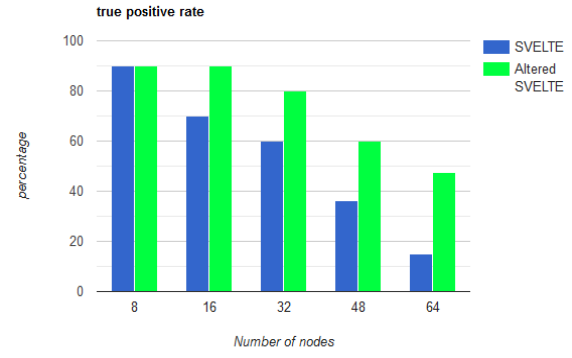

Fig. 11: True positive rate

5.4 True negative rate

Fig. 12 shows the graph for the average true negative rate measured and calculated for each configuration from section 6.

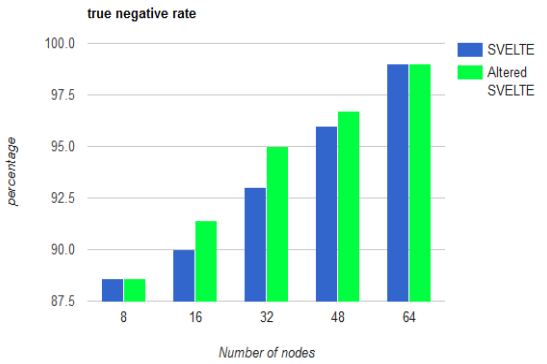

Fig. 12: True negative rate

5.5 Accuracy

Fig. 13 shows the graph for the average accuracy calculated for each configuration from section 6. In our proposed scheme, parent switching threshold attunes the metric that the objective function uses to stabilise the network and also discourage to the frequent switching of parents. Consequently, it leads to an increase the network accuracy.

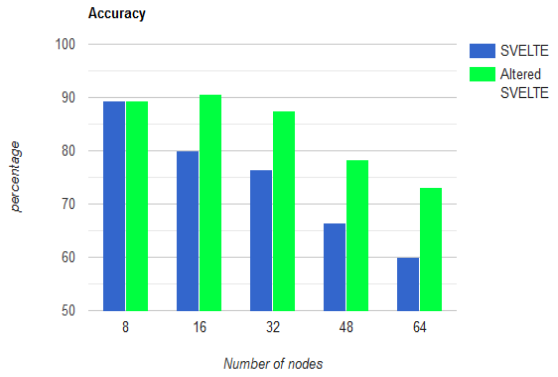

Fig. 13: Accuracy

5.6 Average Delay

Fig. 14 shows the average delay comparison between the proposed scheme and SVELTE. The Altered SVELTE gives a better performance in terms of average delay because it discourages frequent switching of a parent, and the network remains stable. It can see that when the number of nodes is less, its delay deference is low; however, by increasing the number of nodes, the Altered SVELTE is outperforming as shown in Fig. 14.

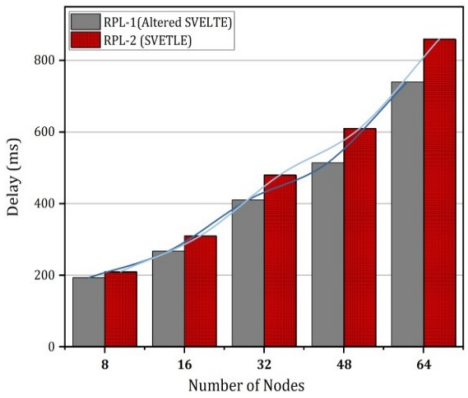

Fig. 14: Average Delay

5.7 Energy and Power Consumption

A small battery serves as a power source for each node. Therefore, energy is a scarce resource in the Internet of Things (IoTs). Consequently, we measure the power consumption of the proposed scheme at the system and node levels. Contiki Powertrace \cite{corporaton2006tmote} is used to measure power consumption determined by various components of a system. The power consumption and energy usage are measured using normal values, and the Tmote sky parameter values are shown in Table 1. It shows different values based on the Tmote sky condition, such as "MCU on, Radio Tx," "MCU on, Radio Rx," "MCU on, Radio off," "MCU idle, Radio off," and "MCU standby." CPU time occurs when the MCU is "on" while the radio is "off." Transmission and reception time refer to the time when the MCU is turned on, and the radio is transmitting and receiving data, respectively. Our study explores the energy usage of a 6LoWPAN network during its duty cycle (when the ratio is generally off). In a duty cycle network, we estimate the network energy consumption of the RPL network using 6Mapper and an intrusion detection mechanism. Contiki employs a duty cycle MAC protocol. We use eight wakeups\/s that is by the default setting of contikiMAC. We took 8, 16, 32, 48, and 64 Tmote sky nodes to evaluate the network performance for experimental work.

Figure 15 shows the energy consumption of the entire network node over a 15-minute period. It is calculated as follows:

Table 1: Tmote sky operating conditions

Parameter	Min.	Normal	Max.	Unit
Voltage		2.1	3.6	V
Temp (Free Air)	-40		85	C
MCU on, Radio Tx		19.5	21	mA
MCU on, Radio Rx		21.8	23	mA
MCU on, Radio off		1800	2400	µA
MCU idle, Radio off		54.5	1200	µA
MCU Standby		5.1	21.0	µA

Energy=Transmit*(MCU on, Radio Tx)+ listen * (MCU on, Radio Rx) + CPU *

(MCU on, Radio off) +LPM * (MCU idle, Radio off) * \frac{Voltage}{Data}
* WakeupsPerSecond

From the network energy usage, the average power can be calculate as:

Where the Power is in mW; Energy is in mJ, and time in ‘s’.

Figure 15 & 16 depicts the total energy consumption of the network and the average power consumption. For example, when the number of nodes is 8 and 16, as shown in Fig a, both RPL-1 and RPL-2 consume about the same amount of energy. However, RPL-1 performs better when the number of nodes is 32, 48, and 64 and has similar behaviors in average power consumption. As shown in Fig. b, the proposed system performs better as the number of nodes increases.

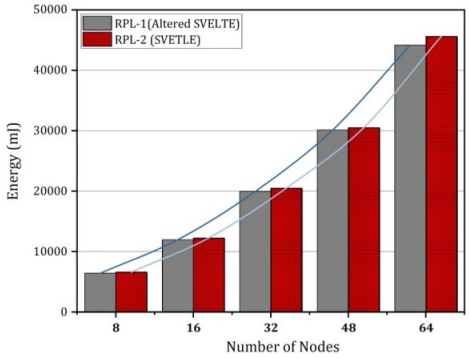

Fig. 15: Network Energy Consumption

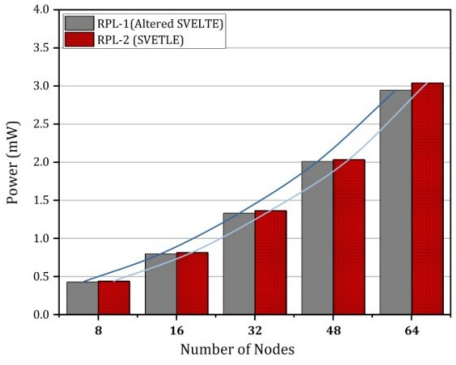

Fig. 16: Average Power Consumption

Table 2: Memory overhead

Configuration	Total ROM (byte)	SVELTE Over-head(byte)	Altered SVELTE Over-head(byte)
6Mapper Client	44264	1414	1397
Packet loss Improvement	43264	0122	0108
6Mapper Server (8 nodes, 1 neighbor)	46798	3846	3712
6Mapper Server (16 nodes, 1 neighbor)	46800	4152	4023
6Mapper Server (16 nodes, 8 neighbor)	46924	4724	4687
6Mapper Server (32 nodes, 8 neighbor)	46948	5312	5234

5.8 Memory consumption

In experimental work, memory consumption is also taken into account. For data processing, each node has a small amount of memory. Memory overhead is calculated for both SVELTE and altered SVELTE, as given in Table 2. It demonstrates that the ROM requirements for each SVELTE and altered SVELTE module vary depending on the Contiki system. Compared to SVELTE in Contiki, Altered SVELTE has a low memory overhead, as seen in the Table \ref{tab-Memory}. We looked at four different scenarios for overhead memory observation and altered SVELTE outperformed in all of them.

To prove that we have enacted the same system as the original SVELTE for testing and comparing the results, Figure 17 shows the true positive rate of the SVELTE IDS for a sinkhole attack as calculated by the original authors and by us after running for ten minutes. It is observed that our re-enactment is faithful to the original and is well within the margin of error.

It can be observed from the Fig.s 9 - 13 that our modified version of the SVELTE algorithm performs marginally better in most situations. For example in Fig. 8 the graph for the true positive rate a visible trend is present. The difference between the performance of both algorithms starts out negligible at first but it increases to nearly 20 percent when the network size increases from 8 to 16 nodes. The trend continues as such until we

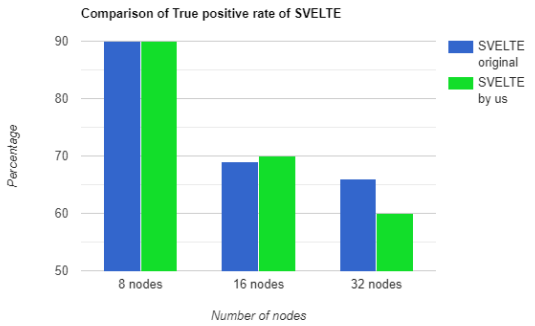

Fig. 17: Comparison of true positive rate of original and re-enacted SEVLTE

reach the 32 nodes value where the difference increases slightly (around 22 percent) and then the difference increases once again for the 64 nodes configuration reaching almost 30 percent.

In summary we can say that the disparity between the performance of the original and the modified algorithm increases when the network size increases. This behaviour can also be observed in the rest of the graphs especially in Fig. 10 (false negative rate) and Fig. 13 (accuracy). The difference is negligible for the 8 node configuration, but it increases to 20 percent for 16 and 32 node configurations and the disparity increases further for the 48 and 64 node configuration. In Fig. 13 (accuracy comparison) we can observe that:

- For 16 node configuration, the algorithm performed approximately 10% better.
- For 32 node configuration, the algorithm performed approximately 12% better.
- For 48 node configuration, the algorithm performed approximately 13% better.
- For 64 node configuration, the algorithm performed approximately 14% better.

The reason for such behaviour is the 'valid inconsistency' described in section 3. The original SVELTE is very susceptible to fall for the valid inconsistencies, and the probability for the valid inconsistency to occur increases with the size of the network. That is why the disparity increases from 20 percent to nearly 30 percent because the performance of the original SVELTE is affected more in comparison to the modified algorithm which was introduced to reduce the impact of the valid inconsistencies. The extra condition introduced in the algorithm serves a dual purpose; it tries to decrease the false positive rate and

improve the true positive rate at the same time. The valid inconsistency will not be flagged as a real malicious node because it will not trigger the second condition of the algorithm due to the fact that the difference in rank would be within normal limits because it occurred due to normal operation while a real malicious node will trigger it because the rank disparity will be too large to be considered as naturally occurring.

In Fig.s 9 and 12 we can observe that the difference in performance is non existent for two configurations, both the 8 node configuration and the 64 node configuration have no difference in performance for the SVELTE and altered SVELTE algorithms. It can also be seen that the difference between the performance is smaller for the 48 node configuration. This behaviour can be explained by the size of the network. From the results it can be seen that this approach for detecting the rank attack is not very scale-able, it performs very well on smaller networks but its performance drops significantly when the number of nodes in the network increase. Since our algorithm uses the original as a base it suffers from the same drawback of not being scale-able. The culmination of this phenomenon is reflected in the graph at the value for sixty four nodes where the value for both the original and altered algorithms is the same.

6. Conclusion and Future Work

We have modified the algorithm of SVELTE for the detection of the sinkhole/rank attack. Through the results obtained from our experiments we have seen that the results obtained for SVELTE correspond with the results they have shown in their research. Furthermore we have seen that with the modifications to their algorithm that have been introduced by us, the performance of the algorithm has been increased. That is characterised by the increase in both the true positive rate and reduction in the false positive rate.The system was run with the same configuration as was the original SVELTE, meaning that the connections were lossy, duty cycling was on and the border router was always in listening mode. We did this in order to better compare with them and have a valid comparison.

It is concluded that the modification of the algorithm has been very beneficial. It is however completely possible for this algorithm to improve further. It is regrettable that we were not able to improve other aspects of the IDS due to several reasons. The number one reason is the lack of documentation on Contiki OS and cooja which severely hampered our ability to understand the implementation of the system level code and also reduced the chances for making further meaningful changes. We are very grateful to the authors of SVELTE for making their project open source and facilitating us in our research.

The IoT market is rapidly expanding and with new features there will be new flaws to exploit. The research in this field is only going to increase from here on out as IoT becomes more popular with the general public and with this project we hope to have contributed in this noble endeavor. \textbf{In the future this project can be enhanced by incorporating defence against more attacks such as wormhole attack. It might also be fruitful to look into methods to increase the scalability of the IDS.

6. References

[1] Shelby, Z., & Bormann, C. (2011). 6LoWPAN: The wireless embedded Internet (Vol. 43). John Wiley & Sons.

[2] Kothmayr, T., Schmitt, C., Hu, W., Brünig, M., & Carle, G. (2013). DTLS based security and two-way authentication for the Internet of Things. Ad Hoc Networks, 11(8), 2710-2723.

[3] Raza, S., Duquennoy, S., Chung, T., Yazar, D., Voigt, T., & Roedig, U. (2011, June). Securing communication in 6LoWPAN with compressed IPsec. In 2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS) (pp. 1-8). IEEE.

[4] Brandt, A., Designs, S., Hui, J., Kelsey, R., Levis, P., Pister, K., & Alexander, R. (2012). Internet Engineering Task Force (IETF) T. Winter, Ed. Request for Comments: 6550 Category: Standards Track P. Thubert, Ed.

[5] Ammar, M., Russello, G., & Crispo, B. (2018). Internet of Things: A survey on the security of IoT frameworks. Journal of Information Security and Applications, 38, 8-27.

[6] Raza, S., Wallgren, L., & Voigt, T. (2013). SVELTE: Real-time intrusion detection in the Internet of Things. Ad hoc networks, 11(8), 2661-2674.

[7] 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.

[8] Dvir, A., & Buttyan, L. (2011, October). VeRA-version number and rank authentication in RPL. In 2011 IEEE Eighth International Conference on Mobile Ad-Hoc and Sensor Systems (pp. 709-714). IEEE.

[9] Wallgren, L., Raza, S., & Voigt, T. (2013). Routing attacks and countermeasures in the RPL-based internet of things. International Journal of Distributed Sensor Networks, 9(8), 794326.

[10] Raoof, A., Matrawy, A., & Lung, C. H. (2018). Routing attacks and mitigation methods for RPL-based Internet of Things. IEEE Communications Surveys & Tutorials, 21(2), 1582-1606.

[11] 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), 429-438.

[12] Semedo, F., Moradpoor, N., & Rafiq, M. (2018, September). Vulnerability assessment of objective function of RPL protocol for Internet of Things. In Proceedings of the 11th International Conference on Security of Information and Networks (pp. 1-6).

[13] Stephen, R., & Arockiam, L. (2018, November). E2V: Techniques for detecting and mitigating rank inconsistency attack (RInA) in RPL based Internet of Things. In Journal of Physics: Conference Series (Vol. 1142, No. 1, p. 012009). IOP Publishing.

[14] Ashraf, J., Keshk, M., Moustafa, N., Abdel-Basset, M., Khurshid, H., Bakhshi, A. D., & Mostafa, R. R. (2021). IoTBoT-IDS: A Novel Statistical Learning-enabled Botnet Detection Framework for Protecting Networks of Smart Cities. Sustainable Cities and Society, 103041.

[15] Kalyani, S., & Vydeki, D. (2018, September). Survey of Rank Attack Detection Algorithms in Internet of Things. In 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI) (pp. 2136-2141). IEEE.

[16] Kaliyar, P., Jaballah, W. B., Conti, M., & Lal, C. (2020). LiDL: Localization with early detection of sybil and wormhole attacks in IoT Networks. Computers & Security, 94, 101849.

[17] Airehrour, D., Gutierrez, J., & Ray, S. K. (2018). A trust-based defence scheme for mitigating blackhole and selective forwarding attacks in the RPL routing protocol. Journal of Telecommunications and the Digital Economy, 6(1), 41-49.

[18] Ma, G., Li, X., Pei, Q., & Li, Z. (2017, October). A security routing protocol for Internet of Things based on RPL. In 2017 International Conference on Networking and Network Applications (NaNA) (pp. 209-213). IEEE.

[19] Kamgueu, P. O., Nataf, E., & Ndie, T. D. (2018). Survey on RPL enhancements: a focus on topology, security and mobility. Computer Communications, 120, 10-21.

[20] Conti, M., Kaliyar, P., Rabbani, M. M., & Ranise, S. (2018, October). SPLIT: A Secure and Scalable RPL routing protocol for Internet of Things. In 2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob) (pp. 1-8). IEEE.

[21] Verma, A., & Ranga, V. (2020). Mitigation of DIS flooding attacks in RPL-based 6LoWPAN networks. Transactions on emerging telecommunications technologies, 31(2), e3802.

[22] 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.

[23] Taghizadeh, S., Bobarshad, H., & Elbiaze, H. (2018). CLRPL: context-aware and load balancing RPL for IoT networks under heavy and highly dynamic load. IEEE Access, 6, 23277-23291.

[24] Taghanaki, S. R., Jamshidi, K., & Bohlooli, A. (2019, October). DEEM: A Decentralized and Energy Efficient Method for detecting sinkhole attacks on the internet of things. In 2019 9th International Conference on Computer and Knowledge Engineering (ICCKE) (pp. 325-330). IEEE.

[25] Sharma, M., Elmiligi, H., Gebali, F., & Verma, A. (2019, October). Simulating attacks for rpl and generating multi-class dataset for supervised machine learning. In 2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON) (pp. 0020-0026). IEEE.

[26] de Souza, C. A., Westphall, C. B., Machado, R. B., Sobral, J. B. M., & dos Santos Vieira, G. (2020). Hybrid approach to intrusion detection in fog-based IoT environments. Computer Networks, 180, 107417.

[27] A. Dunkels, "Contiki: The open source os for the internet of things."

[28] Osterlind, F., Dunkels, A., Eriksson, J., Finne, N., & Voigt, T. (2006, November). Cross-level sensor network simulation with cooja. In Proceedings. 2006 31st IEEE conference on local computer networks (pp. 641-648). IEEE.

[29] Österlind, F. (2011). Improving low-power wireless protocols with timing-accurate simulation (Doctoral dissertation, Acta Universitatis Upsaliensis).

[30] Polastre, J., Szewczyk, R., & Culler, D. (2005, April). Telos: Enabling ultra-low power wireless research. In IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005. (pp. 364-369). IEEE.

[31] M. Corporaton, "Tmote sky: Datasheet," 2006.

[32] Younus, M. U. (2018). Analysis of the impact of different parameter settings on wireless sensor network lifetime. Int. J. Adv. Comput. Sci. Appl, 9(3), 16-21.