BLACKHOLE ATTACKS IN IOT NETWORKS

A Thesis Submitted to
the Graduate School of Engineering and Sciences of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

in Computer Engineering

by
Barış SOKAT

December 2020
İZMİR

ACKNOWLEDGMENTS

I would like to express my gratitude to my advisor Prof. Yusuf Murat ERTEN for his continuous support and effort during my graduate studies. He has always mentored me with all his sincerity and kindness. I am grateful that I have the opportunity to work with him. I would also like to thank Assoc. Prof. Dr. Tolga AYAV for his support and supervision during my thesis work.

I would like to thank Prof. Dr. Cüneyt F. BAZLAMAÇCI and Assoc. Dr. Deniz KILINÇ for accepting to be a part of my thesis defense jury and sparing their valuable time to read and evaluate my thesis and provide me with feedbacks.

Also, I would like to thank my family. They always supported me at to do better all times and under all circumstances.

ABSTRACT

BLACKHOLE ATTACKS IN IOT NETWORKS

IoT technologies are very popular today, and they are used in almost every field. Therefore, the number of IoT devices used is increasing day by day. Like every field in computer networks, security is quite important in IoT networks. However, the resource-constrained nature of IoT devices makes the study of security measures for IoT networks very challenging.

In the literature research conducted before this thesis study, it was seen that it can perform a wide variety of RPL-based attacks on IoT networks. One of these attacks is the blackhole attack. Although the black hole attack is functionally simple, the damage it causes in the network can be extremely destructive. As far as it is known, in addition to the limited number of studies in this field of attack, the black hole attack used in the studies in this field has also basic features. The basic feature mentioned here is that the attacker node that will perform a black hole attack drops all the packets that come to it. When the attacker node drops all incoming packets, it causes the topology to change in the network, the number of control messages to increase and the attacker node to be isolated from the network in a short time. However, blackhole attack can be combined with different attacks. Therefore, in this thesis, the node that will perform the black hole attack is designed to allow control messages to pass, while dropping all other packets. Here, it is aimed that the attacker node remains on the network for a longer time. As a result, as long as the attacker node is active, it will be able to drop more packets and the number of control messages in the network will be controlled since the topology does not change. With the black hole attack developed as a result of the simulation tests, the number of control messages released in the network was taken under control and it was observed that the attacker node could remain in the network throughout the simulation period. Thus, the effect of different types of black hole attacks on the network that can be developed has been revealed.

ÖZET

IOT AĞLARINDA KARA DELİK SALDIRILARI

IoT teknolojileri günümüzde oldukça popüler olmasının yanında neredeyse her alanda kullanıldığı görülmektedir. Bunun sonucunda da kullanılan IoT cihazları sayısı günden güne katlanarak artmaktadır. Bilgisayar ağlarındaki her alan gibi IoT ağları da güvenliğe fazlasıyla ihtiyaç duymaktadır. Ancak IoT cihazlarının kaynak kısıtlı yapısı, IoT ağları için yapılan güvenlik önlemleri ile ilgili çalışmaları oldukça zorlu hale getirmektedir.

Bu tez çalışması öncesi gerçekleştirilen literatür araştırmasında IoT ağlarında RPL tabanlı çok çeşitli saldırılar gerçekleştirebildiği görülmüştür. Bu saldırılardan biri ise ağ içerisinden gerçekleştirilen karadelik (blackhole) saldırısıdır. Karadelik saldırısı işlevsel olarak basit bir yapıda olmasına karşın, ağda neden olduğu zararlar fazlasıyla yıkıcı olabilmektedir. Bilindiği kadarıyla bu saldırı alanında yapılan çalışmalar az olmasının yanında, bu alanda yapılan çalışmalarda kullanılan karadelik saldırısı temel özelliklere sahiptir. Burada konusu geçen temel özellik karadelik saldırısı gerçekleştirecek saldırgan düğümün kendisine gelen tüm paketleri düşürmesidir. Saldırgan düğümün kendisine gelen tüm paketleri düşürmesi ağda topolojinin değişmesine, kontrol mesajlarının sayısının artmasına ve kısa sürede saldırgan düğümün ağdan izole olmasına neden olmaktadır. Ancak karadelik saldırısı her ağ saldırısında olabileceği gibi farklı saldırılar ile birleştirilebilir. Bu nedenle gerçekleştirilen bu tez çalışmasında karadelik saldırısını gerçekleştirecek düğüm kontrol mesajlarının geçmesine izin verirken, diğer tüm paketleri düşürecek şekilde tasarlanmıştır. Burada saldırgan düğümün ağda daha uzun süre kalması hedeflenmektedir. Bunun sonucunda da saldırgan düğüm aktif olduğu sürece daha fazla paket düşürebilecek ve topoloji değişmediği için ağda kontrol mesajlarının sayısı kontrol altına alınacaktır. Yapılan simülasyon testleri sonucunda geliştirilen karadelik saldırısı ile ağda salınan kontrol mesajlarının sayısı kontrol altına alınmış ve saldırgan düğüm simülasyon süresi boyunca ağdan izole olmadan kalabildiği görülmüştür. Böylece geliştirilebilecek farklı tip karadelik saldırısının ağdaki etkisi ortaya çıkarılmıştır.

LIST OF CONTENTS

LIST OF FIGURES

Figure **Page**

LIST OF TABLES

Table **Page**

LIST OF ABBREVIATIONS

6LoWPAN	IPv6 over Low-Power Wireless Personal Area Networks
AMI	Automated Metering Infrastructure
DAG	Directed Acyclic Graph
DAO	Destination Advertisement Object
DIO	DODAG Information Object
DIS	DODAG Information Solicitation
DODAG	Destination-Oriented Directed Acyclic Graph
ETX	Estimated Transmission Count
ICMPv6	Internet Control Message Protocol for IPv6
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IDS	Intrusion-Detection Systems
ICMPv6	Internet Control Message Protocol for IPv6
IPv6	Internet Protocol version 6
IoT	Internet of Things
LLN	Low-Power and Lossy Networks
RAM	Random Access Memory
ROM	Read-only Memory
RPL	Routing Protocol for Low-Power and Lossy Networks
OF	Objective Function
UDP	User Datagram Protocol
WSN	Wireless Sensor Network

CHAPTER 1

INTRODUCTION

Computer networks have been developing from the first years of their emergence to the present day. While they were used as military technologies in the first period of its emergence, it has been adopted to personal and commercial use today, thus it has found a place for itself in every aspect of our lives. The biggest contributors to this development are academic studies in the field. Due to the development of computer networks and the benefits coming from this development, to increase the usage area of the Internet has been one of the goals of academic studies and private-sector research. One of the technologies developed for this purpose is Wireless Sensor Networks (WSN). With the creation of WSNs, easy cross-device communication with WSN networks is provided where traditional computer network technologies cannot be implemented.

1.1. The Internet of Things (IoT)

The Internet of Things (IoT) is a communication network that can transfer data over a network without any need for human interaction. Devices in IoT networks generally send a sensor data they collect from their environment to a sink node used as an IoT gateway. While these sink nodes can store and process the data they receive, they can also send them to other media over the Internet. IoT devices communicate with each other in some applications and exhibit the behaviors they are programmed with according to the data they receive from each other. These devices can also interact with humans, even though they do most of their tasks without human intervention. IoT devices can easily be linked with each other or with larger systems. IoT provides Internet connectivity to restricted devices by marking them with unique identifiers.

The main purpose of IoT, built on a complex network that connects billions of devices and people to multi-technology, multi-protocol, and multi-platform infrastructure, is to bring together physical and digital, and many other areas of our daily life to create smart environments that provide more intelligence (Liñán Colina et al. 2016). However, standard and protocol development studies for IoT networks are quite challenging due to resource-constrained and different types of device in the network. On the other hand, increasingly, organizations in a variety of industries are using IoT technology to work more efficiently, improve decision making, and increase the value of the business. Therefore, the popularity of IoT studies has been increasing nowadays.

Low Power and Lossy Networks (LNNs) which can be created by IoT devices, are formed by the combination of many resource-constrained (energy, memory, processing) embedded networking devices (Vasseur 2014). These devices can be linked to each other in different ways. These links can be Institute of Electrical and Electronics Engineers (IEEE) 802.15.4 or low-power Wi-Fi (Vasseur 2014).

1.2. Wireless Sensor Networks (WSN)

The network formed by many sensors that communicate with radio signals is called Wireless Sensor Networks (WSN). Devices in this type of network autonomously create the network and flow their data. In addition, they can change the network topology according to the state of the network. Sensors used in WSNs are small, cheap, and uncomplicated devices, and generally resources are limited. They work with the small batteries on them and these batteries have limited energy. They also have a low-capacity microcontroller, Random Access Memory (RAM), and Read-only Memory (ROM). Therefore, they have low computing and storage capacity. (Dunkels, Grönvall, and Voigt 2004)

Due to the nature of wireless connections and resource-constrained devices, connections between sensors have high loss rates and low throughput. High loss rate and low throughput can be caused by overloaded, selfish, malicious, and broken nodes, as detailed in the study (Marti et al. 2000). In addition, since the network topology is

created automatically, there is no traffic pattern defined before. Therefore, the traffic pattern between devices can be point-to-point or point-to-multipoint (Mayzaud, Badonnel, and Chrisment 2016).

In a typical WSN implementation, multiple sensor nodes are randomly distributed across the network and these nodes can be easily relocated. WSN networks are generally created to monitor and record physical and environmental conditions. The data collected by the nodes are sent to the sink node. It is then used here for editing and processing. The sink nodes have higher energy and computation power than other nodes in the network. These networks act as a bridge between the physical world and the virtual world.

The devices in WSN addressed using Internet Protocol version 6 (IPv6) are small sensor nodes with limited computing resources, energy, and memory. The protocols used in such networks are created by adapting from existing networks to suit the structure of the constrained network (Gutierrez et al. 2001). For example, a special task group by the Internet Engineering Task Force (IETF) has defined the header compression and framing technique to make it possible to communicate between sensor nodes. This technique is called IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) (Gutierrez et al. 2001).

IP-connected WSNs connect directly to the untrusted Internet (Raza, Wallgren, and Voigt 2013). WSNs can be considered as a subnet of IoTs. Many applications that are created with IoT devices have this feature. However, being able to connect to IoT networks over the Internet makes these resource-constrained devices vulnerable to external attacks (Raza, Wallgren, and Voigt 2013). Since IoT applications can be used in important areas such as health and automation, it is very important to take security measures against external attacks, in spite of the structural challenges they have.

1.3. Motivation

It is often very challenging to try to integrate protocols used in traditional computer networks into devices with limited resources, such as IoT devices because trying to add all the features of the protocols to such devices causes the low resources of

the devices to be consumed in a short time. Although the IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) in the Contiki-ng operating system, which was developed as open-source, guarantees reliable and continuous packet transmission over the shortest path, it has been revealed by studies (Dunkels, Grönvall, and Voigt 2004) that it is also vulnerable to many attacks from within the network (Mayzaud, Badonnel, and Chrisment 2016; Pongle and Chavan 2015; Yang et al. 2019). For this reason, security-related features are usually ignored or tried to be provided in different ways.

Although studies of Blackhole Attack on IoT networks have gained popularity in recent years, the threat of blackholes against RPL networks has always existed. As far as it is known, the most basic form of blackhole attack has been used in most studies. Network analyzes have been made and Intrusion-Detection Systems (IDS) systems have been created over such types of blackhole attacks. As the popularity of IoT technologies grows with each passing day, it will interest hackers more and more, and thus new types of weaknesses will continue to emerge. For this reason, in addition to creating new IDS systems, it is very important to carry out studies that identify new vulnerabilities. Therefore, in this thesis study, the results of the effects of the blackhole attack on the network have been analyzed by adding new features to the basic blackhole attack. In this study, we introduced a blackhole which only drops User Datagram Protocol (UDP) packets but passes the control packets untouched which makes the attack more difficult to detect. We have analyzed the behavior of the network under these circumstances and presented the results of network performance in the proceeding chapters.

1.4. Outline of the Thesis

The thesis is organized as follows: Chapter 2 gives information about RPL's basic features and blackhole attack logic. Chapter 3 provides the general literature overview about blackhole attack effect on RPL networks and IDS mechanism against blackhole attack. The experimental results are described in Chapter 4. Chapter 5 gives details about performance analysis and results. We conclude the paper in Chapter 6.

CHAPTER 2

BACKGROUND

The IoT devices mentioned earlier can communicate with each other autonomously. Routing protocols play an important role so that these devices can communicate with each other. Although there are many types of routing protocols that can be chosen to be implemented on a network, they all serve the same purpose. This goal is to ensure that the packets reach the correct destination using the most optimal path (Winter et al. 2012). When the Internet of Things (IoT) is considered these optimal paths may be the paths from the node to the RPL Root with the least Estimated Transmission Count (ETX). However, routing protocols to be used in networks such as LLN should consist of lightweight protocols suitable for the constrained structure of the network.

2.1. Routing Protocol for Low-Power and Lossy Networks

RPL is a distance vector routing protocol used on LLN networks. RPL is designed to use IPv6. Devices connected using RPL form the Destination Oriented Directed Acyclic Graph (DODAG) topology. This topology is loop resistant and the data generated by the nodes flows towards a single destination (DODAG Root) (Winter et al. 2012).

Contiki-NG provides two RPL implementations with different features: RPL Classic and RPL Lite (“Contiki-Ng Wiki” n.d.). The RPL Classic was created as a continuation of the original ContikiRPL. However, it has been complicated and has a large ROM footprint because it supports too many RPL features. Some of these features include multiple instances and DODAGs support, storing and non-storing mode, multicasting (“Contiki-Ng Wiki” n.d.). RPL Lite is built to cover basic features to be more lightweight than RPL Classic. In this implementation, it eliminates the complexity

of RPL Classic by removing features such as storing mode, multiple instances and DODAGs. In this way, a more stable and better performance RPL implementation with smaller ROM footprints has been created. Also, RPL Lite is the default RPL implementation for Contiki-NG ("Contiki-Ng Wiki" n.d.).

As mentioned above, there are several modes used in RPL implementations. These modes specify how the downward routing process will be done. These mods cause various results such as ROM usage and package size, depending on their features ("Contiki-Ng Wiki" n.d.). The first of these modes is the storing mode. In this mode, each node on the network stores routing tables with route information. However, this feature causes extra resource usage for nodes. The second mode is the non-storing mode. However, in this mode, if there is more than one intermediate node in the path of the packet, this downward routing may increase the size of the packets.

DODAG is created according to the metrics and defined constraints contained in Objective Function (OF). That is, OF defines how to select routes between nodes in the network and ensures that these routes are selected in an optimized way (Winter et al. 2012). Thus, nodes choose the most suitable neighbor node as parent according to the metrics in OF.

OF needs to know the position of nodes relative to root to avoid routing loop when choosing paths. For this, it calculates a Rank value for each node. This Rank shows the relative position of nodes in relation to Root and its value increases as it gets further away from the Root. Although the rank value is calculated according to the metrics and constraints in the OF, it can usually be equal to simple hop-count distance (Tsvetkov and Klein 2011). Accordingly, the increase in Rank value seen as you move away from Root should reflect the increase in metric value.

2.1.1. ICMPv6 Control Messages

As explained in detail in the study (Winter et al. 2012), four auxiliary control messages have been defined on RPL networks. These control messages play a major role in establishing and maintaining the topology. The first control message is DODAG Information Object (DIO). This control message stores the RPL instance information

and distributes it to other nodes. This information contains information such as RPL instance information, the current Rank of the node, and the Root IPv6 address. DIO is first created by the Root node and broadcast to the entire network. The second control message is DODAG Information Solicitation (DIS). The node wishing to join an RPL network starts broadcasting DIS messages to request DIO messages from neighbor nodes. In response to the DIS message, the neighboring node which receives DIS message begins sending a DIO message which contains RPL information of its network. The third control message is Destination Advertisement Object (DAO). This message is sent directly to the Root by the nodes that selected it as its parent. The message contains information about the selected parent and this information is recorded in the RPL routing table by Root. DAO-ACK is the last control message generated by the root in response to the DAO message indicating that the selected parent has been approved by Root. Joining the network process is finished after the DAO-ACK message is received by the node. All these control messages are defined in RPL and are used for establishing and maintaining the RPL network.

2.1.2. DAG Creation

The DODAG creation process begins with the periodic advertising of DIO messages by the predetermined Directed Acyclic Graph (DAG) Root. These intervals are determined by the Trickle timer. A trickle timer mechanism is used to reduce overhead in traffic by regulating the transmission of control messages (Levis et al. 2011). It does this process by dividing the time periodically. When a node's data does not match its neighbors, that node increases the rate of control messages to resolve the inconsistency, otherwise, they will increase the window size exponentially, slowing down the sending rate. Therefore, instead of filling a network with packets, the algorithm controls the sending rate so that each node receives packets small enough to remain consistent.

Other nodes that hear DIO messages can choose to join the DAG. Those nodes that want to join the DAG must first choose a parent in the network that has communication with Root. If the node can connect directly with root, it will select root

as a parent. Before selecting a parent from among neighbors, the node must be sure of the quality of the connection with his parent that will be chosen. This quality is estimated with the help of Objective Function and metrics.

After the node has chosen a parent for itself, the path it chose [non-storage mode] to join the network must be registered by Root ("Contiki-Ng Wiki" n.d.). For this purpose, it must send the DAO control message to the Root. After the root receives the DAO message, it stores the relationship between the node and its parent in the routing table. A DAO-ACK message that states that the DAO message has been approved is then sent back to the node from Root. The node that receives the DAO-ACK message now knows that it is now included in an RPL network. It then starts sending its own DIO messages to the network so that new nodes can be added to the network. This process continues until there are no more accessible nodes outside of the network.

If a new node wants to join an existing network, it starts broadcasting DIS messages. DIS message increases the probability of the node receiving a DIO message. If the DIS message is heard by any node in the network, it starts broadcasting the DIO message in response. Then the node that evaluates the DIO messages it receives from its neighbors joins the network by choosing a parent for itself. Thus, a DODAG is created in all aspects.

2.1.3. DODAG Maintenance

Another feature of the RPL network is its reliability (Winter et al. 2012). For this, it ensures that the paths created between the nodes are kept up to date. Nodes in LLNs consist of source constrained devices. For this reason, deterioration may occur in the selected paths. At the same time, the mechanisms to detect these deteriorations should not burden the resources of these restricted devices. Two repair mechanisms have been designed in RPL to maintain the created network (Winter et al. 2012). The broken link is fixed by the Global Repair mechanism initiated by DODAG Root to rebuild the entire topology by sending a new sequence number to the network. After receiving the new DIO message, the nodes restart the parent selection and update their rank. If a local node suffers from a broken connection and does not want to wait too

long for Global Repair, the Local Repair mechanism is executed. The node that requests repair to update information about its parents' requests new DIO control messages from neighboring nodes.

2.2. Blackhole Attack

A wide variety of attacks can be performed on LLNs over different layers (Le et al. 2012). In addition, it is very challenging to take security measures in protocols developed for LNNs due to the fact that different types of devices in LNNs can be together and the limited structures of these devices. Cryptographic defense systems, on the other hand, only protect against external attacks, as well as bringing additional payload to nodes within the network. This causes created networks to be vulnerable to many attacks.

Since the priority in RPL is to ensure that packets are forwarded to the correct destination through an optimal path, it is vulnerable to attacks except having a few security mechanisms (global/local repair mechanisms, loop avoidance and detection techniques) (Mayzaud, Badonnel, and Chrisment 2016). Although RPL attacks are very diverse, it has been observed that they are performed against resources, topology, and traffic (Mayzaud, Badonnel, and Chrisment 2016). A blackhole attack on the topology of the network causes nodes or subtree to be isolated from the network or packets to be delayed (Mayzaud, Badonnel, and Chrisment 2016). The blackhole attack over the network layer is basically carried out by the attacker node by silently dropping all the packets sent to it to be forwarded to another node.

Although the blackhole attack looks structurally simple, it is very difficult to notice because the nodes performing the attack do nothing but drop packets. The attacker node can remain unnoticed in the network for a long time while performing packet dropping. Also, when combined with different types of attacks, the effects of the attack on the network can be quite devastating. Although the Blackhole attack is difficult to notice, it leaves a footprint on the network like any network attack. Therefore, the functioning and structure of the RPL network must be analyzed very well and any anomalies that may occur must be defined.

Nodes that perform a blackhole attack are usually placed on the network to drop all packets that come to them. For this purpose, the nodes placed in the network cause the RPL topology to be corrupted, as they will also drop the ICMPv6 control messages. This way, the path where the malicious node is located can be treated as a broken path by the neighboring nodes. This change in topology pushes the nodes in this region to search for a new parent. Nodes in search of a new parent will advertise plenty of DIO packages in the network. That is, the increase in the number of DIO messages can be considered as a sign of instability in the network (Chugh, Lasebae, and Loo 2012). For this type of attack, the change in DIO messages can be used as a parameter to detect the attack, although it is not the only parameter to be used for attack detection.

Another option is that the malicious node drops only the specific packet type. This can be accomplished by transmitting ICMPv6 control messages and dropping other packets. This method allows the attacker node to drop packets without damaging the topology. Thus, an increase in the number of ICMPv6 control messages is not observed in the network, making it difficult to detect the malicious node.

Studies in the field of blackhole attacks on RPL networks may be considered new compared to other fields. For this reason, as far as it is known, blackhole attacks, which has generally basic features, have been analyzed or an IDS system has been created against this type of attack, except for a few studies. In this study, we have modified the RPL protocol to implement the blackhole attacks where only the data packets are dropped. We then analyzed the network behavior to devise a blackhole attack detection method.

CHAPTER 3

RELATED WORK

As stated before, RPL is a routing protocol that is used as a standard in IoT networks. Although there are specific defensive features brought by the RPL structure, the RPL is very vulnerable to attacks from within the network (Mayzaud, Badonnel, and Chrisment 2016; Yang et al. 2019). Due to the difficulties of the resource-constrained structure of IoT devices, it is very challenging to develop detection mechanisms for these attacks. Nevertheless, in recent years, many attacks have been identified through RPL (Mayzaud, Badonnel, and Chrisment 2016; Yang et al. 2019) and detection mechanisms have been developed for these attacks (Raza, Wallgren, and Voigt 2013; Ahmed and Ko 2016; Jiang, Liu, and Dezfouli 2019; Sahay et al. 2018; Neerugatti and Reddy 2019; Zhang et al. 2019; Airehrour, Gutierrez, and Ray 2017). One of the internal attacks on the RPL is the Blackhole Attack. Although the logic of the Blackhole Attack is simple, its impact on the network can be devastating.

The articles in reference (Chugh, Lasebae, and Loo 2012; Kumar, Matam, and Shukla 2016), one of the effects of Blackhole Attack on the network is that it causes nodes to change parents. Nodes looking for a new parent begin sending DIO packets to the network. Therefore, it is possible to comment on the state of the network via DIO packets. The increase in the number of DIO control messages on networks using RPL routing protocol reflects instability in network routing topology, while the decreasing frequency of DIO control messages reflects a stable network routing topology. At the same time, the scenario that sends the data packets produced by the attacker node shows an 8% increase in the total number of DIO packets exchanged between nodes, while fewer DIO packets are exchanged in the scenario in which the attacker node does not create a data packet, resulting in a faulty stable network routing topology (Chugh, Lasebae, and Loo 2012).

Another effect of the Blackhole Attack on the network is the delays in the arrival of packets to the Root node. Nodes that have already determined their parent may be affected by the attack and have to choose a less optimal route than the one they

previously selected. In addition, the waiting times of packages may increase with the increase in buffer queues on some paths. This leads to increased delays in packages. This delay occurs approximately 4.3 times more (Chugh, Lasebae, and Loo 2012; Kumar, Matam, and Shukla 2016).

Based on the impact of the Blackhole Attack on IoT networks, an Intrusion Detection System (IDS) can be created to prevent the Blackhole Attack. The IDS mechanism to be created detects an attack by analyzing network activity. IDS mechanisms can be a wide variety (Raza, Wallgren, and Voigt 2013). These IDS systems can be placed on IoT nodes on the network when needed.

In Ref. (Ahmed and Ko 2016), the study divided the IDS system into a two-step process: Local Decision and Global Verification. In the Local Decision process, a node identifies the suspicious node based on information gathered about the behavior of its neighbors. If a node does not hear the packet transmitted from its neighbor at a time greater than the threshold, that node is identified as a suspect. Once the suspicious node is identified, it undergoes a second process for further investigation by its neighbors. During this process, data packets are transmitted using an alternative route. A trust-based IDS mechanism proposed in (Airehrour, Gutierrez, and Ray 2017). The basis of this mechanism is based on calculating a trust value for all nodes on the network. Depending on this trust value, nodes perform parent selection. In the article (Neerugatti and Reddy 2019), the packet drop ratio of each node is calculated and sends the threshold value to the Root node with the help of DAO control message. DAO-ACK packets are then sent by the root node to test the threshold values. If a node has a high packet drop ratio, the malicious node is detected and removed from the network.

No matter how many IDS systems that can successfully detect Blackhole Attacks are created, it is not a good practice in real life to assign tasks to IoT nodes in the network to perform the IDS mechanism. This is because bringing additional loads to IoT devices that already have a limited structure will force these devices and they will not be able to perform the tasks they need to do (Jiang, Liu, and Dezfouli 2019). In addition, relying on the neighboring nodes of the attacker node is not a 100% effective practice. This is because adding multiple nodes to the network to collaborate with each other to perform a Blackhole Attack will make it difficult for these IDS systems to detect attacks (Zhang et al. 2019). Such reasons have led researchers to search for new IDS system models.

Another method in which the IDS system can be created is to perform detection through the root node. The reason for using this method is that the detection system to be added will not affect the operation of the Root node, since the Root node is a more powerful node than other nodes in the network. In Ref. (Jiang, Liu, and Dezfouli 2019), root regularly monitors the network packet flow and looks for anomalies. It performs this tracking process by assigning a sequence number to each message and checking the correctness of that sequence numbers. If any mismatch is detected in the message sequence numbers, this indicates an anomaly. If the root node detects the attacker node, it broadcasts the IP address of the attacker node to the entire network. Thus, the attacker node is excluded from the network.

In addition, applications of different disciplines can be combined when creating IDS mechanisms. In this study (Sahay et al. 2018), an algorithm is proposed to detect malicious nodes that trigger blackhole attacks in real-time using the Exponential smoothing technique. The purpose of this algorithm is to estimate the time of the next node that the root node will receive and to call "detection event" to detect malicious nodes on the network based on deviations at this time. In another study, a cuckoo filter based RPL has been proposed to protect the Automated Metering Infrastructure (AMI) network against Blackhole Attacks (Zhang et al. 2019). A whitelist is created with the hash table created with this method. This list is then sent to the entire network with the root signature. Nodes are not allowed to change this table. Whenever a DIO message arrives at a node, the source address of the DIO message is compared with the list. DIO messages that are not in the list are dropped by the node.

To detect the Blackhole Attack, the impact of the attack on IoT networks and the structure of the RPL routing protocol must be well examined. Awareness of these features is crucial to creating IDS systems based on anomalies that the attack will give to the network. To this day, studies have generally developed IDS mechanisms on attacks that contain basic Blackhole Attack features. However, every day the attacks on the network are evolving, combining different purpose attacks. Therefore, in this study, it has been our main goal to develop the more complicated Blackhole Attack and analyze the new effects that may occur in the network. Thus, we carried out such a study in order to be a resource for the IDS systems to be created in future studies.

CHAPTER 4

EXPERIMENTAL DESIGN AND METHODOLOGY

Although studies of Blackhole Attack on IoT networks have gained popularity in recent years, the threat of blackholes against RPL networks has always existed. As far as it is known, the most basic form of blackhole attack has been used in most studies. Network analyzes have been made and IDS systems have been created over such types of blackhole attacks. As the popularity of IoT technologies grows with each passing day, it will interest hackers more and more, and thus new types of weaknesses will continue to emerge. For this reason, in addition to creating new IDS systems, it is very important to carry out this type of studies that identify new vulnerabilities. Therefore, in this thesis study, the results of the effects of the blackhole attack on the network have been analyzed by adding new features to the basic blackhole attack.

Contiki-NG and Cooja simulator, which is the current version of the Contiki-OS operating system, were used in this study. Contiki-ng is an operating system used for resource-constrained devices in IoT applications. Since Contiki-NG is open-source, a malicious operating system to perform the blackhole attack has been created by modifying the source code. The changes made on this operating system were made in a way that does not obstruct the natural functioning of the system, and at the same time, they are minor changes that do not cause a burden on the system. Nodes with this malicious operating system have been created to drop packets that arrive at them. However, the ability to activate/deactivate the blackhole attack has been added so that the malicious node compiled with the malicious operating system can be used in every scenario created. Therefore, in cases where the blackhole attack is not requested, the attack can easily be stopped.

The malicious operating system that will perform the blackhole attack has been tested in the Cooja simulation environment. There are many types of motes that can be chosen in the Cooja environment. These motes can be selected according to the type of mote desired to be used in real life. However, the resource of each node in the ooja environment is different. In this thesis, the code snippet performing the blackhole attack

alone does not put a load on the Contiki-NG operating system. However, when the code snippets that will give the log outputs required for the creation of the test environment are added to the compiled operating system, it causes the ROMs of the selected motes to be overloaded. For this reason, a simulation has been carried out on "Cooja mote", which is the native mote type of Cooja, in order to prevent the ROM overload by selecting the motes such as SKY and Z1. However, regardless of the thesis work, if only the developed blackhole attacks are desired to be performed, choosing the desired type of node will not be a problem.

The client and server nodes, from nodes created for testing in the Cooja environment are compiled with the original Contiki-ng, while the malicious node is compiled with malicious Contiki-ng. The network created on the Cooja simulator has 1 root (server) node, 1 malicious node and 6 client nodes. The Malicious node is compiled using the malicious Contiki-ng, while the remaining nodes are compiled with the original Contiki-ng. The network topology is as shown in Figure 3.1. The node in which is green with ID: 1 indicates the Root (server) node, the purple node with ID: 2 indicates the malicious node, the remaining yellow nodes with ID: 3 to ID: 10 indicates the client nodes. These nodes with the specified IDs are used in the same way in all scenarios. In the following parts of the article, the node used as Node-1 represents the Root (Sink) node, the node used as Node-2 represents the Malicious node, and the representations between Node-3 and Node-10 represent all the remaining client nodes. The task of the client nodes created for this study is to send UDP packets to the Root node. These UDP packets are set to send 1 packet per minute. These nodes are randomly distributed within the network.

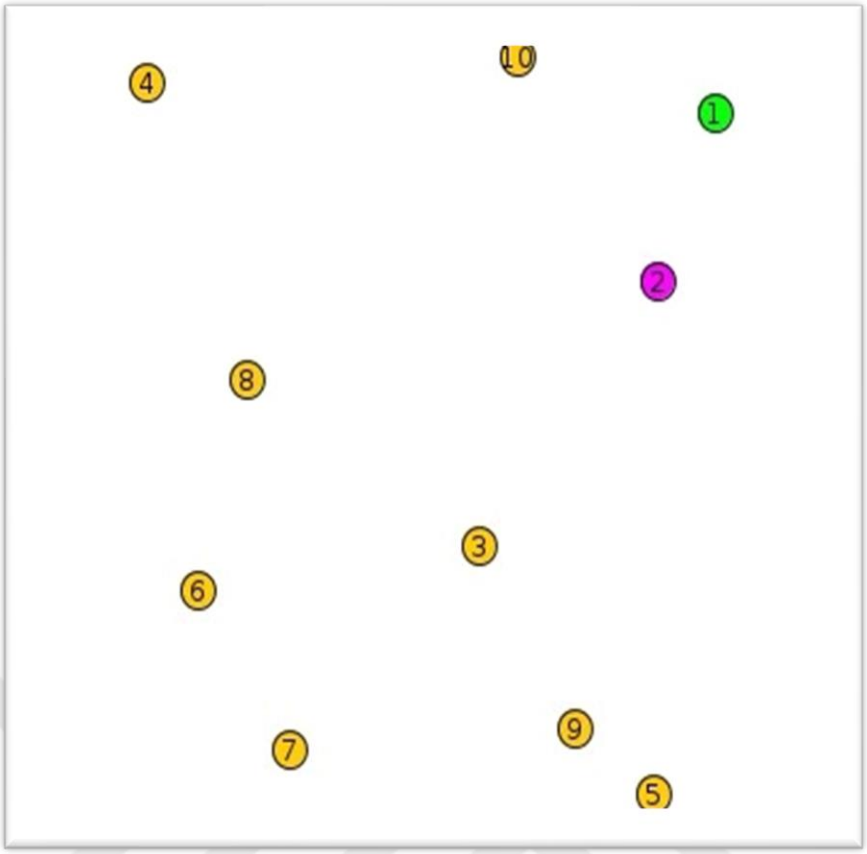

Figure 3.1. Network Topology

A screenshot of the Cooja simulator is shown in Figure 3.2. The "Network" window, shown with 1 in Figure 3.2, is where nodes are placed, and network topology is created. Here, nodes can be placed in any desired position. The "Mote Output" window shown with 2 is the Log Listener plug-in, where messages from nodes are written. The node messages here can then be saved collectively as a text file. In this study, text files collected in each run of scenarios were used for later analysis. The "Simulation Control" window shown with 3 is where the simulation is controlled. Here, at the same time, the speed of operation of the simulation can be adjusted to the desired extent. The "Notes" window shown with 4, is where notes can be kept in the simulator. These windows described above are sufficient for this thesis study.

In addition to what is mentioned in the above paragraph, a new simulation can be created, the previously created simulation can be reopened or saved with the "File" tab. With the "Simulation" tab, the simulation can be run or reloaded. With the "Motes"

tab, a new mote of any type can be created and added to the network. Tools owned by Cooja can be opened with the "Tools" tab. Windows such as "Network", "Mote Output" mentioned before are opened with this "Tools" tab. Finally, you can add/remove new extensions to the Cooja environment with the "Settings" tab. As we can see, although Cooja has a simple structure, it is a well-prepared simulator for testing IoT networks.

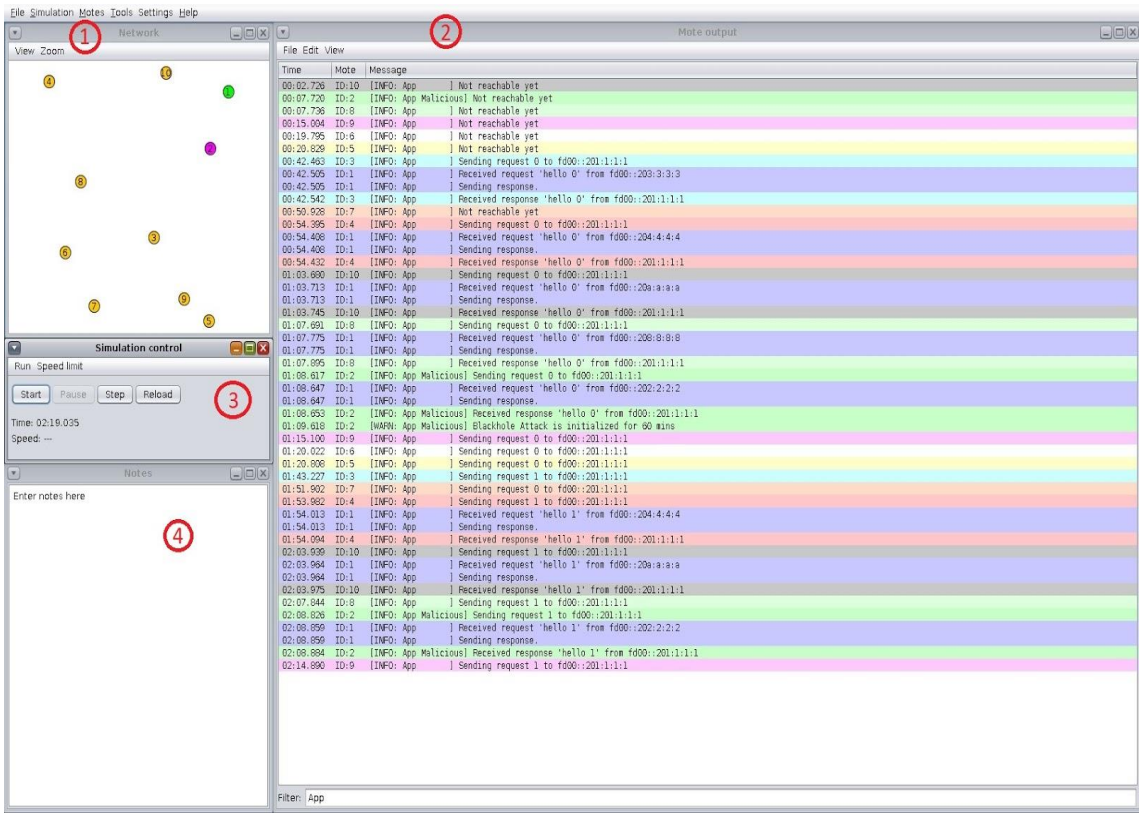

Figure 3.2. Screenshot of the Cooja Simulator

Three scenarios were tested in the network created in the Cooja simulator. The average of the data obtained by running each scenario 10 times for 60 minutes was used to show the results of the scenarios. In the network operated for 60 minutes, logging is done with the Log Listener plug-in by writing the number of packets entering and leaving each node every 3 minutes. For example, if we look at the UDP packets generated by a client node, it is expected to send 3 packets to the Root node every 3 minutes. This means that 60 UDP packets are produced after 60 minutes. However, when the simulation is run, this number may not be 60 for each node due to the delay of packets.

In Scenario 1, the network was run without the blackhole attack. The malicious node in this scenario forwards the incoming packets to the Root node without dropping any packets. Scenario 1 is created to understand the natural operation of the network and compare it with other scenarios.

The malicious node in Scenario 2 performs the blackhole attack. The blackhole attack here is done by dropping all packets coming to the node. The attack here has the basic blackhole structure, which drops all packages coming into the malicious node. To create this attack, firstly, how Contiki-ng reacts when a packet arrives at a node was analyzed. Then, as a result of the reactions of Contiki-ng, the place where the malicious code should be used was determined. Since the attack here takes place without any packet separation, the malicious code that will cause the packets to be dropped is set to run on every packet arrival. This scenario was created to observe the effects of the blackhole attack used in Scenario 3 on the network according to the basic blackhole attack. In the following parts of the study, this type of blackhole attack used in scenario 2 will be mentioned as basic blackhole attack.

On the other hand, the malicious node in Scenario 3 performs the blackhole attack in a way that drops all other packets while transmitting ICMPv6 control messages. In the attack in this scenario, Contiki-ng was first allowed to decapsulate packets coming to the node. After Contiki-ng identified the package for us, the malicious code was placed to drop the packets of the desired type. In both scenarios, it allowed Contiki-ng to execute its main tasks, allowing the blackhole attack to occur with a few lines of code. Thus, the overflow of Contiki-ng's source code is prevented. As a result, this data obtained from Scenario 2 and Scenario 3 are compared with Scenario 1. In the following parts of the study, this type of blackhole attack used in Scenario 3 will be mentioned as extended blackhole attack.

In order for the blackhole attack to be performed on a network to be successful, the attacker node must be chosen as the selected parent by the surroundings nodes. Regardless of the attacker node's position in the network topology, the Rank value of the node has been manipulated to ensure that the attacker node is chosen as the selected parent. For this reason, when testing the extended blackhole attack, the Rank value of the node that will perform the attack is prevented from automatically calculating, and the lowest possible value is directly assigned to it. Thus, the chance of the attacker node being selected as the selected parent is greatly increased. Although this method is not a

completely recommended way, it has been found to work because it significantly reduces the time taken to calculate the optimal rank value for the node itself. Also, this method does not have any effect on the success of the extended blackhole attack, it is only an attempt to increase the impact of the attacker node on the network.

Malicious nodes in Scenarios 2 and 3 initiate the blackhole attack from the moment it is included in the network. However, due to the structure of the developed attack code, the blackhole attack can be started at the desired time. This attack can switch to standby mode for a specified time after a defined time. However, this feature was not used actively in this study. In the 57th second after the simulation starts, the malicious node is accessible to the network and the blackhole attack is launched in the 58th second. The client nodes in all scenarios send a UDP message to the Root node every 1 minute starting from the 1st minute. These UDP messages correspond to real-life data generated by sensor nodes measuring environmental conditions. Each time the client node sends a UDP message, it also adds an output to the log record. The Root node that receives the UDP message also adds an output to the log record that it received the message. Thus, it is understood that the UDP message coming out of the client node has reached the Root. The Root node, which receives UDP messages, sends the same message back to the client node in response and reports this by adding output to the log record. The client node that receives this reply message adds an output to the log record when it receives the message. As a result, we understand that a package from Root has reached the client node. The effectiveness of the Blackhole attack is seen with missing outputs. Finally, Java code that can be transferred to Excel tables in the collected Log files has been developed for this thesis study.

CHAPTER 5

EXPERIMENTAL RESULTS

In this study, the following results were generally obtained as a result of all scenarios being carried out through Cooja simulation. As mentioned in the previous section, each scenario has been run 10 times and for 60 minutes. In addition, logging was performed with 3-minute periods. The generated graphs and tables show packets processed for 3 minutes each. First, the data of the network that is run with Scenario 1 is put together. The data from Scenario 1 shows the results of the normal operation of the network and was used for benchmark purposes to compare with other scenarios. Scenario 2 was then run to analyze the results of the basic blackhole attack with basic characteristics. Finally, the data was collected by running Scenario 3, which includes the extended blackhole attack with more complex features. As a result, the data of Scenario 3 were compared with scenario 2 and scenario 1, revealing the effects of the developed attack with complex features on both the network and the consequences that could not be seen in the basic blackhole attack.

5.1. Dropped Packets

Since the purpose of the malicious node created to perform the attack in this thesis is to drop the packets sent to it, the number of dropped packets should be examined first. The total number of packets dropped in Scenario 2 in Table 4.1 is more than Scenario 3. However, it should be noted here that all the packets sent to the malicious node are dropped in Scenario 2, while only UDP packets are dropped in Scenario 3. In Figure 4.1, it is seen that in Scenario 2, an average of 22 packages was dropped until the 30th minute, while an average of 47 packages was dropped after the 30th minute. In order to understand where the dropped packet increase is coming from, it is necessary to examine the nodes affected by the attack. This is because UDP packets sent by nodes affected by the attack, as well as other control messages, are dropped by

the malicious node. However, in scenario 3 the attacker node only dropped UDP packets, and the average of 18 dropped packets came from 6 nodes. It was also observed that the number of packets dropped in scenario 3 in Figure 4.1 did not change during the running period of the simulation.

Table 4.1. Dropped Packets by Malicious Node in Scenarios

Scenarios	**Scenario 1**	**Scenario 2**	**Scenario 3**
Total	0	724	355

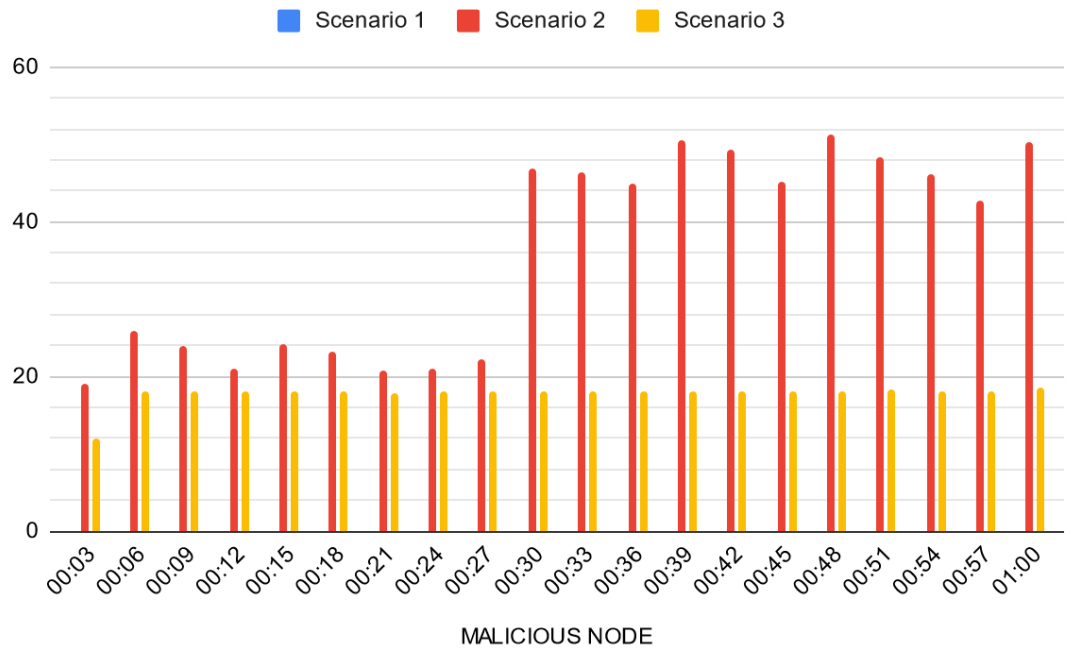

Figure 4.3. Dropped Packets by Malicious Node in Scenarios

5.2. What Happened in 30. Minute?

It was observed that the RPL protocol in the scenarios implemented, renewed the preferred parent selection of the nodes in the network with Global Repair in the 30th minute. Nodes that re-select their preferred parent notify their choices to the Root with DAO packages. Therefore, the effect of preferred parent selections on the network is seen by the increase in the number of DAO packets coming to Root. This can be seen in Figure 4.2 with the increase in the DAO packets at 30th and 60th minutes. As can be seen, there was an increase in the number of DAO packets coming to Root in these minutes. While the number of DAO packages received by the Root in Scenario 1 and Scenario 3 was equal to each other in the 3rd and 30th minutes, the number of packages received in Scenario 2 increased and even DAO package production continued at the 33rd and 42nd minutes. In order to understand the reason for this situation, the change in preferred parent selection of nodes should be examined. In addition, the effects of Global Repair, which is performed in the 60th minute, are ignored because the simulation is completed at that time.

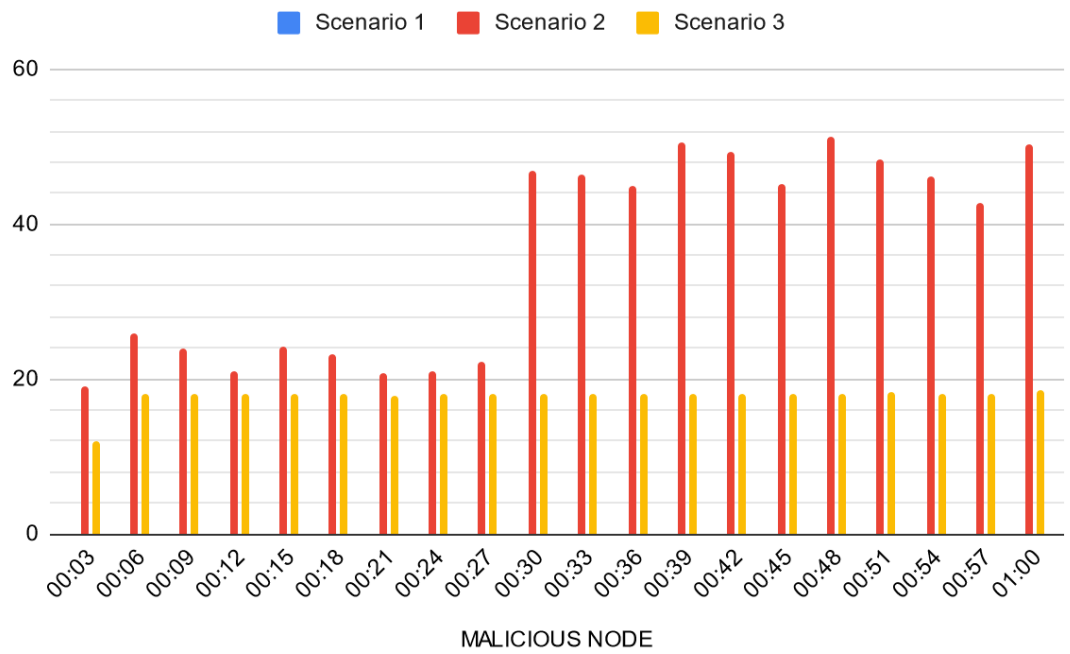

Figure 4.4. DAO Messages Received by Root in Scenarios

5.3. Why Have Dropped Packets Increased After the 30. Minute?

In the previous paragraph, it was mentioned that there will be an increase in DAO packages coming to Root with Global Repair. In addition, when the Table 4.2 is examined, it is seen that the total number of DAO messages sent to Root and the total number of DAO messages received by Root are not the same for Scenario 2. Nodes that select preferred parents again with Global Repair usually reselect their old parent unless there is a change in the network topology. This situation is also seen in Scenario 2. However, when nodes that choose the malicious node as the preferred parent again, attempt to send DAO messages through the malicious node, the DAO messages will be dropped by the malicious node. Nodes that do not receive a response to the DAO message will then try to send 5 more DAO packets as seen in Figure 4.1. If there is no response to these DAO messages, it will be understood that that preferred parent is not in use.

The node will then go in search of a new preferred parent with Local Repair. Therefore, in Scenario 2, it is seen that the number of DAO messages sent to the Root is high, as well as the number of DAO messages sent by the nodes. In addition, the increase in DAO messages seen here can be shown as one of the reasons for the increase in the total number of packets dropped in Scenario 2.

Table 4.2. DAO Messages Released by Nodes and Received by Root in Scenarios

Scenarios	Released by Nodes	Received by Root
Scenarios 1	27	27
Scenarios 2	63	29
Scenarios 3	27	27

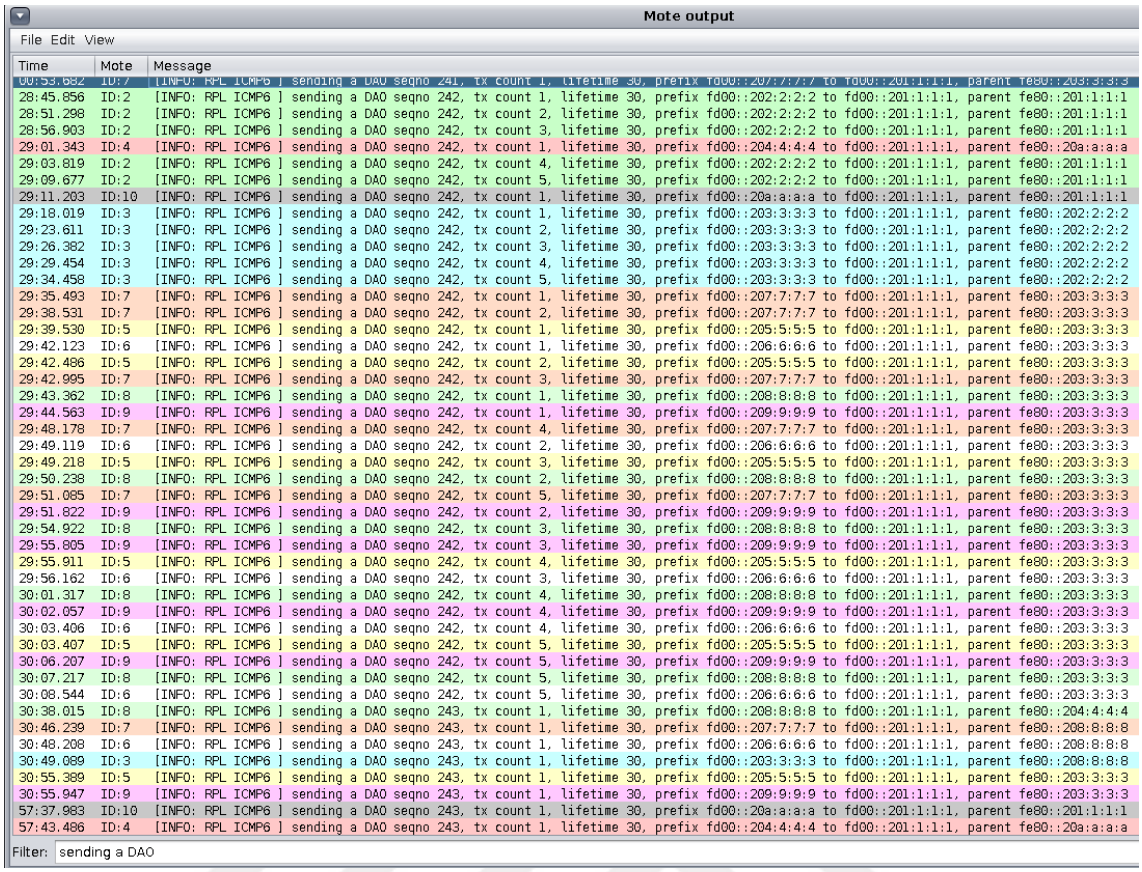

Figure 4.5. Repetitive DAO Packet Delivery before Local Repair

5.4. Global Repair's Impact on Preferred Parents

In order to understand the effect of Global Repair on preferred parents mentioned above, when the Table 4.2 is examined, only one parent is selected in Scenario 1 and Scenario 3, while in Scenario 2, parent selection is seen three times for the nodes affected by the attack. This explains that the number of DAO packages produced in Scenario 1 and Scenario 3 specified in the above paragraph remains unchanged. However, as mentioned in the previous paragraphs, nodes remain without any preferred parents by the effect of Global Repair and Local Repair, so we can see three different parent selections for the affected nodes in Scenario 2. Since it causes the DIS and DIO control messages to be released to the network before the new preferred parent selection, one more reason for the packet increase seen after 30 minutes of

Scenario 2 in Figure 4.1 is explained. Also, since the malicious node will drop the DAO-ACK packets that it will receive in response to the DAO packets it sends, the last selected parent value of the malicious node will remain NULL and cause the node to be disconnected from the network. In addition, we can also include DAO-ACK messages to the package type dropped by the malicious node in scenario 2 of Figure 4.1. Thus, it can be said that the malicious node in Scenario 2 causes the number of all control messages on the network to increase. Besides, in Scenario 1 and Scenario 3, as a result of Global Repair, no change was observed in the preferred parents. Therefore, nodes affected by the attack in Scenario 3 continued to be affected throughout the runtime.

Table 4.3. Preferred Parents in Scenarios

Scenarios	**Scenario 1**	**Scenario 2**			**Scenario 3**
Parents	**1. Parent**	**1. Parent**	**2. Parent**	**3. Parent**	**1. Parent**
Node-2	1	1	NULL	-	1
Node-3	2	2	NULL	8	2
Node-4	10	10	-	-	10
Node-5	3	3	NULL	3	3
Node-6	3	3	NULL	8	3
Node-7	3	3	NULL	8	3
Node-8	3	3	NULL	4	3
Node-9	3	3	NULL	3	3
Node-10	1	1	-	-	1

5.5. Preferred Parents in Scenarios

It is seen that the packages dropped in Scenario 2 in Table 4.1 are more than twice the packages dropped in Scenario 3. However, since the packets dropped in Scenario 2 include control messages, the Table 4.4 should be examined to understand the impact of the attack on UDP packets. While the total number of UDP packets produced in Scenario 1 and Scenario 3 is almost equal to each other, it is seen that the

number of packages produced in Scenario 2 decreased. The reason for this situation is that the attack carried out in Scenario 2 blocks the connection of the nodes affected by the attack with the network. As seen in Figure 4.2, it is seen that the affected nodes are disconnected from the network during Global Repair at the 29th and 30th minutes. This interruption due to the nodes still trying to choose the malicious node parent. We also saw again this in Scenario 2 in Table 4.3 when nodes set their parent to NULL. Therefore, nodes cannot send UDP packets until they re-join the network. As a result, the UDP nodes produced in scenario 2 of Table 4.4 are less than in other scenarios. As a result, the produced UDP packets in scenario 2 of Table 4.4 are less than the other scenarios.

Figure 4.2 shows the situation in which the aforementioned malicious node is completely disconnected from the network due to the constant drop of DAO-ACK messages it must receive to join the network. In the topology created in the study, when other nodes can find other parents to reach the Root, they will join the network after a while and continue to produce UDP packets. On the other hand, in a new topology to be created in a different study, nodes that are affected by the attack and cannot find a new parent other than the malicious node will be completely isolated from the network.

As seen in Table 4.4, when the drop rates of the released UDP packets are examined, the packet loss in Scenario 2 is 43%, while the loss in Scenario 3 is 66%. Also, from another point of view, it has been observed that while the attacks can only perform dropping until the first Global Repair in Scenario 2, they are not affected by Global Repair in Scenario 3 and can perform dropping during the running time. Therefore, if the attack used in Scenario 3 is taken at a certain time frame where it is active, the packet loss there will be 100%. In addition, in Table 4.4, the number of UDP packets sent by the nodes and the number of those packets that can reach the Root are shown in detail. As can be seen, while the dropped packets of the nodes affected by the attack were reduced by half in Scenario 2, almost all UDP packets were dropped in Scenario 3.

Table 4.4. UDP Packet Delivery Fraction Across in Scenarios

Scenarios	UDP Sent	UDP Received	Dropped UDP
Scenario 1	534	534	0
Scenario 2	468	269	199 (43%)
Scenario 3	534	179	355 (66%)

Table 4.5. UDP Packets in Scenarios

	Scenario 1		Scenario 2		Scenario 3	
Nodes	**Released**	**Received**	**Released**	**Received**	**Released**	**Received**
Node-2	60	60	29	1	60	60
Node-3	59	59	52	24	60	0
Node-4	59	59	60	60	59	59
Node-5	59	59	53	24	59	0
Node-6	59	59	53	24	59	0
Node-7	59	59	53	24	59	0
Node-8	59	59	58	30	59	0
Node-9	59	59	52	24	59	0
Node-10	60	60	59	59	59	59
TOTAL	**534**	**534**	**468**	**269**	**534**	**179**

Time	Mote	Message
00:02.726	ID:10	[INFO: App] Not reachable yet
00:07.720	ID:2	[INFO: App Malicious] Not reachable yet
00:07.736	ID:8	[INFO: App] Not reachable yet
00:15.004	ID:9	[INFO: App] Not reachable yet
00:19.795	ID:6	[INFO: App] Not reachable yet
00:20.829	ID:5	[INFO: App] Not reachable yet
00:50.928	ID:7	[INFO: App] Not reachable yet
29:43.472	ID:3	[INFO: App] Not reachable yet
29:53.310	ID:7	[INFO: App] Not reachable yet
30:06.209	ID:2	[INFO: App Malicious] Not reachable yet
30:08.960	ID:8	[INFO: App] Not reachable yet
30:10.824	ID:9	[INFO: App] Not reachable yet
30:19.386	ID:6	[INFO: App] Not reachable yet
30:23.270	ID:5	[INFO: App] Not reachable yet
30:43.173	ID:3	[INFO: App] Not reachable yet
31:06.120	ID:2	[INFO: App Malicious] Not reachable yet
32:05.522	ID:2	[INFO: App Malicious] Not reachable yet
33:06.097	ID:2	[INFO: App Malicious] Not reachable yet
34:05.146	ID:2	[INFO: App Malicious] Not reachable yet
35:05.994	ID:2	[INFO: App Malicious] Not reachable yet
36:05.291	ID:2	[INFO: App Malicious] Not reachable yet
37:04.530	ID:2	[INFO: App Malicious] Not reachable yet
38:04.124	ID:2	[INFO: App Malicious] Not reachable yet
39:03.156	ID:2	[INFO: App Malicious] Not reachable yet
40:02.839	ID:2	[INFO: App Malicious] Not reachable yet
41:01.990	ID:2	[INFO: App Malicious] Not reachable yet
42:02.087	ID:2	[INFO: App Malicious] Not reachable yet
43:01.120	ID:2	[INFO: App Malicious] Not reachable yet
44:02.062	ID:2	[INFO: App Malicious] Not reachable yet
45:02.098	ID:2	[INFO: App Malicious] Not reachable yet
46:03.054	ID:2	[INFO: App Malicious] Not reachable yet
47:03.748	ID:2	[INFO: App Malicious] Not reachable yet
48:03.123	ID:2	[INFO: App Malicious] Not reachable yet
49:03.268	ID:2	[INFO: App Malicious] Not reachable yet
50:03.052	ID:2	[INFO: App Malicious] Not reachable yet
51:02.274	ID:2	[INFO: App Malicious] Not reachable yet
52:02.607	ID:2	[INFO: App Malicious] Not reachable yet
53:01.617	ID:2	[INFO: App Malicious] Not reachable yet
54:00.800	ID:2	[INFO: App Malicious] Not reachable yet
55:00.137	ID:2	[INFO: App Malicious] Not reachable yet
55:59.239	ID:2	[INFO: App Malicious] Not reachable yet
56:59.035	ID:2	[INFO: App Malicious] Not reachable yet
57:58.897	ID:2	[INFO: App Malicious] Not reachable yet
58:59.309	ID:2	[INFO: App Malicious] Not reachable yet
59:59.490	ID:2	[INFO: App Malicious] Not reachable yet
1:01:00.483	ID:2	[INFO: App Malicious] Not reachable yet

Filter: Not

Figure 4.6. Nodes are Isolated from the Network in Scenario 2

Another effect that Global Repair causes on the network is the increase in DIS messages in the network due to the disconnection of nodes mentioned in earlier paragraphs from the network. Because the nodes that want to join the network constantly send DIS messages around them while seeking a network for them. As seen in Table 4.6, in Scenario 2, as a result of the nodes disconnecting their connection with the network, they released quite a lot of DIS packets to re-join the network. The node that produced the most DIS packets was the malicious node with 65. Because, as mentioned earlier, because the malicious node that is disconnected from the network dropped all packets that come to it, after the Global Repair, the malicious node continued to produce the DIS packet even though it could not connect to the network

again. If there are nodes that cannot find parent other than malicious nodes in another topology to be created, the number of DIS packets on these nodes will also be quite high. In contrast, the number of DIS packets in Scenario 3 is similar to Scenario 1.

Table 4.6. DIS Packets Released in Scenarios

Nodes	Scenario 1	Scenario 2	Scenario 3
Node-2	0	65	0
Node-3	1	13	1
Node-4	1	1	1
Node-5	1	14	1
Node-6	1	14	1
Node-7	1	15	2
Node-8	1	2	1
Node-9	1	15	1
Node-10	0	0	0
TOTAL	6	139	7

As mentioned in previous articles (Marti et al. 2000; Winter et al. 2012) the increase in DIO packages seen as the effect of blackhole attack is also seen in Scenario 2 in this thesis study. As seen in Table 4.7, while the total number of DIO packages created in Scenario 1 was 325, in Scenario 2 this number increased by 153% and reached 823. In Scenario 3, this number increased by only 14% to 371. As it is known, after a node sends a DIO message to the network, a DIO packet is sent to it in response from the perimeter nodes. Therefore, the increase in the total number of DIO packets sent is reflected in the total DIO packets received by the nodes. As can be seen in the Table 4.7, the increase in Scenario 3 is much less than in Scenario 2. However, the fact that the number of DIOs in Scenario 3 is not the same as in Scenario 1 is because the simulations are generated randomly, and this is an expected difference. As a result, the increase in the number of DIO packets seen in the blackhole attack, which drops all packets, has been eliminated to a certain extent with the extended blackhole attack.

Table 4.7. Summary of Total DIO Messages Released and Received in Scenarios

	Released			**Received**		
Scenarios	**Scenario 1**	**Scenario 2**	**Scenario 3**	**Scenario 1**	**Scenario 2**	**Scenario 3**
TOTAL	325	823	371	572	1806	613
Increase	Benchmark	+153%	+14%	Benchmark	+216%	+7%

Table 4.8. DIO Messages in Scenarios

	Scenario 1		**Scenario 2**		**Scenario 3**	
Nodes	**Released**	**Received**	**Released**	**Received**	**Released**	**Received**
Node-1	10	29	176	194	10	29
Node-2	28	49	22	15	32	48
Node-3	22	83	178	118	58	85
Node-4	23	30	26	201	27	31
Node-5	38	53	41	198	38	60
Node-6	38	52	44	203	38	64
Node-7	55	98	56	250	55	104
Node-8	47	77	43	219	48	83
Node-9	38	54	41	198	38	63
Node-10	28	48	195	210	27	47
TOTAL	**325**	**572**	**823**	**1806**	**371**	**613**

CHAPTER 6

CONCLUSIONS

RPL is a very useful routing protocol for Low power and Lossy Networks. In addition, it is also vulnerable to many insider attacks that can be carried out in the network. One of these attacks is the blackhole attack. Blackhole attack has a simple structure. The only purpose of the malicious node is to drop the packets coming to it rather than forwarding them. For this reason, when it is not noticed, it has a destructive effect on the network.

As can be seen in the second scenario of the thesis study, basic blackhole attacks are proved once again that ICMPv6 control messages increase and the network has an unstable structure. However, an IDS mechanism created based on the change of the number of ICMPv6 control messages alone for the detection of the blackhole attack would be misleading. We can easily say this by looking at the results of the extended blackhole attack we developed in scenario 3. As seen in this thesis, the increase in control messages seen in basic blackhole attack has been eliminated in extended blackhole attack. As a result of the basic blackhole attack, nodes fall off the network and become isolated. After that, there is an increase in ICMPv6 control messages in the network due to the desire to rejoin the network or to find a different selected parent. However, since the connection between nodes and root is always protected in the extended blackhole attack, isolation is carried out by dropping the packets. Thus, there is no change in the topology and no increase in control messages count that released in the network.

As stated above, in a basic blackhole attack, the attacker node and its sub-tree are isolated from the network. The reason for this is that the basic blackhole attack also drops the control messages that ensure the formation and continuity of the network. The attacker node, which also dropped the control messages, will then lose its connection with the network when the parent selections are refreshed after Global Repair. However, with the extended blackhole attack, it has been observed that the packets can

be dropped while the connection between the nodes in the network is maintained. Thus, the effect of the blackhole attack will continue to be observed as long as the attack occurs, not until Global Repair occurs. Therefore, this situation should be taken into consideration when developing IDS mechanisms.

The difference between the extended blackhole attack from the other attack was that it prevented an increase in the number of ICMPv6 control messages. However, allowing nodes to stay on the network and drop their packets causes them to be easily detected by some types of IDS mechanisms developed in the literature. In such an IDS mechanism, each node controls network traffic at perimeter nodes. Since the attacker node performing the extended blackhole attack does not forward the packets sent to it to the parent node, it causes it to be detected by neighboring nodes. However, this type of IDS mechanism using nodes in the network as a defense mechanism will cause the nodes to run out of resources in a short time. Therefore, although such IDS mechanisms work theoretically, their use in real life is not recommended by other studies in the literature. In addition, the combination of two attacker nodes to cooperate will make this type of IDS mechanism very difficult to detect correctly.

As a result, the first problem caused by a basic blackhole attack is that the node and subnet performing the attack are isolated from the network. These nodes that are lost their connection from the network will cause the topology to change or prevent the new node from joining the network. As a result of attempts to eliminate this problem, an extended blackhole attack was developed. After the extended blackhole attack is developed, preventing the node that will perform the attack from being isolated from the network has prevented the topology from changing. Hence the proposed blackhole attack mechanism requires the current defense mechanisms to be reconsidered.

REFERENCES

Ahmed, Firoz, and Young Bae Ko. 2016. “Mitigation of Black Hole Attacks in Routing Protocol for Low Power and Lossy Networks.” *Security and Communication Networks* 9 (18). https://doi.org/10.1002/sec.1684.

Airehrour, David, Jairo Gutierrez, and Sayan Kumar Ray. 2017. “Securing RPL Routing Protocol from Blackhole Attacks Using a Trust-Based Mechanism.” In *26th International Telecommunication Networks and Applications Conference, ITNAC 2016*. https://doi.org/10.1109/ATNAC.2016.7878793.

Chugh, Karishma, Aboubaker Lasebae, and Jonathan Loo. 2012. “Case Study of a Black Hole Attack on 6LoWPAN-RPL.” In *SECURWARE 2012 - 6th International Conference on Emerging Security Information, Systems and Technologies*. https://www.researchgate.net/publication/315672940_Case_Study_of_a_Black_Hole_Attack_on_6LoWPAN-RPL.

“Contiki-Ng Wiki.” n.d. Accessed November 25, 2020. https://github.com/contiki-ng/contiki-ng/wiki/Documentation:-RPL.

Dunkels, Adam, Björn Grönvall, and Thiemo Voigt. 2004. “Contiki - A Lightweight and Flexible Operating System for Tiny Networked Sensors.” In *Proceedings - Conference on Local Computer Networks, LCN*, 455–62. IEEE. https://doi.org/10.1109/LCN.2004.38.

Gutierrez, José A., Marco Naeve, Ed Callaway, Monique Bourgeois, Vinay Mitter, and Bob Heile. 2001. “IEEE 802.15.4: A Developing Standard for Low-Power Low-Cost Wireless Personal Area Networks.” *IEEE Network*. https://doi.org/10.1109/65.953229.

Jiang, Jun, Yuhong Liu, and Behnam Dezfouli. 2019. “A Root-Based Defense Mechanism Against RPL Blackhole Attacks in Internet of Things Networks.” In *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA ASC 2018 - Proceedings*. https://doi.org/10.23919/APSIPA.2018.8659504.

Kumar, Arvind, Rakesh Matam, and Shailendra Shukla. 2016. “Impact of Packet Dropping Attacks on RPL.” In *2016 4th International Conference on Parallel, Distributed and Grid Computing, PDGC 2016*. https://doi.org/10.1109/PDGC.2016.7913211.

Le, Anhtuan, Jonathan Loo, Aboubaker Lasebae, Mahdi Aiash, and Yuan Luo. 2012. “6LoWPAN: A Study on QoS Security Threats and Countermeasures Using Intrusion Detection System Approach.” *International Journal of Communication Systems* 25 (9). https://doi.org/10.1002/dac.2356.

Levis, P., T. Clausen, J. Hui, O. Gnawali, and J. Ko. 2011. “RFC 6206 - The Trickle Algorithm.” *Internet Requests for Comments*. https://tools.ietf.org/html/rfc6206.

Liñán Colina, Antonio, Alvaro Vives, Antoine Bagula, Marco Zennaro, and Ermanno Pietrosemoli. 2016. “Internet of Things IN 5 DAYS,” 227.

Marti, Sergio, T. J. Giuli, Kevin Lai, and Mary Baker. 2000. “Mitigating Routing Misbehavior in Mobile Ad Hoc Networks.” In *Proceedings of the Annual International Conference on Mobile Computing and Networking, MOBICOM*. https://doi.org/10.1145/345910.345955.

Mayzaud, Anthéa, Rémi Badonnel, and Isabelle Chrisment. 2016. “A Taxonomy of Attacks in RPL-Based Internet of Things.” *International Journal of Network Security*. Vol. 18. https://hal.inria.fr/hal-01207859.

Neerugatti, Vikram, and A. Rama Mohan Reddy. 2019. “Detection and Prevention of Black Hole Attack in RPL Protocol Based on the Threshold Value of Nodes in the Internet of Things Networks.” *International Journal of Innovative Technology and Exploring Engineering* 8 (9 Special Issue 3): 325–29. https://doi.org/10.35940/ijitee.I3060.0789S319.

Pongle, Pavan, and Gurunath Chavan. 2015. “A Survey: Attacks on RPL and 6LoWPAN in IoT.” In *2015 International Conference on Pervasive Computing: Advance Communication Technology and Application for Society, ICPC 2015*. https://doi.org/10.1109/PERVASIVE.2015.7087034.

Raza, Shahid, Linus Wallgren, and Thiemo Voigt. 2013. “SVELTE: Real-Time Intrusion Detection in the Internet of Things.” *Ad Hoc Networks* 11 (8). https://doi.org/10.1016/j.adhoc.2013.04.014.

Sahay, Rashmi, G. Geethakumari, Barsha Mitra, and V. Thejas. 2018. “Exponential Smoothing Based Approach for Detection of Blackhole Attacks in IoT.” In *International Symposium on Advanced Networks and Telecommunication Systems, ANTS*. Vol. 2018-December. https://doi.org/10.1109/ANTS.2018.8710073.

Tsvetkov, Tsvetko, and Alexander Klein. 2011. “RPL: IPv6 Routing Protocol for Low Power and Lossy Networks.” *Seminar Sensorknoten: Betrieb, Netze Und Anwendungen*. https://doi.org/10.2313/NET-2011-07-1_09.

Vasseur, JP. 2014. “Terms Used in Routing for Low-Power and Lossy Networks.” *RFC7102*.

Winter, T., P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, JP. Vasseur, and R. Alexander. 2012. “Rfc 6550: Rpl.” *RFC 6550, IETF*. https://tools.ietf.org/html/rfc6550.

Yang, Wei, Yuan Wang, Zhixiang Lai, Yadong Wan, and Zhuo Cheng. 2019. “Security Vulnerabilities and Countermeasures in the RPL-Based Internet of Things.” In *Proceedings - 2018 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, CyberC 2018*, 49–54. Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/CyberC.2018.00020.

Zhang, Tianchen, Taimin Zhang, Xiaoyu Ji, and Wenyuan Xu. 2019. “Cuckoo-RPL: Cuckoo Filter Based RPL for Defending AMI Network from Blackhole Attacks.” In *Chinese Control Conference, CCC*. Vol. 2019-July. https://doi.org/10.23919/ChiCC.2019.8866139.