Hindawi
Security and Communication Networks
Volume 2018, Article ID 3672698, 15 pages
https://doi.org/10.1155/2018/3672698

Research Article

Hybrid Internal Anomaly Detection System for IoT: Reactive Nodes with Cross-Layer Operation

Nanda Kumar Thanigaivelan, Ethiopia Nigussie, Seppo Virtanen, and Jouni Isoaho

Department of Future Technologies, University of Turku, Finland

Correspondence should be addressed to Ethiopia Nigussie; ethnig@utu.fi

Received 30 December 2017; Accepted 30 July 2018; Published 15 August 2018

Academic Editor: Jesús Díaz-Verdejo

We present a hybrid internal anomaly detection system that shares detection tasks between router and nodes. It allows nodes to react instinctively against the anomaly node by enforcing temporary communication ban on it. Each node monitors its own neighbors and if abnormal behavior is detected, the node blocks the packets of the anomaly node at link layer and reports the incident to its parent node. A novel RPL control message, Distress Propagation Object (DPO), is formulated and used for reporting the anomaly and network activities to the parent node and subsequently to the router. The system has configurable profile settings and is able to learn and differentiate between the nodes normal and suspicious activities without a need for prior knowledge. It has different subsystems and operation phases that are distributed in both the nodes and router, which act on data link and network layers. The system uses network fingerprinting to be aware of changes in network topology and approximate threat locations without any assistance from a positioning subsystem. The developed system was evaluated using test-bed consisting of Zolertia nodes and in-house developed PandaBoard based gateway as well as emulation environment of Cooja. The evaluation revealed that the system has low energy consumption overhead and fast response. The system occupies 3.3 KB of ROM and 0.86 KB of RAM for its operations. Security analysis confirms nodes reaction against abnormal nodes and successful detection of packet flooding, selective forwarding, and clone attacks. The system's false positive rate evaluation demonstrates that the proposed system exhibited 5% to 10% lower false positive rate compared to simple detection system.

1. Introduction

Internet-of-Things (IoT), where intelligent "things" are interconnected to monitor and exchange information, is becoming increasingly prevalent across different application areas. Most of IoT applications facilitate our daily life, such as healthcare [1–5], smart home and cities [6–10], and intelligent transportation systems [11–13]. IoT consists of a huge number of devices that monitor the desired physical events/parameters and communicate the sensed data with the remote users through edge-router/base stations. The monitoring devices are mostly battery powered and have limited processing and short range radio communication capability. The introduction of an adaptation layer (6LoWPAN) in the network protocol stack enabled the integration of low power networks of IoT with IPv6, maximizing the utilization of available resources while benefiting from the huge address space of IPv6.

As in other similar technologies, security is an afterthought in IoT technology and becoming a main barrier in the wider adoption of IoT based services [14–16]. To address this issue, a number of security solutions for IoT have been proposed [17–20]. Most of these solutions focus on the use of cryptography for preventing external attacks, such as message alteration and eavesdropping. If some of the nodes are compromised and become internal attackers, cryptographic techniques cannot detect these malicious nodes since the adversary may have a valid key or the necessary information to perform activities within the network. The open wireless channels and unattended operation of the network make compromising and capturing sensor nodes easy which breaks the trust relationship that has been established beforehand. Usually internal attackers establish malicious nodes as legitimate nodes within the network. Experiments showed that an intruder could effectively interpose itself in the low power networks within five minutes and almost all attackers exploit the weaknesses in routing protocols [21, 22]. The attacker can easily retrieve all security information from compromised/captured node and launch internal attacks, such as data alternation, selective forwarding, jamming, and clone attacks. These attacks are destructive to network operation. For example, an internal attacker can keep injecting false data to cause network congestion and even denial of service. Thus, the capability to detect intrusions and malicious activities within IoT networks is critical for maintaining the functionality of the network [23].

Intrusion detection systems (IDS) are categorized into signature-based and anomaly-based detection [24] based on their detection technique. Due to the diversity of devices deployed in IoT networks, using signature-based IDS is impractical since they rely on predefined patterns. Unlike signature-based IDS, anomaly-based IDS are capable of detecting unknown attacks and thus appropriate for IoT. Anomaly detection systems that are developed for wireless networks are not suitable for IoT due to very limited energy resource, memory, and processing capability of nodes. Following a centralized approach where the base station monitors the entire network for malicious activities in the network is not efficient either as this approach lacks scalability in case of large networks. Even in distributed approach usage of traditional initiation-responder communication and relying on cluster heads alone to carry out entire IDS operations may incur overhead among the nodes and network. In either approach, nodes cannot react or make any countermeasures against suspicious node. Hence, in this paper a hybrid internal anomaly detection system is proposed. The detection system operation is integrated with Routing Protocol for Low power and lossy networks (RPL), which is a standard routing protocol for IPv6 based IoT.

In the proposed detection system, nodes monitor only their one-hop neighbors and all observed information is analyzed and managed locally in the nodes without the need for communication between nodes and nodes to gateway. This allows each node to react instinctively against threats. Communication with parent and subsequently to the gateway occurs only if an event has to be reported. The network and neighbor activity related information exchange will occur through the use of the designed novel control message. The system follows a cross-layered approach, operating at both data link and network layers. This enables the subsystems to perform their tasks independently by sharing a common repository. In addition, this allows discarding the packet of abnormal nodes at data link layer, preventing the packets from reaching to higher layers and congesting the network. The system is scalable, extendable, and interoperable with other security mechanisms due to its modular property. Thus, the contributions of this work are as follows:

(1) Architecturally reactive-adoption of reactive behavior in the system design enables every node to respond and enforce temporary countermeasures against threats without the involvement of router or cluster head in the decisions.

(2) Design and integration of Distress Propagation Object (DPO) messages: reporting of anomaly detection to parents is done through the use of newly developed DPO messages. It is an *ICMPv*6 message and integrated with the RPL protocol. The gateway processes received control messages (DPO) along with *UDP* messages (contextual data) and then correlate the information as well as performing periodic consistency checks to verify the network functionality and make decisions to raise appropriate alerts.

(3) Network fingerprinting is used in order to detect and track network changes as well as to determine the position of threat source.

(4) Reduction in network congestion and energy consumption: due to avoidance of broadcast type messages, the reactive type communication is employed instead of initiator-responder type and the dissemination of network changes to the gateway without its intervention occurs.

(5) Learning and grading are used to prevent the requirement for loading prior node's behaviors, to reduce the number of false positives, and to assist nodes in imposing temporal communication ban on threat sources.

The paper is organized as follows. The next section discusses existing relevant work and compared the features of the proposed internal anomaly detection system with related work. The architecture and the design of the proposed system including discussion of its subsystems, operation phases, cross-layer design, and integration of anomaly reporting messages are presented in Sections 3 and 4 accordingly. The implementation details of the system are discussed in Section 5. The test-bed and emulation environment setup that are used for evaluating the developed detection system are presented in Section 6. The analyses of the proposed system's overhead and detection capability are discussed in Section 7. Finally, the conclusion of the work is presented in Section 8.

2. Related Work

Intrusion Detection Systems (IDS) can be classified depending on its implementation (host-based vs network-based) or by detection methodologies (anomaly-based and misuse) [24]. As the proposed detection system is anomaly-based, similar existing works which operate as distributed or hybrid approach are discussed here.

In [25], a detection model where every node monitors its neighbor nodes has been proposed. The collected information is filtered and then the existence of an outlier is detected using Mahalanobis distance and confirmed based on neighbor's voting result. The authors claimed that the model has a very low false positive rate but the result is based on statistical analysis without appropriate network protocol emulation. There is no information about the energy consumption and memory requirements, though it is clear that there are computation and energy overheads due to monitoring, filtering, and voting using broadcast communication. Also, the system's implementation feasibility and gateway involvement were not discussed. In [26], an IDS employing a generic algorithm based on collaboration among nodes has been proposed. Each node is loaded with alert module and a set of one-way hash key chain from preassigned key. During operation, nodes publish their key and engage in voting to determine the trust-ability of neighbor nodes by broadcasting the observation up to 2 hops. Low memory footprint is achieved due to the use of external flash for key storage. Two-hop wide broadcasts consume additional resources and the system relies on exchanged information rather than activity monitoring.

In [27], the use of routers to carry out detection operations rather than nodes has been discussed. For the detection, the authors introduced a new *ICMPv*6 heartbeat message, which are sent periodically from router to nodes in order to complement the intrusion detection system by defending against selective forwarding attacks. The possible ways to eliminate the malicious nodes by isolating them with the maintenance of blacklist/whitelist content have been presented. A centralized detection system specifically targeted for IPv6 enabled wireless sensor networks using network-based approach is implemented in [28]. Network-based IDS (NIDS) is implemented in selected nodes and they act as watchdogs to overhear the conversations of the nearby nodes. These NIDS watchdogs are loaded with rules and belong to predefined group. Each watchdog node may have different set of rules and can be updated over the air. It is implemented using the readily available Finger2 middleware. The NIDS lacks dynamic features and learning ability and relies on preloaded rules for operation.

SVELTE, a hybrid IDS, is presented in [29]. The implementation of the IDS is divided into two: the resource intensive tasks are implemented in border-router and the nodes are given with least complex tasks. The nodes are responsible for sending network information to the router along with malicious incidents if detected. The border-router will make decision based on the information received from nodes. Another IDS work is proposed in [30]. The network is divided into several clusters and each cluster has a cluster head consisting of IDS instance. All the cluster members are in direct communication with cluster head and report neighborhood details. The cluster head is responsible for gathering the activities in the networks and determining the occurrence of malicious activities. In [31], a detection system in which nodes will send the detected changes to border-router is implemented. The nodes perform various relevant activities, such as sending neighbor and RSSI information while the border-router performs threat detection by analyzing the received information. The proposed system incurs overhead in memory as well as energy due to intense message exchanges in the network.

Our IDS implementation follows a similar task division principle to [29, 31] where the resource intensive tasks of IDS are handled by the border-router. In addition, it has other novel features. First, it responds to a threat immediately without the involvement of a border-router. When a node in neighborhood began to behave differently, it will be isolated by the neighboring node by blocking its packets at the data link layer and the parent node will notify this suspicious activity to the border-router. Though the isolation of the malicious node by its neighbor nodes is temporary, this act will contain the threat from making serious damage, e.g., packet flooding. Secondly, anomaly is reported to parents using a novel control message (Section 5.3) that is integrated with the RPL protocol. The border-router analyzes the received control messages and makes a decision on the detection. Thirdly, it has the ability to control the operation of the developed IDS system through the profile settings. Through adjustment of profile, the memory requirements, execution of IDS subsystems, and control messages dissemination can be reconfigured at any time. Finally, it has grading mechanism to control the false positives rate. Furthermore, our approach follows cross-layer design which prevents unnecessary processing of packets originated from the malicious nodes. In our IDS, router will never request neighbor or network related information from the nodes. All the activities in the network will be gathered and forwarded by the nodes through newly designed *ICMPv*6 control messages. In addition to the control messages, nodes also send the contextual data to router via *UDP*. The comparisons of the approaches used in the related works and the proposed IDS are summarized in the Table 1.

3. Architecture and Assumptions

In this section, the architecture of the detection system and details of its subsystems are presented. The principle of the proposed internal anomaly detection system is to look for any discrepancies in the network by monitoring the activities of sensor nodes, such as data rate and packet size. As discussed above, both the nodes and router take part in the detection process. The edge-router performs computationally heavy tasks since the sensor nodes are resource constrained devices. The sensor nodes monitor their neighbors and the participation of every node in the monitoring process is mandatory. All observed information is analyzed and managed locally in the nodes without the need for communication between nodes. Reporting to parent occurs only if a malicious activity is observed by the node. The neighbor and network activity information exchange occurs through the use of the newly designed control message (Section 5.3). Contextual data are sent to the router as *UDP* messages. The router analyzes the received anomaly reports along with the UDP messages and makes the final decision on isolating the anomaly node permanently.

To perform the detection of anomaly, each node in the network has three subsystems: monitoring and grading subsystem (MGSS), reporting subsystem (RSS), and isolation subsystem (ISS). These subsystems operate independently without interfering with each other. The subsystems share the same data repository as the base for their processes. The system specifies profile for operation and in the profile criteria for subsystem's phases are defined, such as process interval, dissemination interval of behavior reporting to parent, memory, threshold, and grading parameters. It does not need any prior information about the neighboring nodes since it is able to learn some features while the network is operational. This is advantageous since the behavior of the nodes and the threat pattern can change dynamically. The system also

Table 1: Comparison between related approaches and our work.

	Decision making	**Network monitor**	**Message exchange among nodes**	**Message exchange between nodes & router or cluster head**	**Protocol**	**Compulsory node participation**	**Evaluation**
[25]	distributed (through vote)	distributed	yes	-	-	yes	simulation
[26]	distributed (through vote)	distributed (every node)	yes	yes	MultihopLQI	yes	simulation
[27]	centralized	distributed (router through heart beat msg.)	no	yes	ICMPv6	no	simulation
[28]	centralized	distributed (watchdog nodes)	no	yes	BLIPv2	yes	testbed
[29]	centralized (router)	distributed (every node)	yes	yes	ICMPv6	yes	simulation
[30]	distributed (cluster heads)	distributed (cluster heads)	no	yes	-	yes	simulation
[31]	centralized (router)	distributed	no	yes	UDP	yes	simulation
this work	hybrid (node and router or cluster head)	distributed (every node)	no	yes	ICMPv6	yes	testbed & simulation

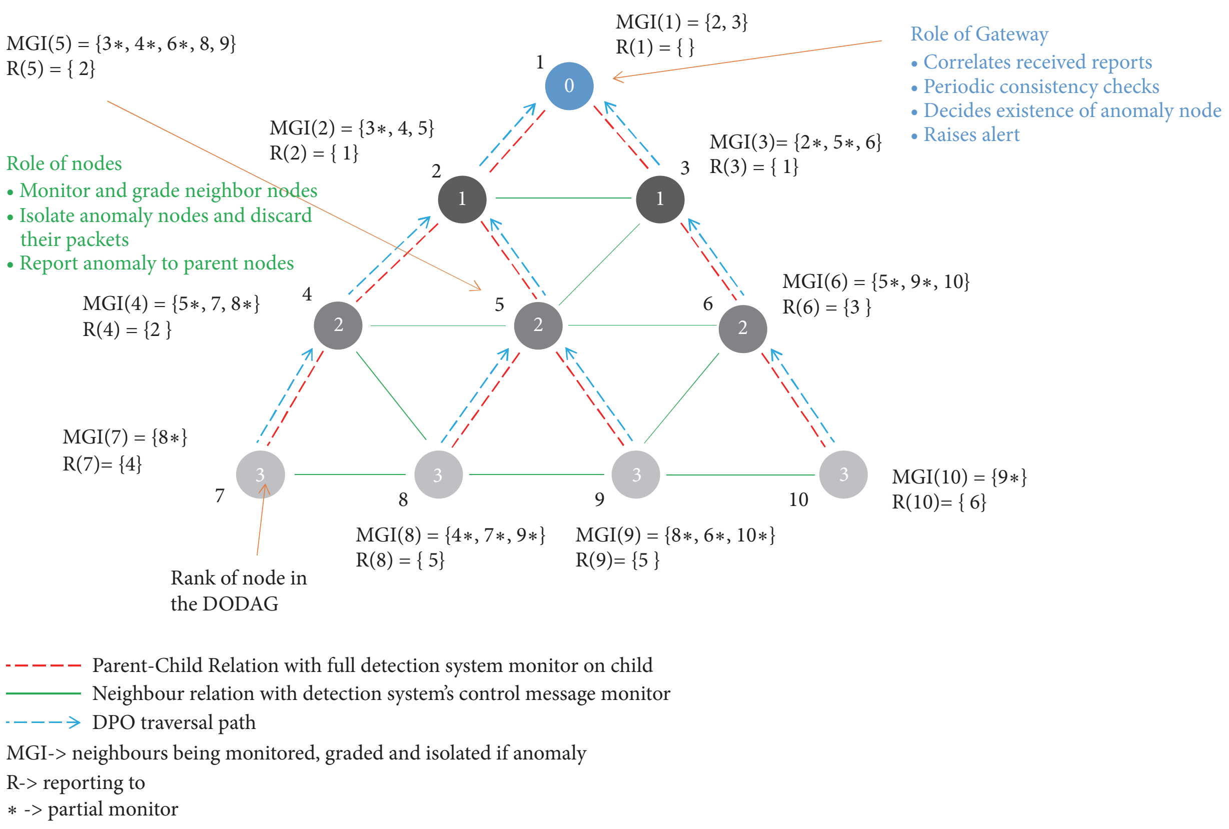
MGI(1) = {2, 3}
R(1) = { }
Role of Gateway
• Correlates received reports
• Periodic consistency checks
• Decides existence of anomaly node
• Raises alert
MGI(5) = {3∗, 4∗, 6∗, 8, 9}
R(5) = { 2}
MGI(2) = {3∗, 4, 5}
R(2) = { 1}
MGI(3)= {2∗, 5∗, 6}
R(3) = { 1}
Role of nodes
• Monitor and grade neighbor nodes
• Isolate anomaly nodes and discard their packets
• Report anomaly to parent nodes
MGI(4) = {5∗, 7, 8∗}
R(4) = {2 }
MGI(6) = {5∗, 9∗, 10}
R(6) = {3 }
MGI(7) = {8∗}
R(7)= {4}
MGI(8) = {4∗, 7∗, 9∗}
R(8) = { 5}
MGI(9) = {8∗, 6∗, 10∗}
R(9)= {5 }
MGI(10) = {9∗}
R(10)= { 6}
Rank of node in the DODAG
Parent-Child Relation with full detection system monitor on child
Neighbour relation with detection system's control message monitor
DPO traversal path
MGI-> neighbours being monitored, graded and isolated if anomaly
R-> reporting to
∗ -> partial monitor

Figure 1: Monitoring and DPO traffic in RPL's DODAG.

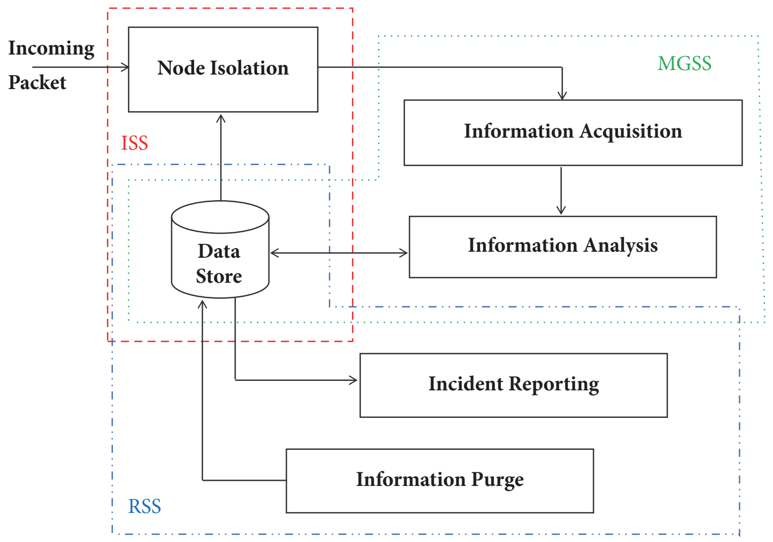

Figure 2: Node's subsystems and operational phases interrelation. MGSS: monitoring and grading subsystem, ISS: isolation subsystem, and RSS: reporting subsystem.

provides provision to control the memory requirement of the node's repository through appropriate settings in the profile configuration. The detection part of the router has three subsystems: intermediary, web API, and subroutine.

A destination oriented directed acyclic graph (DODAG) highlighting the process in the detection system is shown in Figure 1. The system has five major operational phases: information acquisition, information analysis, incident reporting, information purging, and node isolation as shown in Figure 2.

The proposed detection system is modular and extendable in order to monitor a number of characteristics of the neighboring nodes. The system supports mobility to a certain extent. The intimation of new and inactive neighbors to root helps to understand the network dynamics. The network fingerprinting feature allows the router to know the position of the nodes in the network without the use of geolocation hardware.

Apart from the network observation and dissemination of observed events, sensor nodes are tasked with an additional responsibility of containing threat independently. With the observed information, a node can deduce the suspicious activity originated from a child or neighbor node. After the deduction, the node reacts instinctively by imposing temporal communication ban of the anomaly node and notifies the router about the suspicious node. This is one of the unique features of our internal anomaly detection system that enables isolation of the threat by the nodes before routers involvement. The router, based on the collective information, will raise notifications to appropriate remote user and extend the communication ban throughout the network, if the analysis by the router also confirms the anomaly. Packet flooding threat scenario is taken in this work to prove the nodes ability to react instinctively against threats.

Assumptions. Edge-router is a fully secured entity such that it can withstand/identify and report/mitigate any attack targeted towards itself. It also has sufficient processing power to manage network-wide cryptographic key configurations and memory for storing information about nodes' data and activities for prolonged time period. Furthermore, it is assumed that the likelihood of achieving unauthorized access to the edge-router is exceedingly unlikely. We also assume that network is protected from external attacks through the employment of AES-128 based cryptographic protocol.

4. Design of Detection System

The detection system is designed to operate as two entities: one in the router and the other in the nodes. The design of the nodes operational phases, cross-layer functionality, and router operations are elaborated below.

4.1. Profile. In the profile, reporting intervals, log size, grading tolerance, rating ranges, and threshold values are defined. Profile will also define the time limit of temporal communication ban. Optionally, normal behavior can be included in the profile, if needed. If normal nodes/overall network activity is not available, the system will learn from the collected log and act upon them. For instance, the transfer rate and size of contextual data being transmitted from the neighboring node can be deduced from the log. Profile can be defined on node basis or as a whole either during flashing nodes with program or through router.

The primary motivation for defining a profile with limited and optional configuration settings and grading by grouping specific number of monitoring records is to reduce the false positive rate. Anomaly detection systems may generate considerable number of false positives since the outcome depends on variable network behavior and lack of normal behavior awareness. With the introduction of profile, the system became aware of the normal behavior boundary; this makes the system very malleable during assessment. The tolerance and rating ranges provide further flexibility during monitoring the activities in the network. Moreover, this flexible feature will provide additional time to correct the behavior of the node before classifying them as an abnormal one. These customizations in the system play a significant role in mitigating false positive rate. The MGSS is ingrained with these customizations and will automatically come into force when dealing with network activities.

4.2. Detection Subsystems of a Node. As discussed in Section 3, each node has four subsystems including repository that are responsible for performing the detection process. The monitoring and grading subsystem (MGSS) in each node is responsible for information gathering by observing different parameters of the neighboring nodes, such as data rate and packet size, and it grades the observed event based on the collected information in order to identify the anomalies. MGSS creates and maintains the primary structure of a detection system in the node. The primary structure is defined in the node's data store and it contains node's link layer address, grade and status information, and other relevant information. MGSS also creates logs structure in the data store of the node. The log structure contains extracted information from the activities in the network, such as packet received time, time difference, type, and group grade status. The reporting subsystem (RSS) handles the communication of the decision made by the analysis phase in the MGSS to its parent node. It also periodically cleanses the data store to manage the available memory efficiently. RSS is responsible for purging unnecessary logs using the primary structure statuses. During purge, it will go through the log to make sure that the statuses are correctly reflected by the MGSS. The isolation subsystem (ISS) oversees the packet reception and decides either to discard or allow packet to the upper layers for further processing based on the available grade (indicative of the anomaly activity) information in the data store.

4.3. Operation Phases of a Node

Information Acquisition Phase. Identical copy of the received message is created in this phase. The information required by the system for further processing will be extracted and elicited from the copy. The extracted information can be type of message, received time stamp, payload size, and sender details. Once the required information is available, they will be sent to successive phases for further processing.

Information Analysis Phase. The grading of the one-hop neighbor nodes is part of the analysis phase. The information received from the information acquisition stage will be compared against previously stored values of the same node. If any changes are noticed in the pattern, appropriate flags and constants will be set/reset and incremented/decremented. Based on the flag settings, the rating of the node will be quantified and if the quantified values reach minimum

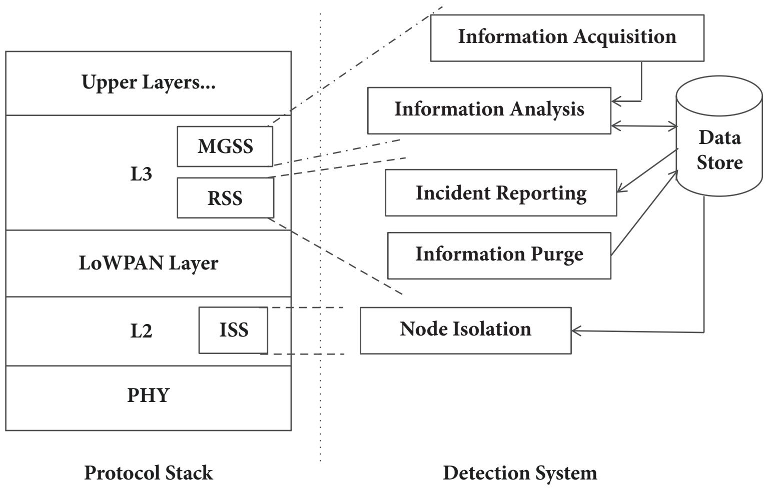

Figure 3: Cross-layer architecture.

threshold limit, respective flags will be affected. The received information along with the flags will be stored temporarily in the node memory.

Incident Reporting and Purging Phases. These are responsive phases and the flow entirely depends on the flags which are manipulated by the previous information analysis phase. The anomalies marked by the flags will be identified and reported to the node's parent. The receiving parent node relays the same information to its parent until it reaches the edge-router. Due to memory limitation, the observed behavior will be purged on predefined regular intervals on node-by-node basis. The incidents are reported using *ICMPv*6 message.

Node Isolation. The isolation phase is responsible for discarding the packets from the neighboring nodes which are failed to maintain their behavior above the provided threshold level from the monitoring node's perspective. Upon reception of messages from the neighboring nodes, the node will check the appropriate flags for the corresponding node and decide on either allowing the packet to upper layer or discarding it.

4.4. Cross-Layer Design. The adoption of cross-layer approach enables information sharing between the data link and network layers of the protocol stack. The isolation subsystem operates in the data link layer while the rest operates in the network layer. The cross-layer architecture of the detection system is shown in Figure 3. The primary objective for following cross-layer approach is to avoid unnecessary packet processing at LoWPAN layer since the LoWPAN layer is responsible for header compression and decompression tasks. The pieces of information such as IP header and message type are encoded during transmission and they will be decompressed by the receiving node's LoWPAN layer. Without knowing the IP addresses or the message type, the system does not work properly. Therefore, the monitoring and reporting subsystem phases such as information gathering and incident reporting operate in the network layer since they require those information for their operations. For the node isolation phase, link layer address is sufficient since the corresponding IP address for the link layer address is available in the routing table. This avoids the unnecessary and resource consuming header decompression process of the packets from anomaly nodes and prevents the traversal of these packets to higher layers by discarding them at data link layer.

4.5. Integration of Anomaly Reporting with RPL. The reporting of anomaly to parents is done through the use of the newly designed Distress Propagation Object (DPO) messages. The DPO is an *ICMPv*6 message and integrated with the RPL protocol. Its format is similar to the RPL control message format except for the unavailability of options field. It has different payload format to represent various observations. All the DPO messages are unicast messages which propagate to their preferred parent. DPO messages have the capacity to carry multiple neighbor nodes information objects in a single dispatch and use reactive type communication. Furthermore, control messages will be dispatched only when the changes have to be reported. These abilities allow avoiding frequent dispatching of message from node and thus reducing the overhead and congestion of network even after inclusion of new control messages. Section 5.3 gives in-depth information about the message formats and its appropriate usage.

4.6. Edge-Router. The edge-router receives reported anomaly information, processes and correlates the information, makes decision, and raises an alert, if necessary. Unlike sensor nodes, edge-router has no limitation on storage or processing and hence it can be used for process intensive tasks. The received incidents will be stored as long as the memory permits and a webserver can be configured to deliver the content over browser to remote end-users. As discussed in Section 3, intermediary, web API, and subroutine subsystems are the detection subsystems of the router. The intermediary subsystem is responsible for receiving packets from the nodes and stores them in the repository. The API subsystem provides data services to remote end-users through API. The

```
for every incoming IPv6 packet {
  if (sender exists in nbr_monitor) {
    if (normal node) {
      extract required information;
      generate appropriate information from extracted info;
      manipulate status flags based on current and available information;
      optimize and store information as log;
      allow packet to upper layers;
    } else {
      discard packet;
    } end if
  else {
    extract required information;
    generate appropriate information from extracted information;
    assign default operational settings;
    allow packet to upper layers;
  } end if
}
```

Pseudocode 1: MGSS and ISS overview.

subroutine subsystem is the core component of the detection system and it is responsible for various tasks, such as data correlation and threat deduction. It is also responsible for the creation and maintenance of list of active nodes under the router network. The data correlation task is executed on predefined intervals to check the inconsistency in the nodes reporting. If an abnormal incident is reported by a node, which is not a parent to the anomaly node, then router will check the reports of its neighbor nodes and correlate their findings before raising an alert. If the reported node is a parent then it will raise suspicious activity alert immediately provided that the node position information from all neighbors is satisfied. These tasks are handled by the subroutines. Though the nodes send anomaly reports, it is up to the router to make a decision based on the correlation output to classify abnormal behavior by raising an appropriate alert. Therefore, edge-router plays a crucial role in the detection system.

5. Implementation

The implementation details of the detection system are discussed in this section. The detection mechanism is designed in such a way that it does not raise an alert whenever an abnormal activity gets detected. Instead, it will record those activities and derive a grade by grouping certain number of records altogether. When the derived grade reaches the threshold limit defined in the profile as abnormal, then the exhibited activities are classified as anomaly and the node which exhibits the series of those activities will be perceived as an abnormal node. Once the detection system becomes conscious of an abnormal node in the network, the appropriate measures come into force automatically to deal with the classified node.

5.1. Detection Subsystems of Node and Router. The operations of MGSS and ISS are described in Pseudocode 1. The ISS is responsible for allowing the relay node to receive the packet or not and its functions are based on the outcomes of MGSS. Initially every neighbor is considered as a normal node and their traffic is allowed. From the received traffic, the required information is extracted, formatted, and compared with normal behavior including forbearance and previous statuses by the subsystems of a node. The appropriate flags are configured at the end of status assessment and finally grading based on the flags and assessments is carried out. The grading process determines the next action of the ISS and RSS phases.

The report flag is one of the flags in the detection system and it is used by the MGSS and RSS phases. By default it is set to false. In incident reporting phase, the status of the report flag determines the generation and dispatch of the Distress Propagation Object (DPO) messages. The flag is set to true, whenever a DPO message is dispatched. The resetting of the same flag will take place when a node is categorized as abnormal for certain activity. The setting and resetting of the flag ensure that the sensed activities are reported to the parent. The overview of incident reporting phase is presented in Pseudocode 2.

The generation of message payload depends on the other flags involved in the subsystems. The report flag also plays a role in purge phase. Purge phase is responsible for cleaning of unnecessary logs to free the memory for future storage of logs including clearing of classified node after certain time. In order to clear notification logs, it will check the status of report flag before commissioning the cleansing process. An outline of a purge phase's task is given in Pseudocode 3.

As discussed in Section 4.6, the subroutine subsystem of the edge-router is responsible for core detection functions. Different parts of the subsystem operate consecutively within the subroutines to set appropriate flags in the data repository which are essential for raising an alert. An example subroutine task of edge-router is given in Pseudocode 4. Based on the reported data from the nodes, the edge-router decides whether there is anomaly or not in the network.

```
while (report_timer expired) {
  for every node in nbr_monitor {
    if (status == new neighbour) {
      if (not reported to parent) {
        generate NIO;
        dispatch DPO;
        assign reported status;
      }end if
    } else if (status == inactive || anomaly) {
      if (not reported) {
        generate ONIO;
        dispatch DPO;
        assign reported status;
      } end if
    } end if
  }
  reset report_timer;
}
```

Pseudocode 2: RSS-incident reporting phase outline.

```
while (purge_timer expired) {
  for every node in nbr_monitor {
    if (status == inactive) {
      if (reported to parent) {
        purge corresponding node information;
      } end if
    } end if
  }
  reset purge_timer;
}
```

Pseudocode 3: RSS-information purge phase: inactive node outline.

5.2. Network Fingerprinting. To detect internal anomaly in the network, it is important to understand the structure of the network as well as any changes while the network is running and this can be achieved through network fingerprinting. Fingerprinting operation is reflexively handled by the MGSS and RSS operational phases. Whenever there is a change in the network, the operational phases of the MGSS will identify those changes including the addition of new neighboring nodes. The nodes which were once neighbors may no longer exist after certain period due to mobility, being taken for maintenance, or running out of energy. These nodes are identified by the RSS since its operation is not entirely dependent on the manipulated flags by the analysis phase. RSS can also review the monitoring logs and manipulate certain flags to mark them as needed or not by the detection system and act upon them. The purge phase will take those flags into consideration while executing its own tasks for cleaning the data store. Though the current version of the system does not support full mobility in ascertaining the anomalies, it recognizes changes in the network and give notification instinctively.

Table 2: List of DPO flags and payload type.

Value	Type of Payload
1	Neighbor Information
2	Inactive Neighbor Information
3	Anomaly in contextual data (UDP)
4	Anomaly in control messages (ICMPv6)

5.3. Distress Propagation Object (DPO). DPO messages are used to report changes in the network by communicating the activity of their neighbors including anomaly activities. The generic structure of DPO message is shown in Figure 4. The $8-bit$ "RPL Instance ID" field contains the id of RPL instance. The "version number" is the DODAG version number. Both the RPL instance id and version number field do not contain any new values which are unknown to the receiving node. The "flag" is an $8-bit$ field which decides the packet format in the sender and packet processing at the receiver ends. The list of available flag values is shown in Table 2.

The payload structure and its length vary depending on the type of flag used. If a node receives a DPO message, it will check the flag before processing the payload since the flag determines the payload content. If the flag is other than the assigned value, the packet will be discarded. The received DPO will be further relayed by the receiving node to its parent until it reaches to the root. The "payload length" field can hold the count of neighbor information object included in the payload if the flag is set to 1; else it contains the length of the payload.

RPL defines code to segregate the purpose of control message usage and the codes. The newly proposed DPO messages also have different purposes and they are classified based on the flag which is listed in Table 2. Separate codes are assigned for DPO messages to differentiate them from other RPL messages, such as DIO, DAO, and DIS. These codes are used for sending DPO message and DPO acknowledgment and their values are "0x04" and "0x05", respectively. The "code" field of $ICMPv6$ header carries these values during packet transmission.

5.3.1. Neighbor Information. The neighbor information message is used to disseminate the neighboring nodes information in upward direction towards its parent. The format is shown in Figure 5. The payload part of the message has three fields collectively known as neighbor information object (NIO). These fields are "pFlag", "MAC", and "Local Scope IP" fields. The "pFlag" is an $8 - bit$ flag field which is used to identify whether the node is parent to the sender or not. The $64 - bit$ "MAC" and $128 - bit$ "Local Scope IP" fields contain neighbor node's link layer address and local scope $IPv6$ address, respectively. The $8 - bit$ "payload length" holds the number of NIO in the payload content.

5.3.2. Inactive Neighbor Information. The inactive neighbor message will be propagated when a node was previously active and no longer transmitting any kind of messages. The message body contains two fields: $64-bit$ "MAC" and $128-bit$ "Local Scope IP" that are collectively referred to as observed

```
while (sbrt_timer expired) {
  for every node (n_i) in network {
    for every nbr (n_j) of n_i {
      if (n_i reported anomaly of n_j && open) then {
        if (n_i is parent) then {
          new_nbr:= count(nbr[n_j] reported dpo_type_1 after n_i anomaly report);
          new_nbr_an:= count(new_nbr reported anomaly);
          if (new_nbr > 0) then {
            if (new_nbr_an < new_nbr) then {
              //do nothing;
            } else {
              set data_anomaly of n_j;
            } end if
          } else {
            set data_anomaly of n_i;
          } end if
        } else {
          if (count(nbr[n_j] reported inactivity) < (count(nbr[n_j]) {
            //do nothing;
          } else {
            set unreported_data_anomaly of n_i;
          } end if
        } end if
        set closed;
      } end if
    }
  }
  reset sbrt_timer;
}
```

Pseudocode 4: Data anomaly deduction from nodes observation.

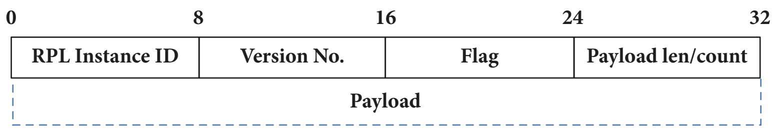

Figure 4: DPO message generic structure.

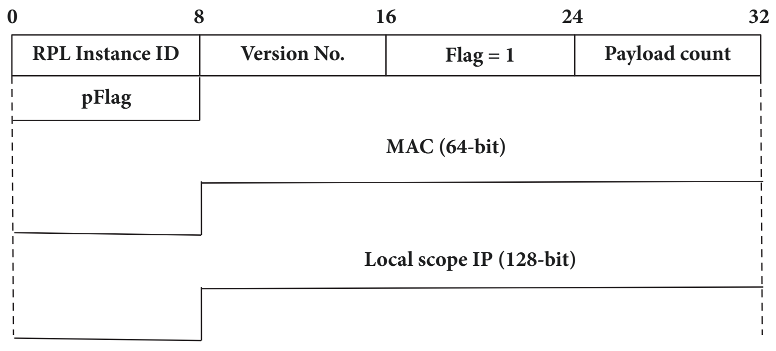

Figure 5: DPO message: neighbor information format.

neighbor information object (ONIO). The message structure is given in Figure 6. The 8 – *bit* "payload length" field holds the length of the entire message body. Unlike other DPO messages, it requires an acknowledgment from the root as confirmation that the message reached the root properly. The acknowledgment is obligatory since the message concerns the disappearance of a node or inactivity. The disappearance may be due to one of the three reasons: mobility, node running out of energy, or unauthorized disabling/removal of the node physically. The latter two cases may pose considerable

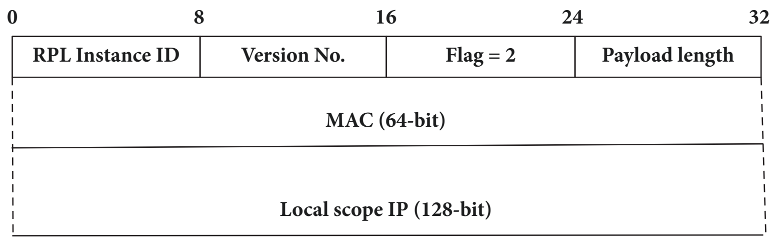

FIGURE 6: DPO message: inactive neighbor information format.

threat to either the network or the application and hence the acknowledgment of inactive NIO is required from the edge-router.

5.3.3. Anomaly Neighbor Information. This message is similar to inactive neighbor information message (shown in Figure 6) except for the flag field value which ascertains the specific anomaly behavior in the node. If the neighboring node exhibits variation in sending contextual data, the flag will be set to 3 and if the variation is in the control messages, then it is set to 4. This message does not require acknowledgment from the root.

6. Test-Bed and Emulation Setup

The developed detection system is evaluated using both test-bed and simulation/emulation environment. In this section, the details of the test-bed and emulation environment are presented.

Test-bed environment is constructed using Z1 node and PandaBoard. The sensor node part of the system is implemented in Z1 mote by Zolertia which is an IEEE 802.15.4 compliant low power node. The primary reason behind the selection of Z1 is to prove the system openness to any kind of cryptographic based data security and to keep the implementation hardware requirement as low as possible. Z1 has 16-bit ultralow power microcontroller, 92 KB flash, and 8 KB RAM. Z1 node is shown in Figure 7 as part of the gateway implementation.

Z1 has on board integrated temperature sensor. It is programmed with *UDP* client program to send the temperature information to the root on regular intervals. Also, it contains a part of the detection system (MGSS, RSS, and ISS) as shown in Figure 3. Along with the initialization of *IPv*6, the detection system will be initiated; after the creation of routes and neighbor information, base for MSS will be established first and ISS commences its operation. Once MSS is established, the detection is ready to perform its tasks whenever a packet from a neighbor is received.

Edge-router/gateway requires two separate hardware types to fulfill its operations: one to have communication with 6LoWPAN network and the other to provide interconnection between 6LoWPAN and external IP networks. The gateway solution is developed using the PandaBoard hardware and Z1 node attached to gateway to communicate with sensor nodes using IEEE 802.15.4 radio protocol. The setup of gateway using PandaBoard and Z1 node is shown in Figure 7. The

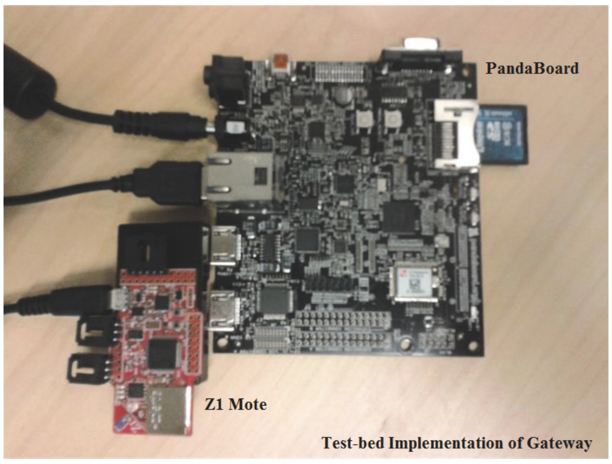

FIGURE 7: Test-bed implementation of gateway.

memory requirement and packet flooding attack detection are investigated using the test-bed.

Emulation environment is established using Contiki OS in Cooja. Contiki OS is an open source, lightweight, event driven, and multitasking operating system specifically developed for low power and memory constrained embedded systems [32]. Cooja allows displaying the status of the process or the execution flow if needed. It also comes with several built-in hardware motes for emulation of the network. Cooja was set to emulate Zolertia nodes and use *UDP* over *IPv*6 in network communication. Outgoing *IPv*6 packets flow from the *uIPv*6 layer to the 6*LoWPAN* layer for header compression and fragmentation. The 6*LoWPAN* layer sends outgoing packets to the MAC layer. ContikiMAC with 8 Hz channel check rate is employed as CSMA/CA mechanism. Packets from the queue are transmitted in order through the radio duty cycling (RDC) layer. The RDC layer in turn transmits the packets through the radio link layer. The MAC layer will retransmit packets until it sees a link layer acknowledgment from the receiver. If a collision occurs, the MAC layer does a linear back-off and retransmits the packet. Outgoing packets have a configurable threshold for the maximum number of transmissions. Two network topologies are simulated in Cooja: one has 20 nodes and the other has 50 nodes in addition to the DODAG root. The network topology of 50 nodes is shown in Figure 8. The energy consumption, detection capability of selective forwarding,

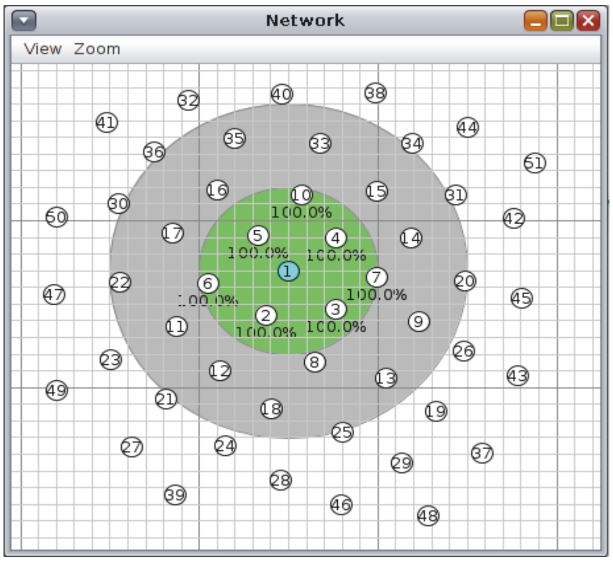

Figure 8: Simulated network: 50 nodes.

and clone attacks, as well as the false positive rate of the system are analyzed using emulation in Cooja.

7. Results Analysis

The overhead of the developed anomaly detection system in terms of memory requirement and energy consumption is analyzed. The capabilities of detecting packet flooding, selective forwarding, and clone attacks are also evaluated and presented in this section. In addition, the analysis of false positive rate of the detection system is also discussed.

7.1. Memory Requirements. The developed detection system libraries have occupied 448 bytes of RAM during compilation time and 3315 bytes of ROM after compilation. The detection system is configured to monitor up to 8 neighbors and the memory required for maintaining neighbor statuses and monitoring logs was allocated and released during runtime. Neighbor statuses required 28 bytes and log needed 24 bytes per node. Thus, for monitoring 8 nodes, a total of 416 bytes of RAM were required during runtime. The total RAM consumption for the detection system was 864 bytes.

The memory efficiency of the developed detection system is compared against the works mentioned in [26, 28, 29, 31]. The memory requirement of [26] was 12.23 KB of ROM and 720 bytes of RAM excluding policies (each policy definition is expected to consume a maximum of 50 bytes of RAM). And [28] needed 583 and 9473 bytes of RAM and ROM, respectively. The memory requirements comparison of the three detection systems is shown in Table 3, assuming that [26] employed 8 policies and the nodes in the developed detection system monitored 8 nodes. These comparison results clearly confirmed the efficient memory usage of the developed detection system.

Table 3: Memory requirements comparison.

Memory	[28]	[26]	**This work**	[31]	[29]
ROM (bytes)	12524	9473	3315	42,636	1,760
RAM (bytes)	1120^{+}	583	864$^{\#}$	2,886	365*

+ denotes requirement for handling 8 policies.
# denotes requirement for handling 8 neighbors.
∗ denotes requirement for handling single event.

7.2. Energy Consumption. The energy consumption of the detection system was estimated using Energest library provided in Contiki. Two cases were considered: all nodes are legitimate and adversary nodes existed in the network. The simulation is run for an hour and the energy consumption is recorded by employing the Energest library. Higher energy consumption is recorded when nodes detect and report anomaly because energy is consumed due to reception of packets as well as transmitting of the DPO message.

Energy is calculated using $E = V * I * t$, where V is voltage, I is current, and t is time. The node voltage is assumed to be 3V. The current values in transmission, reception, and processing modes are taken from the datasheet of Zolertia node. The time under the different operation mode is recorded from the simulation. The energy required to handle the DPO messages is listed in Table 4. These values are the energy overhead of the developed detection system.

7.3. Security Model and Analysis. To assess the capability of the detection system, selective forwarding, clone, and packet flooding attacks are evaluated. All legitimate nodes are programmed to transmit data every 5 seconds. The placement of adversary can be anywhere in the network since the participation from every node is obligatory.

7.3.1. Packet Flooding. Packet flooding attack is performed by sending huge number of packets to deplete the available resources. This test is done in test-bed. Four nodes are selected to carry out this attack. Two of these nodes are programmed to transmit 300 packets per minute and the other two 1200 packets per minute. The first packet flooding attack starts at 181st second and the successive adversaries start 10 seconds after the other. Initially, the neighbor nodes begin to accept the packets due to unavailability of prior knowledge about normal behavior of nodes in the detection system. After 25 seconds, the nodes classify the flooder nodes as abnormal nodes. Nodes instinctively enforced 3-minute communication ban against the flooder nodes. This activity contained the effects of massive packet injection into the network and restored the normal traffic. The DPO message of ONIO type was sent to the gateway by the intermediate parent nodes though these transmissions consume twice the energy of the normal transmission. Figure 9 clearly shows the normal packet movement before the flooder nodes injection and after detection of flooder nodes as abnormal ones and the congestion in the network during the attack.

7.3.2. Selective Forwarding. In selective forwarding attack the adversary node allows packets from certain nodes and

Table 4: Energy consumption for transmission and reception of DPO messages.

DPO message type	Total size(bits)	Transmission energy (mJ)	Reception energy (mJ)
Single NIO payload	200	3.159	3.794
Multiple NIO payload (6 objects)	1200	21.115	22.767
Single ONIO payload	192	3.378	3.643

```
if source ip_address == 2001:5c0:1508:4001:c30:0:0:4 then
  drop the UDP packet;
else
  forward the UDP packet;
```

Pseudocode 5: Selective forwarding.

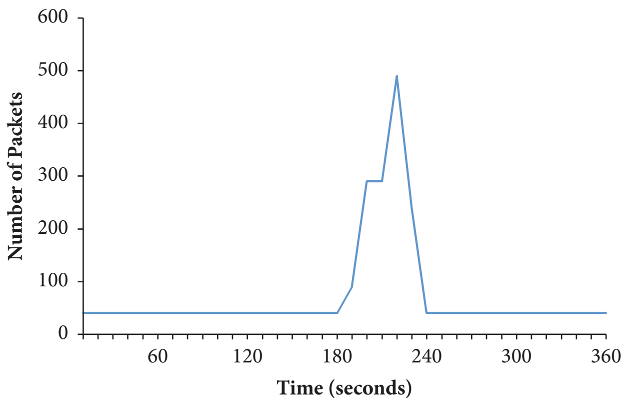

Figure 9: Congestion in the network due to packet flooding.

discards the rest. To simulate selective forwarding attack, a node without the detection system is configured to drop the packets of a genuine node. This adversary node is introduced to the network around 10 minutes after network starts running. It is necessary that the adversary node should not have the detection system; if it exists, it will send DPO message of NIO type. The detection system running at gateway may fails to detect the attack due to reception of NIO and might consider that the event has occurred due to mobility. The algorithm used for modeling selective forwarding attack by an adversary is given in Pseudocode 5.

The gateway's detection system executes a subroutine every five minutes except in the first execution which will occur only 15 minutes after network formation. The subroutine's logic is straightforward; it will check whether it receives contextual data from every node in the network by going through the list of active nodes under the router. This active list is constructed through the reception of DPO messages. If router finds that it does not receive context data from an active node, then it will check the corresponding intermediate parent node status. If those statuses are positive then the system raises an event "unreported inactive status" for the node. It is also easy to find the point of occurrence since the router has every information about the nodes and their neighbors. The simulated selective forwarding threat is detected within 10 minutes from the introduction of the adversary. Moreover, selective forwarding attacks require the adversary to be part of the network; this will trigger an NIO type message from the intermediate parent node. The parent node will be the same parent for the legitimate node whose packets are discarded by the adversary in most cases and thus makes it easy to approximate the range of threat existence.

7.3.3. Clone Attack. Clone attack is similar to Sybil attack except for the presence of multiple IDs in a single physical node. In order to create clone attack in the network, two adversary nodes with the same identity of two legitimate nodes are created. They are placed in four hops away from the copied legitimate nodes. Since disseminating DPO message for inclusion of every new neighbor is mandatory, the neighbor nodes raise and forward DPO towards the gateway. The gateway will process the DPO as usual and considers them as genuine nodes. The subroutine which runs every 5 minutes in the router will correlate nodes relationship structure; it will cross verify the relationship status by validating the DPO reported by neighbor nodes and if it founds any inconsistency in the neighbor reporting, the detection system raises an event "inconsistent neighbor relation". Due to the availability of nodes and their neighbor information, it is not a complex task to approximate the position of a node in the network and with this position approximation clone attacks can be mitigated. The detection system took 10–15 minutes to identify the simulated clone threats.

7.4. Analysis of False Positive Rate. False positive refers to the identification of legitimate nodes as abnormal and it is one of the important performance evaluation factors for an anomaly detection system. As we discussed in earlier sections, the detection system offers features to configure the grading system, the aggregated log grading, and the flexibility setting to handle the false positive rate. The grading system configuration can be adjusted based on the requirements. The aggregated log grading is set to two by default. The system learns the data rate by analyzing the log. A flexible setting is provided in the profile in order to give a sufficient window space for comparison. In the simulation, the flexible setting is set to two seconds. If the system learns that the data rate of the

Table 5: False positive numbers.

	20-Node Network	50-Node Network
Simple DS	361	553
DS (1st run)	44	37
DS (2nd run)	34	-
DS (3rd run)	21	-

nodes in the network is 12 packets per minute (1 packet per 5 seconds), then the comparison window will be between (5-2) and (5+2) seconds. Two scenarios were employed to find the false positive rate: the developed detection system (DS) and simple anomaly detection system capable of learning but without profile and aggregated log grading (simple DS). These two scenarios are evaluated using Cooja emulation of networks consisting of 20 and 50 Zolertia nodes. The evaluations are executed for about six hours. The grading system configuration was as follows: the grade step was set to 1, the threshold was set to 3, and the default maximum and minimum of the grade were 10 and 1, respectively. The number of false positives for the two emulated topologies employing the developed detection system and the simple DS is listed in Table 5. In the 20-node case, the simple anomaly-based system generated more than 350 false positive numbers while the DS generated around 40 numbers. To confirm the reputability of the low false positive number in the DS, three more runs were carried out. Each time the number of generated false positives is relatively smaller than the previous execution and in 3rd time the number reached around 20.

The above results indicate that the false positive rate of the developed detection is lower compared to the simple DS (between 5%-10% of the simple DS). The use of flexibility setting, configurable grade system, and aggregated log grading clearly made the difference in minimizing false positive rate. The only limitation in the current implementation is the use of just two configurations for aggregated log grading (1 or 2 logs per node). If the aggregated log grading has similar settings to grading (max, min, default, step, and benchmark), it might further reduce the false positive rate. This is not implemented currently due to the high memory overhead. We will investigate effective way of implementing this in future. One significant factor encountered during result analysis is that the possibility of a normal node getting classified as abnormal is likely higher if it is classified at least twice previously. It is also noted that the current node's activity learning process using three features (packet rate, packet size, and control packet) can be increased further to improve the accuracy of the detection.

8. Conclusion

In this work, a hybrid internal anomaly detection system that enables nodes to respond instinctively against threats has been presented. The main principle of the system is that nodes monitor and grade their one-hop neighbors and report them to their parent only if they detect an anomaly. The monitoring node isolates the anomaly node instinctively and discards its packets at the data link layer to avoid the unnecessary packet processing overhead. A novel *ICMPv*6 control message, Distress Propagation Object (DPO), has been developed and integrated with RPL protocol for propagating the anomaly observations and network changes to the immediate parent, to subsequent parents, and ultimately to the root. The edge-router performs periodical consistency checks, processes the received report, correlates the information, makes a decision, and raises an alert when necessary. The system is modular and extendable with configurable profile and subroutine settings. The system's network fingerprinting feature allows the edge-router to be aware of network changes and approximate threat locations without any assistance from a positioning system. The system evaluation has been carried out in test-bed and emulator. The system occupied 3315 bytes of ROM and 864 bytes of RAM in a node when the node was responsible for monitoring 8 nodes, confirming its low memory requirement. The developed system's capability of detecting packet flooding, selective forwarding, and clone attacks has been evaluated by introducing these attacks in the network. The nodes were able to detect the packet flooding attacks within 25 seconds and successfully ban the flooding nodes. Selective forwarding and clone attacks were also successfully detected and blocked through cooperation between nodes and router. Furthermore, the analysis of networks consisting of 20 and 50 nodes demonstrated that the developed detection system has low false positive rate. All these results confirmed the developed detection system is capable of detecting internal anomalies successfully with lower overheads, which makes it more appropriate for IoT.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] V. M. Rohokale, N. R. Prasad, and R. Prasad, "A cooperative internet of things (iot) for rural healthcare monitoring and control," in *Proceedings of the 2011 2nd International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology (Wireless VITAE)*, pp. 1–6, IEEE, 2011.

[2] Y. J. Fan, Y. H. Yin, L. D. Xu, Y. Zeng, and F. Wu, "IoT-based smart rehabilitation system," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1568–1577, 2014.

[3] B. Farahani, F. Firouzi, V. Chang, M. Badaroglu, N. Constant, and K. Mankodiya, "Towards fog-driven IoT eHealth: Promises and challenges of IoT in medicine and healthcare," *Future Generation Computer Systems*, 2017.

[4] L. Catarinucci, D. de Donno, L. Mainetti et al., "An IoT-aware architecture for smart healthcare systems," *IEEE Internet of Things Journal*, vol. 2, no. 6, pp. 515–526, 2015.

[5] S. M. Riazul Islam, D. Kwak, M. Humaun Kabir, M. Hossain, and K.-S. Kwak, "The internet of things for health care: a comprehensive survey," *IEEE Access*, vol. 3, pp. 678–708, 2015.

[6] B. Li and J. Yu, "Research and application on the smart home based on component technologies and internet of things," *Procedia Engineering*, vol. 15, pp. 2087–2092, 2011.

[7] S. D. T. Kelly, N. K. Suryadevara, and S. C. Mukhopadhyay, "Towards the implementation of IoT for environmental condition monitoring in homes," *IEEE Sensors Journal*, vol. 13, no. 10, pp. 3846–3853, 2013.

[8] A. Zanella, N. Bui, A. P. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, 2014.

[9] K.-L. Tsai, F.-Y. Leu, and I. You, "Residence energy control system based on wireless smart socket and IoT," *IEEE Access*, vol. 4, pp. 2885–2894, 2016.

[10] P. Sotres, J. R. Santana, L. Sanchez, J. Lanza, and L. Munoz, "Practical lessons from the deployment and management of a smart city internet-of-things infrastructure: The smartsantander testbed case," *IEEE Access*, vol. 5, pp. 14309–14322, 2017.

[11] Z. Yang, X. Wang, and H. Sun, "Study on urban its architecture based on the internet of things," in *LTLGB 2012*, pp. 139–143, Springer, 2013.

[12] J. A. G. Ibáñez, S. Zeadally, and J. Contreras-Castillo, "Integration challenges of intelligent transportation systems with connected vehicle, cloud computing, and Internet of Things technologies," *IEEE Wireless Communications Magazine*, vol. 22, no. 6, pp. 122–128, 2015.

[13] J. E. Siegel, D. C. Erb, and S. E. Sarma, "A survey of the connected vehicle landscape–architectures, enabling technologies, applications, and development areas," *IEEE Transactions on Intelligent Transportation Systems*, 2017.

[14] M. U. Farooq, M. Waseem, A. Khairi, and S. Mazhar, "A critical analysis on the security concerns of internet of things (iot)," *International Journal of Computer Applications*, vol. 111, no. 7, 2015.

[15] Z.-K. Zhang, M. C. Y. Cho, C.-W. Wang, C.-W. Hsu, C.-K. Chen, and S. Shieh, "IoT security: ongoing challenges and research opportunities," in *Proceedings of the 7th IEEE International Conference on Service-Oriented Computing and Applications (SOCA '14)*, pp. 230–234, IEEE, Matsue, Japan, November 2014.

[16] I. Cvitić, M. Vujić, and S. Husnjak, "Classification of security risks in the IoT environment," in *Proceedings of the 26th International DAAAM Symposium on Intelligent Manufacturing and Automation*, pp. 731–740, 2016.

[17] S. R. Moosavi, T. N. Gia, E. Nigussie et al., "End-to-end security scheme for mobility enabled healthcare internet of things," *Future Generation Computer Systems*, vol. 64, pp. 108–124, 2016.

[18] T. Kothmayr, C. Schmitt, W. Hu, M. Brunig, and G. Carle, "A DTLS based end-to-end security architecture for the internet of things with two-way authentication," in *Proceedings of the IEEE 37th Conference on Local Computer Networks Workshops (LCN '12)*, pp. 956–963, IEEE, October 2012.

[19] S. R. Moosavi, T. N. Gia, A.-M. Rahmani et al., "Sea: a secure and efficient authentication and authorization architecture for iot-based healthcare using smart gateways," *Procedia Computer Science*, vol. 52, pp. 452–459, 2015.

[20] T. Markmann, T. C. Schmidt, and M. Wählisch, "Federated end-to-end authentication for the constrained internet of things using ibc and ecc," *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 4, pp. 603-604, 2015.

[21] K. Ioannis, T. Dimitriou, and F. C. Freiling, "Towards intrusion detection in wireless sensor networks," in *Proceedings of the 13th European Wireless Conference*, pp. 1–10, 2007.

[22] G. Huo and X. Wang, "DIDS: A dynamic model of intrusion detection system in wireless sensor networks," in *Proceedings of the 2008 International Conference on Information and Automation (ICIA '08)*, pp. 374–378, IEEE, Changsha, China, June 2008.

[23] A. Abduvaliyev, A.-S. K. Pathan, J. Zhou, R. Roman, and W.-C. Wong, "On the vital areas of intrusion detection systems in wireless sensor networks," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1223–1237, 2013.

[24] T. Grandison and E. Terzi, "Intrusion detection technology," in *Encyclopedia of Database Systems*, pp. 1568–1570, Springer, 2009.

[25] F. Liu, X. Cheng, and D. Chen, "Insider attacker detection in wireless sensor networks," in *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM '07)*, pp. 1937–1945, IEEE, May 2007.

[26] I. Krontiris, Z. Benenson, T. Giannetsos, F. C. Freiling, and T. Dimitriou, "Cooperative intrusion detection in wireless sensor networks," in *Proceedings of the European Conference on Wireless Sensor Networks (EWSN '09)*, vol. 5432 of *Lecture Notes in Computer Science*, pp. 263–278, Springer.

[27] L. Wallgren, S. Raza, and T. Voigt, "Routing attacks and countermeasures in the RPL-based internet of things," *International Journal of Distributed Sensor Networks*, vol. 2013, Article ID 794326, pp. 1–11, 2013.

[28] J. P. Amaral, L. M. Oliveira, J. J. P. C. Rodrigues, G. Han, and L. Shu, "Policy and network-based intrusion detection system for IPv6-enabled wireless sensor networks," in *Proceedings of the 2014 1st IEEE International Conference on Communications (ICC '14)*, pp. 1796–1801, IEEE, Sydney, NSW, Australia, June 2014.

[29] S. Raza, L. Wallgren, and T. Voigt, "SVELTE: Real-time intrusion detection in the Internet of Things," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2661–2674, 2013.

[30] A. Le, J. Loo, K. K. Chai, and M. Aiash, "A specification-based IDS for detecting attacks on RPL-based network topology," *Information (Switzerland)*, vol. 7, no. 2, p. 25, 2016.

[31] P. Pongle and G. Chavan, "Real time intrusion and wormhole attack detection in internet of things," *International Journal of Computer Applications*, vol. 121, no. 9, pp. 1–9, 2015.

[32] Contiki: The open source operating system for the internet of things, http://www.contiki-os.org/.