Computers & Security 94 (2020) 101849

Contents lists available at ScienceDirect

Computers & Security

journal homepage: www.elsevier.com/locate/cose

LiDL: Localization with early detection of sybil and wormhole attacks in IoT Networks

Pallavi Kaliyar[a], Wafa Ben Jaballah[b], Mauro Conti[a], Chhagan Lal[c,*]

[a] University of Padova, Padova, Italy
[b] Thales, France
[c] Simula Research Laboratory, Norway

ARTICLE INFO

Article history:
Received 5 September 2019
Revised 17 April 2020
Accepted 20 April 2020
Available online 28 April 2020

Keywords:
RPL
Internet of Things
Security
Sybil attack
Wormhole attack
IPv6
6LoWPAN

ABSTRACT

The Internet of Things (IoT) is recognized as a disruptive innovation that has been led by industry leaders and researchers. IoT promises to improve our daily life based on smart objects interacting with each other, and that can be connected to the Internet. Building a security framework into this new paradigm is a significant technical challenge today. It is mainly due to the low-cost and resource-constrained nature of IoT devices. In most of the IoT application scenarios, the routing is done by the de-facto standard protocol called routing protocol for low power and lossy networks (RPL). The use of RPL is suitable due to its energy-efficient schemes, availability of secure and multiple communication modes, and adaptivity to work in various IoT network scenarios. Hence, many researchers are now focusing on RPL related security issues. To this end, our work provides a concise description of two major threats to RPL called sybil and wormhole attacks. Moreover, we propose two solutions to detect these attacks in RPL-based IoT networks. Specifically, our proposed techniques exploit the concept of Highest Rank Common Ancestor (HRCA) to find a common ancestor with the highest rank among all the ancestors that a pair of nodes have in the target network tree. Our two detection algorithms not only detect an ongoing attack but also localizes the position of the adversary in the network. Thus, it makes the mitigation process lightweight and fast. We implement the two approaches in Cooja, the Contiki network emulator. The results obtained from our experiments demonstrate the feasibility of the proposals concerning true positive rate, detection time, packet loss ratio, memory consumption, and network overhead. Our techniques show promising to cover more complex scenarios in the future.

1. Introduction

The proliferation of the IoT is a new wave of innovation with recent forecasts suggesting massive deployment of several IoT devices to reach 24 billion in 2020 Online. In general, an IoT framework could be defined as the use of heterogeneous technologies, systems, and TCP/IP protocols, with the growing paradigm of device-to-device communications and the contextual environment. IoT networks are considered as an example of Low-Power and Lossy Network (LLN), which consist of tiny, heterogeneous devices with limited power, memory, and processing resources. These specific networks have been used in a broad scope of real-world application areas such as smart home, health care, urban sensor networks, water management, smart cities, and industrial smart grid systems (Miettinen et al., 2017). However, the use of IoT devices imposes strict resource constraints regarding energy and memory as well as considering the high loss rate of the communication links in these networks (Miettinen et al., 2018).

Routing in IoT network facilitates the connection of network elements in different applications. So far, the only standardized protocol available for IoT is the RPL (Winter et al., 2012) protocol. However, RPL provides little security against different routing attacks (Hwang, 2015; Kharrufa et al., 2019; Raoof et al., 2018). In particular, the design flaws of secure network formation processes of the standard RPL exposes the network to various attacks (Ghaleb et al., 2019) (Tahir et al., 2018) such as sybil, blackhole (Glissa et al., 2016), rank (Sahay et al., 2018), and wormhole (Pongle and Chavan, 2015b). These attacks hinder the enforcement of basic security services such as confidentiality, data integrity, authenticity, and access control (Le et al., 2013), and could be exploited by an adversary to run more powerful attacks. There-

* Corresponding author.
E-mail addresses: pallavi@math.unipd.it (P. Kaliyar), wafa.benjaballah@thalesgroup.com (W.B. Jaballah), conti@math.unipd.it (M. Conti), chhagan@simula.no (C. Lal).

https://doi.org/10.1016/j.cose.2020.101849
0167-4048/© 2020 Elsevier Ltd. All rights reserved.

fore, data routing in such networks is considered as one of the weakest links in the adoption of these networks in real-world applications (Kharrufa et al., 2019).

1.1. Motivation and ccontribution

An adversary could easily capture, tamper, or even destroy devices in an IoT network. It is due to the lack of physical protection and tamper resistance in LLNs. Although RPL provides confidentiality by using simple cryptographic mechanisms that ensure authenticity and integrity of its control messages, a legitimate node captured by an adversary can still eavesdrop, duplicate, or alter packets, leading to significant problems such as power outages in smart grid networks or widespread system failures. Even though many papers have extensively addressed the impact of attacks in traditional networks such as vehicular networks, wireless sensor networks (Tomi and McCann, 2017), and mobile ad hoc networks (Abusalah et al., 2008), they could not be applied directly to the IoT. It is because of IoT networks' specific characteristics such as device heterogeneity, resource constraints, few standard protocols, context-dependence, and cross-device dependencies. Therefore, researchers have started working on the design and development of novel solutions that can be efficiently used in the IoT networks to improve their communication reliability (Pu and Hajjar, 2018; Thulasiraman and Wang, 2019a), and security (Raoof et al., 2018; Raza and Magnsson, 2019). Although sybil and wormhole attacks have been well investigated in the literature for traditional networks, there are still no suitable solutions that could fully address these attacks by considering the unique characteristics of IoT and RPL protocol.

In this paper, we propose novel techniques to detect two of the most destructive attacks known as sybil and wormhole in RPL based IoT networks. Our proposed techniques exploit the concept of Highest Rank Common Ancestor (HRCA), which aims to find a common ancestor with the highest rank among all the ancestors that a pair of nodes have in the target network tree. As HRCA being a tree-based approach and the RPL routing protocol also logically creates a tree topology for routing, we show that the HRCA approach becomes a perfect match for securing large-scale RPL based IoT scenarios from the sybil and wormhole attacks. Our proposals not only detect an ongoing attack but also localizes the position of the attacker in the network. Thus, it makes the mitigation process lightweight and fast. In particular, several network nodes periodically execute our detection algorithms on their specified cached entries. The aim of these periodic executions on the root and specifically identified intermediate nodes in the RPL's Destination Oriented Directed Acyclic Graph (DODAG) tree (Winter et al., 2012) is to discover anomalies, which provides an indication of ongoing sybil and wormhole in the network. Once an anomaly is detected, the root node takes adequate actions to remove the identified malicious node(s) from the network. To this end, the key contributions of this paper are as follows.

- We propose two new techniques to detect sybil and wormhole attacks in RPL based IoT networks. Both solutions make efficient use of the HRCA scheme for attack detection and localization. The localization process provides rapid mitigation of an ongoing attack with low overhead.
- We formulate a novel mathematical model to perform a simple yet effective detection of the wormhole attack in the IoT network. The proposed model uses minimum additional information for the detection process. It dramatically reduces the attack detection overhead when executed in the non-storing routing mode of RPL, which is one of the most used routing modes in the real-world applications of IoT.
- We fully implement our proposed techniques in Cooja, the Contiki network emulator, which also works as an emulator. The correctness of our techniques has been shown through result evaluations concerning the relevant metrics such as true positive rate, packet loss ratio, memory consumption, and attack detection overhead in various IoT scenarios with different network load and size. Finally, we make available the open-source implementation of both the approaches on Git repository[1].

1.2. Organization

The organization of the rest of this paper is as follows. Section 2 provides a brief overview of the RPL protocol, with a description of the sybil and wormhole attacks followed by the related work concerning the state-of-the-art attacks and solutions in RPL-based IoT networks. In Section 3, we discuss the system and adversary models. In Section 4, we present the details of our proposed solutions. The simulation setup details and performance evaluation are presented in Section 5. Finally, Section 6 concludes our work.

2. Background and related Work

In this section, first, we present a brief overview of the RPL protocol and working methodology of sybil and wormhole attacks. Then, we will discuss state-of-the-art attacks and solutions on data communication in RPL-based IoT networks along with their limitations to show how our proposed work improves the state-of-the-art.

2.1. Routing protocol for low power and lossy networks (RPL)

RPL (Winter et al., 2012) is based on a virtual routing topology called Destination-Oriented Directed Acyclic Graph (DODAG) on top of the underlying physical topology. The DODAG is a directed graph-oriented towards a root node without loops. In DODAG, each node has multiple parents towards the root. However, a node selects only a preferred parent based on the routing metric and objective function. The parent node will be used for forwarding data packets. The structure of DODAG supports multipoint-to-point communication in RPL, which provides communication from the nodes to the root. Each node receives a rank ID that depends on its distance from the root. The creation and the maintenance of the DODAG are done through ICMPv6 control packets known as DODAG Information Objects (DIO). Each node in RPL disseminates DIO messages containing the link, node metrics, and an objective function that is used by each node to select the preferred parent among its neighbors. The node metrics contain values such as the expected transmission count (ETX) and the residual energy. To maintain the DODAG, the DIO packets are rebroadcast by each node based on the Trickle algorithm (Levis et al., 2011). Another control packet known as DODAG information solicitation (DIS) packet is triggered when a new node wants to join an existing DODAG. The DODAG node receiving the DIS will send a DIO packet.

RPL supports different types of communications such as point-to-multipoint, point-to-point, and multipoint-to-point, and it provides two modes known as storing and non-storing. In storing mode, which is based on table-driven routing, the non-root nodes create and maintain a routing table for all their descendant nodes. While in the non-storing mode, which is based on source routing, only the root node creates and maintains the routing information about all the network nodes. The creation and maintenance of routing tables in both the RPL modes are done with the help

[1] https://github.com/pallavikaliyar/LiDL.

of RPL's control packets called Destination Advertisement Object (DAO). Each non-root node sent these packets towards the root to announce itself as a possible destination to the root. During their way towards the root, these packets pass through their ancestors, thus establishing "downwards" routes along the way. The full implementation details of RPL and its design goals are out of the scope of this paper. Hence, we direct the interested readers to more comprehensive literature on RPL protocol, which is given in (Kim et al., 2017; Winter et al., 2012).

2.2. Sybil and wormhole attacks in IoT

In this section, we present an overview of sybil and wormhole attacks concerning the IoT networks.

- *Sybil Attack* - sybil attack consists of manipulating network devices to harm the data communication process (Evangelista et al., 2016). In particular, a sybil node illegitimately claims multiple identities and executes this attack to achieve various advantages, such as usage of unauthorized resources, harming the confidentiality of users, disseminating false control information, and publishing private information of network users. Since a sybil node has legitimate information (code and cryptographic material), it may participate in the network operations in the same way as a non-compromised node. Hence, sybil nodes can launch a variety of attacks and become even more critical.
- *Wormhole attack* - In wormhole attack (Khabbazian et al., 2009a), an attacker first *tunnels* the packets that it receives at one point in the network to another point in the network. Later, it replays these packets into the network from that point. This attack aims to give the attacker advantages relative to other nodes, and the attacker could exploit it in a variety of ways, such as to misdirect traffic of other nodes or privilege its own traffic. The success of the wormhole attack is independent of the fact that the network communication is confidential and authentic, and it uses cryptographic keys to perform secure communications.

2.3. State-of-the-art on security in RPL-based IoT networks

There are very few efforts in the past that addresses the security issues in RPL protocol against various routing attacks, such as rank, sybil, and wormhole attacks. The authors in (Raza et al., 2013) propose an Intrusion Detection System (IDS) called SVELTE. In SVELTE, the IDS is installed at the root and a set of network nodes to analyze the traffic to identify anomalies, which helps in detecting various routing attacks. However, the network scalability and routing overhead remains the biggest drawback of the SVELTE scheme. Additionally, unlike LiDL, it does not provide attack localization, and it does not evaluate SVELTE against sybil and wormhole attacks.

In (Zhang et al., 2014), sybil attacks are categorized into three different forms. Based upon the geographical and social scope of the attacking nodes, the countermeasures proposed are social graph-based sybil Detection (SGSD), behavioral classifies-based sybil Detection (BCSD), and mobile sybil detection. These techniques are quite relevant in mitigating most of the sybil attacks but faces specific challenges such as the lack of global social and historical graphs, location information of mobile devices, the unauthorized study of users patterns, and the difference in security capability of different devices. Additionally, the authors do not implement and evaluate their proposed scheme, which questions its correctness and deployment feasibility. SecTrust-RPL, a secure RPL protocol proposed in (Airehrour et al., 2018), is based on a trust-based mechanism that stands out from all the earlier works. It mainly relies on cryptographic techniques for securing IoT networks. It is effective against both the rank attack and the sybil attack by the adoption of a trustee-trust or relationships for all its nodes. However, the proper integration of recouped nodes back in the network remains a problem in this approach. Recently, authors in (Thulasiraman and Wang, 2019a) propose a trust-based security architecture for RPL in which parameters such as nonce value, network white-list, and timestamp are used to ensure the integrity of the control messages in transit. Authors in (Conti et al., 2018) exploit the use of a lightweight remote attestation scheme to improve the security of the data communication process in RPL-based IoT networks.

Wormhole attacks, as described in (Perazzo et al., 2018), are one of the sophisticated attacks to detect. The attack becomes more critical when it is launched in combination with other attacks like eavesdropping, selective forwarding, and blackhole. In literature, there are effective proposals to tackle wormhole attack in Wireless Sensor Networks (WSNs) (Khabbazian et al., 2009b; Pongle and Chavan, 2015a). Wormhole attack on short-path length routing protocol is considered in (Khabbazian et al., 2009b) for wireless ad hoc networks, where they propose a timing-based detection. The approach considers packet transmission time (PTT) to detect wormhole attacks, and it is an improvement to its predecessors. In (Pongle and Chavan, 2015a), the authors make use of neighbor information stored by nodes in an RPL based network to detect wormhole tunnels, and the root provides the validation. The monitoring algorithm based on RSSI eventually discovers the wormhole. The main drawbacks of the mechanism consist of the cost overhead and memory allocation for neighbors. Also, the scalability of the proposed approach remains questionable as it is only evaluated in small networks (i.e., 24 nodes). In particular, the work concerning the impact and countermeasures of a wormhole in RPL-based IoT networks is still in its early stages. Therefore, it requires significant attention from the research community.

In (Ahsan et al., 2017), the authors propose a technique to detect and mitigate wormhole attack in LLNs. The technique uses two special types of motes, namely, Area Border Router (ABS) and Sensing Aware Nodes (SAN), to detect wormhole attacks. The proposed approach mainly monitors the signal strength of nodes, and if the distance between two nodes is found greater than the default distance, than an attack is detected. Both types of special motes act as a backup for each other, such that if one type of mote fails, others will detect the attack. The results are good, and it does not require excessive power, which is quite useful in a resource-constrained environment. However, the main disadvantage is that the detection of approach is directly proportional to the special types of motes used in the network, which makes it less effective in small IoT networks.

3. Problem formulation

In this section, we present the problem formulation which includes system and adversary modelling. Later, we use these models to describe and evaluate our proposed security solutions. Table 1 includes the list of sybmbols that we used to describe LiDL's working methodology.

3.1. System model formulation

In our work, the system model has the following properties.

- The target IoT network consists of a set $N = \{N_1, N_2, \dots N_m\}$ of size m resource-constrained devices (i.e., sensors and actuators) called nodes, and $R = \{R_1, R_2, \dots R_n\}$ resourceful nodes called LLN border routers (LBR). The nodes in set N exhibit the same amount of resources. However, the functionalities of these

Table 1
Symbol table.

Symbol	Definition
N	Set of nodes in network
m	Number of IoT nodes in N
S_{tab}^{det}	sybil Detection table
W_{tab}^{det}	wormhole Detection table
$< S_i, D_i >$	i^{th} Source-Destination pair
I	Identity of a benign node
SID	Sybil identity
G	Communication Graph (DODAG)
wh	Wormhole attack
Γ	Attack intensity
S_{id}^{i}	i^{th} Source node ID
D_{id}^{i}	i^{th} Destination node ID
$Prev_{id}^{hop}$	Previous hop Node ID
F_{id}^{hop}	Forwarding next hop Node ID
S_{rank}^{i}	i^{th} Source node Rank
D_{rank}^{i}	i^{th} Destination node Rank
T_{hops}^{i}	Total path-length (in hops) of i^{th} Source-Destination pair
R_{node}	Root Node
D_{pkt}	Data Packet
D_{pkt}^{new}	First Data Packet from a new Data session
N	Random node in DODAG
NL_{node}	Non leaf Node
t_i	time at i^{th} instance
P_{timer}	Network level periodic timer
N_{child}	Child of random node N
L_{node}	Leaf Node
N_{rank}	Rank of random node N
V_{hop}^{count}	Verified hop count of particular Source-Destination pair
W_{det}^{pkt}	wormhole detection packet

nodes could differ from each other depending upon the type of sensors (e.g., light, pressure, and temperature) that are installed in them. Based on the type of IoT application running on top of the networking infrastructure, the nodes with similar functionalities could be grouped to create a multicast group in the network. For routing, the RPL protocol supporting storing and non-storing modes has been configured at the network layer of all the nodes. The other layers (e.g., application, link layer, and physical) have been configured using the standard protocol stack of IoT (Al-Fuqaha et al., 2015).

- At the initialization phase, all nodes are deployed randomly in the given network area, and the RPL protocol creates a virtual DODAG over the physical topology. As mentioned before that, along with the m network nodes, the IoT network also contains n LBR nodes, which are considered as resourceful nodes and act as a root for a DODAG. We assume the *rank* value of the LBR nodes is set to one, and the rank of the other nodes is calculated based on their distance from the LBR. Depending upon the network configuration and requirements of IoT applications running on the top, a network could have more than one virtual DODAG instances over the same physical network, each represented by a DODAG ID (DID_i). To ensure that our system model covers a broad range of IoT domains, it provides support for all types of communications (i.e., multipoint-to-point, point-to-multipoint, and point-to-point) in both the RPL modes.
- Our proposed security solutions define the following additional or enhanced data structures at nodes (i.e., root or non-leaf) in the target network.
  - *Sybil Detection Table (S_{tab}^{det}):* Irrespective to RPL routing mode (i.e., storing or non-storing), each non-leaf node in the network stores S_{tab}^{det}, which consists of 2-tuple $< S_{id}^{i}, Prev_{id}^{hop} >$. Here, S_{id}^{i} is the ID of the i^{th} network node, which is also acting as a source node, and $Prev_{id}^{hop}$ is the ID of the previous forwarding node of a received data packet. For each received packet at a non-leaf node, the receiving node will check the S_{tab}^{det} entries based on our proposed sybil attack detection technique. At any point of time, the number of entries in S_{tab}^{det} will not exceed the total number of data sessions in the network, otherwise the first-in-first-out (FIFO) replacement algorithm is used to store a new entry in the table.
  - *Wormhole Detection Table (W_{tab}^{det}):* Depending upon the RPL routing mode, our wormhole detection algorithm operates differently on the W_{tab}^{det}. The selected network nodes execute our wormhole detection algorithm on the W_{tab}^{det}, which consists of the 5-tuple $< S_{id}^{i}, D_{id}^{i}, S_{rank}^{i}, D_{rank}^{i}, T_{hops}^{i} >$, here i is the i^{th} node in the network, S_{id}^{i} and D_{id}^{i} are the source and destination IDs, S_{rank}^{i} and D_{rank}^{i} are the rank of source and destination nodes, and T_{hops}^{i} is the hop count between the $< S_{id}^{i}, D_{id}^{i} >$ source-destination pair. In non-storing mode, each data packet passes through root node, due to which the root node has the global topology view. Thus, it can easily gather the information required to fill all the entries in W_{tab}^{det}. In case of storing mode, we use a different approach to perform the wormhole detection, which we describe in detail in Section 4.3.

3.2. Adversary model formulation

The rapid increase in the use of IoT networks for a vast array of user-centric applications makes these networks a high-profit target for attackers. In our target IoT scenarios, the attackers are assumed to have the following characteristics.

- We are concerned about malware infection, i.e., predominantly Software Adversary (*SA*). In our adversary model, we assume that a *SA* is powerful enough to capture and manipulate one or more legitimate nodes remotely. Once those nodes are infected with malicious code, then the *SA* can disrupt data communication functionality by launching sybil or wormhole attacks.
- The primary goal of the *SA* is to disrupt network routing protocol and interfere with ongoing communications. In the case of the sybil attack, the malicious node acquires the identity of a legitimate node and try to disrupt the whole routing process. While in the case of the wormhole attack, the malicious nodes can create a tunnel between any two nodes to mislead network traffic by causing more delay or loops or perform selective packet drop in the network. This situation can lead the sender node to transmit a single packet multiple times. This scenario causes energy loss since previously sent packets to roam in the network from one node to another node until they reach its destination or outlives.
- In this paper, we primarily focus on the mitigation of adversarial scenarios when single, or multiple malicious nodes try to make multiple fake IDs of the legitimate nodes or create tunnels. The aim is to cause communication delay and disrupt network topology view in routing/forwarding tables. We assume that the adversary will not interfere with the proper functioning of a network, such as altering data packets, destroying network devices, and tampering with the cryptographic key management operations. An IDS can detect such activities. Hence it puts the attacker at risk of being detected (Kim et al., 2017).

Next, we mathematically formulate the two attacks that our proposed approaches identify and mitigate in a target IoT network scenario.

- Wormhole Attack Formulation: We assume that G be a communication graph of target IoT network that consists after the DODAG formation, and wh be a wormhole attack on the G. Thus, G_{wh} will be the resultant DODAG after the attack. Assume

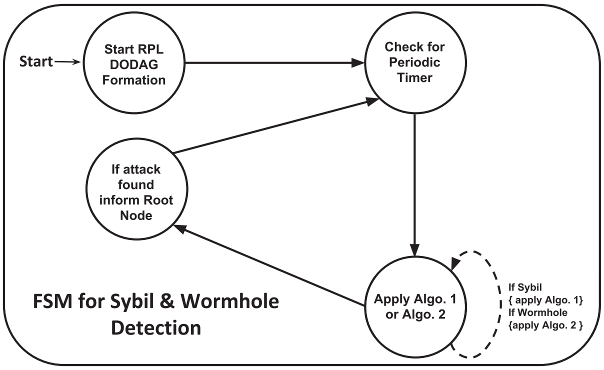

Start
Start RPL DODAG Formation
Check for Periodic Timer
If attack found inform Root Node
Apply Algo. 1 or Algo. 2
If Sybil { apply Algo. 1} If Wormhole {apply Algo. 2 }
FSM for Sybil & Wormhole Detection

Fig. 1. FSM for sybil and wormhole Attack Detection.

that $L(S_i, D_i)$ and $L_{wh}(S_i, D_i)$ are the lengths of the shortest paths between a random source-destination pair $S_i, D_i \in V(G) \cap V(G_{wh})$ on G and G_{wh}, respectively. If $L_{wh}(S_i, D_i) < L(S_i, D_i)$, it indicates that there exists a wormhole attack on G_{wh}. If, $\Gamma_{S_iD_i} = L(S_i, D_i)$ - $L_{wh}(S_i, D_i)$ quantifies the shortened path length of wh between S_i and D_i, then the intensity of wh can be defined as $\Gamma_{S_iD_i} = max\ \{\Gamma_{S_iD_i} | S_i, D_i \in V(G) \cap V(G_{wh})\}$. The intensity of the attack depends on the number of attackers in the network and their positions with respect to the source-destination pair that they target. In particular, when the attack intensity is high, the topological distortion in the target IoT network will be important.

- Sybil Attack Formulation: Let m be the number of nodes in network, and $I = \{ID_1, ID_2, ID_3, \ldots ID_m\}$ be the set of identities of these nodes. An attacker needs to possess one or more valid identities (called sybil identities) to perform the attack. Assume $SI = \{SID_1, SID_2, SID_3 \ldots SID_n\}$ be the identities an attacker possess, where $n < m$, and SID is a sybil identity. The possession of identity could be done by compromising a benign node or by fabricating the identity of a benign node. The fabrication process can be executed by acquiring a set of identities (i.e., SID) in a way that all the SIDs fall in the category of valid identities, i.e., $SID_i \in SI$, and $SID_i \in \{SID_{\min}, SIDmax\}$, where $SID_{\min}$ and $SID_{\max}$ are minimum and maximum valid range of IDs. The ID fabrication might not be possible if there exists a secure communication channel between the communicating nodes, in such cases, the node compromise could be used to launch the sybil attack.

4. LiDL: Sybil and wormhole attack detection approaches

In this section, we first present the working methodology of our proposed security solutions for sybil and wormhole detection. To better understand the functioning of our proposed solutions, later, we use an example of an IoT scenario which consists of benign and malicious nodes.

4.1. Overview

The finite state machine (FSM) model of both the proposed approaches is shown in Fig. 1. Here, we perform the following essential functions.

- *Initial Joining:* Wait for RPL DODAG formation. Once done, all nodes will become part of the network.
- *Verify Periodic timer:* After network formation, a network-level periodic timer is being set.
- *Apply detection algorithm:* Upon each expiry of the periodic timer, one or more nodes are selected to execute our wormhole and sybil detection algorithms. Based on the result of the algorithm's execution, the node(s) that identify misbehavior will trigger an alarm message to the root node.
- *Result:* Based on the alarm message content, the root node will take further actions to mitigate the attack(s) in the provided locality.

4.2. Sybil attack detection

Our sybil attack detection and localization technique use S_{tab}^{det}, which resides at each non-leaf node in the RPL DODAG. The proposed detection approach works implicitly during the S_{tab}^{det}'s creation and maintenance processes, which are as follows. Initially, the S_{tab}^{det} is empty. Whenever an i^{th} non-leaf node receives a data packet, it extracts the source ID (S_{id}^{i}) from the packet and identifies the previous-hop ID ($Prev_{id}^{hop}$), and it checks the existing S_{tab}^{det} entries for a match with the condition $S_{id}^{i} \rightarrow Prev_{id}^{hop}$ (Algorithm 1, lines 2 – 3). In particular, upon reception of a data packet at a non-leaf node, the following functional dependency (FD) is checked:

Algorithm 1 Sybil attack detection process.

```
 1: for each $D_{pkt}$ received at N do
 2:   if $N \equiv NL_{node} \vee N \equiv R_{node}$ then
 3:     Extract $< S_{id}^{i}, Prev_{id}^{hop} >$ from $D_{pkt}$
 4:     if detection process is based on $P_{timer}$ then
 5:       Add a new entry in the $S_{tab}^{det}$
 6:     else
 7:       if $S_{id}^{i} \in S_{tab}^{det}$ then
 8:         Check FD: $S_{id}^{i} \rightarrow Prev_{id}^{hop}$ in $S_{tab}^{det}$
 9:         if FD $\equiv$ TRUE then
10:           Process the $D_{pkt}$ normally
11:         else
12:           Notify $R_{node}$ for sybil detection
13:         end if
14:       else
15:         Add $< S_{id}^{i}, Prev_{id}^{hop} >$ entry in $S_{tab}^{det}$
16:       end if
17:     end if
18:   else
19:     Process the packet normally, i.e., the node is the leaf node
20:   end if
21: end for
22: for each expiry of $P_{timer}$ do
23:   Check FD: $S_{id}^{i} \rightarrow Prev_{id}^{hop}$ in $S_{tab}^{det}$
24:   if FD $\equiv$ TRUE then
25:     Process the $D_{pkt}$ normally
26:   else
27:     Notify $R_{node}$ for sybil attack detection
28:   end if
29: end for
```

FD: $X \rightarrow Y$, it means that the values of X determine the value of Y. In our network scenario, X is the source node (S_{id}^{i}) of the received data packet, and Y is $Prev_{id}^{hop}$ of a data packet at the current node. If any two entries (including the current entry extracted from the received packet) shares the same values of X, then it is mandatory to have the same values of Y to hold the FD to TRUE.

In the case where no entry matching with S_{id}^{i} is found in S_{tab}^{det}, a new entry is created with the information extracted from the received data packet (Algorithm 1, lines 7 and 15). While, if the S_{id}^{i} extracted from received data packet matches with some S_{id}^{i} in S_{tab}^{det} and it satisfies the FD, then the packet is processed normally (Algorithm 1, lines 8 – 10). On the other hand, if a match has been found in the table such that FD is not satisfied, then the node will raise a sybil attack detection alarm by informing the root

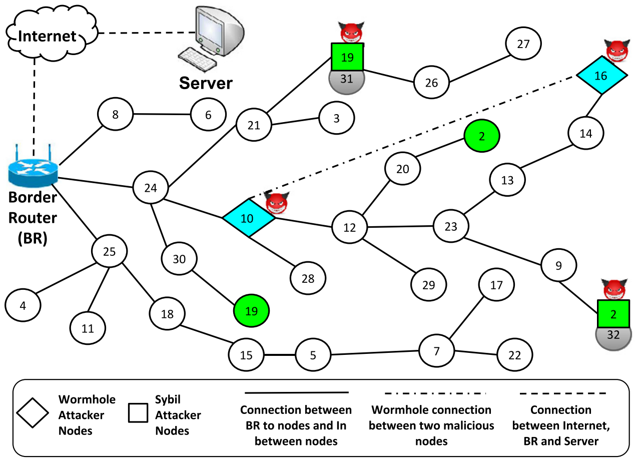

Internet
Server
Border Router (BR)
8
6
21
3
19
31
26
27
16
14
2
20
13
24
10
12
23
9
25
30
28
29
17
4
11
18
19
2
32
15
5
7
22
Wormhole Attacker Nodes
Sybil Attacker Nodes
Connection between BR to nodes and In between nodes
Wormhole connection between two malicious nodes
Connection between Internet, BR and Server

Fig. 2. Sybil and wormhole Attacks in RPL IoT Network.

Table 2
Sybil Example: S^{det}_{tab} at node 24.

S^i_{id}	$Prev^{hop}_{id}$	Remark (this entry is not stored at nodes)
19	21	packet originated by node 31 at t_1
2	10	packet originated by node 2 at t_2
23	10	packet originated by node 23 at t_3
.	.	.
.	.	.
.	.	.
26	21	packet originated by node 26 at t_{x-1}
19	30	packet originated by node 19 at t_x

node about it (Algorithm 1, line 12). The use of the 2-tuple (i.e., $< S^i_{id}, Prev^{hop}_{id} >$) in the matching process helps the localization of the attacker. It is because this particular 2-tuple ensures that the attacker will always be a descendant of the node at which the attack is detected, i.e., the node that informed the root node about the sybil attack. Additionally, this non-leaf node is by default the Highest Rank Common Ancestor (HRCA) of the sybil node and its corresponding genuine node to which it is cloning. To reduce the overhead caused due to processing (i.e., the FD check) of all the received data packet at non-leaf node, each non-leaf node extracts the 2-tuple and stores it in the S^{det}_{tab} (Algorithm 1, lines $4-5$). The non-leaf node only performs the FD check on all the current entries in its S^{det}_{tab} once a predefined periodic network timer (P_{timer}) expires (Algorithm 1, lines $22-29$). The value of P_{timer} should be carefully selected, and it is a function of traffic load, memory at IoT nodes, and attack detection time. Algorithm 1 provides the steps of our proposed sybil attack detection approach, and it includes the detection process with timer and without timer.

The size of the S^{det}_{tab} (i.e., number of entries) greatly affects the detection process and communication overhead. Let us assume that S^{det}_{tab} could store x number of entries at any point in time, and once the table is full, then the new entries will replace the oldest entry in the table. A larger value of x results in longer delays in the searching process performed at each expiry of P_{timer}. Thus, larger x increases the energy consumption (due to high processing power), as well as memory requirements for the resource, constrained IoT nodes. On the other hand, a small value of x could weaken the attack detection capabilities of the proposed detection approach. It is because once the S^{det}_{tab} is full at a node, the entries created by the data packets received from the sybil nodes might get overwritten by the new entries that are caused by the First In First Out (FIFO) replacement algorithm at that node. As depicted in Algorithm 1, line 4, the per received packet FD check-in S^{det}_{tab} can be avoided at the expense of the reduced attack detection probability by performing the FD check operation periodically. In case of the periodic timer (P_{timer}) detection approach, each non-leaf node (including root) will check FD for each entry in the S^{det}_{tab} upon every expiry of the timer.

To better understand the working methodology presented in Algorithm 1, we describe it with the help of Fig. 2, which shows a RPL DODAG constructed on an IoT scenario of 33 nodes. Let us assume that nodes 31 and 32 are sybil attackers and these nodes use the ID of nodes 19 and 2 for packet transmissions. Table 2 depicts an instance of node 24's S^{det}_{tab} at time t_x. Let us suppose that at time t_1 (where $t_x > t_1$) the HRCA (i.e., node with ID 24) of nodes 31 (attacker) and 19 (benign) receives a data packet from node 31, then as per Algorithm 1, node 24 checks if the sybil detection process is periodic or not (line 4 in Algorithm 1). If it is periodic, then it extracts the $< S_{id}, Prev^{hop}_{id} >$ and creates a new entry in its S^{det}_{tab}. The extracted values are $S^i_{id} = 19$ and $Prev^{hop}_{id} = 21$. While, for non periodic detection process, it will check the FD S^i_{id} $S^i_{id} \to Prev^{hop}_{id}$ in S^{det}_{tab} and creates a new entry only if FD is TRUE. Now, if at time t_x, node 24 receives a packet from node 19, it will extract the 2-tuple (i.e., $< 19, 30 >$) and checks for the *FD* in S^{det}_{tab}. Due to the existence of entry $< 19, 21 >$ in 24's S^{det}_{tab}, the *FD* will not satisfy and node 24 suspects an ongoing sybil attack in its sub-tree and notify the same to the root node by sending an alarm message. Upon receiving an alarm message, the root could easily identify the sybil node by looking into the global topology information that it has collected at the time of DODAG creation. Our detection technique works for storing and non-storing modes of RPL, moreover, it supports all the communication modes including point-to-point, point-to-multipoint, and multipoint-to-point. However, the attack localization can only be possible in storing mode with point-to-point and point-to-multipoint communications.

4.3. Wormhole attack detection

In this section, we present the working methodology of our novel wormhole detection technique for RPL-based IoT network scenarios. The proposed technique makes efficient use of HRCA nodes in RPL-DODAG to execute the wormhole detection technique periodically. Each non-leaf node in the DODAG will identify and maintain a list of source-destination pairs for which it is HRCA. In storing mode of RPL, a node stores route for all its descendants with the help of DAO messages. To keep the network overhead low, we use a HRCA identification process which is local to a node, i.e., each node can identify whether it is HRCA for one or more communicating pairs in the network, by using the following approach: *when a node receives a data packet forwarded by one of its child and the forwarding next-hop (F^{hop}_{next}) node towards the destination of the packet is also one of its other children. The current node becomes the HRCA for this communicating pair (Algorithm 2, lines* $1-5$*).*

For the sake of clarity, we refer the reader to Fig. 2, where a data session begins between the nodes 3 and 19, then the data packets will follow the route 3 → 21 → 24 → 30 → 19. In this route, when node 21 forwards the packet to node 24, the node 24 will forward it to node 30. Here, node 24 satisfies the HRCA property (i.e., receiving the packet from one child and forwarding it to another child) for a source-destination pair (i.e., node 3 and node 19). However, our wormhole detection algorithm does not consider a node as HRCA if the next-hop node from the HRCA to the source node and the destination node is the same. For example, in Fig. 2 node 18 is HRCA for $< 5, 22 >$ source-destination pair because the next-hop node to reach to nodes 5 and 22 from node 18 is the same node (i.e., node 15). Thus, our wormhole detection approach does not consider node 18 as HRCA for nodes 5

Algorithm 2 Wormhole attack detection process.

```
1:  for each $D^{new}_{pkt}$ that $N$ receives from a $< S_i, D_i >$ pair do
2:    if $(N \neq D_i) \wedge (N \neq L_{node}) \wedge (F^{hop}_{next} \equiv N_{child} \wedge F^{hop}_{next} \neq Prev^{hop}$ of $D^{new}_{pkt})$ then
3:      $N$ is HRCA for $< S_i, D_i >$ pair
4:    end if
5:  end for
6:  for each $D^{new}_{pkt}$ that $N$ receives from $S_i$ do
7:    if $N \equiv$ HRCA $< S_i, D_i > \vee N \equiv D_i$ then
8:      $N$ add an entry in $W^{det}_{tab}$ with 5-tuple $< S^i_{id}, D^i_{id}, S^i_{rank}, D^i_{rank}, T^i_{hops} >$
9:      if $N \equiv$ HRCA $< S_i, D_i >$ then
10:       $N$ extract values from $D^{new}_{pkt}$ and fill 3-tuple $< S^i_{id}, D^i_{id}, S^i_{rank} >$ in newly added entry $W^{det}_{tab}$
11:     end if
12:     if $N \equiv D_i$ then
13:       $N$ extract values from $D^{new}_{pkt}$ and fill 5-tuple in newly added entry $W^{det}_{tab}$
14:     end if
15:   end if
16: end for
17: for each expiry of $P_{timer}$ do
18:   each $D_i$ creates $W^{pkt}_{det}$ that consists of $S^i_{id}, D^i_{id}, S^i_{rank}, D^i_{rank}, T_{hops}$
19:   $D_i$ sends $W^{pkt}_{det}$ to $R_{node}$ for each $S^i_{id}, D^i_{id}$ entry in its $W^{det}_{tab}$
20:   for each $W^{pkt}_{det}$ received by a $N$ do
21:     if $N \equiv$ HRCA $< S^i_{id}, D^i_{id} >$ then
22:       $N$ calculates $V^{count}_{hop} = (\mid (D^i_{rank} - N_{rank}) \mid + \mid (S^i_{rank} - N_{rank}) \mid)$
23:       if $V^{count}_{hop} \equiv T^i_{hops}$ then
24:         No wormhole tunnel detected between $< S^i_{id}, D^i_{id} >$
25:       else
26:         wormhole tunnel detected between $< S^i_{id}, D^i_{id} >$
27:         $N$ notifies the $R_{node}$ for mitigation of the attack
28:       end if
29:     end if
30:   end for
31: end for
```

and 22. Finally, in the non-storing mode of RPL, only the root will act as HRCA because all the data packets will be routed through it.

In order to detect the wormhole attack in the RPL DODAG, the steps that are given in our proposed Algorithm 2 are executed in the network upon expiry of the network level timer (P_{timer}). We also present the flowchart in Fig. 3, which provides a simplified explanation of the steps taken in our proposed wormhole detection approach. Below, we detail these steps.

1. Step 1: Whenever a destination node (say D^i) or HRCA of $< S_i, D_i >$ pair receives the first data packet (D^{new}_{pkt}) generated by a new data session, the node creates a fresh entry in its wormhole Detection Table (W^{det}_{tab}) by storing the 5-tuple $< S^i_{id}, D^i_{id}, S^i_{rank}, D^i_{rank}, T^i_{hops} >$, here i represents the i^th source-destination pair in the network, and T^i_{hops} is the total number of hops between S^i_{id} and D^i_{id}, which are recorded in the received data packet header (Algorithm 2, lines 7 – 8). To fill values in the fields of this new entry, the destination node extracts $< S^i_{id}, D^i_{id}, S^i_{rank}, T^i_{hops} >$ from the received packet header, and it already knows its own rank, i.e., D_{rank} (Algorithm 2, lines 12 – 14). While the HRCA node can only fill the values in the new entry for $< S^i_{id}, D^i_{id}, S^i_{rank} >$ fields because the packet header does not contain D^i_{rank}, and the T^i_{hops} is not the total path length yet (Algorithm 2, lines 9 – 11).
2. Step 2: After each expiry of the predefined periodic timer (P_{timer}) with time interval say τ (Algorithm 2, line 17), the following events occur.
   - At any non-root node say N, for each entry in its W^{det}_{tab} where the D_i is equal to N, it will create a custom wormhole detection packet (W^{pkt}_{det}) and send it to root (Algorithm 2, lines 18 – 19). The total number of entries in W^{det}_{tab} at N is always less than or equal to the number of source-destination pairs in which it is acting as a destination node. The W^{pkt}_{det} sent by N is lightweight as it only include $S^i_{id}, D^i_{id}, S^i_{rank}, D_{rank}, T^i_{hops}$ information.
   - Each intermediate node that receives W^{pkt}_{det} during its way towards the root, checks whether it is the HRCA for the pair of nodes S^i_{id}, D^i_{id} specified in the W^{pkt}_{det} or not (Algorithm 2, lines 20 – 21). The check is done by searching the W^{det}_{tab} for the pair of nodes S^i_{id}, D^i_{id}. If the node is HRCA node, then

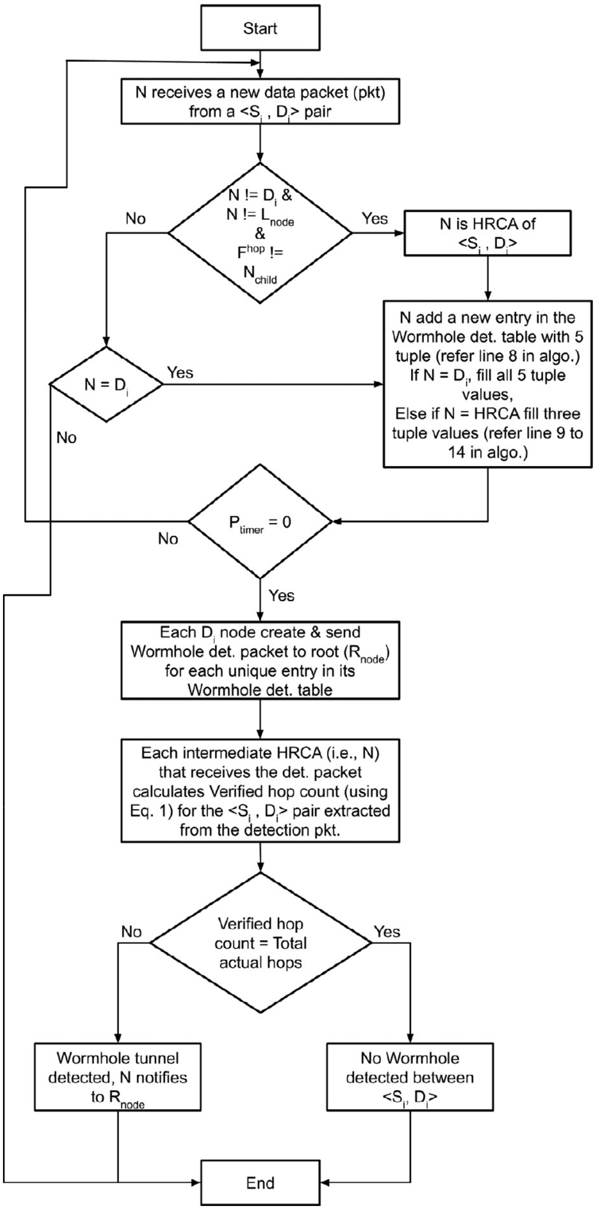

Start
N receives a new data packet (pkt) from a <S_i, D_i> pair
N != D_i & N != L_node & F^hop != N_child
No
Yes
N is HRCA of <S_i, D_i>
N add a new entry in the Wormhole det. table with 5 tuple (refer line 8 in algo.) If N = D_i, fill all 5 tuple values, Else if N = HRCA fill three tuple values (refer line 9 to 14 in algo.)
N = D_i
Yes
No
P_timer = 0
No
Yes
Each D_i node create & send Wormhole det. packet to root (R_node) for each unique entry in its Wormhole det. table
Each intermediate HRCA (i.e., N) that receives the det. packet calculates Verified hop count (using Eq. 1) for the <S_i, D_i> pair extracted from the detection pkt.
Verified hop count = Total actual hops
No
Yes
Wormhole tunnel detected, N notifies to R_node
No Wormhole detected between <S_i, D_i>
End

Fig. 3. Flowchart for wormhole attack detection

it will run the following wormhole detection steps before discarding the packet (Algorithm 2, line 22):
It will compute the verified hop count (V_{hop}^{count}) value using the following equation:

$$V_{hop}^{count} = (|\ (D_{rank}^{i} - HRCA_{rank})\ | + |\ (S_{rank}^{i} - HRCA_{rank})\ |). \quad (1)$$

  - If the value of V_{hop}^{count} for the pair of nodes S_{id}^{i}, D_{id}^{i} is same as the value of T_{hops}^{i} that is extracted from the received W_{det}^{pkt}, then it indicates the absence of a wormhole tunnel between the S_{id}^{i}, D_{id}^{i} (Algorithm 2, lines 23 – 24).
  - If the calculated V_{hop}^{count} differs from the T_{hops}^{i}, then the HRCA notifies the root about the existence of a wormhole tunnel between the S_{id}^{i}, D_{id}^{i} pair, upon which the DODAG root will take the attacker identification and mitigation measures (Algorithm 2, lines 26 – 27).
- If the node receiving the W_{det}^{pkt} is not the HRCA, then it will forward the packet towards the DODAG root. Finally, when the root receives a W_{det}^{pkt} and it is the HRCA, then it will computes the $V_{hop}^{count} = (|\ (D_{rank}^{i} - S_{rank}^{i})\ |$. After that, it compares the value of V_{hop}^{count} with T_{hops}^{i} that is given in the received W_{det}^{pkt} to identify the presence of a wormhole as depicted in (Algorithm 2, lines 21 – 23).

The process mentioned above will be used when RPL is working in the storing mode. In the non-storing mode, where the intermediate nodes have no memory, and all the packets need to be forwarded through the root, our wormhole detection process is simple. At each expiry of P_{timer}, if the root is the destination node for a source-destination pair then it will compute the value of V_{hop}^{count} as follows: $V_{hop}^{count} = (|\ (D_{rank}^{i} - S_{rank}^{i})\ |$. Else, the root will compute V_{hop}^{count} using the following equation:

$$V_{hop}^{count} = (|\ (D_{rank}^{i} - 1)\ | + |\ (S_{rank}^{i} - 1)\ |), \quad (2)$$

where 1 is the root rank. Later, in both the above cases, the V_{hop}^{count} is compared with the actual T_{hops}^{i}. The root stores T_{hops}^{i} for each source-destination pair in the network. It can simply calculate T_{hops}^{i} by using its global topology view.

To better understand the working methodology of our wormhole detection technique, we consider the DODAG depicted in Fig. 2 as an example, which has the following pairs $< S_i, D_i >$ in the network: $< 30, 29 >$, $< 19, 2 >$, $< 15, 24 >$, $< 26, 28 >$, and $< 3, 30 >$. Let us assume that the wormhole attacker pair of nodes is $< 10, 16 >$. We illustrate how the wormhole attack impacts the communication between a genuine pair $< S_i, D_i >$ (say $< 19, 2 >$). Also, we show that how our proposed approach identifies the presence of this wormhole attack. When node 19 sends a data packet to node 2, due to the wormhole attack, the packet follows the path: 19 → 30 → 24 → 10 → 16 → 14 → 13 → 23 → 12 → 20 → 2, which has the total path length (T_{hops}) of 10. However, without wormhole attack, the path is: 19 → 30 → 24 → 10 → 12 → 20 → 2. Without attack, the value of T_{hops} is equal to 6.

Table 3
Example: Node 24 W_{tab}^{det} table at time t_1.

S_{id}^{i}	D_{id}^{i}	S_{rank}^{i}	D_{rank}^{i}	T_{hops}^{i}
30	29	3	-	-
19	2	4	-	-
15	24	4	2	4
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
26	28	5	-	-
3	30	4	-	-

In order to clearly explain our detection algorithm, we make use of Fig. 2 and W_{tab}^{det} (i.e., Table 3) at node 24. Table 3 shows an instance of node 24's W_{tab}^{det}. The first entry in Table 3 is created when node 24 receives a data packet, which is initiated by source node 30, and node 29 is acting as the destination for the packet. The node 24 only creates a new entry for the first data packet of a new data session between a S_{id}^{i}, D_{id}^{i} pair, if node 24 is HRCA of S_{id}^{i}, D_{id}^{i} pair. As it can be seen from Fig. 2 that 24 is HRCA for $< 30, 29 >$ pair, so it will create a new entry for source node (i.e., 29) in its W_{tab}^{det}, only if such an entry is not already available in the table. In particular, node 24 extracts the values of $S_{id}^{i}, D_{id}^{i}, S_{rank}^{i}$ from the packet and stores them in the newly created entry. The columns D_{rank}, and T_{hops}^{i} will remain empty as the packet does not have these values. Similarly, when node 24 receives new packets from other source-destination pairs for which it is HRCA (e.g., $< 19, 2 >$, $< 26, 28 >$, and $< 3, 30 >$), it will create new entries in its W_{tab}^{det}. For $< 15, 24 >$ entry in Table 3, it is seen that all the columns are filled. It is because when node 24 receives the first packet for a new flow between 15 and 24, then the destination address in the received packet will match with 24 itself. Thus, it knows the value of D_{rank} (i.e., its own rank), it can calculate the T_{hops}^{i} by using the ranks of 15 and itself.

By taking into account the network scenario given in Fig. 2 where, if $< 19, 2 >$ is a benign source-destination pair, and $< 10, 16 >$ acts as a wormhole tunnel in the network, then our wormhole detection technique works in the following steps:

1. Source node 19 will initiate a new data session by sending the first data packet (say P_1) to the destination node 2.
2. Upon reception of P_1, each node in the path between 19 to 2 will execute lines 1 to 16 of Algorithm 2, to check if the node is HRCA for the $< 19, 2 >$ pair, and then process the packet accordingly. In our example, only node 24 will satisfy the *IF* condition given in line 2 of Algorithm 2 because it is HRCA for $< 19, 2 >$ pair.
3. Nodes 24 and 2 create an entry in the W_{tab}^{det} as per line 7. The entry created at node 24 is shown in Table 3 (2_{nd} entry). Table 3 also shows the entries for other source-destination pairs for which node 24 is either acting as HRCA or destination. It is seen in Table 3 that when node 24 is HRCA, then it can only store $< S_{id}^{i}, D_{id}^{i}, S_{rank}^{i} >$ information (e.g., 1_{st} and 2_{nd} entries in W_{tab}^{det}) for a $< S_i, D_i >$ pair. However, when node 24 is a destination node (e.g., 3_{rd} entry in W_{tab}^{det}), then it can store $< S_{id}^{i}, D_{id}^{i}, S_{rank}^{i}, D_{rank}^{i}, T_{hops}^{i} >$.
4. Let us assume that the periodic network timer (P_{timer}) expires at time t_1. Node 2 sends the W_{det}^{pkt} towards root (line 19), which goes through the path 2 → 20 → 12 → 10 → 24 → *root*. The W_{det}^{pkt} carries the 5-tuple $< S_{id}^{i}, D_{id}^{i}, S_{rank}^{i}, D_{rank}^{i}, T_{hops}^{i} >$ (i.e., $< 19, 2, 4, 6, 10 >$) that was stored in 2's W_{det}^{pkt} upon the reception of the first data packet from node 19. During its way to the root, when node 24 receives the W_{det}^{pkt}, it extracts the $< S_{id}^{i}, D_{id}^{i} >$ (i.e., $< 19, 2 >$) and verifies whether this pair is in its W_{tab}^{det}.
5. If the pair $< 19, 2 >$ exists in the W_{tab}^{det} of node 24, then it performs the checks (we refer the reader to lines 22 and 23 in Algorithm 2). The output of the Eq. (1) in line 22 will be $V_{hop}^{count} = (|\ (6-2)\ | + |\ (4-2)\ |)$, which is 6. When it is matched with the T_{hops}^{i} (i.e., 10) that is extracted by node 24 from the received W_{det}^{pkt} (line 20), then the given condition will not satisfy, and the lines 26 and 27 will get executed, which in-

Table 4
Simulation parameters.

Parameters	Values
Simulator	Cooja on Contiki *v*2.7
Simulation time	5 to 30 Minutes (for time varying scenario)
Simulation time	10 Minutes
DODAG root rank	1
Scenario Dimension	200 x 200 to 800 x 800 sq.meter
Number of nodes	61 sky motes (including 1 root for fixed scenario)
Number of nodes	21 to 101 sky Motes (for node varying scenario)
Transport layer protocol	UDP
Routing Protocols	RPL and LiDL
P_{timer}	1 Minute
Radio Medium	Unit Disk Graph Medium (DGRM)
PHY and MAC Layer	IEEE 802.15.4 with CSMA and ContikiMAC
RNG Seed	25 iterations each with new seed
Application protocol	CBR
Transmission Range	25m
Number of attacker nodes	2% to 10%
Traffic rate	0.50 pkt/sec - 500 packets

dicates the presence of a wormhole, and a notification about it will be sent to the root node.

5. Implementation and evaluation

In this section, we present first the simulation setup details that we used for implementing and evaluating our proposed countermeasures. Then, we provide an analysis of the simulation results that we have obtained from multiple IoT network scenarios. The results of our proposed solutions have been compared with the traditional RPL protocol only since there is no existing research that specifically addresses any of these two attacks in RPL-based IoT network scenarios. Although, few trust-based (Thulasiraman and Wang, 2019b) (Mehta and Parmar, 2018) and Intrusion Detection System (IDS) (Verma and Ranga, 2019) security solutions for RPL exists, however, these works are not suitable for comparison purposes since they do not address any specific routing attack.

5.1. Simulation setup

We have fully implemented both sybil and wormhole attack detection approaches on top of the available open-source code of RPL. The implementation is performed in *Cooja*, the Contiki network emulator (Romdhani et al., 2016),Get Started with Contiki, which is being widely used for deploying resource-constrained devices in LLNs (e.g., IoT networks), and we make available open-source implementation of both the detection techniques. We compare the performance of our detection techniques with the traditional RPL protocol and the state-of-art work in (Ahsan et al., 2017), in different scenarios. The existing results of the approach in (Ahsan et al., 2017) have been taken on very small network (i.e., 6 to 10 nodes, including one root node and one attacker node), which is not feasible for a scalable approach, so we take and compare results by increasing the same ratio of attacker nodes with respect to node density in the network used in (Ahsan et al., 2017) in different network scenarios. In order to show that our detection approaches are effective and scalable, we consider the different target scenarios with high node density. Additionally, for all IoT scenarios that we consider in our evaluation, we use the storing mode of RPL. The main reason is that in the non-storing mode, all the additional processing and storage consumption due to periodic execution of the detection algorithms will be performed by a resource-full node (i.e., root/LLN router). All the simulation results are generated by taking the average of 25 random simulations on each scenario. Table 4 provides the details of various parameters along with the values that we have used to configure different target IoT network scenarios in *Cooja* emulator Dunkels.

5.2. Performance evaluation and discussion

We present the result analysis of our proposed detection approaches with considering two metrics namely the *Average Packet Delivery Ratio* (APDR) and the *True Positive Rate* (TPR). We create and randomly deploy sybil and wormhole attacker nodes in the target scenario. The attacker nodes adversely affect the APDR by altering different network parameters (e.g., routing table information, and global topology view at the root node) that lead to the disruption of the data communication process. We also evaluate how the execution of detection schemes affect the energy and memory consumption of nodes in the network.

When a wormhole attack is executed in the network, the attacker aims to maliciously take part on a route, so it could degrade the network performance by using one or more of the following ways: (i) selectively drop the packets (i.e., selective packet discarding), (ii) delay the packets by changing their shortest path or creating loops, and (iii) drop all the packets (i.e., blackhole attack). As performing the blackhole attack increases the risk of being detected, therefore, in our scenarios, the wormhole attack aims to accomplish selective packet dropping and unceasing packet end-to-end delay. On the other hand, the sybil attacker intends to send and receive packets by cloning other nodes in the network. In our target scenarios, the aim of a sybil attack is to increase the network traffic by sending unwanted packets or to receive packets that are designated to other nodes.

To demonstrate the scalability of the detection techniques, Figs. 4 and 5 present the percentage of APDR and TPR with increasing the network size. In this network scenario, the number of network nodes is increased from 21 to 101 nodes with an interval of 20 nodes, and the network size is also increased from 200 x 200 to 800 x 800 sq.meter with an increase of 150 sq.meter each time. Moreover, the number of attacker nodes is kept 10% of the total network nodes. The attacker nodes are randomly selected in the network. The simulation time (i.e., 10 minutes) and the value of P_{timer} (i.e., 1 minute) are kept constant. As seen in Fig. 4, the APDR of our detection approaches remains higher than without any detection approach scenarios. However, there is a slight decrease in APDR with an increase in network density. When there are no detection algorithms deployed, the APDR is low since the packets are dropped by the wormhole attackers or due to their loops that induce expiry in the time-to-live value of a packet. Additionally, the presence of sybil attacks increases the network congestion, and

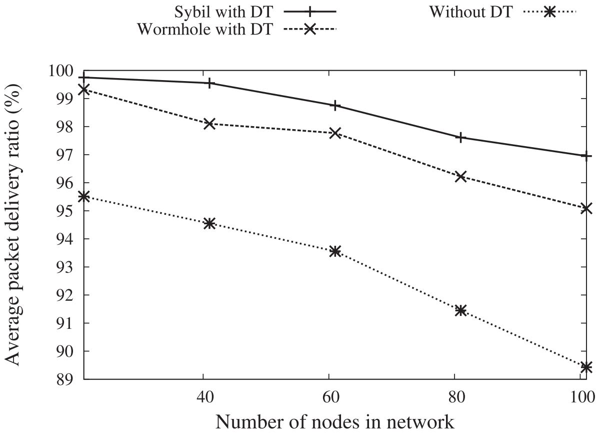


Fig. 4. APDR with increase in network size.

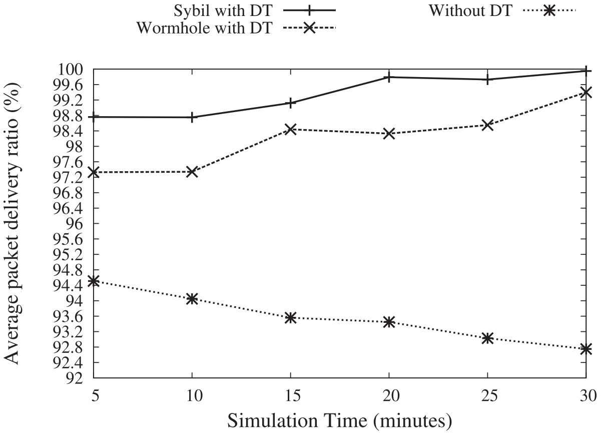


Fig. 6. APDR with increasing simulation time.

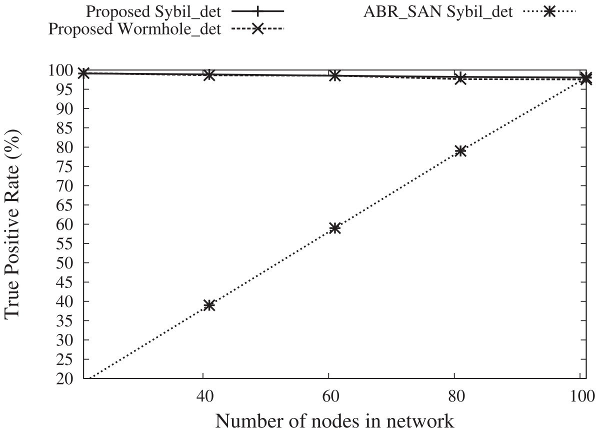


Fig. 5. Effect on true positive rate with increase in network size

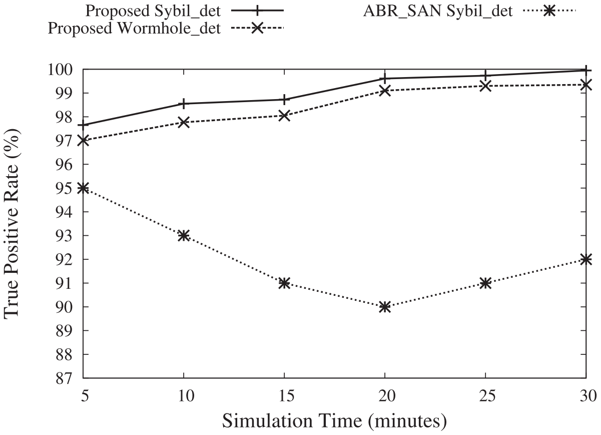


Fig. 7. Impact on true positive rate with increasing the simulation time.

there are cases where a packet is only received by the cloned node, thus further decreases the APDR. When we deploy our detection approaches in RPL, the slight drop in PDR shown in Fig. 4 is due to the time taken by the detection approaches to detect the attacks; and the possible increase in route length is due to increased node density.

The simulation results for both the attacks in a lossless network configuration presents approximately 100% true positive rate (TPR). Particularly, the availability of the required network information that is needed by the nodes executing the algorithms 1 and 2 resulted in nearly zero false positives. However, as the network configuration becomes lossy, due to the loss of some packets that are needed for accurate functioning of our algorithms 1 and 2, the TPR starts to decrease. For instance, with the increase in the size of the network, the TPR decreases, as it is shown in Fig. 5. This behavior is due to the fact that the global topology at the root node, as well as the full information about descendent nodes at a non-leaf node takes a little time to become stable in larger network configurations. Another reason for having lower values of TPR in large networks is due to the generation of high traffic at certain detection nodes (i.e., HRCA) during some detection periods, and the attacker entries in wormhole and sybil detection tables got overwritten. Similarly, when the number of attackers are increased in the target network, the TPR decreases due to the same aforementioned reasons. Fig. 5 also shows the TPR results for (Ahsan et al., 2017) approach, and it can be seen that the TPR is low in networks with a lower number of nodes. It is because (Ahsan et al., 2017) uses two special types of motes (called SAN and ABR) that helps it to detect wormhole presence in the network. Thus, the detection rate of (Ahsan et al., 2017) is directly dependent on the number of these special types of nodes present in the network. The number of these nodes depends on the total number of nodes in the network, and therefore, as the network size increases, these special types of nodes also increases, which increases the TPR as shown in Fig. 5.

Figs. 6 and 7 illustrate the impact of increasing the simulation run time on the two metrics APDR and TPR. We fixed the following parameters: (i) the number of nodes is 60 (including root nodes), (ii) the number of attacker nodes is 10% of total network nodes, and (iii) P_{timer} is 1 min. Fig. 6 shows that the APDR remains high when our detection approaches are active in the network, and the APDR increases with the time due to the increased TPR in the network, as shown in Fig. 7. The TPR has lower values in the case of low simulation time (i.e., 5 min) as it can be shown in Fig. 7. This behavior is because the nodes that execute the Algorithms 1 and 2 do not have enough network information within such a short simulation time to raise the alarm if the same node is misbehaving. Without our detection technique, the APDR remains almost constant (i.e., approximately 94%). For (Ahsan et al., 2017), the TPR values vary between 89% to 95%, and it is the average detection rate when it uses both special types of nodes (i.e., ABR and SNR) in the network. As shown in Fig. 7 our approach outperforms (Ahsan et al., 2017) concerning the TPR. The main reason is that unlike (Ahsan et al., 2017), our approaches do not depend on the presence of special nodes to detect a wormhole.

Figs. 8 and 9 show the percentage of APDR and TPR with varying the number of attackers. For these scenarios, the percentage

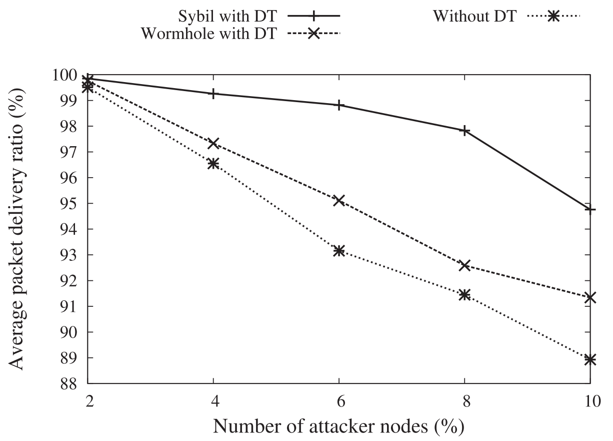


Fig. 8. APDR with increase in percentage of attacker nodes.

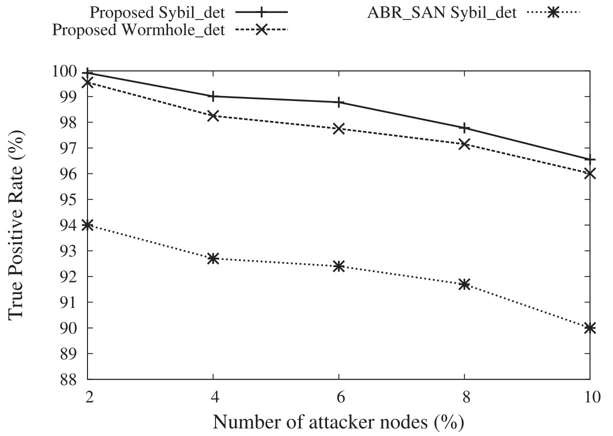


Fig. 9. Effect on true positive rate with increasing the number of attacker nodes

of the attacker nodes is increased from 2% to 10% in the network, while the other parameters are kept constant such as the simulation time (10 min), P_{timer} (1 min), and the number of nodes (61 including the root node). As shown in Fig. 8, the APDR of our sybil detection approach remains higher than 95%, however, for the wormhole detection approach, it is slightly decreasing with the increase in the number of attacker nodes. The APDR of RPL drops from 97 to 89% when the network has 10% of total network nodes as attacker nodes, and it continues to decline as the percentage of attacker nodes increases further.

As shown in Fig. 9, the TPR value for sybil attack is between 97% to 99%. This minor decrease in TPR of sybil detection algorithm is due to following reasons: (i) during the sybil detection execution period in the network if a sybil or its corresponding benign node (whose identify the sybil node is using) leaves the network, and then quickly rejoin from a different position, and (ii) if a HRCA node does not receive data or control packet from the benign node which is being attacked (i.e., forging identity) by the sybil node during the detection period, then the sybil node will not be detected in that detection cycle. However, the above two scenarios are rare in the network. Therefore, the overall TPR for sybil detection does not go lower than 97%. Fig. 9 shows that the performance of the wormhole detection technique decreases more than sybil as the number of attacker nodes increases. The reason for this behavior is two-folded: (i) a small number of attacker nodes might never get a chance to re-route data (i.e., become part of an active root) because the traffic is limited in the network. Thus, these attackers are not investigated by our detection algorithms, and (ii) due to the static nature of DODAG, all nodes have their fixed routes to send and receive data packets (unless there are any leave or join operations in DODAG). Thus, the attacked nodes will not be detected until either the DODAG changes or new traffic flows pass through these left out attackers. In both the cases mentioned above, the attackers will not be able to degrade the network performance because they are not part of any active data communication routes. In the detection technique proposed in (Ahsan et al., 2017), the TPR value varies from 90% to 94%. It has been seen that when both techniques (i.e., SAN and ABR) work together, the TPR increases. In a few cases, if one fails, then the other type of node can still detect the wormhole attacks, but the TPR in such cases will be lower. Our wormhole detection approach provides higher TPR than the work in (Ahsan et al., 2017), and the TPR of (Ahsan et al., 2017) decreases with an increase in the number of attackers in the network. This behavior could be justified since some attackers are successfully able to avoid detection from the SAN and ABR nodes, as they might reside out of their detection area.

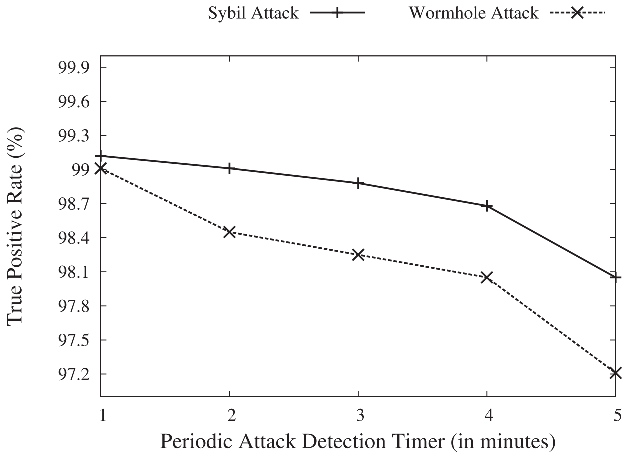


Fig. 10. Effect on true positive rate with increase in P_{timer} value.

Fig. 10 illustrates the attack detection rate with varying the P_{timer} interval. The value of P_{timer} is increased from 1 to 5 minutes with 1 minute interval, while the other parameters like simulation time (10 minutes), number of nodes (61 including root), and number of attacker nodes (6% and 10% of the total nodes) are kept constant. Fig. 10 shows that when the P_{timer} interval increases, then the performance of both the detection approaches decreases. The main reason for this behavior is that larger values of P_{timer} causes more entries in the detection tables to be over-written, which causes a delay or avoid the detection of attacker nodes. However, for low values of P_{timer} this will generate high energy consumption in the network. Hence, an optimal value of P_{timer} should be selected based on the specific target scenario. This optimal value of P_{timer} should consider parameters such as resource availability (i.e., node energy and memory), attacker detection time, and the percentage of detected attackers.

5.3. Energy and memory consumption analysis

We compute and present the overall energy consumption in the network in the presence of our detection techniques. Let $E^{wormhole}_{consume}$ be the energy consumed for processing the wormhole detection technique and $E^{sybil}_{consume}$ be the energy consumed for processing the sybil detection technique for all the source and destination pairs, which are represented by i and j variables in our below energy calculation equations. The detection process runs only at the HRCA nodes. Thus the energy consumption for the wormhole detection

Table 5
Average Power Consumption (APC) with using Sky motes.

PC Factors (In mW)	Traditional RPL	RPL with sybil	RPL with sybil detection	RPL with wormhole	RPL with wormhole detection
LPM	9.211	9.212	9.216	9.214	9.301
CPU	24.949	25.121	25.350	25.202	25.786
Listen	27.938	27.975	27.979	27.974	27.988
Transmit	4.88	5.01	5.12	5.06	5.22
Total	66.978	67.317	67.665	67.450	68.295

Table 6
Memory Usage.

	Flash [Bytes]	RAM [Bytes]
ContikiRPL	41,498	8246
sybil Detection Technique	192 (+0.5%)	96 (+1.2%)
wormhole Detection Technique	480 (+1.2%)	232 (+2.8%)
LiDL	42,170 (+1.7%)	8574 (+4.0%)

technique is depicted as follows.

$$E^{wormhole}_{consume} \approx (\forall P_{timer}) \sum_{n=1}^{j} HRCA(j) * \frac{W^{det}_{tab}}{\sum_{y=1}^{l} W^{det}_{tab}} \tag{3}$$

Similarly, the energy consumption for sybil detection technique is as follows.

$$E^{sybil}_{consume} \approx (\forall P_{timer}) \sum_{m=1}^{i} HRCA(i) * \frac{S^{det}_{tab}}{\sum_{x=1}^{k} S^{det}_{tab}} \tag{4}$$

Furthermore, we consider the total energy consumption based on standard Contiki measurement that is described in Table 5. Table 5 shows the comparison of average energy consumption for traditional RPL with our proposed detection approaches. These energy consumption values are computed based on a scenario which consists of a total of 61 nodes (54 benign nodes, 6 attackers, and 1 root), and it has one minute for P_{timer}. The simulation runs for 10 minutes with both detection approaches and without any detection approach (only traditional RPL). The values in Table 5 show minor increment in energy consumption when executing our proposed approaches. The main reason is that only a small subset of network nodes (i.e., HRCA) execute our simple (yet practical) detection approaches. Therefore, there exist a set of nodes in the network that do not have to execute the detection approaches. Thus, they will save their energy during the attack detection process. In particular, only the HRCA and root nodes execute the detection algorithms periodically in the network. The number of HRCA nodes that execute the detection algorithms depends on the number of active source-destination pairs and their positions concerning each other in the target DODAG. Please note that in a DODAG, it is possible that two or more source-destination pairs share the same HRCA, in such cases, the number of HRCA will be lower than the number of source-destination pairs in the network, and it will further lead to a small energy consumption. For instance, as shown in Table 3, node 24 is HRCA for four different source-destination pairs. Hence, if we consider the Fig. 2 and Table 3, then by executing the detection algorithms only on node 24, we can identify, if there exists a sybil or wormhole attack that adversely affects the nodes that include 30, 29, 19, 2, 15, 26, 28, 3, and 30. The energy consumption can be further reduced significantly if the length of the detection tables and the source-destination pairs in the network are kept lower, and the value of P_{timer} is kept higher. However, this will also decrease the TPR in the network. The low energy consumption of our detection techniques makes them eligible for the large-scale network deployment considering energy as one of the constraints for the IoT devices.

Table 6 shows ContikiRPL memory consumption Corporation, and the overall code and data memory increase when implementing our detection techniques. The cost of sybil detection technique is 192 Byte of Flash and 96 Byte of RAM. For the wormhole detection, the memory cost is 480 Byte of Flash and 232 Byte of RAM. As per our approach, we consider several entries in the table between 95 to 100, which is sufficiently a large amount compared to a large IoT network. Thus the memory consumption mentioned above is considerably low with respect to the security features that it provides when compared to the traditional RPL protocol.

6. Conclusion and Future Work

Data routing in IoT networks is a challenging process in unattended and insecure environments. To alleviate the major security threats and risks for RPL-based IoT networks, we propose two efficient detection approaches to counterfeit against two specific networking attacks, namely, sybil and wormhole attacks. Our algorithms benefit from the use of the HRCA concept that considers a local common ancestor in order to perform the detection process and localization of attackers in a specific (possibly small) area in the network. Hence, this simplifies the identification and removal of the attackers. We assess the feasibility of our algorithms by a thorough set of simulations scenarios. We show that our proposed detection techniques are lightweight (i.e., low resource consumption), scalable, and effective (i.e., high PDR and attack detection rate). As the networking attack detection and mitigation in resource-constrained LLNs are still in its early stages, in the future, we will investigate possible solutions for other networking attacks in the RPL-based IoT networks. Moreover, we plan to build a generic intrusion detection system to identify and mitigate multiple attacks and improve the overall network security.

CRediT authorship contribution statement

Pallavi Kaliyar: Writing - original draft, Validation, Software. **Wafa Ben Jaballah:** Conceptualization, Methodology, Investigation. **Mauro Conti:** Supervision, Conceptualization. **Chhagan Lal:** Writing - review & editing.

Acknowledgements

Pallavi Kaliyar is pursuing her Ph.D. with a fellowship for international students funded by Fondazione Cassa di Risparmio di Padova eRovigo (CARIPARO). Wafa Ben Jaballah is partially supported by the H2020 COLLABS under grant H2020-ICT-08-2019-871518. Mauro Conti were supported in part by EU LOCARD Project under Grant H2020-SU-SEC-2018-832735. The work of M. Conti was supported by the Marie Curie Fellowship through EC under Agreement PCIG11-GA-2012-321980.

References

Abusalah, L., Khokhar, A., Guizani, M., 2008. A survey of secure mobile ad hoc routing protocols. IEEE Communications Surveys Tutorials 10 (4), 78–93.

Ahsan, M.S., Bhutta, M.N.M., Maqsood, M., 2017. Wormhole attack detection in routing protocol for low power lossy networks. In: 2017 International Conference on Information and Communication Technologies (ICICT), pp. 58–67. doi:10.1109/ICICT.2017.8320165.

Airehrour, D., Gutierrez, J.A., Ray, S.K., 2018. Sectrust-rpl: A secure trust-aware rpl routing protocol for internet of things. Future Generation Computer Systems 1–18.

Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., Ayyash, M., 2015. Internet of things: A survey on enabling technologies, protocols, and applications. IEEE Communications Surveys Tutorials 17 (4), 2347–2376.

Conti, M., Kaliyar, P., Rabbani, M.M., Ranise, S., 2018. 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.

Corporation, M.,. Ultra low power IEEE 802.15.4 compliant wireless sensor module. http://www.crew-project.eu/sites/default/files/tmote-sky-datasheet.pdf.

Dunkels, A.,. Contiki OS. http://www.contiki-os.org/download.html.

Evangelista, D., Mezghani, F., Nogueira, M., Santos, A., 2016. Evaluation of sybil attack detection approaches in the internet of things content dissemination. In: 2016 Wireless Days (WD), pp. 1–6.

Ghaleb, B., Al-Dubai, A., Ekonomou, E., Qasem, M., Romdhani, I., Mackenzie, L., 2019. Addressing the dao insider attack in rpls internet of things networks. IEEE Communications Letters 23 (1), 68–71.

Glissa, G., Rachedi, A., Meddeb, A., 2016. A secure routing protocol based on RPL for internet of things. In: 2016 IEEE Global Communications Conference (GLOBECOM), pp. 1–7.

Get Started with Contiki. http://www.contiki-os.org/. (Access Date: 28.03.2020).

Hwang, Y.H., 2015. Iot security and privacy: Threats and challenges. In: Proceedings of the 1st ACM Workshop on IoT Privacy, Trust, and Security.

Khabbazian, M., Mercier, H., Bhargava, V.K., 2009. Severity analysis and countermeasure for the wormhole attack in wireless ad hoc networks. IEEE Transactions on Wireless Communications 8 (2), 736–745.

Khabbazian, M., Mercier, H., Bhargava, V.K., 2009. Severity analysis and countermeasure for the wormhole attack in wireless ad hoc networks. IEEE Transactions on Wireless Communications 8 (2), 736–745.

Kharrufa, H., Al-Kashoash, H.A.A., Kemp, A.H., 2019. Rpl-based routing protocols in iot applications: A review. IEEE Sensors Journal 19 (15), 5952–5967.

Kim, H.S., Ko, J., Culler, D.E., Paek, J., 2017. Challenging the ipv6 routing protocol for low-power and lossy networks (rpl): A survey. IEEE Communications Surveys Tutorials 19 (4), 2502–2525.

Le, A., Loo, J., Luo, Y., Lasebae, A., 2013. The impacts of internal threats towards routing protocol for low power and lossy network performance. In: 2013 IEEE Symposium on Computers and Communications (ISCC), pp. 789–794.

Levis, P., Clausen, T., Hui, J., Gnawali, O., Ko, J., 2011. The trickle algorithm.

Mehta, R., Parmar, M.M., 2018. Trust based mechanism for securing iot routing protocol rpl against wormhole grayhole attacks. In: 2018 3rd International Conference for Convergence in Technology (I2CT), pp. 1–6.

Miettinen, M., Marchal, S., Hafeez, I., Asokan, N., Sadeghi, A., Tarkoma, S., 2017. Iot sentinel: Automated device-type identification for security enforcement in iot. In: 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), pp. 2177–2184.

Miettinen, M., Nguyen, T.D., Sadeghi, A.-R., Asokan, N., 2018. Revisiting context-based authentication in iot. In: Proceedings of the 55th Annual Design Automation Conference, pp. 1–6.

Online Available: https://www.gartner.com/newsroom/id/3869181.

Perazzo, P., Vallati, C., Varano, D., Anastasi, G., Dini, G., 2018. Implementation of a wormhole attack against a rpl network: Challenges and effects. In: 2018 14th Annual Conference on Wireless On-demand Network Systems and Services (WONS), pp. 95–102.

Pongle, P., Chavan, G., 2015. Real time intrusion and wormhole attack detection in internet of things. Inter-national Journal of Computer Applications 121 (9).

Pongle, P., Chavan, G., 2015. A survey: Attacks on rpl and 6lowpan in iot. In: 2015 International Conference on Pervasive Computing (ICPC), pp. 1–6.

Pu, C., Hajjar, S., 2018. Mitigating forwarding misbehaviors in rpl-based low power and lossy networks. In: 2018 15th IEEE Annual Consumer Communications Networking Conference (CCNC), pp. 1–6.

Raoof, A., Matrawy, A., Lung, C., 2018. Routing attacks and mitigation methods for rpl-based internet of things. IEEE Communications Surveys Tutorials. 1–1

Raza, S., Magnsson, R.M., 2019. Tinyike: Lightweight ikev2 for internet of things. IEEE Internet of Things Journal 6 (1), 856–866.

Raza, S., Wallgren, L., Voigt, T., 2013. Svelte: Real-time intrusion detection in the internet of things. Ad Hoc Networks 11, 2661–2674.

Romdhani, I., Al-Dubai, A., Qasem, M., Thomson, C., Ghaleb, B., Wadhaj, I., 2016. Cooja Simulator Manual. Technical Report.

Sahay, R., Geethakumari, G., Modugu, K., 2018. Attack graph based vulnerability assessment of rank property in rpl-6lowpan in iot. In: 2018 IEEE 4th World Forum on Internet of Things (WF-IoT), pp. 308–313.

Tahir, Y., Yang, S., McCann, J., 2018. Brpl: Backpressure rpl for high-throughput and mobile iots. IEEE Transactions on Mobile Computing 17 (1), 29–43.

Thulasiraman, P., Wang, Y., 2019. A lightweight trust-based security architecture for rpl in mobile iot networks. In: 2019 16th IEEE Annual Consumer Communications Networking Conference (CCNC), pp. 1–6.

Thulasiraman, P., Wang, Y., 2019. A lightweight trust-based security architecture for rpl in mobile iot networks. In: 2019 16th IEEE Annual Consumer Communications Networking Conference (CCNC), pp. 1–6.

Tomi, I., McCann, J.A., 2017. A survey of potential security issues in existing wireless sensor network protocols. IEEE Internet of Things Journal 4 (6), 1910–1923.

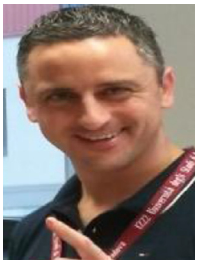
Verma, A., Ranga, V., 2019. Elnids: Ensemble learning based network intrusion detection system for rpl based internet of things. In: 2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU), pp. 1–6. doi:10.1109/IoT-SIU.2019.8777504.

Winter, T., Thubert, P., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, J., Alexander, R., 2012. RPL: IPv6 routing protocol for low-power and lossy networks.

Zhang, K., Liang, X., Lu, R., Shen, X., 2014. Sybil attacks and their defenses in the internet of things. IEEE Internet of Things Journal 1 (5), 372–383.

Pallavi Kaliyar is currently a Ph.D. student in school of Brain Mind and Computer Science at the University of Padova, Italy with a fellowship for international students funded by Fondazione Cassa di Risparmio di Padova e Rovigo (CARIPARO). Here, she is part of the SPRITZ Security and Privacy Research Group research group under the supervision of Prof. Mauro Conti. She received my Masters of Technology in Computer Science and Engineering in 2012 and Bachelor of Engineering in Computer Science and Engineering in 2008. She is conducting research on fields including security and communication reliability related to the Internet of Things and Software Defined Networking.

Wafa Ben Jaballah is a security researcher at Thales Group, France. She received her Ph.D. degree from the University of Bordeaux. Before joining Thales, she held a research position at Orange Labs. She was also a Post-Doc Researcher at the University of Bordeaux, France. Her main research interest is in the area of IoT security and network security. She has been a Visiting Researcher at the University of Padua (2012-2017).



Mauro Conti is Full Professor at the University of Padua, Italy, and Affiliate Professor at the University of Washington, Seattle, USA. He obtained his Ph.D. from Sapienza University of Rome, Italy, in 2009. After his Ph.D., he was a Post-Doc Researcher at Vrije Universiteit Amsterdam, The Netherlands. In 2011 he joined as Assistant Professor the University of Padua, where he became Associate Professor in 2015, and Full Professor in 2018. He has been Visiting Researcher at GMU (2008, 2016), UCLA (2010), UCI (2012, 2013, 2014, 2017), TU Darmstadt (2013), UF (2015), and FIU (2015, 2016). He has been awarded with a Marie Curie Fellowship (2012) by the European Commission, and with a Fellowship by the German DAAD (2013). His research is also funded by companies, including Cisco and Intel. His main research interest is in the area of security and privacy. In this area, he published more than 200 papers in topmost international peer-reviewed journals and conference. He is Area Editor-in-Chief for IEEE Communications Surveys and Tutorials, and Associate Editor for several journals, including IEEE Communications Surveys and Tutorials, IEEE Transactions on Information Forensics and Security, and IEEE Transactions on Network and Service Management. He was Program Chair for TRUST 2015, ICISS 2016, WiSec 2017, and General Chair for SecureComm 2012 and ACM SACMAT 2013. He is Senior Member of the IEEE.

Chhagan Lal is currently working as a postdoctoral research fellow at Simula Research Laboratory, Norway. Previously, he was a postdoctoral fellow at Department of Mathematics, University of Padua, Italy, where he was part of the SPRITZ research group. He received his PhD in Computer Science and Engineering from the Malaviya National Institute of Technology, Jaipur, India, in 2014. During his PhD, he has been awarded with the Canadian Commonwealth Scholarship under the Canadian Commonwealth Scholarship Program to work in University of Saskatchewan, Saskatoon, SK, Canada. His current research areas include applications of blockchain technologies, security in software-defined networking, Information-Centric Networks, and Internet of Things networks.