****

**CSN10108 Security Systems for IOT**

**Coursework Submission**

**David McCahon**

**40214392**

**School of Computing**

**November 2018**

# Contents

[List of Tables 4](#_Toc531286585)

[List of Code 5](#_Toc531286586)

[Chapter 1 Literature Review 6](#_Toc531286587)

[1.1 IoT Networks 6](#_Toc531286588)

[1.2 Blackhole Attack 7](#_Toc531286589)

[1.3 Sinkhole Attack 8](#_Toc531286590)

[1.4 Increased Rank Attack 9](#_Toc531286591)

[1.5 DIO Suppression Attack 10](#_Toc531286592)

[1.6 Hello Flooding Attack 11](#_Toc531286593)

[1.7 Wormhole Attack 11](#_Toc531286594)

[1.8 Version Number Attack 12](#_Toc531286595)

[1.9 DIS Attack 13](#_Toc531286596)

[1.10 IoT Attack Countermeasures 13](#_Toc531286597)

[Chapter 2 Attack Implementation and Simulation 15](#_Toc531286598)

[2.1 Scenarios 15](#_Toc531286599)

[2.2 Attack Implementation 17](#_Toc531286600)

[2.3 Evaluation 23](#_Toc531286601)

[Chapter 3 – Hardware 33](#_Toc531286602)

[3.1 Device Setup 33](#_Toc531286603)

[3.2 DIS Attack Implementation 33](#_Toc531286604)

[3.3 Results 34](#_Toc531286605)

[References 36](#_Toc531286606)

# List of Figures

[Figure 1: DOGAG Tree 7](#_Toc531287330)

[Figure 2: Blackhole Attack 9](#_Toc531287331)

[Figure 3: Sinkhole Attack 10](#_Toc531287332)

[Figure 4: Increased Rank Attack (Nawir, Amir, Yaakob, & Lynn, 2016) 11](#_Toc531287333)

[Figure 5: DIO Suppression Attack (Perazzo, Vallati, Anastasi, & Dini, 2017) 11](#_Toc531287334)

[Figure 6: Wormhole Attack (RGHIOUI, KHANNOUS, & BOUHORMA, 2014) 13](#_Toc531287335)

[Figure 7: Random DODAG 17](#_Toc531287336)

[Figure 8: Random Neighbours 17](#_Toc531287337)

[Figure 9: Linear DODAG 18](#_Toc531287338)

[Figure 10: Version Number Range 21](#_Toc531287339)

[Figure 11: DIO Suppression Mote Output 23](#_Toc531287340)

[Figure 12: Scenario 1 Delivery Ratio 24](#_Toc531287341)

[Figure 13: Unstable Parent Selection 25](#_Toc531287342)

[Figure 14: Scenario 1 Power Consumption 27](#_Toc531287343)

[Figure 15: Scenario 2 Delivery Ratio 30](#_Toc531287344)

[Figure 16: Rank Parent Loop 30](#_Toc531287345)

[Figure 17: Linear Version Successful Motes 31](#_Toc531287346)

[Figure 18: Scenario 2 Power Consumption 32](#_Toc531287347)

[Figure 19: Terminal Power Consumption 35](#_Toc531287348)

[Figure 20: DIS Power Consumption 36](#_Toc531287349)

# List of Tables

[Table 1: Seed Setup 16](#_Toc531286841)

[Table 2: DIO Suppression ID Values 22](#_Toc531286842)

[Table 3: Scenario 2 Parent and Delivery Ratio Comparison 29](#_Toc531286843)

[Table 4: Hardware Setup 34](#_Toc531286844)

# List of Code

[Code 1: Sinkhole Attack 19](#_Toc531286835)

[Code 2: Rank Attack 20](#_Toc531286836)

[Code 3: Version Number Attack 20](#_Toc531286837)

[Code 4: DIO Suppression 23](#_Toc531286838)

[Code 5: DIS Timer Changes 35](#_Toc531286839)

[Code 6: DISCounter 35](#_Toc531286840)

# Chapter 1 Literature Review

Internet of Things (IOT) devices have seen a recent surge due to popularity and technological advancement. IOT devices are every day appliances such as kettles, cookers, fridges, CCTV Cameras and air quality sensors on cars. Moreover, IOT devices also reside within industry as sensors to determine health and machine usage within kitchens, factories and engineering. The security of these devices is essential as the abundance of devices is set to grow as IHS suggests 30.7 billion IOT devices by 2020 whereas Intel suggest 200 billion devices by 2020 (Claveria, 2018).

Famously IoT devices are deceptive to attack as they contributed to the Mirai attack creating a record breaking 620Gbps DDoS attack (Sinanovic & Mrdovic, 2017). This literature review will discuss IoT network attacks and the effects these attacks can have on the network. Moreover, potential countermeasures will be discussed to protect devices against these attacks.

## 1.1 IoT Networks

IoT Networks uses RPL, Routing Protocol for Low Power and Lossy Networks (LLN), to communicate with other devices on the IoT networks. RPL is optimised for LLN due to the limited processing power, memory and resources available within IoT devices and therefore energy efficiency is key (IoTone, n.d.). RPL is designed for multi-point to point communication forming a Destination Oriented Directed Acyclic Graph (DODAG) tree containing the root/sink mote (Figure 1. DODAG Tree).

A close up of a necklace

Description automatically generated

Figure 1: DOGAG Tree

The DODAG tree starts from the root mote at the top of tree and expands as the root mote broadcasts DODAG Information Object (DIO) messages to other motes on the network to determine their ID, location and rank. As motes receive the DIO they then broadcast their own DIO becoming the parent of the child motes which receive their message. The tree is formed dependant on these parent to child relationships and the rank of each mote dependant on distance and energy of the link (Pongle & Chavan, 2015a). Furthermore, if a mote joins the network it broadcasts a DODAG Informational Solicitation (DIS) message to request a DIO updating its metrics and to join the network (Accettura, Grieco, Boggia, & Camarda, 2011). The formation and effectiveness of the DODAG can be maliciously altered by topological attacks in order to affect the availability and performance of the network.

## 1.2 Blackhole Attack

One or more malicious motes can execute a blackhole attack within the IoT network; the more malicious motes attacking the greater the damage is to performance of the network. The blackhole attack causes the malicious motes to drop packets being routed through the mote causing disruption to the traffic. The blackhole attack can be implemented in complete or partial. A complete blackhole attack drops 100% of the packets sent to it whereas a partial or selective attack only drops a certain percentage of motes (Chugh, Lasebae, & Loo, 2012). The blackhole attack results in damage to data integrity and increased risk due to sent packets being blocked to the sink. (Yu, Li, Zhou, & Li, 2012).

The malicious mote sends DIO messages advertising itself as the best route to the root mote or sink supressing communication within the network. Sender/Child motes select the malicious mote as their parent resulting in their packets being entirely or partially dropped dependant on the type of active attack. Furthermore, the blackhole can be seen as a DoS attack as if the malicious motes escalates itself to an important rank some of the network can be taken offline and unstable affecting the availability of the network (Chugh, Lasebae, & Loo, 2012).

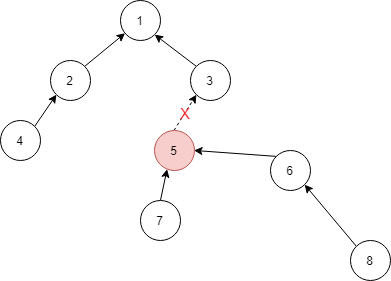


Figure 2: Blackhole Attack

As seen in the above in Figure 2 mote 5 has been selected as the malicious mote executing the blackhole attack. In comparison to the original DODAG Tree in Figure 1 mote 5 has sent a DIO to mote 6 advertising a higher rank and best path to the sink. Therefore, Mote 6 has selected mote 5 as its parent instead of mote 3. All packets forwarded from mote 6,7 and 8 are now being completely or partially dropped by mote 5. Moreover, if a complete blackhole attack is in place the availability of the network is affected as motes 6,7 and 8 are taking offline and unable to connect to the sink.

## 1.3 Sinkhole Attack

A sinkhole attack is like a partial blackhole attack aforementioned in Chapter 1.2.1. A mote is selected to execute the attack with the aim of attracting as much traffic to the mote as possible to harm the reception of data compromising integrity and reliability of data within the network.

As the rank is escalated to the malicious mote other motes within the network select the malicious mote as their parent forwarding all packets to the mote. Once the malicious mote has attracted as many motes to it as possible it modifies itself as another attack such as a partial blackhole or selective forward attack and only partially send data to the sink. Due to this, paths are hijacked and unoptimized leading to higher ETX values and energy consumption to deliver the packets to the sink (Cervantes, Poplade, Nogueira, & Santos, 2015).

A close up of a logo

Description automatically generated

Figure 3: Sinkhole Attack

As seen in Figure 3 like the blackhole attack mote 5 is executing the sinkhole attack. It sends fake information to the motes around it attracting the motes and escalates its rank as the fastest route to the sink. In correspondence, mote 2 has set its parent to mote 5 instead of the sink. The sinkhole then modifies itself to a selective forward attack and has full control over which packets are forwarded to the sink and what packets are dropped. IDS solutions can be used to countermeasure the sinkhole attack as discussed in Chapter 1.10.3.

## 1.4 Increased Rank Attack

The rank attack is an indirect spoofing and alteration attack which increases the rank of a malicious mote to attract child motes and select them as their parent. Within RPL the rank increases in the downward direction from root to child. Therefore, a parent mote should always have a lower rank than its parent. However, within the rank attack the malicious mote advertises a higher rank creating an undetectable loop within the IoT network.

This attack creates various consequences within the IoT network as optimised paths exists but are never used. Unoptimized paths are created causing delay between sending messages and receiving at the sink with increased malicious motes the longer the delay alongside reduced ratio of packet delivery and consent changing topology around the malicious mote dependant on whether the DIO messages are being sent and updated . (Pongle & Chavan, 2015a)

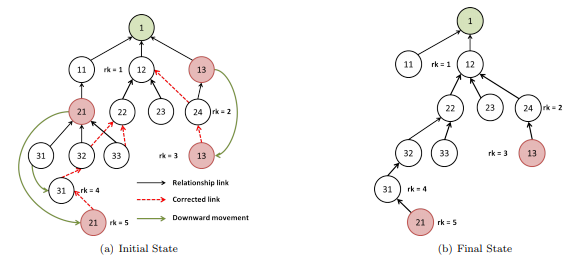


Figure 4: Increased Rank Attack (Nawir, Amir, Yaakob, & Lynn, 2016)

As seen in the above Figure 3 mote 21 and 13 have increased their rank to become the child of their initial children motes (31, 32, 33 and 24). Therefore, the malicious motes select the child motes as their parent creating a loop in the network.

## 1.5 DIO Suppression Attack

DIO Suppression attacks aim to attack and create a malicious mote within the IoT network which sends spoof messages via multicast to other motes on the network. These spoof messages contain continuous sequence numbers to stop motes receiving legitimate messages forcing erasure of cached messages on the mote (Pu & Zhou, 2018). The DIO Suppression attack induces malicious mote to block DIO messages received from other motes on the network disrupting the formation of the DODAG tree. Replay messages are used to make other motes believe that the routing information contained within the DIO messages has already been forwarded several times before allowing ease of routing updates without having to steal cryptographic keys like in other attacks.

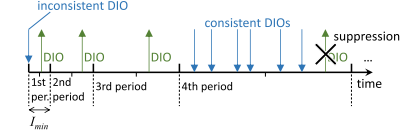


Figure 5: DIO Suppression Attack (Perazzo, Vallati, Anastasi, & Dini, 2017)

This suppression prevents the routing topology being maintained causing degradation of routes which will overtime create partitions within the network. Moreover, the attack is less energy expensive and therefore more effectively degrades the network in comparison to a jamming attack (Perazzo et al., 2017).

## 1.6 Hello Flooding Attack

Internal flooding attacks known as the hello flooding attack exploits the DIO messages which are sent when a mote join the messages otherwise known as “hello messages”. The malicious mote floods these messages to neighbouring motes alongside strong metric information to gain trust within the network.

The attack can send two types of messages – DIS broadcast which sends a message to all motes on the network and resets the trickle timer or DIS unicast which sends a message to a single mote which must reply with a DIO message. These messages create congestion within the network increasing control message overload however all packets are delivered, and the delivery ratio is unaffected. The availability of the network can also be affected as an overload of information sent to motes can cause them to use up their resources and go offline. Currently there is no countermeasure for this attack (Nawir et al., 2016) however this attack can only run for a short period of time as RPL repairs itself causing the attack to be stopped. Although, using this attack in conjunction with other attacks can prevent automatic repair by RPL.

## 1.7 Wormhole Attack

The wormhole attack aims to disrupt the network topology and flow of traffic throughout the network. The attack creates a tunnel between 2 malicious motes and tunnelling traffic between them (Figure 6: Wormhole Attack). This attack can be carried out by encapsulating messages sent between the malicious motes where the attacker takes the payload and re-transmits or by packet delay where a malicious mote transmits messages between distant motes making them believe that they are neighbours which can only be launched by one mote. During a wormhole attack average packet loss increases resulting in motes having to resend messages to ensure reliable delivery which consumes unnecessary resources such as energy and processing power.

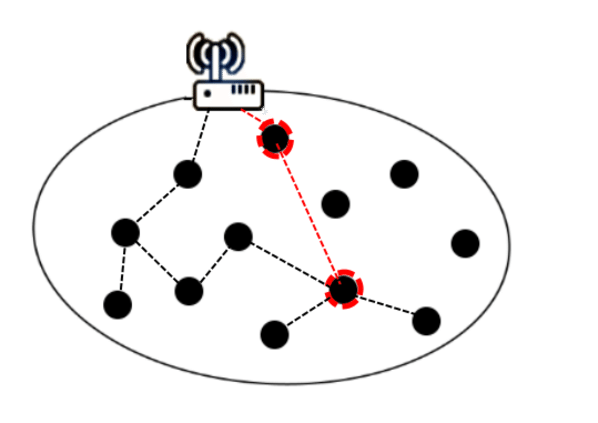


Figure 6: Wormhole Attack (RGHIOUI, KHANNOUS, & BOUHORMA, 2014)

Countermeasures against this attack include hardware based techniques such as synchronising the clocks on each mote to detect anomalies in the network; packet leased techniques which only allows a packet to travel a certain distance similar to TTL; round trip time techniques which detects the send and receive time for packets to detect the wormhole and neighbour information techniques to detect a motes neighbours and detect changes in the network. Statistical information can monitor monitors links returned for multi-path protocols detecting a wormhole by analysing the delay per hop source and destination to detect different paths that a packet may take (Pongle & Chavan, 2015b). Markle Tree Authentication can be used to countermeasure the sinkhole attack as discussed in Chapter 1.10.4

## 1.8 Version Number Attack

The version number is used by the sink/root mote to determine the current version of the DODAG. The version number determines if a mote is aware of the current version of the DODAG and if the update to the topology has been successfully applied. To inform child motes of the update the version number if sent as part of the DIO Message which updates the child with the new DODAG and recalculates the rank of the mote. An old version number being sent from a mote indicates that it is inconsistent with the network and should not be selected as a parent until the version number is updated.

RPL does not support the integrity of version numbers and therefore a malicious mote is able to change its DIO message to its own version number, reset their own timer and advertise the new version to their neighbours causing illegitimate version numbers throughout the network. Furthermore, the DODAG is unnecessarily updated and rebuilt due to new illegitimate version numbers being transmitted which uses up resources such as processing and energy. Moreover, constant updating of the DODAG can cause temporary loops within the network affecting availability within the IoT network. Moreover, the further the malicious mote is away from the sink the longer it takes the sink to realise that the version number has changed. This delay causes more motes on the network to be attacked increasing the severity. Version Number and Rank Authentication (VeRa) aims to detect and prevent the version number attack by one way hashing the message and chain (Aris, Oktug, & Berna Ors Yalcin, 2016).

## 1.9 DIS Attack

Upon joining the IoT network a mote receives a DIO message containing information about the network and the motes metrics. However, if a DIO message is not sent to the mote a broadcast or uni-cast DODAG Information Solicitation (DIS) message is sent to the mote’s neighbours requesting a DIO message in response.

Attackers can exploit this message by periodically sending DIS messages to its neighbours resulting in a spam of response of DIO messages. This does not affect the delivery ratio but can increase the message delay and unnecessary using up resources such as power consumption (Sharma Sr Assistant Professor, Mishra Sr Assistant Professor, & Jain, 2017).

## 1.10 IoT Attack Countermeasures

### 1.10.1 VeRa

VeRa (Version Number and Rank Authentication) prevents illegimate changing of ranks and version numbers within the DODAG. This prevents motes from impersonating the sink, protecting the network from eavesdropping attacks and overall control of what messages reach the sink. Moreover, VeRa stops motes from broadcasting illegitimate version numbers which can flood the network with DIO messages causing flooding and unnecessary resource usage within the network. VeRa achieves this by generating a hash chain which is broadcast across the network for motes to verify the authenticity of the data (Dvir, Holczer, & Buttyan, 2011).

### 1.10.2 TRAIL

TRAIL (Trust Anchor Interconnection Loop) is also used to authenticate the IoT network topology preventing topological inconsistencies. Each mote on the network validates its upwards path towards the sink detecting rank attacks. If TRAIL finds inconsistencies and an attack has been established within the network RPL either locally repairs the malicious mote or disconnects it from the network preventing further damage (Perrey et al., 2015).

### 1.10.3 IDS Solutions

Intrusion Detection Systems (IDS) can be used to prevent attacks within the IoT network. Like traditional networks IDS analyses network activity to detect malicious behaviour or intruders on the network. Suspicious behaviour can either be flagged or logged for review to decide whether the alert is true or a false-positive has been created. Furthermore, IDS can block packets triggering an RPL self-heal or take the malicious mote offline. Two types of IDS can be used on the IoT network – Signature Based and Anomaly Based. Signature based IDS requires in depth knowledge of IoT attacks to be able to detect them, no new attacks can be discovered this way. Anomaly Based IDS detects out of the ordinary behaviour which can detect new attacks however suffers from a high false-positive rate (Wallgren, Raza, & Voigt, 2013).

### 1.10.4 MTA

Markle Tree Authentication (MTA) uses a bottom up approach as it starts at the leaves of the DODAG and works upwards towards the root. MTA uses the mote ID and public keys to calculate a hash used for authentication. If any mote within the network does not authenticate child-motes avoid the unauthenticated note and select a different parent (Sharma Sr Assistant Professor et al., 2017, pg. 7).

# Chapter 2 Attack Implementation and Simulation

This chapter will set out the implementation steps of the sinkhole attack and rank attack against a random and linear topology. The setup will be explained in detail including the properties of the attack, file location, topology setup, code and effect the attack had on the network in terms of packets received by the sink and power consumption.

## 2.1 Scenarios

Two scenarios are set out by different parameters with slight changes within the topology and position of the malicious mote within the network. However, few properties are set the same as transmission range is set to 25,30 and packets per minute are set to 2 by editing the common-common.c and editing the period to 30 on line 53. Moreover, to obtain fair results an average is taken over 3 different seeds as defined in Table 1: Seed Setup below. Within each scenario a total of 9 simulations will be ran as a no attack, rank attack and sink attack with each seed for 20 minutes simulation time.

|  |  |
| --- | --- |
| **Number** | **Seed** |
| 1 | 216889 |
| 2 | 428679 |
| 3 | 843212 |

Table 1: Seed Setup

### 2.1.1 Scenario 1 – Random Topology

The random topology sets mote in random position similar to the position of the motes in a real-life scenario. This allows for a realistic simulation and accurate results to compare for fair analysis. Furthermore, the random topology is setup in a tree like structure to allow for simulation on a mote which acts as a gateway to the sink. This allows for analysis of the effect attacks have on this mote and to determine whether RPL can selfheal in this type of scenario.

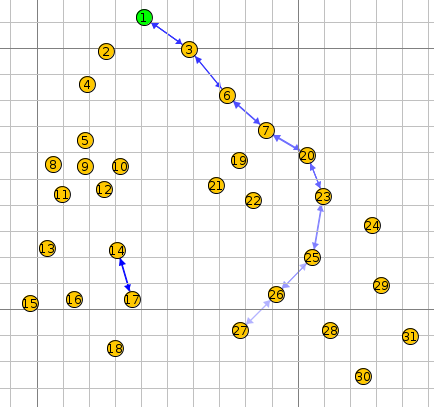


Figure 7: Random DODAG

In the above Figure 7: Random DODAG mote 5 is set as the malicious mote due to the requirement of the mote must have the highest number of neighbours (Figure 8: Random Neighbours). Within this tree like structure this is the most efficient way to alter the topology to meet this requirement. Furthermore, mote 5 acts as a gateway to the left-hand branch of the tree as it’s child motes are out of range of mote 4.

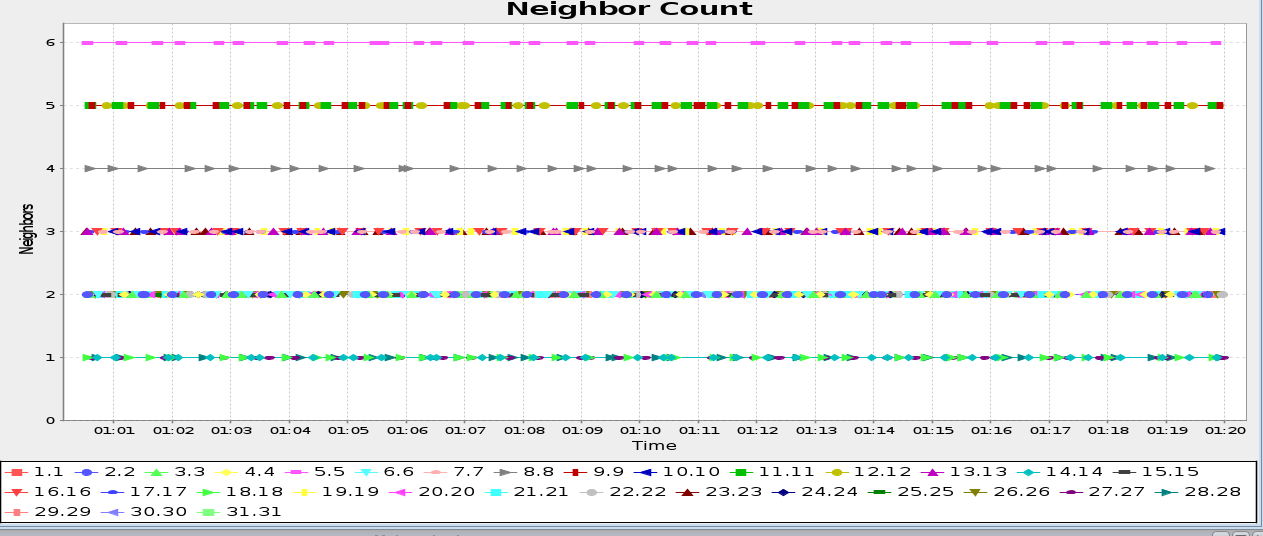


Figure 8: Random Neighbours

### 2.1.2 Scenario 2 – Linear Topology

The linear topology defines the motes into a grid like structure allowing for equal distance, parents and metrics between each mote. Within this structure no mote only has 1 parent and therefore RPL is able to selfheal (Figure 9: Linear DODAG).

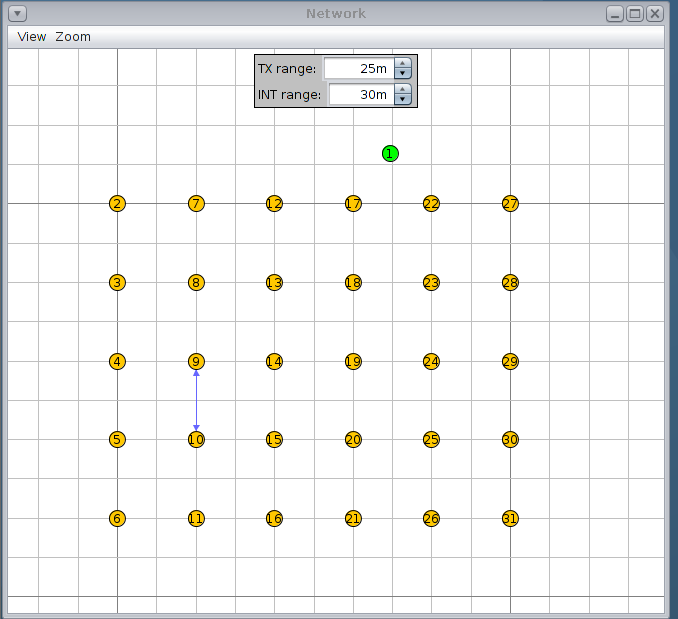


Figure 9: Linear DODAG

Furthermore, this scenario states that the malicious mote must be the furthest away from the sink and therefore mote 6 is selected as the malicious mote.

## 2.2 Attack Implementation

### 2.2.1 Sinkhole Attack

Aforementioned in Chapter 1.3 the sinkhole attack can be viewed as the most destructive attack on the IoT network as integrity, reliability and availability can be affected. However, the sinkhole attack set out within the coursework description and lab can be argued to not be a sinkhole attack at all. As seen below in Code 1: Sinkhole Attack (core/net/ipv6/uip6.c line 1243) the malicious mote/sinkhole takes a flag parameter to decide which packets to forward and which packets to drop. As no count is used to determine this flag1 is set to 0 and then 1 on a packet by packet basis and therefore is expected to drop 50% of the packets sent to the sink. By setting the expected delivery ratio to 50% allows for easily analysis to ensure the attack is mounted correctly.

|  |
| --- |
| **if** **(**flag1 **==** 1 **&&** node\_id **==** 6**){**  flag1 **=** 0**;**  printf**(**"Forwarding From Malicious\n"**);**  **goto** send**;**  **}**  **else** **if** **(**flag1 **==** 0 **&&** node\_id **==** 6**){**  flag1 **=** 1**;**  printf**(**"Forwarding-Blocked\n"**);**  **}**  **else{**  printf**(**"Forwarding-Else \n"**);**  **goto** send**;**  **}** |

Code 1: Sinkhole Attack

However, the sinkhole as a sole attack does not drop any packets. The sinkhole attack merely advertises higher metrics to attracts other motes to it disrupting the network topology. Although the sinkhole can modify itself there is no higher metric advertising and therefore it can be argued that the sinkhole attack is a partial blackhole or selective forwarding attack. With the above code it can be expected to see a 50% drop in packets from the malicious mote and its children.

### 2.2.2 Rank Attack

The rank attack advertises itself to have a higher rank than it is initially assigned. This creates the mote to virtually move down the topology and select its children as its parent. The effect of the rank attack can vary dependant on the position of the attack. For example, if multiple routes are available to the motes RPL can selfheal against the attack by selecting a different route. In this scenario the rank attack has very little impact on the network although by increasing the number of malicious motes can cause delays in the sink receiving packets. The attack can be enhanced by incorporating other attacks into the network. However, if the malicious mote happens to be the main gateway to the sink this can create an undetectable loop within the network potentially taking motes offline and forming an availability attack.

|  |
| --- |
| **if** node\_id **==** 5**{**  set16**(**buffer**,** pos**,** 966**);**  **}**  **else{**  set16**(**buffer**,**pos**,** dag**->**rank**);**  **}** |

Code 2: Rank Attack

The rank attack edits the rank setting within the core/net/rpl/rpl-icmp6.c file by manually setting the rank to edit the position of the mote. Moreover, editing this code can change the rank for every mote in the network or manually setting the mote dependant on node\_id by including the /sys/mote-id.h file.

### 2.2.3 Version Number Attack

As earlier discussed in Chapter 1.8 the version number is used to determine the current topology of the DODAG and DIO. If the DODAG changes the DIO advertises the changes and new metrics for each mote on the network. However, RPL does not implement an integrity check for these version numbers and therefore is changeable on the malicious mote. Adding a counter to the version number for the malicious mote repeatedly changes the version number creating unnecessary DIO broadcasts flooding the network with messages.

|  |
| --- |
| **if** **(**node\_id **==** 5**){**  buffer**[**pos**++]** **=** dag**->**version**++;** /// Increase Version By 1  printf**(**"My DIO Modified Version Is %d\n"**,** dag**->**version**++);**  **}**  **else{**  buffer**[**pos**++]** **=** dag**->**version**;**  printf**(**"My DIO Version is %d\n"**,** dag**->**version**);}** |

Code 3: Version Number Attack

As the DIO messages flood the network this causes a delay within the network which may affect the delivery ratio. Moreover, as DIO broadcasts are unnecessary sent and the constant topology update creating unoptimized paths the power consumption within this attack is expected to increase. The rate of wide variety of version numbers within the network can be viewed below in Figure 10: Version Number Range as mote 6 modifies the version number and sends the updates to the rest of the network.

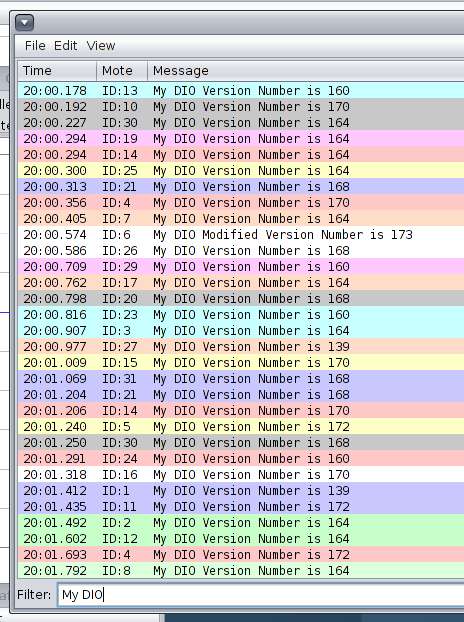


Figure 10: Version Number Range

### 2.2.4 DIO Suppression Attack

The DIO Suppression Attack replays already heard DIO messages to its neighbours making them believe that the DIO has already been sent several times before and therefore making the DIO seem legitimate. To implement this attack the DIO\_INPUT function within the rpl\_icmp6.c file must be altered to send already heard messages. However, another potential way to implement this attack is to analyse the DIO structure and set hard values to each column. By doing this, the malicious mote repeatedly sends the same information to its neighbours to theoretically make them believe that the DIO is real. The DIO message consists of 5 ID’s as follows (Badach, 2018);

* DODAG Version Number – Identifies the version of the DODAG. Incremented by the root to determine current version.
* Rank – Motes are ranked in logical distance from the sink. Required to determine the routes
* DTSN – Destination Advertisement Trigger Sequence Number maintains downwards routes

The above 5 ID’s can be set to a hard code value to send the same DIO regardless of what the received DIO contains on the malicious mote. However, as Rank and Version are both altered this attack could be considered like the rank and version attack with added alterations. These ID’s will be set the values set out in Table 2: DIO Suppression ID Values.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Variable** | **Original Value** | **New Value** | **Line** |
| DODAG Version | version | 240 | 240 | 459 |
| Rank | rank | 1024 | 1024 | 466 |
| DTSN | dtsn\_out | Incremental | 112 | 479 |

Table 2: DIO Suppression ID Values

The values set out above are different from the pre-determined values seen in the no attack simulation to grasp a range of what changing these values will do. As motes within the IoT network are updated by the DIO the DTSN is updated and kept in sync between the motes. By hard-coding this value it is expected that other motes within the network adopt this value instead of incrementing it. Moreover, setting these values to a hard value disregards any topology updates received by the malicious mote as the same data is sent to its neighbours in the illegitimate DIO. However, due to only have 1 malicious mote in the network the other motes could out-rule the malicious mote and ignore its suggestions.

Furthermore, by keeping the Version and Rank ID’s to their original values this excludes the results being put down to a change in these values. Therefore, the results obtained from this attack can only be put down to the repetitive DIO messages being sent across the network. The altered code can be seen below (Code 4: DIO Suppression Code) however it is noted that the various changes are combined into one code block, the specific line numbers can be found in Table 2. Comments are used to line out the variable being edited

|  |
| --- |
| /// DODAG Version  **if** **(**node\_id **==** 5**){**  buffer**[**pos**++]** **=** 240**;**  **}**  **else{**  buffer**[**pos**++]** **=** dag**->**version**;**  **}**  /// Rank  **if** **(**node\_id **==** 5 **){**  set16**(**buffer**,** pos**,** 1024**);**  **}**  **else{**  set16**(**buffer**,** pos**,** dag**->**rank**);**  **}**  /// DTSN  **if** **(**node\_id **==** 5**){**  instance**->**dtsn\_out **=** 111**;**  buffer**[**pos**++]** **=** instance**->**dtsn\_out**;**  **}**  **else{**  buffer**[**pos**++]** **=** instance**->**dtsn\_out**;**  **}** |

Code 4: DIO Suppression

By analysing the mote output below (Figure 11: DIO Suppression Mote Output) it can be confirmed that the above changes (Code 4: DIO Suppression) have been implemented as the values do not change during the simulation.

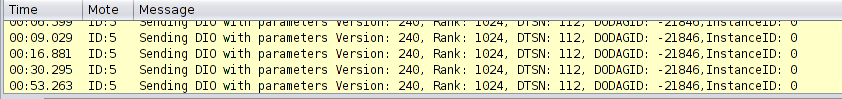


Figure 11: DIO Suppression Mote Output

## 2.3 Evaluation

### 2.3.1 Scenario 1

Within the 12 attacks carried out within this scenario the power consumption and delivery ratio will be analysed an average across all 3 seeds. Aforementioned in Chapter 2.1.1 mote 5 will be selected as the malicious mote as it has the highest number of neighbours.

#### 2.3.1.1 Delivery Ratio

The delivery ratio is calculated as a percentage of the lost: received packets. Within the sinkhole attack it is expected that 50% of the packets sent to mote 5 are lost as flag1 is altered from 0 to 1 on a packet to packet basis. However, as only half of the network is affected it is expected that the average delivery ratio is around 75% as only an estimated 25% of packets are dropped. Moreover, due to the positioning of the malicious mote packets 8 – 18 are expected to also see a drop-in packet as there is no alternative route that can be taken to the mote as seen below in Figure 10: Delivery Ratio Averages.

Figure 12: Scenario 1 Delivery Ratio

As seen above, the rank attack also suffers a greater drop in packets than the sinkhole attack although no packets are written to be dropped by the code. With most of the motes in the network a stable parent is selected and kept for the remainder of the 20-minute simulation. However, as the rank is increased the malicious mote firstly selects mote 4 as its parent and direct route to the sink. However, once mote 5 increases it’s rank the parent selection becomes unstable as it continues to change its parent within the first few minutes to mote 9 (rank 356) -> mote 10 (rank 356) -> mote 18 (rank 356) -> mote 12 (356). Within the rest of the simulation mote 12 remains the parent as mote 5 is compared to mote 4 however never makes the selection of mote 4 as its parent as seen below in Figure 13: Unstable Parent Selection.

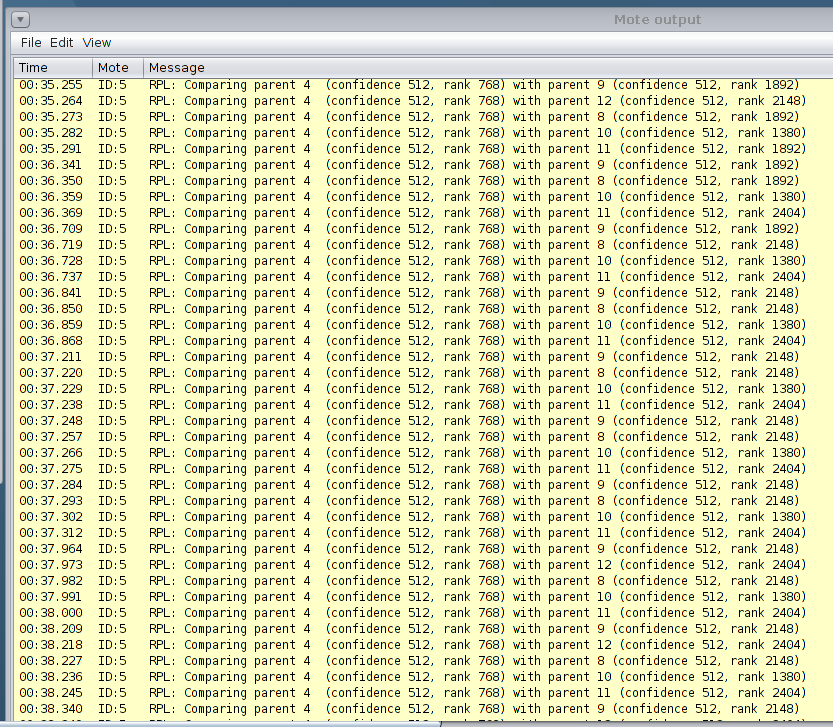


Figure 13: Unstable Parent Selection

Firstly, the unstable selection of parents could be down to each child of the malicious mote having the same rank and therefore the malicious mote cycles through each child to select it as its parent. Moreover, the literature review can back up the theory that the rank attack has caused an undetectable loop within the network and has therefore taken the left-hand branch of the network offline increasing itself to be the sink of the branch. Secondly, the dramatic drop of the delivery ratio on the rank attack in comparison to the sinkhole attack can be due to the branch being taken offline and instead of a partial sending of motes in the sinkhole attack no packets are received by the sink.

The version attack suffers a similar drop of packets compared to the rank attack. However, the parent’s mote of the malicious mote (mote5) are affected as their rank are continually updating. As discussed in the literature review the further the malicious mote is from the sink the longer it takes the sink to release that the attack is taking place. As both motes 2 and 4 are 1-2 hops away from the sink they are still able to deliver packets but at a very low delivery ratio. Furthermore, unlike the rank attack where the right-hand branch of the tree is not effect; the version attack is spread across the entire network causing disruption in the right-hand mote causing the version attack to have a lower delivery ratio than the rank attack. As mote 5 is the only gateway to the sink in the left-hand branch this causes a bottle neck as messages are flooded and unable to make it through the link causing the branch to go offline.

As seen in Figure 12 the DIO Suppression attack has no effect on the random topology as analysing the data shows 100% delivery ratio. This could be down a few factors; the attack is not implemented correctly as there is no reading of received DIO messages which are replayed; the DTSN is not listened to by other motes due to the abundance of legitimate motes on the network and although the malicious mote replays the same DIO message the network parameters do not change within the simulation. Therefore, on a hardware scenario where new motes join the network and move geographical location this attack could affect the network as the incorrect updates are being broadcasted to neighbours.

#### 2.3.1.2 Power Consumption

The power consumption is averaged as the total of the CPU, LPM, Listening and Transmit power. Within each attack the change in power consumption is dependent on the unnecessary sending and receiving on messages sent to the sink. For example, motes are required to repeatedly resend half of their messages within the attack area of the DODAG as only half or partial messages are sent to the sink. Moreover, due to the rank attack taking the branch offline messages are also sent.

Figure 14: Scenario 1 Power Consumption

As seen above in Figure14: Scenario 1 Power Consumption the power usage across the 3 seeds and attacks are similar however pose slight differences. Across the 3 seeds the sink attack uses less power than the default no attack simulation. Furthermore, the rank attack uses more power across the 3 seeds. As discussed in the Delivery Ratio the rank attack creates a loop and constantly compares its parent to mote 4 to attempt to reach the sink. Moreover, during the rank attack the malicious mote changes parents’ multiple times and this could be the reason for the higher power usage.

The sinkhole attack uses less power as packets are dropped and therefore energy is saved as power is not used to send half of the packets on the branch. Although it can be argued that motes will have to resend their packets to reach the sink the packet per minute is set to 2 and therefore this limit cannot be overcome by the motes. Therefore, the motes would need to resend their packets, but the same packet will be sent in 30 seconds instead of new data. This would overall affect the quality of data being sent to the mote and increase delays across the network as motes must send their packets multiple times to be delayed creating a back log of traffic waiting to be sent to the sink.

In comparison, as the version number attack increases the DIO messages sent between the motes forcing an update of the DODAG this attack floods the network with updates causing a spike in power consumption as seen in Figure 11 as the power is almost 3 times higher than the non-attack, sink and rank attack simulations. This is due to the sink and rank attack not having much contact with its neighbours or other motes in the network. Whereas, the version number attack spreads across the entire network.

The DIO suppression attack has little effect on Power Consumption within the random topology. This is expected due to the lack of activity within the delivery ratio. However, the power consumptions decrease although the reason of this cannot be defined as the DIO Suppression Attack is supposed to target the network resources.

### 2.3.2 Scenario 2

Aforementioned in Chapter 2.1.2 this scenario will be setup as a linear topology with the furthest away mote being mote 6. The same format of tests will be carried out as scenario 1 as 12 simulations will be run simulating the rank, sink, version number attack and no attack with 3 different seeds set out in Table 1: Seed Setup.

#### 2.3.2.1 Delivery Ratio

Within this scenario determining the effect of the attacks before simulation can be tricky as unlike the previous scenario there is no single route to the sink. Therefore, it can be assumed that RPL can selfheal and redirect packets to an alternative route. Furthermore, the positioning of the mote must be taken into consideration as the furthest away mote. This creates the question of is there going to be any effect at all? As the mote is furthest away it has no child motes and therefore only forwards packets to its parents (Table 3: Scenario 2 Parent and Delivery Ratio Comparison). Theoretically the only data that will be affected by these attacks would be the data sent by the malicious mote.

|  |  |  |  |
| --- | --- | --- | --- |
| **Mote ID** | **Parent ID** | **Received** | **Lost** |
| **2** | 7 | 39 | 0 |
| **3** | 23 | 39 | 0 |
| **4** | 9 | 39 | 0 |
| **5** | 10 | 39 | 0 |
| **6** | 11 | 39 | 0 |
| **7** | 12 | 39 | 0 |
| **8** | 13 | 39 | 0 |
| **9** | 14 | 39 | 0 |
| **10** | 9 | 39 | 0 |
| **11** | 10 | 39 | 0 |
| **12** | 17 | 39 | 0 |
| **13** | 18 | 39 | 0 |
| **14** | 13 | 39 | 0 |
| **15** | 14 | 39 | 0 |
| **16** | 11 | 39 | 0 |
| **17** | 1 | 39 | 0 |
| **18** | 17 | 39 | 0 |
| **19** | 24 | 39 | 0 |
| **20** | 19 | 39 | 0 |
| **21** | 26 | 39 | 0 |
| **22** | 1 | 39 | 0 |
| **23** | 22 | 39 | 0 |
| **24** | 23 | 39 | 0 |
| **25** | 24 | 39 | 0 |
| **26** | 25 | 39 | 0 |
| **27** | 22 | 39 | 0 |
| **28** | 27 | 39 | 0 |
| **29** | 24 | 39 | 0 |
| **30** | 25 | 39 | 0 |
| **31** | 30 | 39 | 0 |

Table 3: Scenario 2 Parent and Delivery Ratio Comparison

Moreover, the theory that linear topology will see very minimal effects to attack with using the furthest way mote can be confirmed by viewing the delivery ratio below in Figure 15: Scenario 2 Delivery Ratio.

Figure 15: Scenario 2 Delivery Ratio

As seen above only half of the attacks have affected the Delivery Ratio within the linear topology. Firstly, the sinkhole/selective forward attack has made no impact on the network. This could be since the malicious mote is the furthest away from the sink and therefore has no parent selection. Therefore, the only messages that the malicious mote sends on are from itself resulting in no packets being dropped. However, as previously seen by the rank attack within the random topology delivery ratio is reduced by the attack.

Within this attack the rank is set to 1544 which is lower than the malicious mote parent 5, and parent 5’s parent 4. A similar result is obtained as this creates a loop within the network however not as drastic as the random topology as instead of half of the motes being affected only mote 5 and 11 are affected. As the rank is decreased mote 5 and 11 select the malicious mote as their parent. However, mote 6 also selects mote 5 and 11 as its parent and therefore creating a loop between the 3 motes as seen below in Figure 16: Rank Parent Loop.

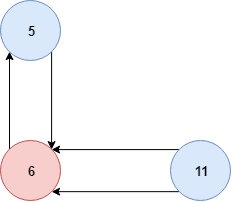


Figure 16: Rank Parent Loop

However, the version attack massively disrupts the linear network as it is placed on the furthest away mote from the sink causing the attack to be more effective as earlier discussed in Chapter 1.8. Due to this the network is highly flooded and owing to the distance causes a delay on the sink realising the version has changed making it impossible for it to keep up with the changes. The sink only receives 12% of the packets sent to the sink with the ratio increasing on the motes closest to the sink. However, only the closest 7 motes to the sink were able to send packets as seen below in Figure 17: Linear Version Successful Motes.

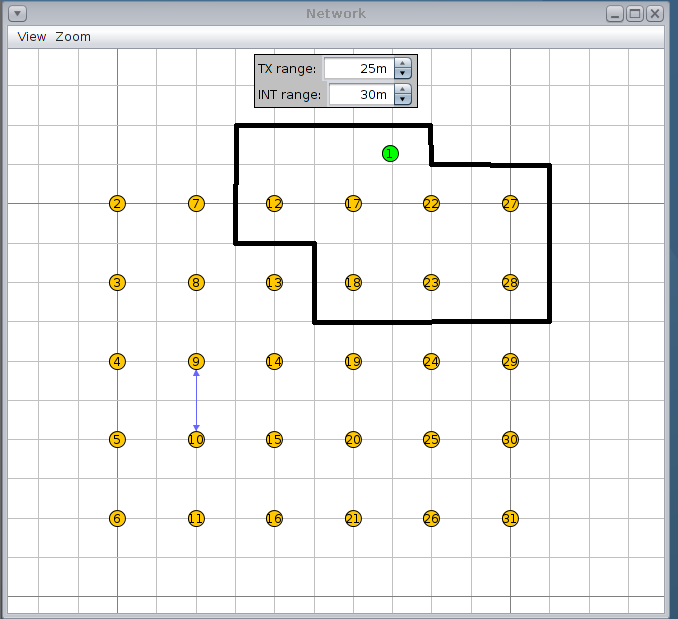


Figure 17: Linear Version Successful Motes

Similarly, to the random topology the DIO suppression attack has no effect to the linear topology. This could be due to similar reasons to the sinkhole attack and the distance makes this attack ineffective. Other reasonings already set out in Chapter 2.3.1.1 could be relevant to the linear topology as well as the random topology.

#### 2.3.2.1 Power Consumption

Within the linear topology power consumption is relatively consistent apart from the version attack. In comparison to the default no attack simulation the power consumption ranges from +- 100 mW. These changes aren’t entirely massive however given the low power and resource availability on IoT devices these consistent small changes over a period could shorten the battery and life of the device. As earlier discussed in the previous chapter the sinkhole/selective forward attack is ineffective as the malicious mote is furthest away from the sink and has no packets to send other than its own. However, the rank attack has a similar impact on the network in terms of power consumption as seen in the random topology as power is decreased. This is due to the loss of packets being transmitted throughout the network although on a smaller scale as only 2 motes are affected compared to the 18.

Figure 18: Scenario 2 Power Consumption

As seen above in Figure 18: Scenario 2 Power Consumption the version attack has the greatest impact on the linear topology as the power consumed is tripled across each seed. This is a similar outcome to the random topology. As earlier discussed in Chapter 1.8 Version Number Attack the effectiveness of the attack is enhanced the further away the mote is from the sink. As within this scenario, the malicious mote is the furthest away mote. Thus, the version attack can spread across the network without the sink noticing until the attack has been fully deployed. Finally, the DIO suppression attack has no effect on the network much the random topology and delivery ratios.

# Chapter 3 – Hardware

## 3.1 Device Setup

3 sky motes will be used to build a small network to test the effect of the DIS attack. The devices will be placed on 3 work-stations – 2 laptops and a PC in C27. The device setup is outlined alongside attack role below in Table 4: Hardware Setup.

|  |  |  |
| --- | --- | --- |
| **Mote ID** | **Role** | **Attack Role** |
| 1 | Server | No Attack |
| 2 | Client2 | DIS Attack |
| 3 | Client3 | No Attack |

Table 4: Hardware Setup

Each device will use the IOTSecurityDIS package from Lab 8 to enable the sending of DIO, DIS and Average Power settings.

## 3.2 DIS Attack Implementation

Aforementioned in chapter 1.9 the DIS attack is a flood attack which uses up resources. The DIS attack periodically sends DIS messages to its neighbours triggering a response of DIO messages updating the mote of the DODAG parameters. The DIS attack can be implemented by resetting the timer to 0 triggering the malicious mote to send a DIS Message to its neighbours and receiving a DIO in return. The timer implementation set out in Code 5: DIS Timer Reset are implemented on the malicious mote (mote 2).

|  |  |
| --- | --- |
| **No Attack Timer** | **Attack Timer** |
|  |  |

Code 5: DIS Timer Changes

A counter is added into dis\_input() function with the rpl-icmp6 file. This function triggers everytime a DIS function is received by the sink and mote3.

|  |
| --- |
| DisCounter **=** Discounter **+** 1 |

Code 6: DISCounter

## 3.3 Results

The DIS attack is expected to affect the power consumption on the victim server and client mode. This is due to the spam in DIS and DIO responses experienced within the network. The modes periodically output their average power consumption to terminal as seen below in Figure 18: Terminal Power Consumption.

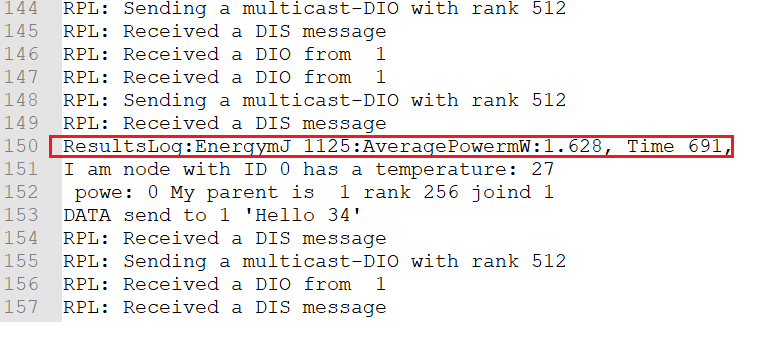


Figure 19: Terminal Power Consumption

By piping the output of the motes into a text file, the average power can easily be extracted by grepping and cutting the output to obtain the results. These results can then be put into excel and the average function used (Figure 20: DIS Power Consumption).

Figure 20: DIS Power Consumption

As seen above the power consumption increases in the malicious mote, server and client3. However, the cost for client2 to mount the attack is at greater consumption in comparison to the server and client 3. This is due to the sending and receiving of DIS/DIO messages to more than one device. Whereas, the server and client3 only send and receive in unicast to the malicious mote. Client3 and server power consumption is increased due to the flood of DIO/DIS messages to them. Also, the increase of power consumption can be down to the rebuild of topology and routing metrics.

# References

Accettura, N., Grieco, L. A., Boggia, G., & Camarda, P. (2011). Performance analysis of the RPL Routing Protocol. In *2011 IEEE International Conference on Mechatronics* (pp. 767–772). IEEE. https://doi.org/10.1109/ICMECH.2011.5971218

Aris, A., Oktug, S. F., & Berna Ors Yalcin, S. (2016). RPL version number attacks: In-depth study. In *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium* (pp. 776–779). IEEE. https://doi.org/10.1109/NOMS.2016.7502897

Badach, A. (2018). RPL Message DIO. Retrieved November 28, 2018, from https://www.researchgate.net/publication/326960645\_RPL\_Message\_DIO

Cervantes, C., Poplade, D., Nogueira, M., & Santos, A. (2015). *Detection of Sinkhole Attacks for Supporting Secure Routing on 6LoWPAN for Internet of Things*. Retrieved from http://dl.ifip.org/db/conf/im/im2015m/137522.pdf

Chugh, K., Lasebae, A., & Loo, J. (2012). Case Study of a Black Hole Attack on 6LoWPAN-RPL. *International Conference on Emerging Security Information, Systems and Technologies*, *6*, 157–**162**. Retrieved from https://www.thinkmind.org/index.php?view=article&articleid=securware\_2012\_6\_20\_30036

Claveria, K. (2018). 13 stunning stats on the Internet of Things. Retrieved November 12, 2018, from https://www.visioncritical.com/internet-of-things-stats/

Dvir, A., Holczer, T., & Buttyan, L. (2011). VeRA - Version Number and Rank Authentication in RPL. In *2011 IEEE Eighth International Conference on Mobile Ad-Hoc and Sensor Systems* (pp. 709–714). IEEE. https://doi.org/10.1109/MASS.2011.76

IoTone. (n.d.). Low Power and Lossy Networks (LLN). Retrieved from https://www.IoTone.com/term/low-power-and-lossy-networks-lln/t343

Nawir, M., Amir, A., Yaakob, N., & Lynn, O. B. (2016). Internet of Things (IoT): Taxonomy of security attacks. *2016 3rd International Conference on Electronic Design (ICED)*, 321–326. https://doi.org/10.1109/ICED.2016.7804660

Perazzo, P., Vallati, C., Anastasi, G., & Dini, G. (2017). DIO Suppression Attack Against Routing in the Internet of Things. *IEEE COMMUNICATIONS LETTERS*, *21*(11). https://doi.org/10.1109/LCOMM.2017.2738629

Perrey, H., Landsmann, M., Ugus, O., Hamburg, H., Wählisch, M., & Schmidt, T. C. (2015). *TRAIL: Topology Authentication in RPL*. Retrieved from https://arxiv.org/pdf/1312.0984.pdf

Pongle, P., & Chavan, G. (2015a). A survey: Attacks on RPL and 6LoWPAN in IoT. In *2015 International Conference on Pervasive Computing (ICPC)* (pp. 1–6). IEEE. https://doi.org/10.1109/PERVASIVE.2015.7087034

Pongle, P., & Chavan, G. (2015b). *Real Time Intrusion and Wormhole Attack Detection in Internet of Things*. *International Journal of Computer Applications* (Vol. 121). Retrieved from https://github.com/pavanpongle/IoT-Wormhole-IDS

Pu, C., & Zhou, X. (2018). Suppression Attack Against Multicast Protocol in Low Power and Lossy Networks: Analysis and Defenses. *Sensors (Basel, Switzerland)*, *18*(10). https://doi.org/10.3390/s18103236

RGHIOUI, A., KHANNOUS, A., & BOUHORMA, M. (2014). Denial-of-Service attacks on 6LoWPAN-RPL networks: Issues and practical solutions. *Journal of Advanced Computer Science & Technology*, *3*(2), 143. https://doi.org/10.14419/jacst.v3i2.3321

Sharma Sr Assistant Professor, D., Mishra Sr Assistant Professor, I., & Jain, S. (2017). *Impact factor: 4.295 A Detailed Classification of Routing Attacks against RPL in Internet of Things*. *International Journal of Advance Research*. Retrieved from www.ijariit.com

Sinanovic, H., & Mrdovic, S. (2017). Analysis of Mirai malicious software. In *2017 25th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)* (pp. 1–5). IEEE. https://doi.org/10.23919/SOFTCOM.2017.8115504

Wallgren, L., Raza, S., & Voigt, T. (2013). Routing Attacks and Countermeasures in the RPL-Based Internet of Things. *International Journal of Distributed Sensor Networks*, *9*(8), 794326. https://doi.org/10.1155/2013/794326

Yu, Y., Li, K., Zhou, W., & Li, P. (2012). Trust mechanisms in wireless sensor networks: Attack analysis and countermeasures. *Journal of Network and Computer Applications*, *35*(3), 867–880. https://doi.org/10.1016/J.JNCA.2011.03.005