Implementing and evaluating variations of the Blackhole attack on RPL

Adam Pettersson
Abstract

The IPv6 Routing Protocol for Low-power and Lossy Networks (RPL) is the most used routing protocol for resource constrained Internet-of-Things (IoT) networks. With the massive increase in number of Internet-connected IoT devices and the fact that they are becoming more common in safety-critical environments such as in health-care and in the industry, security in these networks are of a big concern. A promising approach for threat detection in these networks is the use of intrusion detection systems (IDS) based on machine learning. This thesis focuses on two things: designing new attack variations based on the Blackhole attack which is an attack on RPL and evaluating these variations on an IDS that has been trained on regular blackhole attacks. The attacks that we have implemented are being run in the Cooja network simulator with the Multitrace extension and the data obtained from these simulations are processed and used to train a Random Forest Classifier machine learning model for the IDS. The attack variations that we implemented managed to impact the recall of the Random Forest classifier but not so other metrics such as accuracy and precision which are often used to evaluate machine learning models. The results show that variations on the blackhole attack can be used to avoid detection by an IDS and still be able to reduce the performance of the network.
Acknowledgments

I want to thank my thesis supervisor Andreas Johnson and Amin Kaveh for the time they have invested in me and this thesis during our weekly meetings. Thanks should also go to Christian Rohner for providing valuable comments and corrections to this report. Lastly, I would like to thank my family, Nicole, Theodor and Alicia for encouraging me and cheering me on through this period.
## Contents

1 Introduction 5

2 Background 6
   2.1 IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) 6
   2.2 IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) 7
   2.3 Attacks on RPL 8
      2.3.1 Blackhole Attack 9
   2.4 Intrusion Detection Systems 10
   2.5 Machine Learning 11
      2.5.1 Machine learning classifiers 11

3 Related work 12

4 Methodology 13
   4.1 Framework for generating data traces 14
   4.2 IDS Design 15

5 Attack-variation strategies 18
   5.1 On/off variation (1 malicious node) 18
   5.2 Decreasing rank variation (1 malicious node) 19
   5.3 Introducing a second malicious node 19
      5.3.1 Sequential on/off variation (2 malicious nodes) 19
      5.3.2 Sequential decreasing rank variation (2 attacking nodes) 20

6 Evaluation 21
   6.1 Evaluating a ML classifier 21
   6.2 Attack Impact Metrics 23
   6.3 Evaluation 24

7 Discussion 26

8 Conclusions and future work 27

A IDS Performance and Attack Impact Metrics 31
1 Introduction

As more and more Internet of Things (IoT) devices are connected to the internet and their applications often can be found in safety-critical areas such as medical care, industry, transportation and security [1], security in IoT networks is a major concern. There has been attacks where IoT devices has been hacked and used in other attacks. The Mirai Botnet [2] was a botnet composed of IoT devices among others which wreaked havoc on the internet with massive DDoS attacks in 2016. This shows the importance of IoT security not just for the IoT network itself but for the internet as whole.

The defense of a network often consists of three main components. Prevention, detection and mitigation, see Figure 1. This thesis will focus on the detection part, i.e., when an attacker has managed to circumvent the first component ’prevention’ and the goal is to detect suspicious behavior in the network.

Since nodes in IoT networks often suffer from resource constraints such as a limited power supply, low transmission bandwidth, etc., intrusion detection techniques used in traditional networks are not directly applicable in these settings [3].

Butun et al. in [3] lists three different detection methodologies which IDSs can be categorized into: anomaly-based detection, misuse-based detection and specification-based detection. We will study the anomaly-based detection methodology and more specifically machine learning based IDSs. The use of machine learning when designing IDSs has proven to be a promising approach as we can see in [4] and [5].

One of the many challenges in designing an IDS based on machine learning for IoT networks is data availability. To train machine learning models for IDSs you need data and it is also important to consider that the more diverse the data set, the more knowledge and insights may be learned and captured by the model.

The problem with this approach is that for an IDS to be able to detect an attack it has to been trained on that specific attack type since its model is based on previous attacks. Because of this, some attack types may go under the radar and remain undetected by the IDS because of lack of indicative training samples.

Figure 1: Defense of a network
Research Challenges

This thesis will focus on simulating attack scenarios, and specifically attack scenario variations, to aid in further developing IDSs by providing new data sets from attack simulations and see how much impact variations of the attack has on the performance of the IDS. The attack scenarios will be implemented on motes running the Contiki operating system and simulated in the Cooja network simulator. We are using an IDS model developed by authors in [4] to evaluate our attack variations.

The research challenges this thesis aims to tackle are:

1. How can we design attack variations to try to avoid detection by an IDS?
2. How do these attack variations impact an IDS trained on similar attacks?
3. How much impact does the attack variations have on a network?

2 Background

The first two subsections will cover protocols and standards which relate to IoT networks, following that are a section about attacks on RPL. A subsection is devoted to the Blackhole attack which is the main focus of this thesis. The following sections introduces a high level overview on machine learning and the random forest classifier.

2.1 IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN)

6LoWPAN [6] is a set of IETF standards which makes it possible to use IPv6 in devices with constraints regarding power and processing over low-bandwith wireless networks. This is done by compressing the IPv6 header and using fragmentation and reassembly functionality provided by an adaptation layer that is part of the 6LoWPAN protocol stack [7] depicted in Figure 2.
Networks using 6LoWPAN support multihop communication, meaning that nodes can forward packets on behalf of other nodes. 6LoWPANs are connected to the conventional internet using 6LoWPAN border routers (6BR), an example can be seen in Figure 3.

Figure 3: Example of interconnection between the Internet and a 6LoWPAN

2.2 IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL)

RPL [8] is a standardized protocol for routing in IoT networks. RPL forms a Destination Oriented Directed Acyclic Graph (DODAG) between the reachable nodes in the network. An example of a DODAG can be seen in Figure 4 with the sink node depicted as green. A sink node is called a DODAG root in the RPL specification and can act as a border router (BR6) to connect the 6LoWPAN to the internet [9].
Each node in the graph has a unique id and a rank that defines the node’s position relative to other nodes with respect to a sink node. Rank is strictly increasing downwards from the sink node and strictly decreases upwards towards the sink node. The computation of rank varies depending on the Objective Function (OF) for the DAG and could for example depend on topological distance or link metrics.

The RPL protocol provides a set of new Internet Control Message Protocol for IPv6 (ICMPv6) [10] control messages for exchanging routing graph information.

Table 1 illustrates some of the different control messages used in RPL.

<table>
<thead>
<tr>
<th>ICMPv6 Control Messages</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Advertisement Object (DAO)</td>
<td>Used to advertise information required to support downward traffic towards leaf nodes</td>
</tr>
<tr>
<td>DODAG Information Object (DIO)</td>
<td>Contains information that is used to build the RPL DODAG</td>
</tr>
<tr>
<td>DODAG Information Solicitation (DIS)</td>
<td>Used to request graph related information from neighbor nodes</td>
</tr>
<tr>
<td>Destination Advertisement Object Acknowledgment (DAO-ACK)</td>
<td>Used to inform the sender of a DAO that DAO has been received</td>
</tr>
</tbody>
</table>

Table 1: RPL ICMPv6 control messages.

2.3 Attacks on RPL

Attacks on RPL can broadly be classified into three categories; resource based attacks, traffic based attacks and network topology based attacks [11]. Further
classifications can be done in all categories but the major attack-classifications can be seen in figure 5.

Attacks in the Resource based attacks category can be further classified into direct and indirect attacks. With the flooding attack the intention is to deplete resources and is considered a direct attack. The version number attack is initiated by the global repair mechanism for RPL and increases the control traffic and thus generating costly and unnecessary transmissions, this is an indirect attack.

Traffic based attacks are classified into eavesdropping attacks and misappropriation attacks where the self-explanatory sniffing attack falls under the eavesdropping category and an example of a misappropriation attack is the identity attacks where the attacker node clones the identity of another node to gain traffic intended for the victim node or through the victim node.

The third category is network topology based attacks which can be sub-classified into sub-optimization attacks and isolation attacks. Isolation attacks aim to segregate a particular node from the network and cut off its communication abilities with other nodes.

Sub-optimization attacks aims to minimize the performance of the network by getting involved in the optimal path selection. The blackhole attack belongs to the sub-optimization category and will be examined more in-depth in the next section.

Figure 5: Attacks in RPL

2.3.1 Blackhole Attack

A blackhole attack is when a malicious node in a network drops all packets it is supposed to forward to the sink node. The attack can be disassembled into two steps. One of the steps is to attract neighbors of the malicious node to select it as the parent by advertising a spoofed low rank. This is an attack on
the network topology since it disrupts the optimal path selection and creates a new sub-optimal topology. The other step in the attack is dropping all the packets originating from other nodes.

There are two variations of the blackhole attack called *sink hole attack* and *selective forwarding attack* which both comprises of the spoofed rank step but differs in the packet dropping step. In the sinkhole attack the attacking node is not dropping any packets but instead just routing the traffic through itself, this could be combined with a sniffing attack in order to get hold of more network traffic. The selective forwarding attack on the other hand allows some of the packets to reach the sink node based on predefined rules.

In figure 6 you can see a scenario with 15 nodes that has formed a DODAG and is displaying normal behavior. The same scenario can be seen in figure 7 but in this case the network is under a blackhole attack from the node with id 9 which has disrupted the RPL routing path and attracted its adjacent neighbors to choose it as their parent.

![Normal behavior](image1.png) ![Active attack](image2.png)

**Figure 6: Normal behavior**  **Figure 7: Active attack**

### 2.4 Intrusion Detection Systems

*Intrusion Detection Systems* (IDSs) are one of the resources that can be used in protecting networks from unauthorized or unapproved activities. A protection system for IoT networks typically consists of three components: Prevention, detection and mitigation [4].

Prevention measures are the first line of defense and can involve techniques such as encryption and authentication.

IDSs belong under the detection component which aims to reveal intrusions as soon as possible to prevent undesired disclosures to the attacker.

Since nodes in IoT networks often suffer from resource constraints such as limited power supply, small memory size and data storage, well known and
proven techniques for developing IDSs in traditional networks are not directly applicable in IoT networks [3].

IDSs can be classified into three different categories regarding deployment. They can either be network based, host based or application based depending on the participants characteristics. The method an IDS uses to detect attacks can also be divided into two classes, signature based or anomaly based where a signature based system is trained on specific patterns extracted from previously known attacks and anomaly based attacks are trained on normal data collected from non-attack scenarios [12].

2.5 Machine Learning

Machine learning plays an important role in many different areas today. The concept is to 'learn from data'. In a typical scenario, there is an outcome measurement which could be binary (such as attack/no attack in an IDS scenario) or quantitative (as in stock price predictions). The goal is then to be able to predict this outcome based on a set of features. Some examples of learning problems that machine learning can help with are:

- Image recognition, identify letters in a handwritten letter [13].
- Stock market predictions, forecast stock prices using market metrics [14].
- Designing IDSs, classifying network behavior [15].

The 'learn from data' can be approached in different ways, you have Supervised and Unsupervised Learning for instance where the key difference is that with Supervised Learning you use labeled data to help predict outcomes. Unsupervised Learning on the other hand does not require any labeled data. These two approaches have their own strengths and weaknesses and thus is applicable in different areas. Supervised Learning is good for classification and regression problems while Unsupervised Learning is more fitting for clustering, association and dimensionality reduction tasks [16].

An example of Supervised Learning could be that we have a training set of data where we observe the outcome and the feature measurements for a set of objects. With this data we can build a prediction model which enables prediction of outcome for new unseen objects [17]. With Unsupervised Learning there is no need for human intervention and the machine learning algorithm can make these predictions by discovering patterns in the data on its own.

2.5.1 Machine learning classifiers

Machine learning classifiers are the algorithms that maps input data to a specific category. There exists both Supervised and Unsupervised classifiers, some of the classifiers used for training IDSs are Support Vector Machines (SVM), Deep Neural Networks (DNN), Random Forest Classifier (RFC), K-means and K-Dimensional Tree [4] where the last two are Unsupervised classifiers while the rest are Supervised classifiers.
These classifiers differ in complexity which means that there are situations where a classifier might require too much resources to be an appropriate choice even though it would be the best choice when considering just the prediction accuracy. This is especially relevant when dealing with IoT networks since nodes in these networks are often very resource constrained.

RFC

RFC is a machine learning classifier which works by creating a collection of classification and regression trees [18]. These trees operate as an ensemble and each tree in the ensemble makes a prediction and the class with the most votes becomes the models’ prediction. RFC has shown to be applicable in resource constrained devices due to their simple nature and low memory and performance requirements [19].

3 Related work

Authors in [20] addresses the problem that there is a lack of sufficient data traces which are required to train many machine learning algorithms for intrusion detection systems (IDS) in wireless low-power, multi-hop networks. In multi-hop networks there are no single entity which can overhear all the traffic and this has contributed to the lack of traces. The paper presents Multi-Trace which is an extension to the Cooja network simulator that can generate multi-level traces. Multi-Trace solves the issue of not having one omniscient entity in the network by enabling data logging at different levels which all share a global simulation time. One of the levels enables logging from the radio medium which can provide provide information about which nodes received or were interfered by each radio transmission; thus solving the problem with not having an entity which can capture all the network traffic.

Keipour et al. in [4] utilizes Multi-Trace when they generate a large dataset of attacks on IoT mesh networks which they use to train machine learning algorithms. The paper presents an ML-based IDS for 6LoWPAN which is trained by the dataset generated by Cooja and Multi-Trace. Several different algorithms are evaluated and each of them achieve a high accuracy and precision. Since IoT devices often have resource constraints it is also important to take the complexity of the ML algorithm into consideration when deciding on which model to use.

Authors in [21] presents implementations of most of the known RPL attacks for the Contiki-NG OS and also a framework in Cooja which allows for hybrid RPL attacks, meaning you could have combinations of attacks on multiple nodes. They address the issue that previous work in this field has generated datasets that are not very dynamic or random which affects the machine learning anomaly-based algorithms. By using the framework that the authors present, datasets that are generated are more dynamic and random and could potentially uncover unexpected real scenarios.
Pongle and Chavan in [22] lists possible attacks on both RPL and on the 6LoWPAN protocol, for each attack they specify the effect on the network and methods for counter measures. They also discuss different methods which can be used by an IDS such as event detection based, signature detection based, host based and specification based. Since IoT devices often have limited resources in terms of power and processing power the authors have taken this into consideration when they are discussing the placement for the IDS, they can either be distributed, centralized or hybrid depending on the complexity.

Sharma et al. in [5] analyzes security threats in RPL and possible attacks using Cooja and a self developed framework. The authors implement four different attacks on RPL, two resource consuming attacks (‘Hello Flood’ and ‘DIS’) and two routing attacks (‘Reduced rank’ and ‘Increased version’). With the traffic data generated from the simulations they propose a new machine learning model which achieve a very high (99.3%) classification accuracy on those four attacks using a Random-Forest classifier.

Each of the papers described above (except Multi-Trace [20]) describe implementations of attacks that involve rank decreasing such as blackhole attacks or sinkhole attacks. They do however not introduce any variations to the rank decreasing but instead implement or assume a static decrease to a certain rank. This thesis presents several different scenarios where the attacker or attackers are altering a nodes’ rank in a more dynamic way.

4 Methodology

The Cooja network simulator was used to run the simulations of all scenarios. It is a flexible Java-based simulator designed for simulating networks of sensors running the Contiki operating system [23]. Together with the extension Multi-trace [20] the data needed for evaluating the IDS performance and impact of the attacks were obtained.

Since one of the research questions this thesis aims to answer is how performance of an IDS varies depending on attack parameters, we used the same approach as the authors in [4] when creating simulation scenarios. This enabled us to get a point of reference which we could compare the results of our attack variations with.

Considering the data generated from the simulations are going to be used to train machine learning models for an IDS we want a diverse dataset originating from many different network topologies. This is achieved by creating 64 different scenarios for each type of attack with the number of participating nodes ranging from 5 to 20 with varying placements which we will run the simulations on.

The following sections will describe the methodology and in Figure 8 you can see it illustrated, end-to-end.
4.1 Framework for generating data traces

Creating base scenario

For every attack variation, a base simulation file is created which includes placement of nodes and also parameters for setting up each different attack. There are 16 base scenarios since we run simulations using 5 nodes up to 20 nodes.

Generating new scenarios from the base

A Python script\(^1\) is used to generate 4 new scenarios where the base scenario is used as input to create new scenarios with randomized placements of nodes. There are some limitations on how the nodes are placed to avoid network partitions, we make sure that no node is placed outside the transmission range of any other node.

Running simulations from scenario files

To run all the simulation scenarios generated in the previous step a python script is used. Cooja has a no-GUI functionality which we use in order to speed up and enable automatized simulations. The results for each simulation is stored in separate folders containing data collected during the simulation. Figure 9 shows the resulting files after a successful simulation and a brief description of each log file can be found in Table 2.

\(^1\)Provided by the authors of [4]
File name | Description
---|---
cooja.log | Cooja information with system time logs
cooja.testlog | Cooja script logs
events.log | Logs events during the simulation
mote-output.log | Contains output from all motes throughout the simulation
radio-log.pcap | Contains packet data of the network
radio-medium.log | Logs transmissions from the radio medium
script.log | Logs from events in script

Table 2: Description log files

Parsing data from simulations

The simulations returns logs containing mote output, a radio-log in the .pcap format and information about run-time of the scenario. This is parsed using a python script into a .csv file containing the features presented in table 3. The reason why these features were selected is because the IDS we are using to evaluate our attack variations are expecting these features as input for the machine learning algorithms.

| Time | Time stamp of events in the simulation |
| Mote | ID number of Node |
| Seq | Request number |
| Rank | Rank value of node |
| Version | Version of the DODAG |
| DIS-R | No. of DIS message received (Δ) |
| DIS-S | No. of DIS message sent (Δ) |
| DIO-R | No. of DIO message received (Δ) |
| DIO-S | No. of DIO message sent (Δ) |
| DAO-R | No. of DAO message received (Δ) |

Table 3: Features parsed from each nodes’ RPL statistics

4.2 IDS Design

We relied on concepts and methods developed by the authors of [4] for the design of the IDS with some small modification of the chunking size explained in
the 'Chunking'-subsection. This was done in order to be able to compare our attack variations with the already implemented 'reference attack' and to use that attack as a base line.

Pre-processing data

Data cleaning is the first step performed to make sure the dataset is of high quality. We normalize some of the features in Table 3 in order to make the ML model training faster. The features being normalized are Rank, DIS-R, DIS-S, DIO-R, DIO-S and DAO-R and they are normalized using the generalized min-max scaling method to map the data to the range of $[-1, 1]$ using Equation 1.

\[
x' = 2 \left( \frac{x - x_{\text{min}}}{x_{\text{max}} + 1} - 1 \right)
\]  

(1)

Chunking

After the normalization is done the data is split into 16 equal size time intervals called chunks to analyze the selected features changes between these chunks. Since we have 16 chunks and the simulations run for 5 hours (300 mins), each chunk is a time interval spanning 18.75 minutes. To ensure that the network has had time to stabilize the data traces from the first two chunks are considered as a warm-up period and thus removed from consideration.

The features values in the remaining chunks are then averaged and stored in a vector. This will be used later in the labeling step. A feature matrix is then formed with all of the vectors of each node in each scenario. The features we focus on in Table 3 are Mote and Rank.

Feature engineering

Feature engineering is a method used to categorize the collected data to improve classification performance. The chunking method described above is a form of feature engineering, namely feature extraction. Another category of feature engineering is feature selection which we perform using Principal Component Analysis (PCA) to be able to reduce the feature dimension. This is done to compress the data and reduce the required storage space. It helps to decrease the computation time as well.

Data labeling

The next step is labeling the data. The nodes affected by an attacker is labeled as 'under attack' and the nodes that are not affected by the attacker in attack
scenarios together with all the nodes in non-attack scenarios are labeled as 'non-attack'.

The variance of a node's rank between chunks are used to determine if a node is under attack or not. Since a blackhole attack consists of a rank attack where the malicious node attracts neighbors to choose it as parent by spoofing a low rank, the children of the malicious nodes' rank will also experience a decrease in rank.

**Machine Learning Model**

After the data labeling step is done we have a feature matrix that is ready to use with a ML Model to perform the classification task. The feature matrix is used to train the predictive model which will map the input data to one of the attack or non-attack classes. This thesis will use the supervised RFC ML algorithm for detecting attacks in the network because of the promising results in [4] and because of its simplicity which makes it suitable for resource constrained IoT networks.

**Reference attack**

To be able to evaluate the attack-variations presented in this thesis we use the attack implemented in [4] as a reference attack, the reference attack is labeled as attack with ID 1 in Tables 4, 8 and 9.

In the reference attack one randomly selected node starts a blackhole attack after 15 minutes and the attack is active during the entire 5 hour simulation. The spoofed rank of the attacking node is 128 throughout the simulation.
5 Attack-variation strategies

This section will describe the different attack variations implemented in this project. All of the attack variations are blackhole attacks but with different rank variation tactics. In order to try to avoid detection by an IDS we try two different approaches with respect to rank variation. First we employ a strategy where the attack is turned on and off in fixed intervals. The other strategy is to decrease the rank in steps until a predetermined threshold is reached instead of instantly reducing the rank to that threshold.

In half of the simulations there are one malicious node and in the other half there are two malicious nodes who are performing the attacks in sequence, never at the same time.

This section addresses the first research challenge on how we can design attack variations to try to avoid detection by an IDS.

5.1 On/off variation (1 malicious node)

In the on/off variation scenarios one randomly selected node starts a blackhole attack after 15 minutes. When the attack has been active for a predetermined period of time the attacker-node pauses the attack and starts acting as a non-malicious node. After the same period of time has passed the attack is resumed and this process is repeated until the simulation ends. The spoofed rank of the attacking node is 128 throughout the simulation. These scenarios are labeled with ID 2-6 in Tables 4, 8 and 9.

The motivation for this variation is to reduce the exposure time of the attack by simply turning off the attack in different time intervals and let the malicious node act as a normal node. This will of course reduce the impact of the attack since we have off-periods where we do not drop any packets.

This could however be of interest to an attacker if the attacker is willing to compromise with the impact of the attack for a lower detection rate.

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>Malicious Nodes</th>
<th>Spoofed rank</th>
<th>Rank variation method</th>
<th>Rank variation parameters</th>
<th>Attack start time</th>
<th>Attack stop time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Blackhole</td>
<td>1</td>
<td>128</td>
<td>Static</td>
<td>None</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>2</td>
<td>Blackhole</td>
<td>1</td>
<td>128</td>
<td>On/off</td>
<td>10 min intervals</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>3</td>
<td>Blackhole</td>
<td>1</td>
<td>128</td>
<td>On/off</td>
<td>15 min intervals</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>4</td>
<td>Blackhole</td>
<td>1</td>
<td>128</td>
<td>On/off</td>
<td>20 min intervals</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>5</td>
<td>Blackhole</td>
<td>1</td>
<td>128</td>
<td>On/off</td>
<td>25 min intervals</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>6</td>
<td>Blackhole</td>
<td>1</td>
<td>128</td>
<td>On/off</td>
<td>30 min intervals</td>
<td>15</td>
<td>300</td>
</tr>
</tbody>
</table>

Table 4: Blackhole attack variations (on/off).
5.2 Decreasing rank variation (1 malicious node)

In these scenarios one randomly selected node starts a blackhole attack after 15 minutes and the attack is active throughout the entire simulation. The main difference from the reference attack is that we do not immediately advertise the low (128) rank but instead start at a higher rank (411) and decrement the rank value each minute by a predetermined decrement-size ranging from 1/min to 32/min until the rank reaches 128.

The reason 411 was chosen as initial rank is that this enables attack with ID 7 reach rank 128 right before the simulation ends. These scenarios are labeled with ID 7-12 in Tables 5, 8 and 9.

The motivation behind this variation was that an attack might be less obvious if the decrease in rank was not instant but instead happened gradually. Even though the attack is active throughout the entire simulation we should not expect the same impact as the reference attack since the malicious node require some time before reaching a sufficient rank in order to be able to attract its neighbors. Again, an attacker might settle for a lower impact if more attacks goes undetected.

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th># Malicious Nodes</th>
<th>Rank variation method</th>
<th>Stop decreasing when rank</th>
<th>Rank variation parameters</th>
<th>Attack start time</th>
<th>Attack stop time</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Blackhole</td>
<td>1</td>
<td>Decreasing</td>
<td>≤128</td>
<td>Decrement by 1/min</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>8</td>
<td>Blackhole</td>
<td>1</td>
<td>Decreasing</td>
<td>≤128</td>
<td>Decrement by 2/min</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>9</td>
<td>Blackhole</td>
<td>1</td>
<td>Decreasing</td>
<td>≤128</td>
<td>Decrement by 4/min</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>10</td>
<td>Blackhole</td>
<td>1</td>
<td>Decreasing</td>
<td>≤128</td>
<td>Decrement by 8/min</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>11</td>
<td>Blackhole</td>
<td>1</td>
<td>Decreasing</td>
<td>≤128</td>
<td>Decrement by 16/min</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>12</td>
<td>Blackhole</td>
<td>1</td>
<td>Decreasing</td>
<td>≤128</td>
<td>Decrement by 32/min</td>
<td>15</td>
<td>300</td>
</tr>
</tbody>
</table>

Table 5: Blackhole attack variations (decreasing rank attack).

5.3 Introducing a second malicious node

The following two subsections describe two attack variations that extends the previous two by adding another malicious node in the scenario. The motivation behind this was that attacks might be harder to detect if the origin of an attack is not a fixed point but instead comes from multiple points in a network.

5.3.1 Sequential on/off variation (2 malicious nodes)

The sequential on/off attack variations introduce a second malicious node in the simulations. Two distinct nodes are randomly chosen at the start of the simulation. The attack starts after 15 minutes with one of the malicious nodes.
Then, just as in the other on/off scenario, after a predetermined period of time the attacking node pauses the attack. After the same period of time has passed the attack is resumed but now by the other malicious node. This process is then repeated until the end of the simulation. These scenarios are labeled with ID 13-17 in Tables 6, 8 and 9.

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>#Malicious Nodes</th>
<th>Spoofed rank</th>
<th>Rank variation method</th>
<th>Rank variation parameters</th>
<th>Attack start time</th>
<th>Attack stop time</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>Blackhole</td>
<td>2</td>
<td>128</td>
<td>On/off</td>
<td>10 min intervals</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>14</td>
<td>Blackhole</td>
<td>2</td>
<td>128</td>
<td>On/off</td>
<td>15 min intervals</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>15</td>
<td>Blackhole</td>
<td>2</td>
<td>128</td>
<td>On/off</td>
<td>20 min intervals</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>16</td>
<td>Blackhole</td>
<td>2</td>
<td>128</td>
<td>On/off</td>
<td>25 min intervals</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>17</td>
<td>Blackhole</td>
<td>2</td>
<td>128</td>
<td>On/off</td>
<td>30 min intervals</td>
<td>15</td>
<td>300</td>
</tr>
</tbody>
</table>

Table 6: Sequential Blackhole attack variations (on/off).

5.3.2 Sequential decreasing rank variation (2 attacking nodes)

The last type of attack is a variation on the Decreasing rank variation-attack. Like in the other sequential attack we introduce a second malicious node in the start of the simulation. The initial rank on both nodes is set to 238. This is chosen so that both nodes will be able to reach the specified rank (128) before simulation ends.

The first node initiates the attack after 15 minutes and is active for 30 minutes before pausing and lets the other malicious node initiate the attack. The same process is then repeated until the simulation ends. A 30 minute attack window was chosen because this had shown to reduce performance of the network the most in previous simulations, see Figure 15.

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>#Malicious Nodes</th>
<th>Rank variation method</th>
<th>Stop decreasing when rank</th>
<th>Rank variation parameters</th>
<th>Attack start time</th>
<th>Attack stop time</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>Blackhole</td>
<td>2</td>
<td>Decreasing</td>
<td>≤ 128</td>
<td>Decrement by 1/min</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>19</td>
<td>Blackhole</td>
<td>2</td>
<td>Decreasing</td>
<td>≤ 128</td>
<td>Decrement by 2/min</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>20</td>
<td>Blackhole</td>
<td>2</td>
<td>Decreasing</td>
<td>≤ 128</td>
<td>Decrement by 4/min</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>21</td>
<td>Blackhole</td>
<td>2</td>
<td>Decreasing</td>
<td>≤ 128</td>
<td>Decrement by 8/min</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>22</td>
<td>Blackhole</td>
<td>2</td>
<td>Decreasing</td>
<td>≤ 128</td>
<td>Decrement by 16/min</td>
<td>15</td>
<td>300</td>
</tr>
<tr>
<td>23</td>
<td>Blackhole</td>
<td>2</td>
<td>Decreasing</td>
<td>≤ 128</td>
<td>Decrement by 32/min</td>
<td>15</td>
<td>300</td>
</tr>
</tbody>
</table>

Table 7: Sequential Blackhole attack variations (decreasing rank attack).
6 Evaluation

In order to evaluate the results of the attack variations we consider both the impact the attack has on the network but also the performance of the IDS under our attack variations. What we can see is that there are trade-offs that can be made, an attacker might be satisfied with a low impact if the detection rate by the IDS is low.

It could also be the case that the highest possible impact is desired with no regards to detection rate.

6.1 Evaluating a ML classifier

In appendix A the performance metrics from the RFC machine learning model can be seen in Table 8.

Recall

The main metric we use to evaluate attack variations is recall. Recall is measuring how many of the Actual Positives the model captures through labeling it as a 'True Positive'. (See Equation 2 and Figure 10).

\[
\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}
\]

Recall is a good model metric to select when False Negatives are highly undesirable. This is especially relevant to applications such as IDSs since if an attack (Actual Positive) is predicted as a non-attack (Predicted Negative) that means the IDS was unable to detect the attack.

Accuracy and Precision

Accuracy and precision are two other metrics that can be used to measure the performance of ML models'. Accuracy can be calculated by dividing the number
of correct predictions with all the predictions, see equation 3.

However, with IDSs the focus should lie on minimizing false negatives (e.g., an actual attack remaining undetected) and not maintaining high accuracy since the costs of having those miss-classifications can be high.

\[
\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{True Pos.} + \text{False Neg.} + \text{False Pos.} + \text{True Neg.}}
\]

\[
= \frac{\text{No. of Correct predictions}}{\text{Total no. of predictions}}
\]

Precision is a useful metric when the costs of False Positives are high. It measures how many of the predicted positives actually were positives (see equation 4). In the case with an IDS it is of course desirable to achieve a high precision in order to decrease false alarms but should not be the main focus since false positives are not as critical as false negatives.

\[
\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}
\]

\[
= \frac{\text{True Positive}}{\text{Total Predicted Positive}}
\]

**Area Under Curve**

Area Under Curve (AUC) is used to measure a classifiers ability to distinguish between positive and negative classes by calculating the area under the Receiver Operator Characteristic (ROC) curve [24]. The ROC curve plots the True Positive Rate (Eq. 5) against False Positive Rate (Eq. 6) A higher AUC-score means a higher chance that the classifier will be able to distinguish positive classes from negative classes (i.e., it is able to detect more True Positives and True Negatives than False Negatives and False Positives). Figure 11 illustrates ROC and AUC for the reference attack.

\[
\text{True Positive Rate} = \text{Recall}
\]

\[
\text{False Positive Rate} = \frac{\text{False Positive}}{\text{True Negative} + \text{False Positive}}
\]
6.2 Attack Impact Metrics

In order to measure the impact the different attack variations had, a Python script was written which parsed the mote-output log. Every node (except the sink node) sends an application layer packet containing a sequence number every minute intended for the sink node, this event is logged in the mote-output log file. Upon receiving the packet, the sink node logs that the message was received. In the event that a malicious node drops a packet, a discrepancy occurs in the presumed equality of sent packets vs received packets by sink. Using Equation 7 we can calculate the minimum impact the attack has on the network in the specific scenario. This is calculated and tallied up for all 64 scenarios (total number of simulations for each scenario) running that attack variation and divided by 64 in order to get an average impact value. This helps us answer the third research challenge by calculating the impact an attack has on a network.

\[
\text{Min. impact} = 1 - \frac{\text{Received packets by sink node}}{\text{Sent packets by all nodes}}
\]  

(7)

The reason we call this minimum impact is that this does not capture corner cases where the malicious node is placed as in figure 12. The nodes only generate the application layer packets when they are reachable by the sink node so in this case the impact is actually higher than calculated using equation 7 since the node with ID 6 is not generating the minutely packets and thus not contributing to the denominator in Equation 7.
6.3 Evaluation

Figures 15 and 16 shows the impact and recall for all the attack variations. In figure 15 we can see that none of those attack variations could achieve the same impact as the reference attack which is expected since we have periods where the malicious node is not actively attacking the network.

This however, is not the case with attacks depicted in figure 16. Since the attacks are continuous throughout the simulation we can see some cases where the impact is higher than in the reference attack.

An outlier can be seen in figure 16 in the second bar, the reason behind this could be a very fortunate placement of the malicious node in many of the randomized simulations in that scenario. Figure 13 and 14 displays two similar scenarios where a small change in the placement of nodes result in a big difference in impact of the attack.
Figure 15: Recall and Impact (On/off variations)

Figure 16: Recall & Impact (Decreasing rank variations)
7 Discussion

As we can see from the results in Table 8 the attack variations that we’ve implemented can impact the performance of the IDS with respect to recall but not so much with when examining other metrics. If we look at the results for AUC we can see that its close to 1.0 in each scenario. The accuracy is also high for each scenario. These two correlate since a high AUC means that the model is very accurate and can distinguish between positive and negative classes in a confusion matrix (depicted as green and red in figure 10).

The IDS used in this thesis is designed in a way such that a node is considered to be under attack and thus labeled as ‘under attack’ if the variance in the nodes’ rank is above a given threshold between the intervals called chunks. Given that the nodes’ are stationary throughout the entire simulation, the rank should not fluctuate unless there’s an ongoing attack in the network.

The attack variations implemented in this thesis are all focused on affecting the rank in various ways and should thus not be able to have any major impact on the prediction accuracy. As stated earlier however, accuracy should not be the main focus when evaluating the performance of an IDS since it does not capture how many of the misclassifications were highly undesired false negatives. Instead, recall should be the main model metric when evaluating IDSs since it measures how many actual intrusions remained undetected (false negatives).

The chunks obtained from the chunking method described in Section 4.2 are rather large (each chunk represents 18.75 minutes). Since the IDS is using the variance in a nodes’ rank between these chunks to understand if a node is under attack or not, this causes a delay which could be avoided if smaller chunks were used. Limitations in the IDS used in this thesis restricted the use of smaller chunks.
8 Conclusions and future work

The goals of this thesis was to develop and design new attack variations based on previously known attacks and analyze how these attack variations can impact the performance of an IDS in IoT networks. The attack scenario that we used as a point of reference for our attack variations was a blackhole attack with a static rank decrease which was active throughout the simulation. The variations include toggling the attack on/off in fixed intervals and also gradually decreasing the rank with varying rates. A second malicious node was also introduced in some of the attack variation scenarios.

Running these attack variations in the Cooja network simulator with the Multi-trace extension resulted in datasets which were used to evaluate and train machine learning models for an IDS. Testing the variations on an IDS that had been trained on the reference attack showed that we could impact the IDS performance with respect to recall. We used recall as the main metric when we evaluated the IDS since it highlights the occurrences of false negatives.

We could also see that our attacks all had impact on the performance of the network with varying degrees. This allows an attacker to choose between high impact on the network or low detection rate.

The simulations has been conducted in ideal conditions with no movement in nodes and with guaranteed success in receiving and transmitting data (i.e., no interference between nodes). All nodes has also been placed within reach of each other so that no partitions in the network occur.

In order to get more realistic scenarios, future work could either be done on test beds with physical motes or in simulators using more dynamic settings such as mobile nodes, introducing interference and less restrictions on placement of nodes.

Another proposal for future work is to increase the number of chunks used in the feature engineering step when designing the IDS. This would result in a faster response time for the IDS since it will be able to detect deviant behavior earlier with smaller chunk sizes.
References


# IDS Performance and Attack Impact Metrics

<table>
<thead>
<tr>
<th>ID</th>
<th>Accuracy</th>
<th>Precision</th>
<th>Recall</th>
<th>F1 Score</th>
<th>Cohen Kappa Score</th>
<th>ROC AUC Score</th>
<th>AUC Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.9636</td>
<td>0.9638</td>
<td>0.95</td>
<td>0.93</td>
<td>0.90</td>
<td>0.96</td>
<td>0.99</td>
</tr>
<tr>
<td>2</td>
<td>0.9267</td>
<td>0.9270</td>
<td>0.85</td>
<td>0.89</td>
<td>0.83</td>
<td>0.91</td>
<td>0.98</td>
</tr>
<tr>
<td>3</td>
<td>0.9547</td>
<td>0.9123</td>
<td>0.91</td>
<td>0.91</td>
<td>0.88</td>
<td>0.94</td>
<td>0.99</td>
</tr>
<tr>
<td>4</td>
<td>0.9582</td>
<td>0.9357</td>
<td>0.94</td>
<td>0.94</td>
<td>0.91</td>
<td>0.95</td>
<td>0.99</td>
</tr>
<tr>
<td>5</td>
<td>0.9688</td>
<td>0.9670</td>
<td>0.97</td>
<td>0.94</td>
<td>0.92</td>
<td>0.97</td>
<td>0.99</td>
</tr>
<tr>
<td>6</td>
<td>0.9590</td>
<td>0.9392</td>
<td>0.94</td>
<td>0.94</td>
<td>0.91</td>
<td>0.96</td>
<td>0.99</td>
</tr>
<tr>
<td>7</td>
<td>0.9552</td>
<td>0.9123</td>
<td>0.92</td>
<td>0.92</td>
<td>0.89</td>
<td>0.95</td>
<td>0.99</td>
</tr>
<tr>
<td>8</td>
<td>0.9554</td>
<td>0.9410</td>
<td>0.94</td>
<td>0.94</td>
<td>0.90</td>
<td>0.95</td>
<td>0.98</td>
</tr>
<tr>
<td>9</td>
<td>0.9718</td>
<td>0.9465</td>
<td>0.98</td>
<td>0.96</td>
<td>0.94</td>
<td>0.97</td>
<td>1.0</td>
</tr>
<tr>
<td>10</td>
<td>0.9383</td>
<td>0.9379</td>
<td>0.89</td>
<td>0.91</td>
<td>0.86</td>
<td>0.93</td>
<td>0.99</td>
</tr>
<tr>
<td>11</td>
<td>0.9471</td>
<td>0.9465</td>
<td>0.91</td>
<td>0.93</td>
<td>0.89</td>
<td>0.94</td>
<td>0.99</td>
</tr>
<tr>
<td>12</td>
<td>0.9439</td>
<td>0.9338</td>
<td>0.90</td>
<td>0.92</td>
<td>0.88</td>
<td>0.94</td>
<td>0.98</td>
</tr>
<tr>
<td>13</td>
<td>0.9326</td>
<td>0.9412</td>
<td>0.88</td>
<td>0.91</td>
<td>0.85</td>
<td>0.92</td>
<td>0.98</td>
</tr>
<tr>
<td>14</td>
<td>0.9521</td>
<td>0.9516</td>
<td>0.93</td>
<td>0.94</td>
<td>0.90</td>
<td>0.95</td>
<td>0.98</td>
</tr>
<tr>
<td>15</td>
<td>0.9670</td>
<td>0.9569</td>
<td>0.97</td>
<td>0.96</td>
<td>0.94</td>
<td>0.97</td>
<td>0.99</td>
</tr>
<tr>
<td>16</td>
<td>0.9653</td>
<td>0.9518</td>
<td>0.96</td>
<td>0.96</td>
<td>0.93</td>
<td>0.97</td>
<td>0.99</td>
</tr>
<tr>
<td>17</td>
<td>0.9640</td>
<td>0.9564</td>
<td>0.96</td>
<td>0.96</td>
<td>0.93</td>
<td>0.96</td>
<td>0.99</td>
</tr>
<tr>
<td>18</td>
<td>0.9541</td>
<td>0.9522</td>
<td>0.94</td>
<td>0.94</td>
<td>0.91</td>
<td>0.95</td>
<td>0.99</td>
</tr>
<tr>
<td>19</td>
<td>0.9355</td>
<td>0.9476</td>
<td>0.89</td>
<td>0.92</td>
<td>0.86</td>
<td>0.93</td>
<td>0.99</td>
</tr>
<tr>
<td>20</td>
<td>0.9291</td>
<td>0.9440</td>
<td>0.87</td>
<td>0.91</td>
<td>0.85</td>
<td>0.92</td>
<td>0.97</td>
</tr>
<tr>
<td>21</td>
<td>0.9519</td>
<td>0.9429</td>
<td>0.93</td>
<td>0.93</td>
<td>0.90</td>
<td>0.96</td>
<td>0.98</td>
</tr>
<tr>
<td>22</td>
<td>0.9584</td>
<td>0.9410</td>
<td>0.89</td>
<td>0.92</td>
<td>0.87</td>
<td>0.93</td>
<td>0.98</td>
</tr>
<tr>
<td>23</td>
<td>0.9542</td>
<td>0.9134</td>
<td>0.92</td>
<td>0.92</td>
<td>0.89</td>
<td>0.94</td>
<td>0.98</td>
</tr>
</tbody>
</table>

Table 8: IDS Results for attack variations using Random Forest Classifier
Figure 17: Accuracy & Impact (On/off variations)

Figure 18: Accuracy & Impact (Decreasing rank variations)
Figure 19: Precision & Impact (On/off variations)

Figure 20: Precision & Impact (Decreasing rank variations)
<table>
<thead>
<tr>
<th>ID</th>
<th>Min. Impact</th>
<th>Type</th>
<th>Rank variation parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.2113</td>
<td>Reference attack</td>
<td>N/A</td>
</tr>
<tr>
<td>2</td>
<td>0.1392</td>
<td>On/off variation</td>
<td>10 min</td>
</tr>
<tr>
<td>3</td>
<td>0.1096</td>
<td>On/off variation</td>
<td>15 min</td>
</tr>
<tr>
<td>4</td>
<td>0.1292</td>
<td>On/off variation</td>
<td>20 min</td>
</tr>
<tr>
<td>5</td>
<td>0.1256</td>
<td>On/off variation</td>
<td>25 min</td>
</tr>
<tr>
<td>6</td>
<td>0.1529</td>
<td>On/off variation</td>
<td>30 min</td>
</tr>
<tr>
<td>7</td>
<td>0.1442</td>
<td>Decreasing rank</td>
<td>decrease by 1/min</td>
</tr>
<tr>
<td>8</td>
<td>0.1467</td>
<td>Decreasing rank</td>
<td>decrease by 2/min</td>
</tr>
<tr>
<td>9</td>
<td>0.2148</td>
<td>Decreasing rank</td>
<td>decrease by 4/min</td>
</tr>
<tr>
<td>10</td>
<td>0.2319</td>
<td>Decreasing rank</td>
<td>decrease by 8/min</td>
</tr>
<tr>
<td>11</td>
<td>0.2576</td>
<td>Decreasing rank</td>
<td>decrease by 16/min</td>
</tr>
<tr>
<td>12</td>
<td>0.1999</td>
<td>Decreasing rank</td>
<td>decrease by 32/min</td>
</tr>
<tr>
<td>13</td>
<td>0.0971</td>
<td>Seq. On/off variation</td>
<td>10 min</td>
</tr>
<tr>
<td>14</td>
<td>0.1186</td>
<td>Seq. On/off variation</td>
<td>15 min</td>
</tr>
<tr>
<td>15</td>
<td>0.1280</td>
<td>Seq. On/off variation</td>
<td>20 min</td>
</tr>
<tr>
<td>16</td>
<td>0.1429</td>
<td>Seq. On/off variation</td>
<td>25 min</td>
</tr>
<tr>
<td>17</td>
<td>0.1796</td>
<td>Seq. On/off variation</td>
<td>30 min</td>
</tr>
<tr>
<td>18</td>
<td>0.2717</td>
<td>Seq. Decreasing rank</td>
<td>decrease by 1/min</td>
</tr>
<tr>
<td>19</td>
<td>0.1714</td>
<td>Seq. Decreasing rank</td>
<td>decrease by 2/min</td>
</tr>
<tr>
<td>20</td>
<td>0.1928</td>
<td>Seq. Decreasing rank</td>
<td>decrease by 4/min</td>
</tr>
<tr>
<td>21</td>
<td>0.1518</td>
<td>Seq. Decreasing rank</td>
<td>decrease by 8/min</td>
</tr>
<tr>
<td>22</td>
<td>0.1594</td>
<td>Seq. Decreasing rank</td>
<td>decrease by 16/min</td>
</tr>
<tr>
<td>23</td>
<td>0.1584</td>
<td>Seq. Decreasing rank</td>
<td>decrease by 32/min</td>
</tr>
</tbody>
</table>

Table 9: Impact for various blackhole attacks