

Wormhole Attack Detection in 6LoWPAN RPL-based IoT Network using Machine Learning Approach



MCS

by

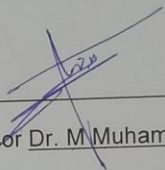
Asim Javed

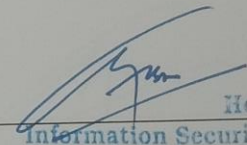
A thesis submitted to the faculty of Information Security Department, Military College
of Signals, National University of Sciences and Technology, Rawalpindi in partial
fulfillment of the requirements for the degree of MS in Information Security

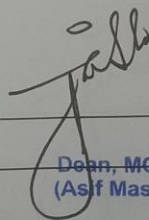
June 2023

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS/MPhil thesis written by Mr/MS Asim Javed, Registration No. 00000319086, of Military College of Signals has been vetted by undersigned, found complete in all respect as per NUST Statutes/Regulations, is free of plagiarism, errors and mistakes and is accepted as partial, fulfillment for award of MS/MPhil degree. It is further certified that necessary amendments as pointed out by GEC members of the student have been also incorporated in the said thesis.

Signature: 
Name of Supervisor Dr. M. Muhammad Waseem Iqbal
Date: _____

Signature (HoD):  HoD
Information Security
Date: _____ Military College of Sigs

Signature (Dean/Principal): 
Date: 20/7/23 Brig
Dean, MCS (NUST)
(Asif Masood, Phd)

Declaration

I hereby declare that no portion of work presented in this thesis has been submitted in support of another award or qualification either at this institution or elsewhere

A handwritten signature in black ink, appearing to be 'S. A. [unclear]', written over a horizontal line.

MS Student

Dedication

“In the name of Allah, the most Beneficent, the most Merciful”

I dedicate this thesis to my parents, brother, and teachers who supported me each step of
the way.

Acknowledgments

All praises to Allah for the strengths and His blessing in completing this thesis.

I would like to convey my gratitude to my supervisor, Dr. Mian Muhammad Waseem Iqbal for his supervision and constant support. His valuable help of constructive comments and suggestions throughout the experimental and thesis works are major contributions to the success of this research. Also, I would thank my committee members; Asst Prof. Mr. Waleed Bin Shahid, and Asst Prof. Dr. Yawar Abbas Bangash for their support and knowledge regarding this topic.

Last, but not the least, I am highly thankful to my parents and brother. They have always stood by my dreams and aspirations and have been a great source of inspiration for me. I would like to thank them for all their care, love and support through my times of stress and excitement.

Abstract

The rapid growth in the Internet of Things (IoT) has led to an increased risk of security attacks. An increase in network traffic has attracted cyber criminals and hackers to inject more network attacks into IoTs. The most common type of RPL-based network attack on an IoT network is a wormhole attack, which can have devastating effects on network performance and reliability. As traditional security approaches like cryptographic protocols, Distance-based methods, and Signal strength-based methods may not be effective against wormhole attacks due to their basic level of security and due to the dynamic nature of the IoT network. In this paper, we propose a machine-learning approach for detecting wormhole attacks in IoT networks. In our approach, we used a new dataset that was generated in the Cooja simulator to train and test a binary classifier that can accurately distinguish between normal network traffic and wormhole attack traffic. A wormhole is a complex type of network attack that depends on the multiple types of features. So instead of using the limited type of features, we have used some additional features compared to the ones already used by the researchers. After preprocessing the dataset, we trained and tested using different classifiers using hit and trial method, but among of them Artificial Neural Network (ANN), Ridge classifier, Deep Belief network, and (RBFN) classifier give the best results. Our results demonstrate that the proposed approach achieves high accuracy in detecting wormhole attacks, making it a promising solution for enhancing the security of IoT networks.

Keywords : Artificial Neural Network, Radial basis function, Deep belief network

Table of Contents

	Page
Abstract	v
Acknowledgements.....	iv
Thesis Acceptance Certificate.....	i
Table of Contents	vi
List of Figures and Tables.....	viii

Chapter

1 Introduction

1.1 Background.....	1
1.2 Motivation and Problem Statement.....	2
1.3 Research Objectives.....	3
1.4 Thesis Contribution.....	3
1.5 Thesis Organization	4

2 Backgrounds and Literature Review

2.1 Overview of the 6LoWPAN Network.....	5
2.2 Routing Protocol for Low Power Lossy (RPL) Network	5
2.3 6LoWPAN RPL-Protocol.....	7
2.4 Attacks on 6LoWPAN RPL-Protocol Stack	8
2.5 Wormhole Attack	9
2.6 Summary of the Literature	10

3 Proposed Methodology

3.1 Introduction.....	14
3.2 Flowchart of The Proposed Architecture	14
3.3 RPL Data Acquisition	15
3.3 Feature Selection Process.....	16
3.3 WH Attack Model Simulation	16

3.3 System Architecture.....	18
4 Experimental Analysis and Findings	
4.1 Experimental Setup.....	19
4.2 Dataset Generation.....	20
4.2.1 Benign Dataset Generation	21
4.2.1.1 Benign Power-trace Dataset Generation	21
4.2.1.2 Benign Network Traffic Dataset Generation	23
4.2.2 Malicious Dataset Generation.....	24
4.2.2.1 Malicious Power-trace Dataset Generation.....	24
4.2.2.2 Malicious Network Traffic Dataset Generation.....	25
4.3 Network Map After Deploying Attacking Nodes	25
4.4 Power Analysis Across Nodes	26
4.5 Data Preprocessing.....	27
4.5.1 Dataset Cleaning	27
4.5.2 Features Selection	28
4.5.3 Data Chunking	30
4.5.4 Dataset Scaling and Cleaning	32
4.5.4.1 Min-Max Normalization	33
4.5.5 NaN Values Removal.....	33
5 Results and Discussion	
5.1 Proposed Models.....	35
5.1.1 Ridge Classifier.....	35
5.1.2 Artificial Neural Network	36
5.1.3 Deep Belief Network	38
5.1.4 Radial Basis Function Network	38
5.2 Simulation Setup.....	41
5.3 Hit and Trial Method using different classifiers	41
5.4 Implementation of Ridge Classifier	42
5.5 Implementation of RNN.....	44
5.6 Implementation of DBN.....	45
5.7 Implementation of RBFN for Binary Classification	47

5.8 Performance Metrics	49
5.8.1 Accuracy	49
5.8.2 Confusion Matrix	49
5.8.3 Precision	50
5.8.4 Recall	50
4.8.5 ROC Curve and Operating Point	51
5.9 Comparative Analysis	51
5.10 Analysis and Recommendations	52
5.11 Discussion	53
6 Conclusion and Future Work	
6.1 Conclusion	56
6.2 Future Work	57
References	58

List of Figures and Tables

Fig 2.1: RPL Network.....	7
Fig 2.2: 6LoWPAN Network stack.....	8
Fig 2.3: Architecture of 6LoWPAN Network.....	8
Fig 2.4: Different Types of RPL-based attacks.....	9
Fig 2.5: Wormhole attack with two malicious nodes.....	10
Table 2.1: Summary of the Literature review	12
Fig 3.1: Flowchart of the proposed architecture	15
Fig 3.2 WH attack within IoT Network	16
Fig 3.3 Overview of the proposed architecture.....	18
Table 4.1: Default Mote interface settings for intercommunication	19
Table 4.3: Parameters setup for network under wormhole attack	20
Fig 4.3: Dataset Generation Types.....	21
Fig 4.4: Benign network setup in Contiki Cooja.....	22
Fig 4.5: Traffic showing using 6LoWPAN Analyzer	24
Fig 4.6: Malicious network in Contiki Cooja.....	25
Fig 4.7: Sensor map after deployment of wormhole attack	26
Fig 4.8: Average power metrics across node	27
Table 4.3: Brief description of the features.....	30
Table 4.4: Data Chunking with each row represents 60 sec duration	31
Table 4.5: Statistical description of the features	32
Fig 4.9: Normalized data after min-max normalization.....	33
Fig 5.1: ANN Architecture.....	37
Fig 5.2: Weight, bias, and activation function in ANN	37
Fig 5.3 RBF Network architecture	39
Table 5.1: Summary of the main models used in wormhole attack detection	40
Table 5.2: Hit and Trial analysis using multiple classifiers	41
Fig 5.6: ROC Operating point of ANN.....	45
Fig 5.7: Training of DBN.....	46

Fig 5.8: Confusion matrix using DBN.....	46
Fig 5.9: ROC Operating point of DBN.....	47
Fig 5.10: Confusion matrix of RBFN.....	48
Fig 5.11: ROC Operating point of RBFN.....	48
Fig 5.12: Binary class confusion matrix	50
Table 5.1: Comparative analysis of different algorithms.....	52

Introduction

1.1 Background

Due to the increasing growth in the wireless industry and rapid technological advancement in the Internet, there has been a significant increase in the number of IoT devices that are connected to the Internet. Due to the exponential growth in the IoT devices the world become smart but we have bigger security threats. The use of IoT devices has rapidly increased in various domains such as healthcare [1], smart homes [2], defense, and transportation systems. However, preserving the security of IoT networks has always been a significant concern due to their vulnerabilities to cyber-attacks.

The WH attack is one of the routing attacks in IoT networks. The RPL Routing attack such as Wormhole has a severe impact on the IoT network as it produces misleading and creates false path for the packets which disrupts the network behavior. In a wormhole attack, the attacking nodes make a secret connection between faraway points in the network, which lets them interrupt communication and modify the traffic between those points. This attack can lead to various security threats, such as data theft, device malfunctioning, and denial of service. Detecting and mitigating wormhole attack is crucial to make sure the security and privacy of IoT networks.

Traditional security mechanisms such as cryptographic protocols, Distance-based methods, and Signal strength-based methods may not be effective against wormhole attacks due to their basic level of security that cannot be effective for sophisticated types of wormhole attacks. Thus, there is a requirement for a more effective and advanced solution to detect wormhole attacks in IoT networks. Machine learning techniques based techniques provide more accurate and effective detection on wormhole attacks

1.2 Motivation and Problem Statement

The extensive usage of the IoTs has resulted to the deployment of large-scale wireless sensor networks (WSNs) in various domains. RPL is a mostly used protocol for resource-constrained devices. RPL works with 6LoWPAN [11], a Wireless Sensor Network (WSN) that employs the IEEE 802.15.4 protocol for data-link and physical layer communication and utilizes a complex version of the IPv6 protocol for networking. When they worked together, they create an IoT protocol specifically designed for routing over energy-efficient networks with limited connectivity. However, these networks are susceptible to security threats due to their low processing power capacity and baseline security features.

One of the most significant security threats in WSNs is the wormhole attack, which allows an attacker to redirect traffic from one node of the network to another node by creating a tunnel, causing network disruption, data tampering, network jamming, and unauthorized access. The wormhole attack in WSNs is more severe than other attacks due to its high impact on the network's functionality and communication reliability.

Hence, there is a need for an effective detection scheme. As various detection techniques have been proposed in the literature. However, majority of these techniques are either based on cryptographic techniques, or signal strength-based methods which are not well applicable for low power source-constrained devices. Although, some machine-learning based detection schemes also have been used but are based on low features.

To address these limitations, in this research, we propose a machine learning-based approach for detecting wormhole attacks in 6LoWPAN IoT networks. We generate a novel dataset using the COOJA [8] simulator, which mimics a real-world IoT network environment. The dataset contains various attack scenarios with different attack patterns and attack intensities. We compare the accuracy and detection rate by using different ML algorithms for multiple different features.

1.3 Research Objectives

The primary aims of the thesis are given as under: -

- To detect wormhole attacks using self-generated datasets in the COOJA simulator.
- Data preprocessing and optimal feature selection.
- Apply different machine learning algorithms by using multiple features to get the best accuracy and F-1 score against wormhole attack.

1.4 Thesis Contribution

This work makes the following main contributions.

- Dataset generation: The research generates a dataset for benign nodes and for wormhole-attacking nodes in 6LoWPAN network based on RPL networks using the COOJA simulator. The dataset includes different network topologies, traffic patterns, and attack intensities, making it a valuable resource for further research on IoT security.
- Data preprocessing and Feature selection: Dataset cleaning has been performed by using different statistical techniques and choosing new optimal features which contribute to the wormhole attack.
- Proposal of a machine learning-based approach: The research proposes a novel approach for detecting wormhole attacks in 6LoWPAN network based on RPL networks using ML algorithms. We performed hit and trial methods by using different ML and DL models. The hit and trial aims to provide an efficient and accurate solution to detect wormhole attacks in real time.
- Finally, we compare the algorithm's efficiency using different performance metrics like accuracy, precision, recall and F-1 score etc.

1.5 Thesis Organization

The thesis is arranged in the following manner.

- Chapter 2 contains the literature reviewed in the thesis which includes the comprehensive analysis of the existing literature. This includes the Overview of IoT networks and their security challenges, Wormhole attack in IoT networks, existing traditional and ML techniques for wormhole attack detection, and a summary of the literature review.
- Chapter 3 contains the proposed scheme for WHA detection, 6LoWPAN RPL-based IoT network architecture, the flowchart of the proposed architecture, and overview of the proposed scheme. The simulation results representing the working of the scheme are covered in Chapter 4.
- Chapter 4 covers implementation of the proposed architecture step wise, built of experimental step for analysis, how we collect the dataset, and perform data preprocessing.
- Chapter 5 contains the algorithms used for the current scenario, pros and cons of the algorithms. This chapter also covers the performance metrics used in the research. At the end we will cover the discussion on the results and findings.
- Chapter 6 will cover the conclusion and future work of the current thesis work.

Background and Literature Review

2.1 Overview of the 6LoWPAN Network

An IoT network is a system of interconnected devices, objects, and sensors that are designed to communicate with each other over the internet. IoT networks can include a wide range of devices, such as smart sensors, actuators, wearable devices, and other embedded devices, all of which are connected to the internet through various network protocols and technologies.

The IETF workgroup has standardized the 6LoWPAN protocol for the purpose of connecting devices in the IoT network. The 6LoWPAN standards allow IPv6 to be used efficiently on basic embedded devices connected to low-power, low-rate wireless networks. This is made possible by using an adaptation layer and optimizing related protocols [5]. This protocol is created specifically for limited-capacity devices in IoT networks, providing a lightweight solution for device connectivity [6]. Fig. 1 illustrates the usage of a 6LoWPAN border router to connect sensors to the internet. The protocol stack used for IPv6 over Low-power Wireless Personal Area Networks is flexible and adaptable, catering to a wide range of applications, including but not limited to environmental monitoring, automation, and smart agriculture [9].

2.2 Routing Protocol for Low Power Lossy (RPL) Network

The RPL is one of the communication protocols used for low-constrained IoT devices. The Internet Engineering Task Force (IETF) [3] developed this protocol to solve the issue of not having a routing standard that works well in IoT networks with limited resources, low power, and poor network conditions. It can handle three different types of network structures, namely, P2P, MP2P, and P2MP [10]. RPL operates based on distance vector routing principles and creates a directed acyclic graph (DAG) or destination-oriented DAG (DODAG) to handle the routing tasks among the nodes. Routing function means how the traffic is routed from one sensor to another sensor in which there is no direct node communication.

There must be a specific protocol using which the communication among nodes establishes between the source node and to the destination node. Thus the network topology in the RPL network is organized by the (DAG) or (DODAG), a graph where the connections between the nodes establish. There is a single node considered to be a root node which serves as the gateway. In a DODAG each node will compute its distance to the route in a distributed manner. This distance is called 'Rank'. The nodes distant from the root node will have high-rank values. RPL networks utilize five control messages to establish and maintain the DODAG structure and communication routes [4]. These controlled messages are strongly affected by the wormhole attack therefore there is a need to be discussed thoroughly.

- DODAG Information Object (DIO)
- DODAG Information Solicitation (DIS)
- Destination Advertisement Object (DAO)
- Destination Advertisement Object Acknowledgment (DAO-ACK)

At bootstrapping, the nodes don't know their rank except the root node. Thus, the root of the network starts transmitting periodically the DIO control packets with a given rank value. Typically, this rank is said to be 256. Any other node has to wait to receive the DIO control packet or they may send a DIS control packet to actually request from their neighbors to send the DIO packet. When the nodes are within the transmission range of the root node, receive its control packet and it will compute its own rank. There are several methods for the node to compute its rank value. For instance, it can be based on the distance between the transmitter and the root node, in terms of the number of hops or it can be based on the quality of the irregular links. The calculated rank of node B is higher than the rank of the root node A.

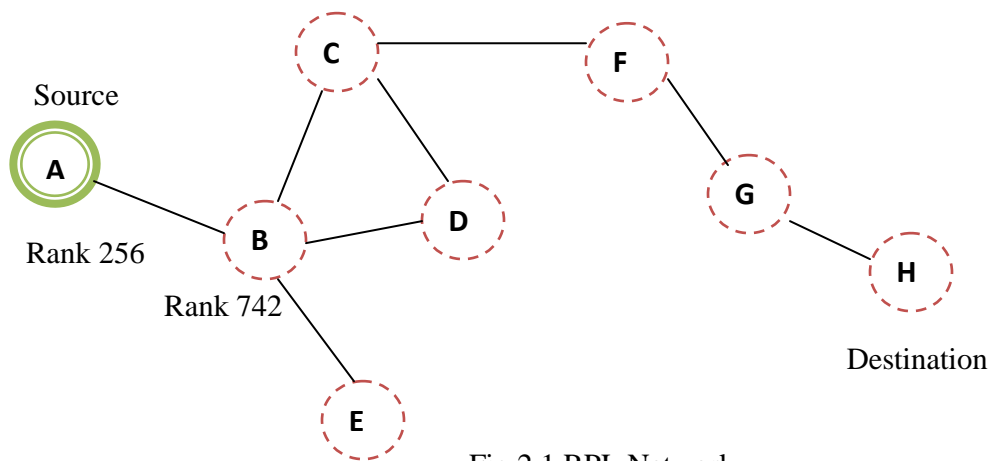


Fig 2.1 RPL Network

Similarly, node B starts periodically transmitting DIO control packets indicating its own rank. Nodes C, D, and E will receive these DIO control packets which were sent by the node B and will perform a similar operations as performed by node B. These all nodes will now compute their own rank values based on node B. This process will continue for the lifetime of the network. For the reverse route, from the route to any non-route nodes, there are two methods i-e (i) storing and (ii) no storing modes and the usage of another control packet (DAO). When a node selects a preferred parent it will send (DAO) to its parent to advertise its position in the topology. This position will be stored only in the root node i-e non-storing mode or intermediate node i-e storing mode. (DAO) is used for source routing and are sent by a node to update its parent node about the status of the path to the destination.

2.3 6LoWPAN RPL-Protocol stack

The 6LoWPAN RPL protocol is used in the IoT network being studied in this research. 6LoWPAN is a protocol that allows IPv6 packets to be sent over wireless networks with low power consumption. The protocol is designed to address the challenges of wireless sensor networks, such as limited bandwidth, minimum power consumption, and low memory resources. 6LoWPAN achieves this by compressing IPv6 packets to make them suitable for transmission over low-power wireless networks. On the other hand, RPL is a

Routing protocol used for transmission of data between nodes in a network. It is particularly useful for limited resource network, such as sensor network, where nodes may have limited resources and power.

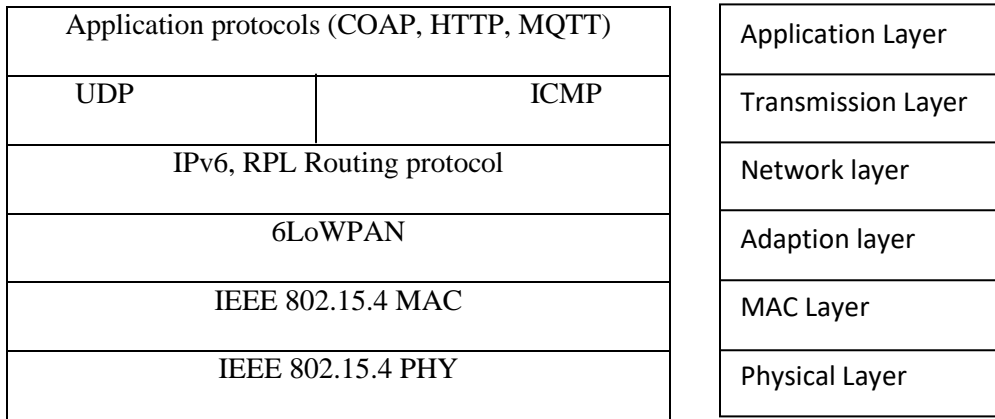


Fig 2.2. 6LoWPAN network stack

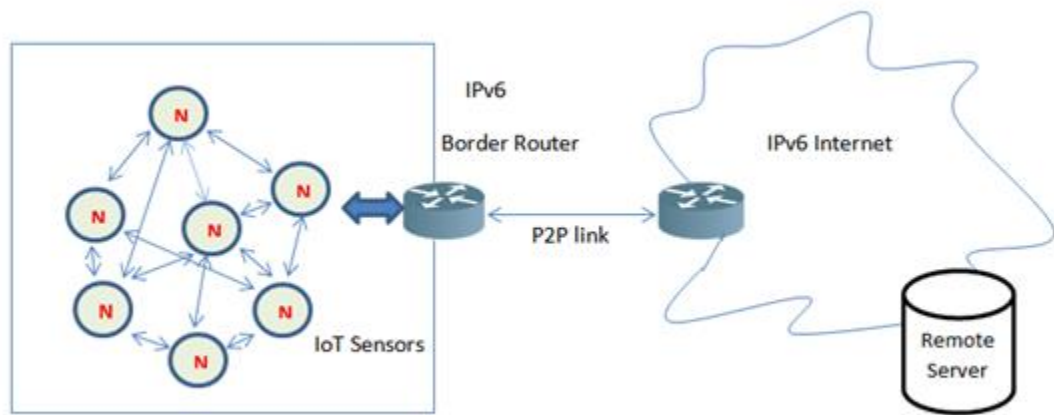


Fig 2.3. Architecture of 6LoWPAN network

2.4 Attacks on 6LoWPAN RPL-based stack

RPL is a widely used routing protocol in networks that have low power and experience signal loss, like 6LoWPAN. However, because of the specific nature of these networks, RPL-based networks are susceptible to different types of attacks. These attacks can be classified based

on several factors such as the topology of the network, available resources, and attack objectives.

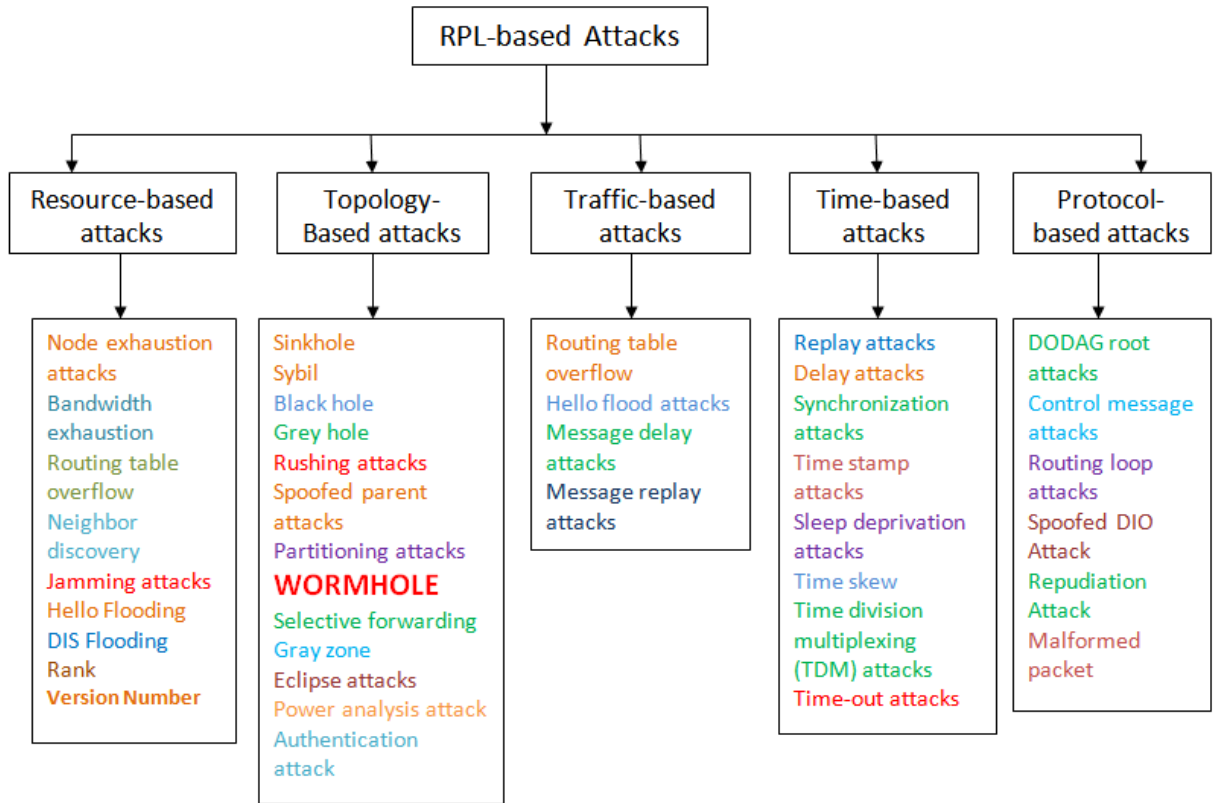


Fig 2.4. Different types of RPL-based attacks

2.5 Wormhole attack

Wormhole is actually a topology-based attack in which two or more than two malicious nodes create a tunnel within the network which disrupts the normal operation of the DAG by creating a shortcut between two points in the network, bypassing intermediate nodes. Marianne Azer et al. (2009) provide a detailed description of the wormhole attack in their paper [7]. This can result in packets being routed along a path that is not optimal, leading to increased latency, reduced throughput, and increased energy consumption.

Detecting and preventing wormhole attacks in RPL-based 6LoWPAN networks can be challenging, but several approaches have been proposed. One approach is to use

cryptographic techniques to authenticate packets and ensure that they have not been tampered with during transmission. Another approach is to use time synchronization to detect inconsistencies in packet delivery times, which can be indicative of a wormhole attack. But we will use Machine learning techniques based techniques provides more accurate and effective detection on wormhole attacks due to resource-constrained nature the nodes.

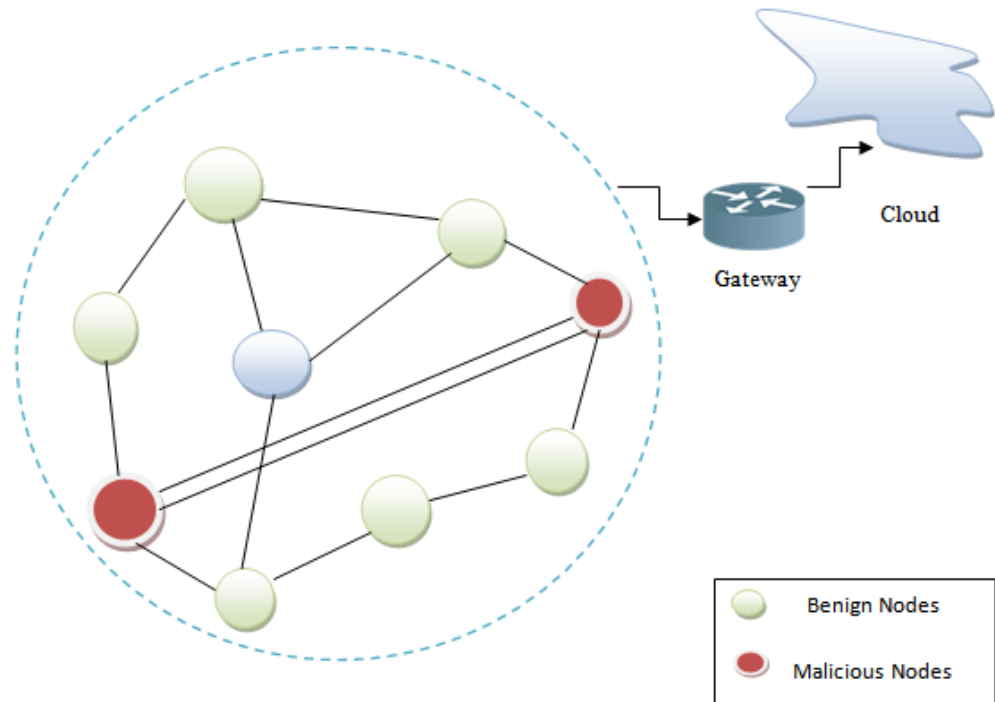


Fig 2.5: Wormhole attack with two malicious nodes

2.6 Summary of the Literature

- Prasad, M., Tripathi, S., & Dahal [12] proposed a ML models for wormhole attack detection in an adhoc networks creating multiple wormhole tunnels. They used three extra features such as multi-rate channel, processing delay, and neighbor monitoring in the previous datasets. Although this approach gives improved accuracy, but they get the manual feature selection which causes increased overfitting risk.

- Zahra, F., Jhanjhi, N.Z., Brohi, S.N., Khan, N.A., Masud, M. and AlZain, M.A [13] proposed a gradient-boosting machine-based algorithm (MC-MLGBM) for a multi-class attacks like Rank and Wormhole attack detection. The authors gathered both static and mobility-based datasets.
- Jhanjhi, N.Z., Brohi, S.N., Malik, N.A. and Humayun [14] proposed a rank and wormhole attack detection using the ML approach. In this paper, researchers proposed the detection of WH and Rank attacks which are launched at the same time on an IoT network.
- Authors in [16] performed two tasks. The First one is to design a Machine learning based IDS for 6LoWPAN attacks and secondly, they performed RPL-based network attacks in the COOJA simulator and get datasets. After preprocessing the raw datasets, they used various machine and deep learning models to detect RPL-based attacks in IoT networks. The authors used three classifiers i-e Random Forest (RF), SVM, SVM-RBF, and deep neural networks. Out of these classifiers, RFC has the highest accuracy and precision of 9.67% and 9.57% respectively. Although they get high accuracy, but they have used just six features (Rank, DIS-S, DIS-R, DIO-S, DIO-R and DAO-R). There are a few other important parameters like energy and power consumption which also critically contribute to the routing attacks as well. Similarly, although the author built an IDS for RPL attacks but actually it is specifically for the black hole attack.
- Snehal A. Bhosale and S.S Sonavane [15] used a hybrid approach for the Intrusion detection for wormhole attacks. They used the location information of node and its neighboring nodes by using Received signal strength (RSSI) values and the number of HOP-Counts to detect the malicious node in the IoT network. RSSI is a range-based localization method and HOP-Count is a range-free localization method. Whenever any node starts sending the request to its neighboring nodes, the distance between these neighboring nodes is calculated using the distance method.

If the RSSI value of the received packet exceeds the transmission range of existing nodes, an alert is generated. Similarly, as in a WHA, attacking nodes make a tunnel which may

may cause a drastic reduction in the HOP-Count. Although given hybrid approach gives improved TPR and FPR results but if there is a cluster of IoTs network this approach is no longer effective. So, we need to switch to machine and deep learning methods.

Table 2.1: Summary of Literature Review

Paper	Year	Network Attacks	Security Mechanism/Results	Simulator Used/Datasets	Future Work
R Mehta <i>et al.</i> [23]		Wormhole & Grayhole Attacks	Mechanism based on Trust computation in terms of performance metrics throughput and packet loss rate which is based on 1) Direct trust & 2) indirect Trust	Cooja	-
M Abdan <i>et al.</i> [24]	2018	Wormhole	ML models i-e KNN, SVM, DT, LDA and NB	MATLAB	
A Kumar <i>et al.</i> [25]	2022	Wormhole	Quantum walk and reinforcement learning are used for routing stage of ad hoc network AND for detection they used Round trip time and packet delivery	-	-
M Ezhilarasi <i>et al.</i> [26]	2023	Selective forwarding, black hole, wormhole, hello flood and identity replication attack	Feed-forward neural network and Fuzzy logic Detection rate : 97.8% and accuracy is 98.8%	Cooja	For future work can use optimization models to detect routing attacks
Tahboush <i>et al.</i> [27]	2022	Wormhole	Used 20 imp attributes to create a dataset and apply SVM and Genetic algorithms for detec	-	-
S Ali, P Nand, <i>et al.</i> [28]	2023	Wormhole	The collected data is pre-processed and the k-NN & Random Forest algorithms are applied to detect WH attack. For attack prevention they used packet lease and cryptographic techniques	NS-3.24.1 simulator	-

			Accuracy 99.196% and 98.66%		
F Zahra <i>et al.</i> [29]	2023	Rank and Wormhole	Dataset is generated by construction different network topologies and light gradient-boosting algorithm optimized for multi-class classification Average accuracy, precision, and recall of 99.7%, 99%, and 99.7%, respectively,	For the multiclass LloTN-RPL dataset	In the future, we can study and create more attack models to simulate and generate datasets for both types of RPL attacks. We can use different models to develop detection.
Stoian [30]	2020	Anomaly detection and attacks in IoT networks	RF, NB, MLP, SVM and AdaBoost in which The RF algorithm has obtained the best results with 99.5% accuracy	IoT-23	-
Rani and Kaushal, [31]	2020	Improve the security and accuracy of intrusion detection system	KNN, NB, Decision Tree, Logistic Regression, RF The proposed simulation has a 99.0% accuracy with less time and energy intrusion detection	NSL-KDD and KDDCUP99	-

Proposed Methodology

3.1 Introduction

In this section, we proposed a detailed research methodology for our work and will show a description of each portion. In this chapter, we propose a methodology that uses machine learning and deep learning techniques to detect WHA in 6LoWPAN RPL-based IoT networks. We start our work with the basic building block of an IoT network, the architecture of a network, how communication establishes within an IoT network, how a network is at risk of various types of attacks across IoT and finally will explain what actually a wormhole attack is and how it affects the network. We present our data collection and pre-processing techniques, as well as our feature selection and extraction methods. Our proposed methodology uses different machine learning models for self-generated datasets. We evaluate the performance of these models using several metrics such as accuracy, precision, recall, F1-score, and area under the ROC curve.

The proposed methodology has several advantages over existing approaches, including its ability to detect wormhole attacks with high accuracy, low false positives, and low false negatives. Furthermore, the proposed methodology is scalable and can be applied to large-scale IoT networks.

3.2 Flowchart of the Proposed Architecture

From the start to the Data preprocessing phase, it involves the process of dataset collection. The proposed methodology begins with the initialization of sensor nodes in the 6LoWPAN RPL-based IoT network. This initialization phase involves configuring and activating the sensor nodes to establish communication within the network.

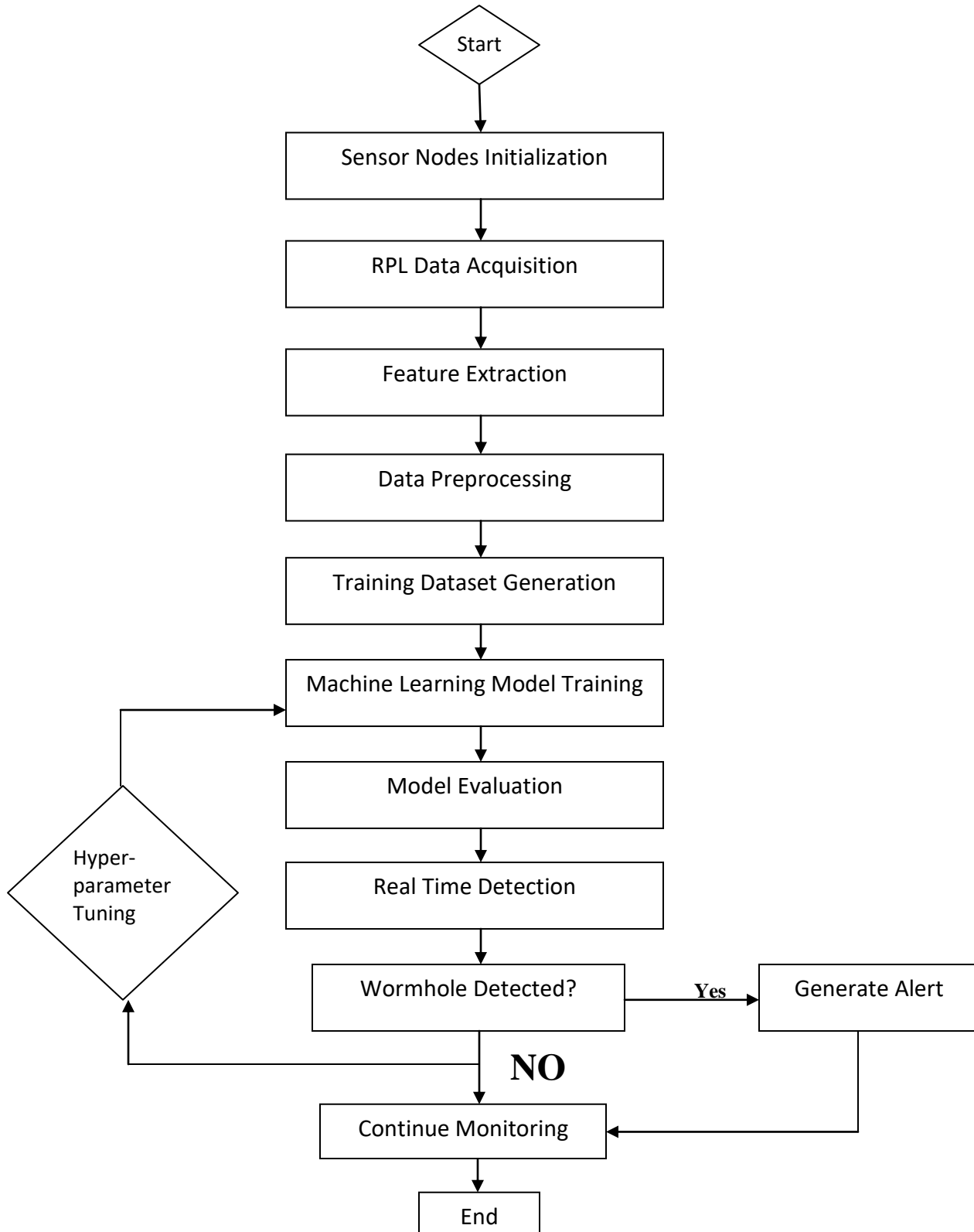


Fig 3.1 Flowchart of the proposed architecture

3.3 RPL Data Acquisition Step

Once the sensor nodes are initialized, the RPL (Routing Protocol for Low-Power and Lossy Networks) is employed to acquire data from the network. This step also involves in detail how the RPL protocol collects network-related information, such as routing tables, topology, and node connectivity.

This step also involves the importance of RPL protocol data in detecting potential wormhole attacks with the help of RPL control messages.

3.4 Feature Selection Process

Feature extraction is a critical step in the proposed methodology as it involves identifying and selecting relevant characteristics from the acquired data that can effectively capture the distinguishing patterns between normal network behavior and potential wormhole attacks. This step aims to transform the raw data into a more compact and representative feature space.

During feature extraction, various techniques and algorithms can be employed based on the nature of the data and the specific requirements of the wormhole attack detection in the 6LoWPAN RPL-based IoT network.

3.5 WH Attack Model Simulation

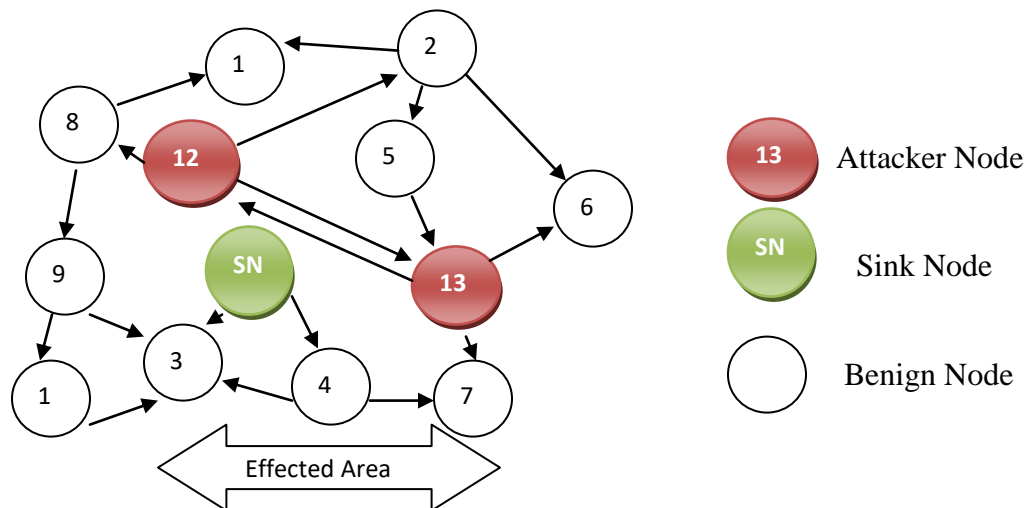


Fig 3.2: WH Attack within IoT Network

In the above scenario, the node 12 and 13 , form a tunnel between each other by probing control messages and similarly starts communication via these control messages i-e DIO , DIS and Ack. The algorithm 1 demonstrates the simulation of wormhole attack in the RPL-based IoT network in the Cooja simulator.

Algorithm 1: Wormhole Attack Scenario

1. Input: wormhole attack building block
 2. Output: attack on RPL network
 3. Begin
 4. True: malicious nodes form tunnel via probing using DIS, DIO and DAO
 5. If
 6. Rank of the nodes abruptly changes,
 7. Receive route requests,
 8. Misdirection of traffic,
 9. Disruption of communication,
 10. Exhaustion in power consumption,
 11. Neighbor nodes overheads fake credentials,
 12. Join the node as child nodes,
 13. Drop the child node packets, then
 14. Nodes = malicious
 15. Else
 16. False: node = benign
 17. Until wormhole attack launched
 18. Network = attacked
 19. End
-

3.6 System Architecture

Our proposed mechanism relies on the following stages: network simulation in the Cooja simulator to collect datasets for benign and malicious nodes, data preprocessing, features engineering, data labeling, deployment of appropriate machine learning algorithm, and statistical results. The complete analysis of each step is given in the next chapter.

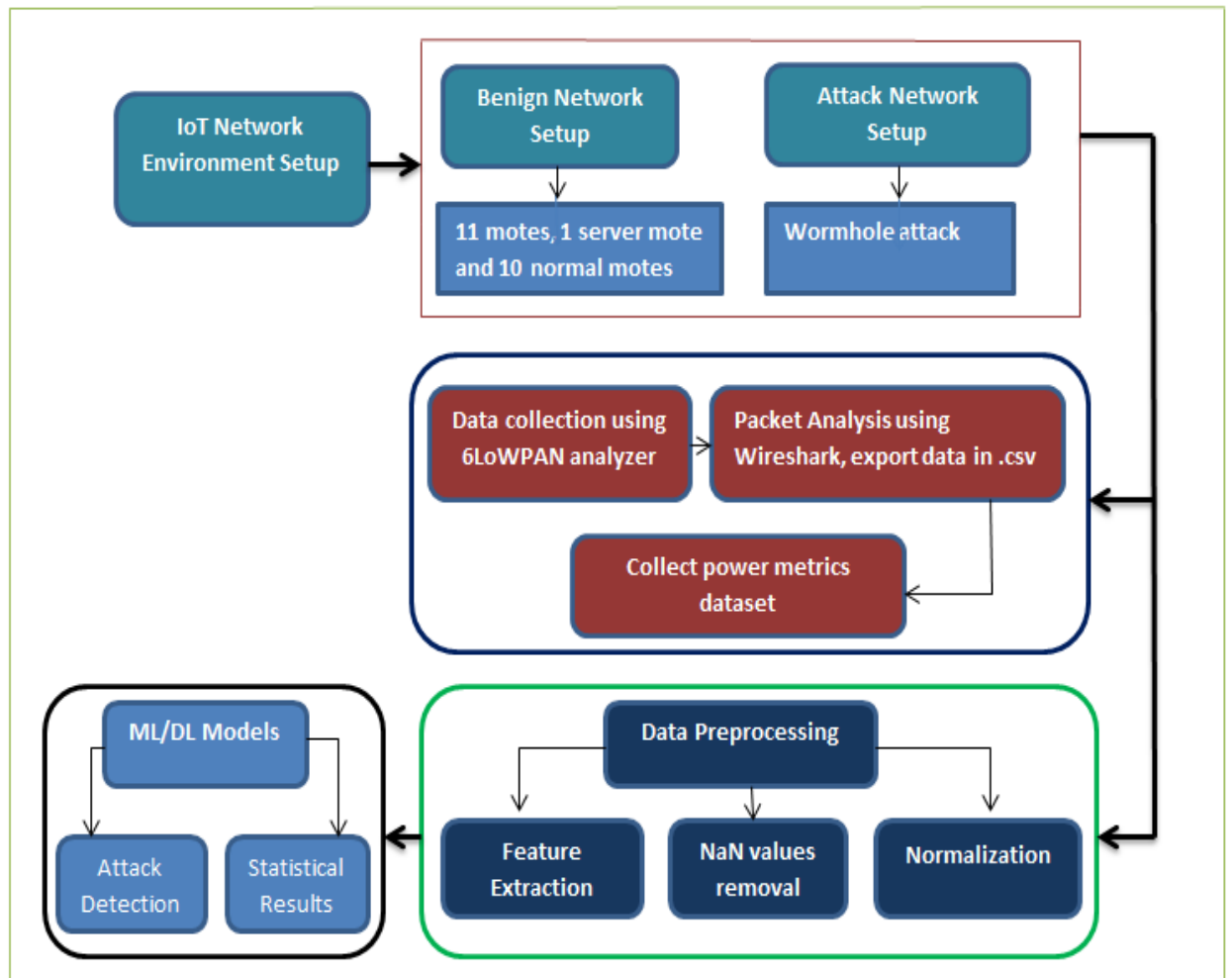


Fig 3.3: Overview of the proposed architecture

CHAPTER 4

Experimental Analysis and Findings

4.1 Experimental setup

Many researchers used online available datasets for Intrusion detection in IoT attacks. In our paper, we built an environment where we deployed sensor nodes and performed attacks. We configure the nodes and their parameters, such as transmission power, channel frequency, and data rate, to represent the real-world network environment as closely as possible for this task we used COOJA, a network simulator built in Contiki OS. Contiki is an open-source OS primarily designed for systems with limited memory, specifically targeting low-power wireless devices. We installed COOJA in VM virtual box. We set up an environment with 10 benign nodes, 01 root node and 02 malicious nodes with a radio medium of Unit Disk Graph Medium (UGDM). The environment consists of 10 yellow UDP-client motes and a green UDP-server mote. We kept a mote startup delay of 1000ms and a random seed was 123,456. COJA interacts with simulated motes via mote interfaces. We kept mote interfaces settings as default and we don't change them.

Table 4.1: Default Mote interface settings for intercommunication

Mote interfaces for intercommunication	
Rime address	✓
IP address	✓
Mote2Mote relations	✓
Mote attributes	✓
Cycle clock	✓
ID	✓
Button	✓
M25P80 Flash	✓
Coffee File system	✓
IEEE 802.15.4 radio	✓
Serial Port	✓

Table 4.1: Cont

	✓
Debugging output	✓
Temperature	✓

The types of mote were sky mote. The detailed environment parameters are highlighted in the following table.

Table 4.2: Parameters setup for network under wormhole attack

Parameters	Values
Nodes Operating System	Contiki NG
Routing Protocol	RPL
Radio Medium Model	Unit Disk Graph Medium (UDGM) Distance Loss
Communication Range	50 m, Interference Range = 100m
Number of Benign Nodes	10, Root nodes =1
Number of Attacker Nodes	02
Network Layer	Contiki RPL
MAC Layer / Sub-layer of Datalink	Contiki MAC
Transport Layer	UDP
Attack Start Time	15 Minutes after start of simulation
Simulation Duration	01 hour

4.2 Dataset Generation

In this section, we present a detailed explanation of how we created a collection of harmless datasets. We accomplished this by simulating a benign IoT network scenario using the Cooja simulator, as depicted in Figure 2. We captured IoT-specific information, such as CPU consumption, from the Contiki plugin called "power trace." Additionally, we collected network traffic data using the COOJA tool's "Radio messages" feature. These data collections allowed us to generate two datasets: one for power consumption and another for network traffic, both representing the simulated benign IoT network

scenario. For the generation of a benign dataset, we deployed 10 nodes at random positions within a communication range of 50m and an interference range of 50m. Now start the simulation and wait at least 15 minutes so that our network gets stable. The stability of the network means that when all the nodes start communicating with each other by the exchange of network packets. Once our network reaches a stable condition, now we properly start monitoring the Radio messages by enabling the radiolog option. For the bulk amount of data, we run the Simulation for one hour and save the radiolog file in PCAP format.

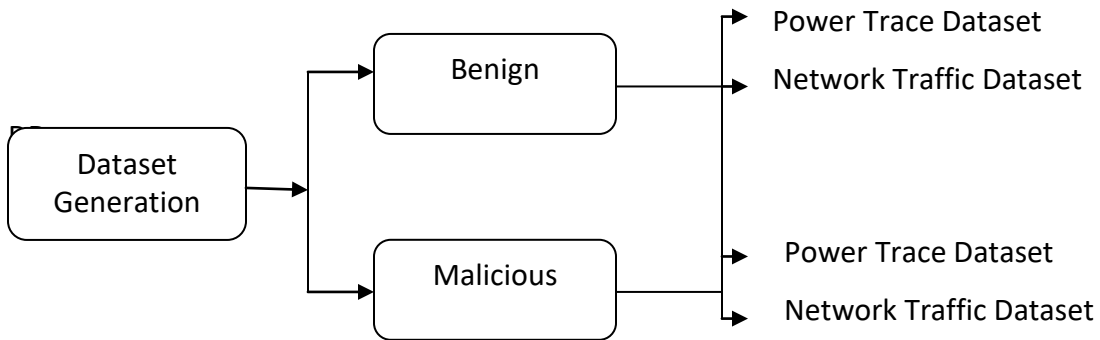


Fig 4.3: Dataset generation types

4.2.1 Benign Dataset Generation

4.2.1.1 Benign Power-trace Dataset Generation

During the simulation, the power trace will record the energy consumption across the nodes and CPU in real time. To obtain the power trace dataset, we configured the benign nodes through programming in java-based source code. The powertrace.h library was used in the source code and simulation time was set to 1 hour or 3,600,000 seconds. The power-trace plugin in the Cooja simulator measures energy consumption in ticks. A tick is a unit of time that is hardware-dependent and corresponds to the CPU clock cycle. The number of ticks consumed by a particular event or operation can be used to estimate the energy consumed by the system. In power-trace mode following power metrics states

have been collected: (i) CPU (ii) LPM (iii) Transmitt (iv) Listen (v) Idle_transmit and (vi) Idle_listen. We used `t_CPU_usage`, `t_LPM_usage`, `t_trx_usage`, and `t_listen_usage` in our dataset to train the models.

In the COOJA simulator, `t_CPU_usage` refers to the amount of CPU time used by a specific node in the simulation. This metric is typically used to measure the computational overhead associated with running a simulation on a particular node, as well as to identify potential performance issues or resource constraints. `t_CPU_usage` is measured in microseconds and represents the total CPU time used by a node since the start of the simulation.

In the COOJA simulator, `t_LPM_usage` refers to the amount of time a specific node in the simulation spends in Low-Power Listening (LPL) mode. LPL is a power-saving mechanism used in wireless sensor networks to reduce energy consumption by allowing nodes to switch their radio transceivers on and off periodically. `t_LPM_usage` is measured in microseconds and represents the total amount of time a node has spent in LPL mode since the start of the simulation.



Figure 4.4: Benign network setup in Contiki Cooja

Similarly, `t_trx_usage` refers to the amount of time a specific node in the simulation spends transmitting or receiving data using its radio transceiver. By monitoring `t_trx_usage` for each node in the simulation, we can identify which nodes are consuming the most radio resources and potentially causing interference or congestion on the network.

And finally, `t_listen_usage` refers to the amount of time a specific node in the simulation spends in passive listening mode, also known as Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA). By monitoring `t_listen_usage` for each node in the simulation, we can identify which nodes are spending the most time listening for incoming data, which can be useful for optimizing the performance of the network

4.2.1.2 Benign Network Traffic Dataset Generation

For the dataset of network traffic, we initially run the simulation for 15 minutes. Once all the nodes started communication to each other it means that our network is now stable. Now start to monitor radio message communication by enabling the 6LoWPAN Analyzer with PCAP.

Similarly, start the simulation and get started to monitor the radio messages by enabling the 6LoWPAN Analyzer with PCAP. Captured the raw traffic data in .pcap file format, to analyze the file use Wireshark and export data in a .csv file.

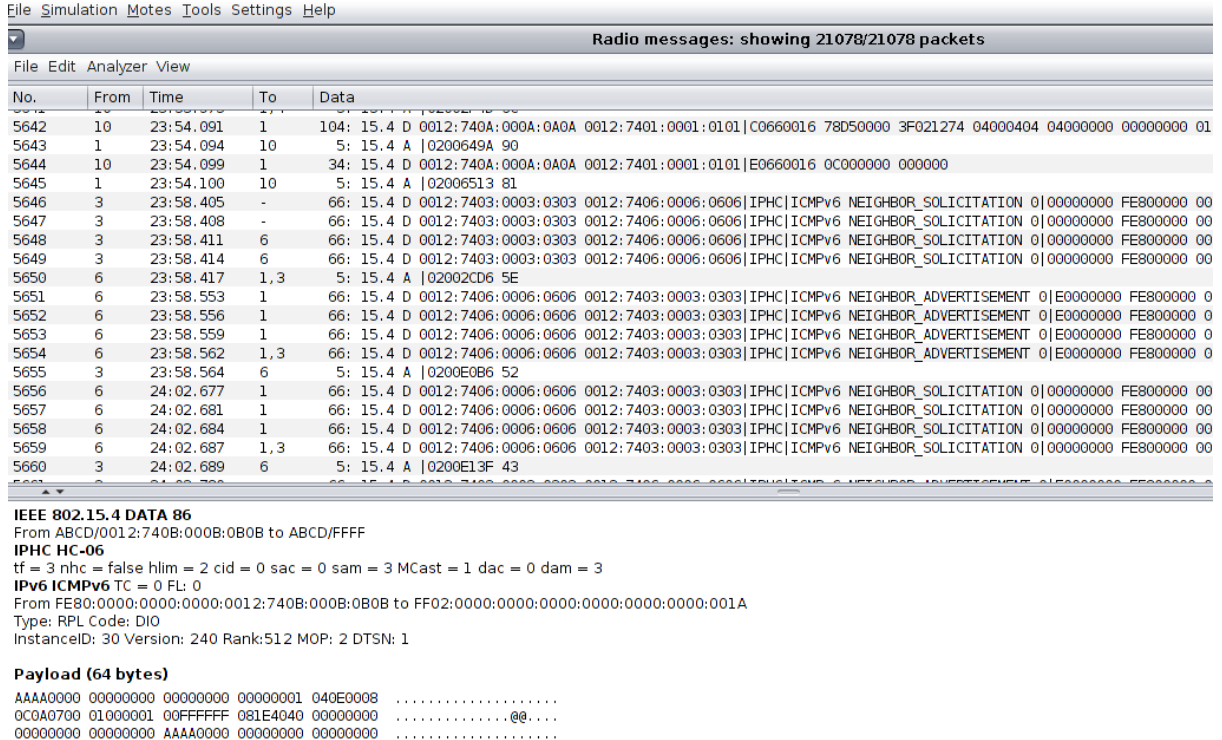


Fig 4.3: Traffic Showing using 6LoWPan Analyzer

4.2.2 Malicious Dataset Generation

4.2.2.1 Malicious Power-trace Dataset Generation

Now we deploy two malicious nodes also we can say attacking nodes within the network at the random position. In this case, node 12 and node 13 are malicious nodes in pink. The power trace plugin was similar for collecting power-trace-related features. In this case simulation is done for 60 minutes duration after once the network achieved stability. In power-trace mode following power metrics states have been collected: (i) CPU (ii) LPM (iii) Transmit (iv) Listen (v) Idle_transmit and (vi) Idle_listen. We used t_{CPU_usage} , t_{LPM_usage} , t_{trx_usage} , and t_{listen_usage} in our dataset to train the models.

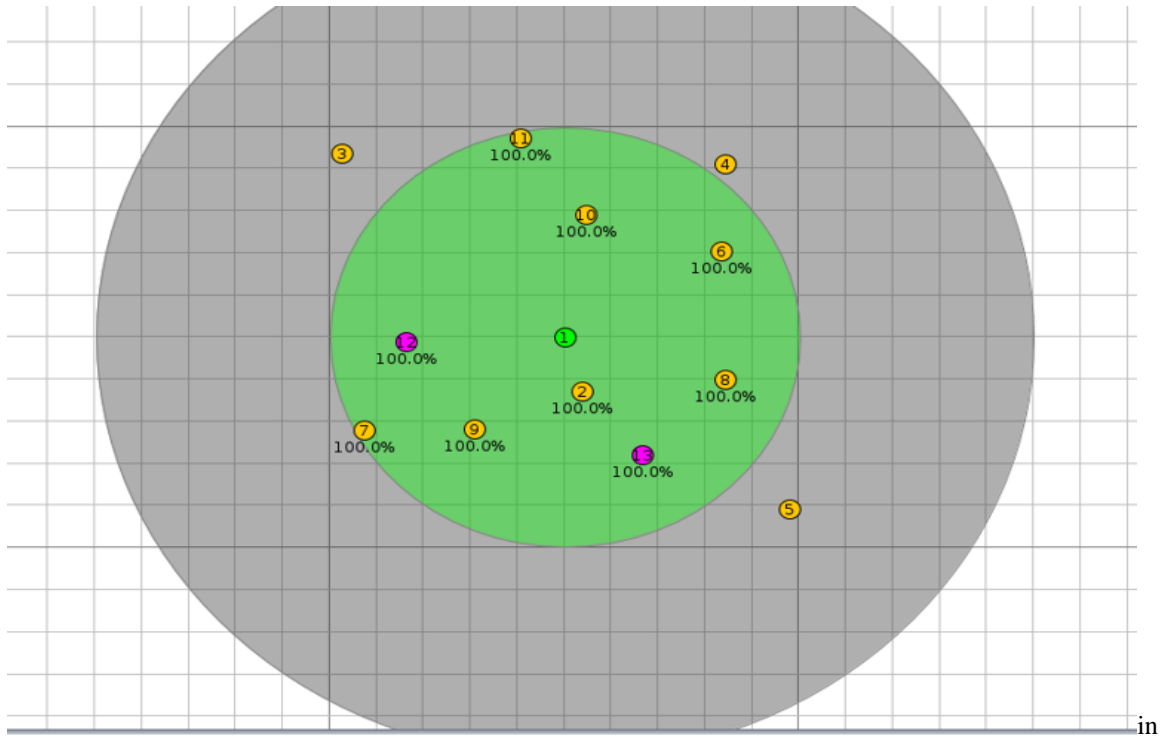


Fig 4.6: Malicious network in Contiki Cooja

4.2.2.2 Malicious Network Traffic Dataset Generation

A Similar strategy has been followed to generate the malicious network traffic dataset as initially used for the benign traffic. The raw dataset contains multiple features like hop count, RPL-control messages, node information, version number, rank value, and instance ID, etc. Wormhole attack has highly impact on mentioned features. The generated malicious power trace file contains 10,794 records.

4.3 Network Map after Deploying Attacking Nodes

When deploying attacking nodes in the Cooja simulator, the network map shows the specific configuration and placement of the nodes.

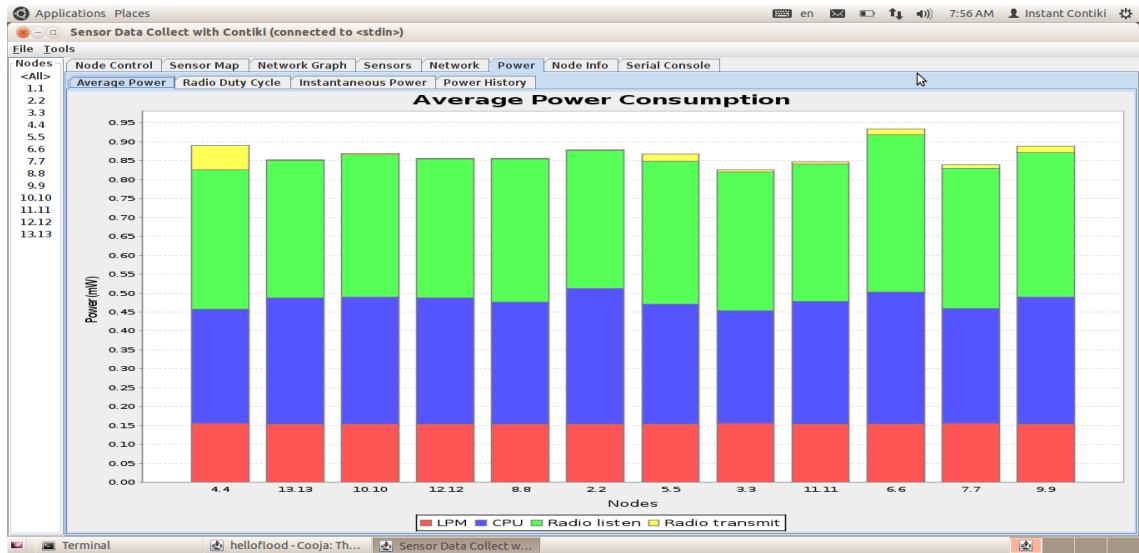


Fig 4.8: Average Power metrics across node

After 1 hour of simulation, the average power across each node, where blue portion represents CPU power consumption, red represents power consumption in Low Power mode, green represents power consumption in radio listen mode, and yellow for power in radio transmit mode.

4.5 Data Preprocessing

4.5.1 Dataset Cleaning

First of all we have cleaned our raw dataset. In our dataset, there are some duplicate values, so removed all duplicate values. The `drop_duplicates()` function is used in python to remove duplicate rows from the DataFrame. The duplicate records in the dataset create redundancy and bias in the analysis, so it is essential to remove duplication in data.

Similarly, some missing values have been handles by imputation technique based on the statistical methods (e.g., mean and median).

Similarly, outliers in the datasets have been removed by using IQR method.

4.5.2 Feature Selection

Different features have been selected by different researchers, but the wormhole attack severely affects the RPL control messages in IoT networks. So we chose the features related to the RPL-control messages. The attacker can create a shortcut between two distant parts of the network by tunneling messages through a covert channel. When applied to RPL control messages, this can result in the attacker intercepting and forwarding DODAG (Destination-Oriented Directed Acyclic Graph) information, causing the network to reroute traffic through a compromised path.

The feature selection using the correlation coefficient method is a way to identify which features in a dataset are most related to a target variable. The feature selection process is done using Python programming in Google Colab. We import common separated file in Google Drive. Then load the dataset in Google Colab.

1. For each feature in the dataset, calculate its correlation with the target variable. Correlation measures how closely two variables are related to each other, and ranges from -1 to 1. A correlation coefficient of 1 means that the two variables have a perfect positive relationship, 0 means that there is no relationship, and -1 means that there is a perfect negative relationship.
2. Take the absolute value of each correlation coefficient, so that we only consider the magnitude of the correlation, not the direction.
3. Rank the features, based on their correlation coefficients, from highest to lowest.
4. Select the top k features with the highest correlation coefficients as the most important features. These features are likely to be most useful for predicting the target variable.

In RPL protocol, DIO-S (DODAG Information Object-Solicited) control message is sent in response to a DAO (Destination Advertisement Object) message received from a node that has no information about the parent in the direction of the DODAG root. In the case of DIO-S messages, a wormhole attacker can capture DIO-S messages from a legitimate parent node and replay them to a victim child node, which can cause the victim node to

choose the wormhole attacker as its parent. This can create a shortcut in the network that bypasses other nodes and can lead to a suboptimal routing path.

DIO-R messages are used to acknowledge the reception of DIO-S messages and to provide information about the topology of the network. The reception of the large number of DIO-R messages by a node indicates that it is receiving messages from the correct upstream neighbor and can communicate with it effectively. However, in a wormhole attack, an attacker may replay DIO-R messages from a different part of the network, which can cause a node to falsely believe that it has a valid upstream neighbor. This can result in the node joining the wrong part of the network or using a suboptimal route, which can increase the latency and reduce the overall efficiency of the network. Therefore, the effect of a wormhole attack on DIO-R messages is generally negative, as it can lead to incorrect network topology information and suboptimal routing decisions.

Similarly, the wormhole attack may lead to a decrease in DIS-R messages if the attack is successful in isolating parts of the network and preventing nodes from communicating with each other. DAO (Destination Advertisement Object) control messages are responsible for conveying the routing information from the parent node to the child nodes in the RPL-based IoT network. The DAO control messages carry information about the parent node, which is used by the child nodes to update their routing tables. In a WHA, the attacker creates a shortcut between two nodes in the network by relaying packets through a high-speed link. When the DAO messages are sent in the network, the attacker can intercept the messages and forward those through the wormhole link, making the messages appear as if they were sent by a node close to the destination.

The detailed information about features is given in the table below:

Table 4.3: Brief description of the features

1	Mote	IOT nodes
2	Rank	Position or order of a node within a network topology.
3	Version	
4	Instance ID	Unique identifier assigned to each simulated node in the network.
5	DIO-S	DIO-R (DODAG Information Object with RPL Source Option) are used for source routing and are sent by intermediate nodes to update the sender about the status of the path to the destination
6	DIO-R	DIO-R (DODAG Information Object with RPL Target Option) provide information about a specific node in the network in response to a DIS-R message
7	DIS-S	DIS (DODAG Information Solicitation with source option) is a RPL control message used to discover the path between two nodes in the network.
8	DIS-R	DIS-R (DODAG Information Solicitation with RPL Target Option) are used to request information about a specific node in the network
9	DAO-S	DAO-R (Destination Advertisement Object with RPL Source Option) are used for source routing and are sent by a node to update its parent node about the status of the path to the destination
10	DAO-R	DAO-R (Destination Advertisement Object with RPL Target Option) contains information about the destination node's address, rank etc.
11	ACK	Acknowledge the receipt of a packet
12	t_CPU_usage	CPU usage of a node during traffic generation,/the total CPU time used by a node since the start of the simulation.
12	t_LPM_usage	measured in microseconds and represents the total amount of time a node has spent in LPM mode since the start of the simulation
13	t_trx_usage	The total amount of time a node has spent transmitting or receiving data since the start of the simulation.
14	t_listen_usage	The total amount of time a node has spent in passive listening mode since the start of the simulation

4.5.3 Data Chunking

After collecting the benign and malicious data we combine all dataset to make chunks of equal time intervals of both malicious and benign traffic separately. For example, for benign traffic of one-hour simulation, we divide or data into 60 chunks. Each chunk corresponds to 1 minute in duration.

Table 4.4: Data chunking with each row represents 60 seconds duration

Mote	Rank	Version	Instance ID	DIO-S	DIO-R	DIS-S	DIS-R	DAO-S	DAO-R	ACK	t_CPU_usage	t_LPM_usage	t_trx_usage	t_listen_usage	Labeling
1	256	240	30	14	87	5	10	0	1	13	728396	6548	844898	0	1
1	256	240	30	32	12	0	0	0	6	16	11725	799683	6548	910434	0
2	394	265	29	29	3	1	0	1	1	7	57008	0	65066	0	1
2	409	260	30	29	1	0	0	2	0	12	6463	59153	0	65066	0
3	390	203	32	0	0	0	1	0	0	0	113813	3274	127300	0	0
3	557	240	30	29	0	0	0	0	0	3	15173	115959	3274	127300	1
11	593	240	30	29	1	0	0	3	-	-	-	-	-	-	-
3	557	180	30	0	0	4	1	0	-	-	-	-	-	-	-
1	256	195	30	0	0	19	31	0	-	-	-	-	-	-	-
4	514	276	30	0	3	0	0	0	-	-	-	-	-	-	-
5	522	195	30	0	6	0	0	0	-	-	-	-	-	-	-
6	384	230	30	0	8	9	3	0	-	-	-	-	-	-	-
2	409	240	30	0	0	5	2	0	-	-	-	-	-	-	-
7	518	173	30	0	1	0	1	0	17	10	56565	34555	345345	766645	1

minute duration of the interval. Now for each interval, we calculate the data with for network traffic features. We chose node-wise the number of RPL control messages within the raw network traffic. For example in the case of DIO messages, DIO-S means the number of DIO messages sent from the specific nodes. Similarly, DIO-R means the number of DIO messages received from the specific node. The number of DIO-S and DIO-R messages was calculated corresponding to each node within the network. Similarly, the number of the RPL-control messages has been calculated in a similar way. Similarly, the rank, instance ID, and version number are calculated within the specific interval. There are a total 1100 number of lines in the prepared dataset containing 14 features containing network traffic and power metrics features. The last column is for data labeling i-e the number 1 used is for malicious traffic and 0 is used for normal traffic.

4.5.4 Data Scaling and Normalization

In this section, we will discuss how our proposed algorithms give the best detection rate by applying different machine learning algorithms. In the literature, different researchers suggested different algorithms for wormhole attack detection. We have used the same algorithms and same features that were used by the researchers but the dataset was generated by our system. So we will compare our results with the previous results. After that, we will use some other features and different machine learning algorithms which were not used by the researchers in the literature. Our scheme is then compared with the existing scheme to show the effectiveness of the proposed system.

The generated dataset is not in the normalized form. So there will be the need to normalize the dataset before applying any ML/DL algorithm. For each feature, We calculated statistical measures such as average, variability, minimum, maximum, and quartiles. The statistical description of the dataset is given in the following diagram.

Table 4.5. Statistical description of the features

	Note	Rank	Version	Instance ID	DIO-S	DIO-R	DIS-S	DIS-R	DAO-S	DAO-R	ACK	t_CPU_usage	t_LPM_usag
count	1005.000000	1005.000000	1005.0	1005.0	1005.000000	1005.000000	1005.000000	1005.000000	1005.000000	1005.000000	1005.000000	1006.000000	1006.000000
mean	6.646766	446.183085	240.0	30.0	5.145274	3.774129	2.537313	1.742289	0.402985	0.307463	6.975124	12884.439026	634.97854
std	3.578600	101.037636	0.0	0.0	12.297393	14.653968	5.138117	4.423796	1.154323	1.521425	7.193566	12977.650237	1021.58208
min	1.000000	256.000000	240.0	30.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	4.000000	384.000000	240.0	30.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	3.000000	327.604750	16.36800
50%	7.000000	409.000000	240.0	30.0	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000	4.000000	9234.314000	36.44800
75%	10.000000	512.000000	240.0	30.0	0.000000	1.000000	4.000000	1.000000	0.000000	0.000000	8.000000	24169.963250	1013.45300
max	13.000000	832.000000	240.0	30.0	177.000000	136.000000	100.000000	42.000000	9.000000	18.000000	55.000000	39109.178000	3461.43900

As it is clear from the statistical data analysis, the range of data values of raw data is of having different scales. In such cases, the classification models will not perform effectively without data normalization.

4.5.4.1 Min-Max Normalization

Data scaling is the process in which we convert the dataset so that it fits within a specific range. Data normalization and scaling are essential preprocessing steps in machine learning that help improve the accuracy and effectiveness of the model. These techniques transform the features of the dataset to a common scale, which helps in reducing the effect of the scale of the features on the performance of the algorithms.

The data is normalized with the following relation.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

But we set the range i-e minimum and maximum between random set of values.

$$x' = a + \frac{(x - \min(x))(b - a)}{\max(x) - \min(x)},$$

```
1 # Fit the scaler to the data and transform the data
2 X_scaled = scaler.fit_transform(X)
3 print(X_scaled)

[[7.69230769e-02 3.07692308e-01 1.00000000e+00 ... 1.89169880e-03
 1.94919017e-02 0.00000000e+00]
 [7.69230769e-02 3.07692308e-01 1.00000000e+00 ... 2.31026171e-01
 1.51063173e-04 2.36092667e-01]
 [1.53846154e-01 4.73557692e-01 1.00000000e+00 ... 0.00000000e+00
 1.50108070e-03 0.00000000e+00]
 ...
 [6.92307692e-01 5.09615385e-01 1.00000000e+00 ... 6.39287591e-01
 2.26571689e-04 6.43111442e-01]
 [7.69230769e-01 4.61538462e-01 1.00000000e+00 ... 6.56605822e-01
 2.26571689e-04 6.60105641e-01]
 [8.46153846e-01 6.33413462e-01 1.00000000e+00 ... 6.73981544e-01
 2.26571689e-04 6.77100359e-01]]
```

Fig 4.9. Normalized data after min-max normalization

4.5.5 NaN values removal

In a dataset, NaN values represent missing or undefined data points. It's a way of

indicating that a value is unknown, unmeasured, or unrecorded for a particular feature in a specific observation or sample. NaN values may occur due to multiple reasons such as data entry errors, faulty sensors, or simply because the data was not collected or recorded for a particular feature in a particular sample. Handling NaN values is important in data analysis as they can lead to biased or inaccurate results if not dealt with appropriately. We used python code to remove NaN values in our dataset.

Results and Discussion

5.1 Proposed Models

We have used various machine learning and deep learning algorithms for classification. In this section, I will explain the reasoning behind the selection and provide a detailed description of each proposed machine learning model. Some commonly used machine learning models for intrusion detection in IoT networks include decision trees, neural networks, SVM, and RF. We have also used some different algorithms according to our dataset.

5.1.1 Ridge Classifier

The Ridge classifier is a type of linear classifier for binary classification tasks. It is a modification of the logistic regression algorithm that includes a regularization parameter called the Ridge parameter.

The ridge classifier works by finding the hyper-plane that maximizes the margin between the two classes while also minimizing the sum of squared weights. The squared weights penalty term is known as the L2 regularization, which helps in mitigating overfitting by reducing the model coefficients towards zero. This regularization term is particularly useful when dealing with high-dimensional data with a large number of features.

It is particularly effective in cases where the data is noisy or where there exist a large number of correlated features.

Let's assume we have a dataset with n samples and d features. We represent the features of each sample as a d -dimensional vector x_i , and the corresponding target class as y_i . The Ridge Classifier aims to find the optimal weight vector b that minimizes the following objective function.

$$\text{minimize } \sum_{i=1}^n (y_i - \hat{y})^2 + \alpha \sum_{j=0}^m b_j^2$$

Alpha is regularization parameter. Larger values of alpha give greater shrinkage, making the coefficients more robust to collinearity. The first term $\|y_i - \hat{y}\|^2$ measures the squared error between the predicted classes (Xw) and the actual classes (y).

5.1.2 Artificial Neural Network

(ANN) can be used for wormhole attack detection in IoT networks. ANNs are capable of capturing non-linear relationships between the input data and the output, which can be useful for detecting subtle patterns that are not easily discernible through traditional methods. This aspect holds significant importance, especially when dealing with wormhole attacks, which can be difficult to detect using conventional methods.

ANNs are generally robust to noise and can handle missing or incomplete data.

The architecture of ANN consists of three types i-e Single-Layer Feed Forward, Multilayer Feed Forward, Recurrent networks, etc. based on the number of hidden layers and feedback mechanisms. ANN typically contains multiple layers of perceptrons, with each layer passing its output to the next layer until a final output is produced. The process of passing the output of one layer to the input of the next layer is known as forward propagation. The input is received by the perceptrons, apply the set of weights and bias then output is calculated. These all passed through the activation function terms of the perceptron are adjusted during the training process to minimize the difference between the predicted output and the actual output. In the process of backpropagation, there involves the calculation of the gradient of the error function with respect to the weights and bias terms and updating them accordingly. The architecture of basic ANN is shown in given figure.

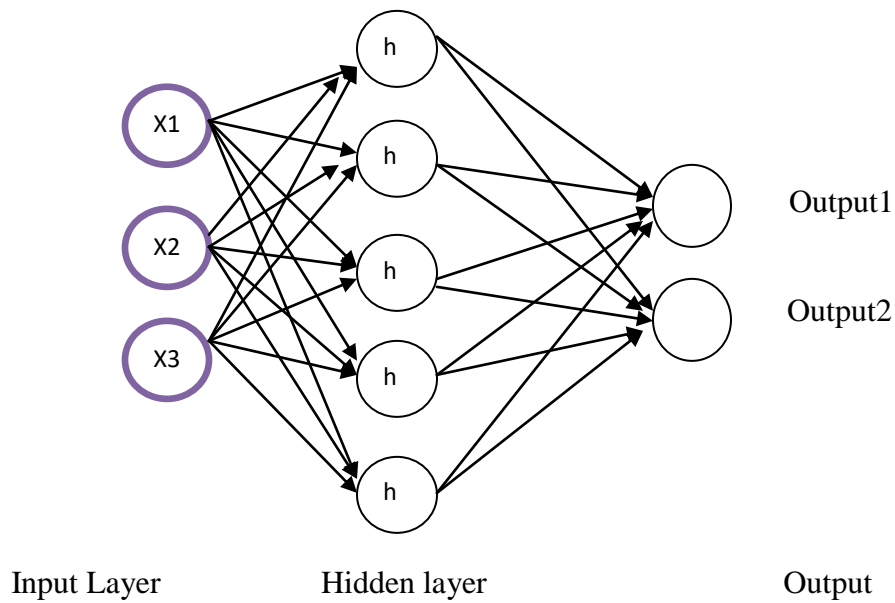


Fig 5.1: ANN architecture

An artificial neural network basically takes up inputs and calculates the weighted sum of these inputs. In addition, a bias term is added to this. This weighted sum with bias is passed to an activation function like sigmoid, RElu, tanh, etc. And the output from one neuron act as input to the next layer in neural networks. A neural network when having more than one hidden layer is called a Deep Neural Network.

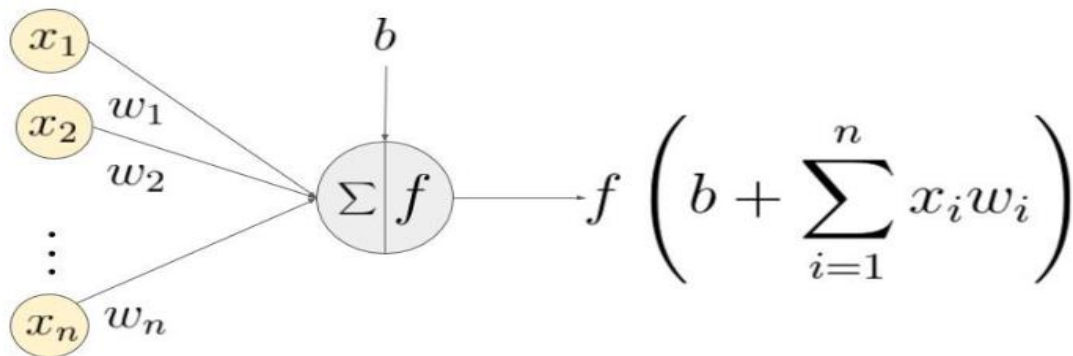


Fig 5.2: Weight, bias, and activation function in ANN

5.1.3 Deep Belief Network

A Deep Belief Network (DBN) is a way to build a complex network by combining smaller networks that learn without being told what the right answer is. We call these smaller networks Restricted Boltzmann Machines (RBMs) or Autoencoders [17]. The idea is to train each small network to get better at recognizing patterns by using the previous network's output as input. By doing this, we can teach the DBN to recognize more complex patterns. The training method used is called contrastive divergence, which helps each small network learn more efficiently. A DBN is a group of RBMs stacked on top of each other to form a bigger network.

In DBN, learns a set of features from the data, and the learned features from one layer are used as input to the next layer [18]. After all the RBM layers have been trained, a process known as fine-tuning is used to further improve the network's performance. Fine-tuning involves training the entire network using supervised learning to optimize the weights and biases of the network for the specific task at hand, such as classification. This helps to improve the accuracy of the network in making predictions on new data [19].

5.1.4 Radial Basis Function Network

The Radial Basis Function (RBF) neural network helps estimate a function using a limited number of features. The program has three layers: the input layer, the hidden layer, and the output layer. The input layer assigns a neuron for each feature and passes them to the hidden layer without any changes. The hidden layer then creates a connection between the input space and a larger space using RBF transfer functions. In the end, the output layer produces a linear output by calculating a weighted sum. For classification tasks, like in our work, the activation function for the output layer is then switched to the sigmoid activation function. [20]

During the training process, the RBF network assigns a neuron to each input feature and directly passes the input layer features to the hidden layer. The hidden layer then creates a non-linear relationship between the input space and a typically higher-dimensional space using RBF transfer functions. In the end, the output layer computes a weighted sum to produce a linear output. For classification tasks like wormhole attack detection,

the activation function of the output layer is changed to the sigmoid activation function31 weighted sum with a linear output. In the case of classification such as wormhole attack detection, the activation function of the output layer becomes the sigmoid activation function.

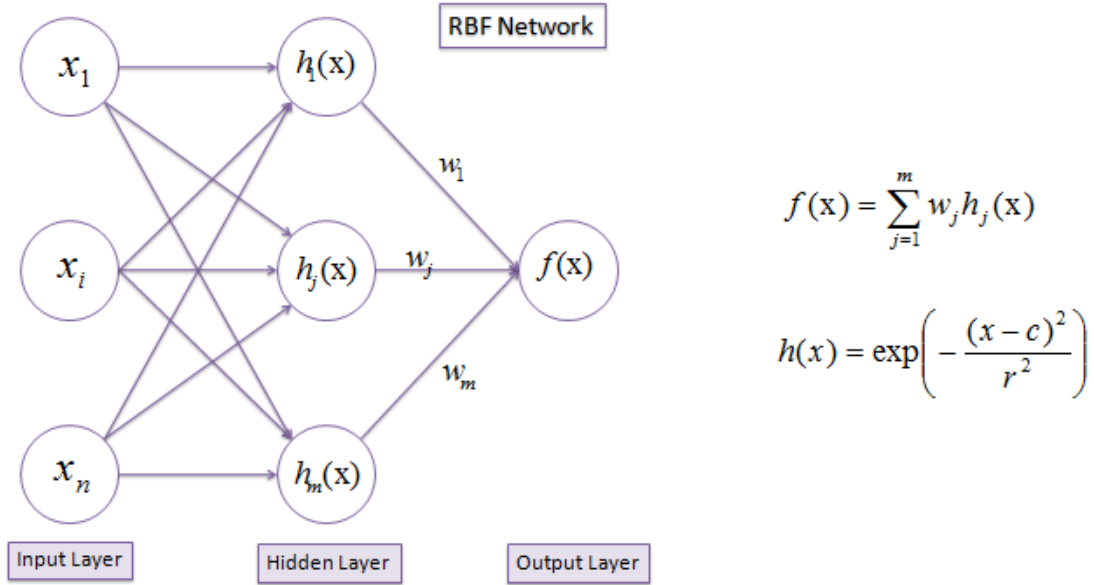


Fig 5.3: RBF Network architecture

$h(x)$ is the Gaussian activation function with the parameters r (the radius or standard deviation) and c (the center or average taken from the input space) [22].

$$r_j = \sqrt{\frac{\sum_{i=1}^k (c_j - c_i)^2}{k}}$$

Table 5.1: Summary of the main models used in wormhole attack detection

Sr. No	Model	Pros				Cons
1	ANN	Scalability	Reduced- False Positives	Adaptability	Low-False Negative rates	<ul style="list-style-type: none"> - Requires a large amount of labeled training data for optimal performance. - May be prone to overfitting if not properly regularized
2	RBFN	Localized Learning	Non-linear Dataset	Work well on Noisy Dataset	Few Hyperparameters	<ul style="list-style-type: none"> - Prone to overfitting if the number of centers is too large
3	Ridge Classifier	Reduced Over fitting	Regularization Control	Multi co linearity	High-dimensionality	<ul style="list-style-type: none"> - Limited capacity to capture complex nonlinear patterns - May struggle with imbalanced datasets without proper class weighting.
4	DBN	Robustness to Noise and Variability	Deep Architectural Depth	Unsupervised Pre-training		<ul style="list-style-type: none"> - Sensitive to hyperparameter selection and network architecture design. - Computationally expensive training

5.2 Simulation Setup

The work is carried out using DELL (inspiron i3-3110M) laptop, with Windows 10 Pro 64-bit operating system installed, Intel® Core™ CPU @ 2.4GHz 6.00GB RAM.

The datasets have been generated in the COOJA simulator which is installed in the Oracle virtual box. The proposed algorithms have been simulated using Google Colab and Jupyter Notebook.

5.3 Hit and Trail Methods Using Different Classifiers

We have tested multiple classifiers to check the different parameters. We have used classifiers i-e Logistics regression, Gaussian naïve Bayes, Random forest, SVM and ada-boost. The overall performance metrics are given in the table.

Table 5.2: Hit and Trail analysis using multiple classifiers

Sr. No	Algorithm	Confusion Matrix Values	Precision	F- Score	Accuracy	Simulator	Remarks
1.	Logistic Regression	FP=42 FN= 17 TP=117 TN=26	73%	80%	70%	Cooja	-
2.	Gaussian Naïve Bayes	FP=67 FN= 2 TP= 132 TN=1	67%	79%	66%	-	-
3.	Random Forest	FP=25, FN=12 TN=43,TP=122	82%	86%	82%	-	-
4.	SVM	FP=53,FN=2 TP=132,TN=14	72%	82%	72%	-	-
5.	Ada-Boost	FP=6,FN=18 TP=93,TN=83	93%	89%	88%	-	-

5.4 Implementation of Ridge Classifier

The above machine learning classifiers didn't provide satisfying results; therefore we have some complex models. For the proposed model to detect wormhole attacks in the RPL-based IoT, we have performed the analysis in Python language. The model is trained on 80% of the training dataset and remaining 20% of the testing dataset. The simulation is performed on the Google Colab, and Jupyter Notebook. Furthermore, we have performed data preprocessing, feature engineering, and hyper parameters tuning to improve the accuracy of the models. We have tested different machine learning and deep learning models on the hit and trials method to check the performance of each model. Additionally, before testing the new models, we also performed analysis on the same models which were used in the literature to check how much the models perform well on our dataset.

For the Ridge classifier, the model achieved an accuracy of 0.97 for the training dataset and an accuracy of 0.96 on the testing dataset for the detection of wormhole attacks.

The model achieved a precision of 0.94 and the recall value is 0.99. Similarly, the F-1 score of the proposed model tends to be 0.9710.

To plot the confusion matrix, we have calculated the FN, FP, TN, and TP values. We have obtained, False Negative value = 0, False Positive = 8, True Positive = 134 and True Negative = 60

The confusion matrix of the Ridge classifier is shown in Fig 5.3

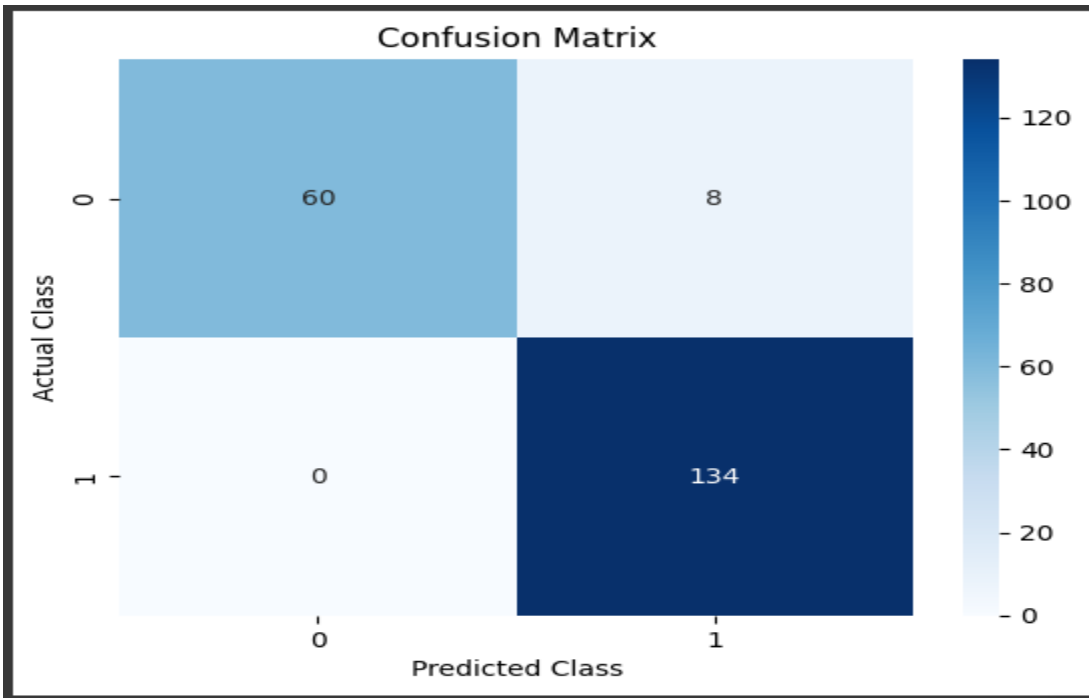


Fig 5.3: Confusion matrix for Ridge Classifier

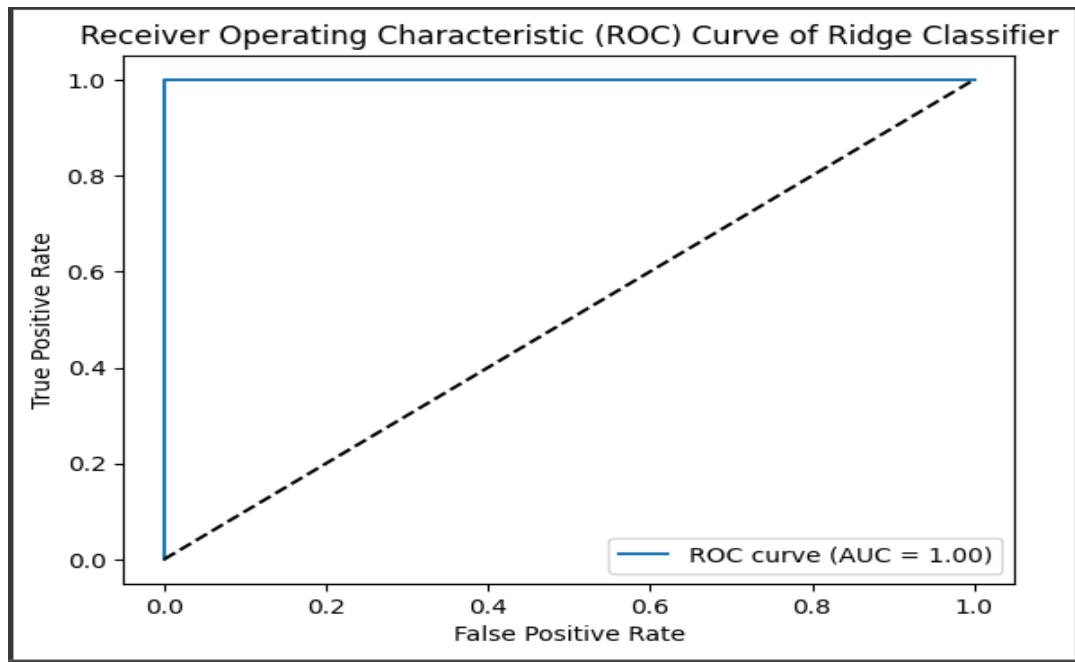


Fig 5.4: ROC curve of Ridge Classifier

5.5 Implementation of ANN

Similarly, for the For ANN classifier, we have used Relu as an activation function. The optimizer was ‘Adam’ and the maximum iteration was 500. The model achieved an accuracy of 0.93 for the training dataset and an accuracy of 0.92 on the testing dataset for the detection of wormhole attacks.

The model achieved a precision of 0.90 and the recall value is 0.97. Similarly, the F-1 score of the proposed model tends to be 0.94.

To plot the confusion matrix, we have calculated the FN, FP, TN, and TP values. We have obtained, False Negative value = 1, False Positive = 10, True Positive = 134 and True Negative = 68

The CM and ROC operating point of the ANN is shown Fig 5.5 and Fig 5.6:

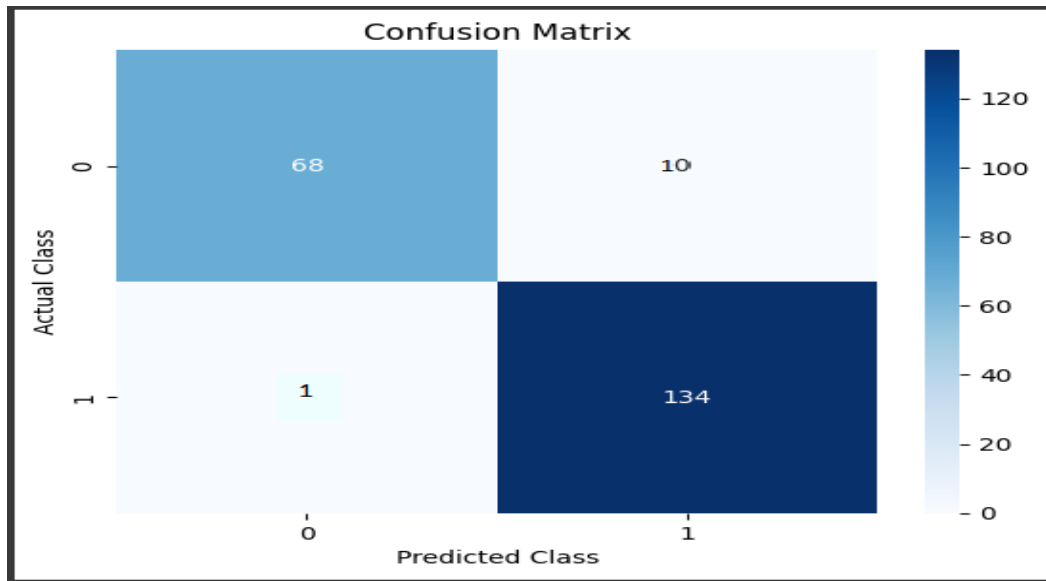


Fig 5.5: Confusion matrix using ANN

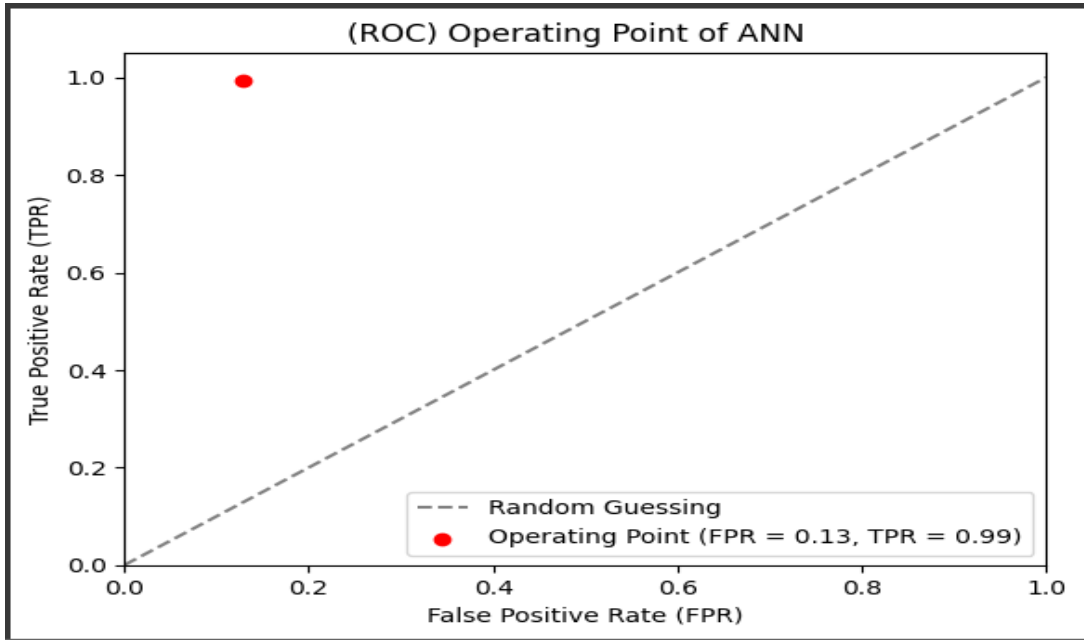


Fig 5.6: ROC operating point of ANN

5.6 Implementation of DBN

To build a DBN, we created a sequential model and set the different parameters. We have used 'Relu' as an activation function. The batch size = 32, the number of epochs = 32, verbose = 1, and used Root mean square propagation (RMS prop) optimizer.

After running the training of the model, we have decreasing loss as the number of epochs increases which means the model is training well as shown in the figure.


```

Epoch 1/20
25/25 [=====] - 1s 13ms/step - loss: 0.6413 - accuracy: 0.6313 - val_loss: 0.5309 - val_accuracy: 0.8150
Epoch 2/20
25/25 [=====] - 0s 4ms/step - loss: 0.5200 - accuracy: 0.7500 - val_loss: 0.4511 - val_accuracy: 0.8150
Epoch 3/20
25/25 [=====] - 0s 3ms/step - loss: 0.4532 - accuracy: 0.7937 - val_loss: 0.4106 - val_accuracy: 0.8250
Epoch 4/20
25/25 [=====] - 0s 4ms/step - loss: 0.4022 - accuracy: 0.8325 - val_loss: 0.3856 - val_accuracy: 0.8250
Epoch 5/20
25/25 [=====] - 0s 6ms/step - loss: 0.3826 - accuracy: 0.8487 - val_loss: 0.3800 - val_accuracy: 0.8300
Epoch 6/20
25/25 [=====] - 0s 3ms/step - loss: 0.3592 - accuracy: 0.8575 - val_loss: 0.3809 - val_accuracy: 0.8300
Epoch 7/20
25/25 [=====] - 0s 4ms/step - loss: 0.3494 - accuracy: 0.8662 - val_loss: 0.3794 - val_accuracy: 0.8300
Epoch 8/20
25/25 [=====] - 0s 3ms/step - loss: 0.3697 - accuracy: 0.8512 - val_loss: 0.3758 - val_accuracy: 0.8350
Epoch 9/20
25/25 [=====] - 0s 4ms/step - loss: 0.3643 - accuracy: 0.8600 - val_loss: 0.3758 - val_accuracy: 0.8350
Epoch 10/20
25/25 [=====] - 0s 5ms/step - loss: 0.3691 - accuracy: 0.8537 - val_loss: 0.3788 - val_accuracy: 0.8350
Epoch 11/20
25/25 [=====] - 0s 3ms/step - loss: 0.3417 - accuracy: 0.8700 - val_loss: 0.3798 - val_accuracy: 0.8300
Epoch 12/20
25/25 [=====] - 0s 3ms/step - loss: 0.3351 - accuracy: 0.8687 - val_loss: 0.3768 - val_accuracy: 0.8300
Epoch 13/20
25/25 [=====] - 0s 3ms/step - loss: 0.3285 - accuracy: 0.8625 - val_loss: 0.3815 - val_accuracy: 0.8350
Epoch 14/20
25/25 [=====] - 0s 4ms/step - loss: 0.3307 - accuracy: 0.8763 - val_loss: 0.3781 - val_accuracy: 0.8250
Epoch 15/20
25/25 [=====] - 0s 4ms/step - loss: 0.3248 - accuracy: 0.8788 - val_loss: 0.3805 - val_accuracy: 0.8300

```

Fig 5.7: Training of DBN

The model achieved a precision of 0.835 and the recall value is 0.84. Similarly, the F-1 score of the proposed model is 0.85. The ROC AUC score is 0.83.

To plot the confusion matrix, we have calculated the FN, FP, TN, and TP values. We have obtained, False Negative value = 19, False Positive = 14, True Positive = 92 and True Negative = 75

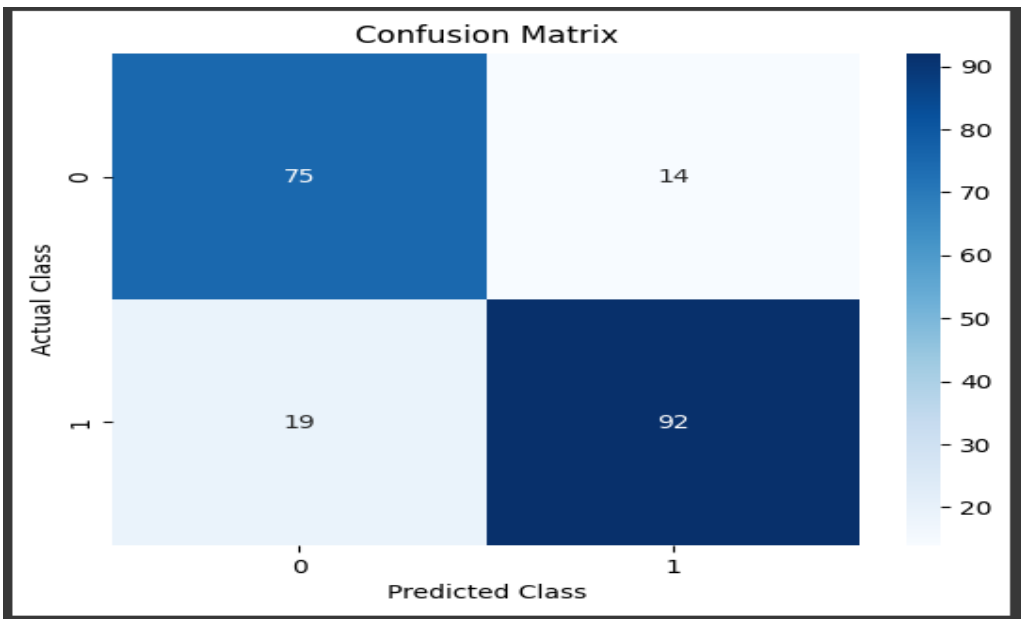


Fig 5.8: Confusion matrix using DBN

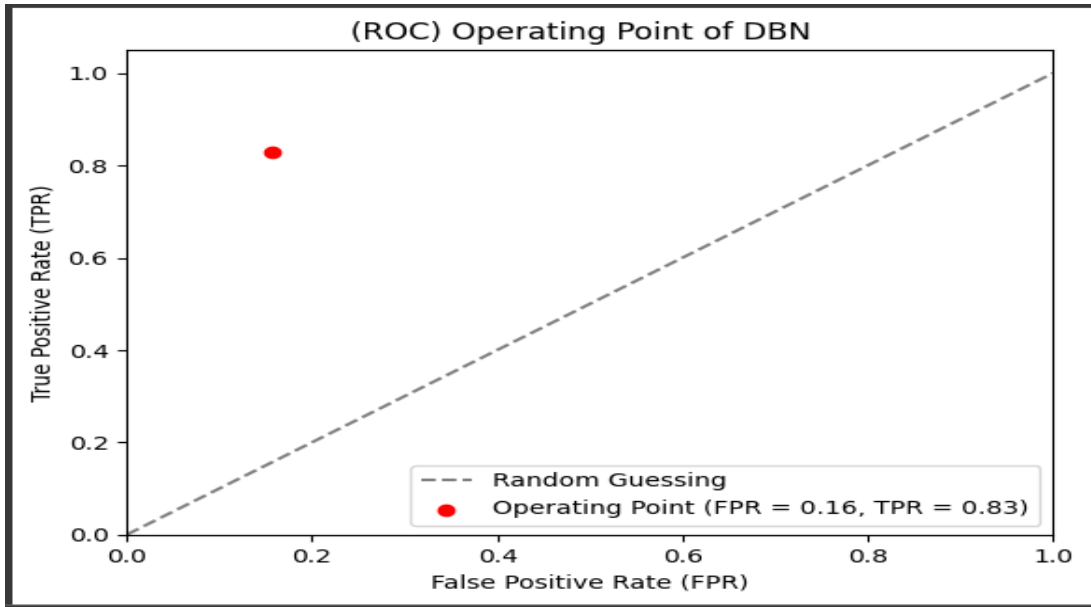


Fig 5.9: ROC operating point of DBN

5.7 Implementation of (RBFN) for Binary Classification

To train the RBFN, Use K-Means clustering to determine the centers of the radial basis functions. Then Calculate the distance between each training example and the RBF centers. Train a linear classifier i-e Logistic Regression based on RBF features. After the training, we got an accuracy of 0.775 and the precision of the model was 0.88. Similarly, the recall value and F-1 score value were 0.88 and 0.77 respectively.

The AUC-ROC curve is a way to assess the performance of classification tasks by examining different threshold settings. The ROC curve is a graphical representation of probabilities, and the AUC indicates how well the classes can be distinguished from each other. The ROC-AUC value in the case of RBFN is 0.785.

To plot the confusion matrix, we have calculated the FN, FP, TN, and TP values. We have obtained, False Negative value = 34, False Positive = 11, True Positive = 77 and True Negative = 78

The confusion matrix curve and ROC operating point of the RBFN classifier is shown in Fig 5.10 and Fig 5.11 respectively:

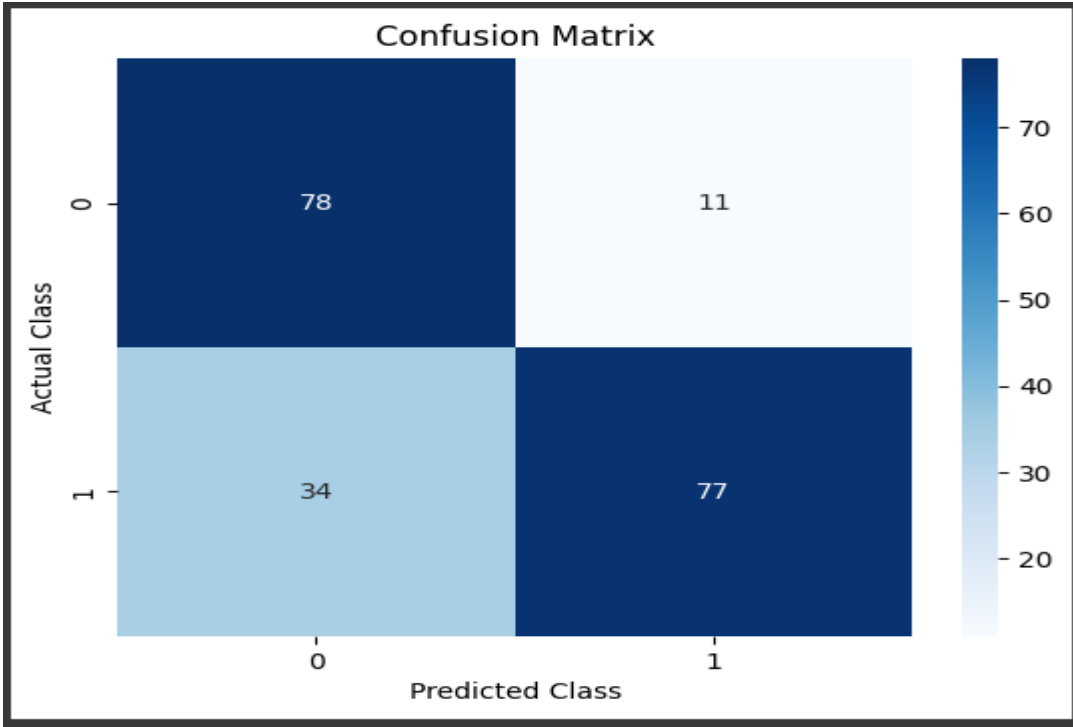


Fig 5.10: Confusion matrix of RBFN

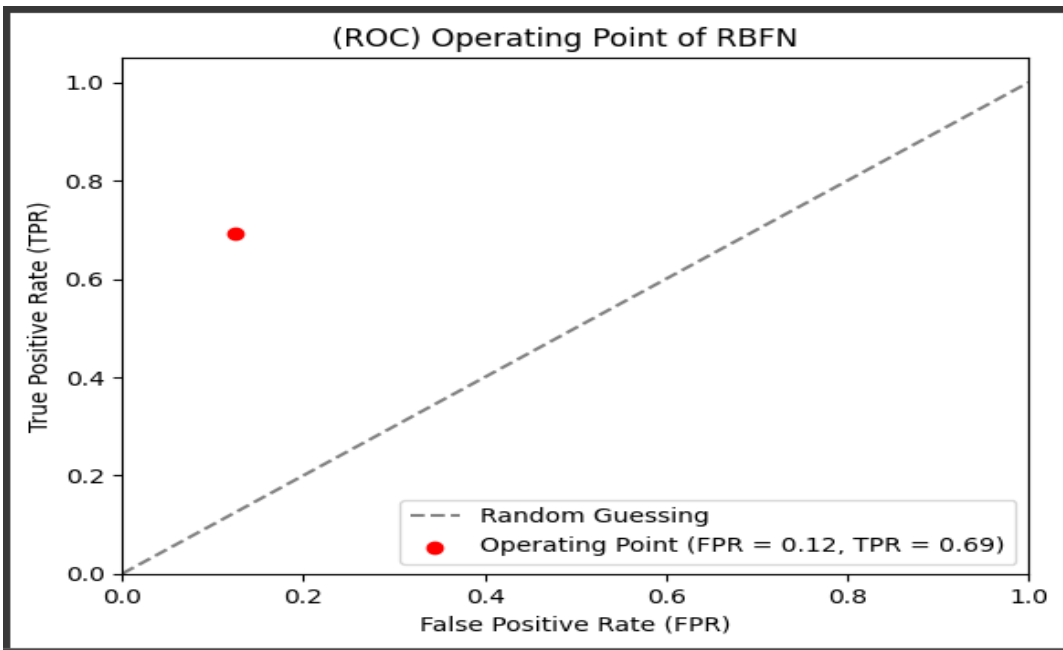


Fig 5.11: ROC operating point of RBFN

5.8 Performance Metrics

To evaluate the effectiveness of the performance of the proposed algorithms, we have chosen some statistical parameters. The results of the proposed algorithms are compared with the algorithms that were used in the literature. When testing a multi-class classifier, there are performance metrics that can be used to measure its performance [21]

5.8.1 Accuracy

Accuracy is a performance metric that measures how well a classification model correctly predicts the class labels of the samples in the dataset. In terms of the confusion matrix, accuracy can be written as:

$$\mathbf{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Accuracy is a commonly used metric for evaluating classification models, but it can be misleading in certain situations. For instance, if our dataset is not balanced, i.e., one class has significantly more samples than the other, then a high accuracy score may not necessarily indicate a good model. In such cases, it is recommended to use other performance metrics, such as precision, recall, and F1-score, to evaluate the model's performance more comprehensively and accurately.

5.8.2 Confusion Matrix

A confusion matrix is a table used to evaluate the performance of a classification model. The matrix compares the predicted class labels of the model to the actual or true class labels of the samples in a dataset.

The entries of confusion matrix consist of four entries:

- True positives (TP): The count of samples that the model accurately identifies as positive.
- False positives (FP): The count of samples that the model incorrectly labels as positive. FP is also called a Type-I error.

- True negatives (TN): The count of samples that the model accurately identifies as negative.
- False negatives (FN): The number of samples that are wrongly classified as negative by the model. FN is also called type-II error.

For the binary classifier, the confusion matrix can be sketched in table form.

		Predicted Class	
		True	False
True Class	Positive	True Positive	False Positive
	Negative	True Negative	False Negative

Fig 5.12: Binary Class Confusion matrix

5.8.3 Precision

Precision is a performance metric in machine learning that measures the proportion of true positives (TP) among all the samples that are predicted to be positive by the model. In other words, precision tells us how often the model's positive predictions are correct. Actually, it is the fraction of true positive predictions among all positive predictions. A higher precision means fewer FP, thus lower chance of making a Type-I error.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

5.8.4 Recall

The Recall is the fraction of true positive predictions among all actual positive examples.

A higher recall means fewer FN but may increase the risk of type-1 error and lower the type-II error.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

5.8.5 ROC Curve and operating point

The ROC (Receiver Operating Characteristic) curve is a graphical representation of the performance of a binary classification model. It illustrates the trade-off between the true positive rate (TPR) and the false positive rate (FPR) at various classification thresholds.

The ROC curve is created by plotting the TPR (also known as sensitivity or recall) on the y-axis against the FPR (specificity) on the x-axis, as the classification threshold is varied. Each point on the curve represents a different threshold, and the curve provides a comprehensive view of the model's performance across all possible thresholds.

The ROC curve is useful for evaluating and comparing different models. A better-performing model will have an ROC curve that is closer to the top-left corner of the plot, indicating a higher TPR and a lower FPR across a range of thresholds.

Similarly, the operating point on ROC determines the balance between the true positive rate and the false positive rate for a particular threshold setting.

5.9 Comparative Analysis

We have performed analysis by using different algorithms. Initially, used those algorithms which were already implemented by the researchers, then after that we have performed analysis on few new algorithms.

Table 5.1: Comparative Analysis of different algorithms

Algorithm	Confusion Matrix	Precision	F-score	Accuracy	Simulator	ROC AUC
Logistic Regression	FP=42 FN= 17 TP=117 TN=26	73%	80%	70%	Cooja	-
Gaussian Naïve Bayes	FP=67 FN= 2 TP= 132 TN=1	67%	79%	66%	Cooja	-
RF	FP=25, FN=12 TN=43, TP=122	82%	86%	82%	Cooja	-
SVM	FP=53, FN=2 TP=132, TN=14	72%	82%	72%	Cooja	-
Ada-Boost	FP=6, FN=18 TP=93, TN=83	93%	89%	88%	Cooja	-
Ridge Classifier	FP=8, FN=0, TP=134, TN=60	94%	97%	96%	Cooja	-
DBN	FP=14, FN=19 TP=92, TN=75	83%	85%		Cooja	84%
ANN	FP=10, FN=1 TP=134, TN=68	90%	94%	92%	Cooja	-
RBFN	FP=11, FN=78 TP=77, TN=34	88%	88%	77%	Cooja	78%

5.10 Analysis and Recommendations

Above table performs the comparison analysis of different algorithms. Analysis has been performed based on the few technical parameters related to dataset.

RF, SVM, Logistic Regression and Ada-Boost perform well on the low amount of dataset. If we increase the dataset and ultimately the features as well then these models not give us optimal results, as highlighted in the table.

Ridge Classifier is a linear model with ridge regularization, which helps to handle multicollinearity and can prevent overfitting. It can work well with high-dimensional data and is computationally efficient. The training of Ridge classifier is computationally easy

and efficient. Also this model has built-in feature selection capability by applying regularization.

As our dataset is non-linear in nature. The distribution of our dataset was log distribution. So, for the non-linear distribution datasets, ANN and RBFN performs well and gives better results. DBNs use unsupervised pre-training to initialize their parameters, allowing them to learn useful representations without the need for labeled wormhole attack instances. From the above table it is clear that our main models perform very well as compared to simple machine learning algorithms like RF, SVM, Logistic Regression and Ada-Boost etc. The accuracy and precision values are much better in our case.

To improve the results of ANN, DBN, RBFN, and Ridge Classifier for wormhole attack detection in IoT networks, consider the following recommendations:

- 1- As ANN and DBN need more dataset for the optimal performance, so gather a diverse dataset that includes a sufficient number of wormhole attack instances. Ensure that the dataset captures various network conditions, device configurations, and attack scenarios encountered in IoT network environment. So if we increase the dataset then our models definitely will give us much better results.
- 2- Conduct in-depth feature engineering
- 3- Model Selection and Hyperparameter Tuning

5.11 Discussion

The results obtained from the experiment show that the proposed models perform well in terms of accuracy, precision, and recall on the dataset generated in the Cooja simulator.

Based on the given results, the Ridge classifier model achieved a high accuracy score of 0.97 for the training dataset and an accuracy of 0.96 for the testing dataset, indicating that the model is performing well and is not overfitting the training data.

The precision value of 0.94 indicates that when the model predicts the wormhole attack, it is correct 94% of the time. The recall value of 0.99 indicates that the model can identify 99% of all wormhole attacks in the testing dataset. The high F1 score of 0.9710 indicates

that the model has a good balance between precision and recall.

To further analyze the model's performance, we can look at the confusion matrix. The confusion matrix provides the counts of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) predicted by the model. In this case, the model correctly predicted 134 wormhole attacks (TP) and correctly predicted 60 non-wormhole attacks (TN). However, it incorrectly classified 8 non-wormhole attacks as wormhole attacks (FP). Importantly, there were no false negatives (FN), which means the model did not miss any wormhole attacks in the testing dataset.

The ANN classifier model with ReLU activation function and 'Adam' optimizer achieved an accuracy score of 0.93 for the training dataset and an accuracy of 0.92 for the testing dataset in detecting wormhole attacks. The precision value of 0.90 indicates that when the model predicts the wormhole attack, it is correct 90% of the time. The recall value of 0.97 indicates that the model can identify 97% of all wormhole attacks in the testing dataset. The F1 score of 0.94 indicates that the model has a good balance between precision and recall.

Looking at the confusion matrix, the model correctly predicted 134 wormhole attacks (TP) and correctly predicted 68 non-wormhole attacks (TN). However, it incorrectly classified 10 non-wormhole attacks as wormhole attacks (FP), and it missed one Wormhole attack (FN).

Overall, these results suggest that the ANN classifier model with ReLU activation function and 'Adam' optimizer is effective in detecting wormhole attacks, with high accuracy, precision, recall, and F1 score. However, there is still room for improvement, particularly in reducing the number of false positives and false negatives.

For the DBN, the model achieved a precision value of 0.835 and the recall value of 0.84, indicating that can correctly identify 83.5% of true wormhole attacks and correctly exclude 84% of non-wormhole attacks. The F1 score of 0.85 that the model maintains a favorable balance between precision and recall. The ROC AUC score of 0.83 indicates that the model has moderate discriminatory ability.

Looking at the confusion matrix, the model correctly identified 92 wormhole attacks (TP) and 75 non-wormhole attacks (TN), but it incorrectly classified 19 wormhole attacks as non-wormhole attacks (FN) and 14 non-wormhole attacks as wormhole attacks (FP).

The model achieved an accuracy score of 0.775, indicating that it correctly classified 77.5% of the data points. The precision value of 0.88 indicates that when the model predicts a wormhole attack, it is correct 88% of the time. The recall value and F1 score of 0.88 and 0.77, respectively, suggest that the model can identify 88% of all wormhole attacks, but there is room for improvement in terms of precision and the balance between precision and recall.

The ROC-AUC score of 0.785 indicates that the model exhibits a moderate capability to differentiate between wormhole and non-wormhole attacks.

Upon analyzing the confusion matrix, the model correctly identified 77 wormhole attacks (TP) and 78 non-wormhole attacks (TN), but it incorrectly classified 34 wormhole attacks as non-wormhole attacks (FN) and 11 non-wormhole attacks as wormhole attacks (FP).

Conclusion and Future Work

6.1 Conclusion

In this paper machine learning and deep-learning based RPL-based wormhole attack detection have been proposed in IoTs. We have tested multiple machine learning algorithms by hit and trial methods, but among them, few algorithms provide the best accuracy and detection rate. These algorithms include Ridge classifier, artificial neural network (ANN), deep belief network (DBN), and radial basis function network (RBFN). Our proposed strategy employs the Cooja simulator for complex dataset generation in real environment. We have deployed 11 normal nodes to generate normal traffic datasets and 2 attacking nodes which are wormhole attack nodes, to generate malicious datasets in RPL network. Further, we have preprocessed the dataset using different statistical techniques to reduce the features and to select the most important features. We have used additional features i-e (i) C.P.U (ii) LPM (iii) Transmitt (iv) Listen (v) Idle-transmit and, and Idle-listen which were not used by the researchers in their analysis. Additionally, some more data preprocessing techniques have been used to clean the dataset. Finally, the simulation results depict the effectiveness of the proposed algorithms in terms of various performance metrics. The proposed algorithms Ridge classifier, ANN, DBN, and RBFN achieved accuracy of 97, 93%, 84% and 77% respectively. Our results shows that our approach achieves high accuracy on complex datasets in detecting wormhole attacks, making it a promising solution for enhancing the security of IoT networks.

6.2 Future Work

In the future, we can perform analysis on the other IoT attacks. For multiple attacks, we will have a huge amount of dataset therefore our intrusion detection system (IDS) can have the ability to detect multiple attacks.

Since the proposed analysis has been performed in the simulator, there will be a clear difference between the simulated environment and real-time environment. Therefore, the same analysis can be performed on the real time IoT network for improved results.

As in our analysis, we have a limited amount of data but Deep learning and deep reinforcement learning models are efficiently trained on a huge of amount of data. Therefore, if we import big data to deep learning models then our accuracy will be much improved.

Due to the limited resources nature of IoT devices, there should be lightweight cryptographic protocols for secure communication in the IoT network.

References

- [1] Selvaraj, S.; Sundaravaradhan, S. Challenges and Opportunities in IoT Healthcare Systems: A Systematic Review. *SN Appl. Sci.* 2020,2,139
- [2] Almusaylim, Z.A.; Zaman, N. A Review on Smart Home Present State and Challenges: Linked to Context-Awareness Internet of Things (IoT). *Wirel. [3] Netw.*2019, 25, 3193-3204.
- [3] Faraj, O.; Megías, D.; Ahmad, A.M.; Garcia-Alfaro, J. Taxonomy and challenges in machine learning-based approaches to detect attacks in the internet of things. In *Proceedings of the 15th International Conference on Availability, Reliability and Security, Virtual Event, Ireland, 25–28 August 2020*; pp. 1–10
- [4] Al-Amiedy, Taief Alaa, Mohammed Anbar, Bahari Belaton, Arkan Hammoodi Hasan Kabla, Iznan H. Hasbullah, and Ziyad R. Alashhab. "A systematic literature review on machine and deep learning approaches for detecting attacks in RPL-based 6LoWPAN of Internet of Things." *Sensors* 22, no. 9 (2022): 3400.
- [5] Deshmukh-Bhosale, Snehal, and Santosh S. Sonavane. "A real-time intrusion detection system for wormhole attack in the RPL based Internet of Things." *Procedia Manufacturing* 32 (2019): 840-847.
- [6] T. Kushalnagar, G. Montenegro, C. Schumacher, IPv6 over Low-power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem State- ment, and Goals, RFC 4919 (2007).
- [7] Marianne Azer, Sherif El-Kassas, Magdy El-Soudani, "A Full Image of the Wormhole Attacks Towards Introducing Complex Wormhole Attacks, in

wireless Ad Hoc Networks”, International Journal of Computer Science and Information Security, Vol. 1, No. 1, May 2009

- [8] Ing PietroGonizzi and Simon Duquennoy. Hands on Contiki OS and Cooja Simulator:Exercises(PartII).https://team.inria.fr/fun/files/2014/04/slides_partI.pdf, 2013. [Online; accessed 10-November-2017]
- [9] E. Kim, D. Kaspar, and J. Vasseur. Design and Application Spaces for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs). RFC 6568, 2012.
- [10] Verma, Abhishek, and Virender Ranga. "Mitigation of DIS flooding attacks in RPL-based 6LoWPAN networks." Transactions on emerging telecommunications technologies 31, no. 2 (2020): e3802.
- [11] N. Kushalnagar, G. Montenegro, and C. Schumacher, “IPv6 over low-power wireless personalArea networks (6LoWPANs): overview, assumptions, problem statement, and goals,” RFC 4919, 2007
- [12] Prasad, M., Tripathi, S. and Dahal, K., 2019, July. Wormhole attack detection in ad hoc network using machine learning technique. In 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT) (pp. 1-7). IEEE.
- [13] Zahra, F., Jhanjhi, N.Z., Brohi, S.N., Khan, N.A., Masud, M. and AlZain, M.A., 2022. Rank and wormhole attack detection model for RPL-based internet of things using machine learning. Sensors, 22(18), p.6765.

- [14] Jhanjhi, N.Z., Brohi, S.N., Malik, N.A. and Humayun, M., 2020, October. Proposing a hybrid rpl protocol for rank and wormhole attack mitigation using machine learning. In 2020 2nd International Conference on Computer and Information Sciences (ICCIS) (pp. 1-6). IEEE.
- [15] Deshmukh-Bhosale, S. and Sonavane, S.S., 2019. A real-time intrusion detection system for wormhole attack in the RPL based Internet of Things. *Procedia Manufacturing*, 32, pp.840-847.
- [16] Keipour, Hossein. "Blackhole Attack Detection in Low-Power IoT Mesh Networks Using Machine Learning Algorithms." (2022).
- [17] Rayeesa Malik, Yashwant Singh, Zakir Ahmad Sheikh, Pooja Anand, Pradeep Kumar Singh, Tewabe Chekole Workneh, "An Improved Deep Belief Network IDS on IoT-Based Network for Traffic Systems", *Journal of Advanced Transportation*, vol. 2022, Article ID 7892130, 17 pages, 2022.
<https://doi.org/10.1155/2022/7892130>
- [18] Deep belief network, 2021,
<https://www.sciencedirect.com/topics/engineering/deep-belief-network>.
- [19] Deep belief networks an introduction, 2021,
<https://medium.com/black-feathers-labs/deep-belief-networks-an-introduction-1d52bb867a25>.
- [20] H. Beitollahi, D. M. Sharif and M. Fazeli, "Application Layer DDoS Attack Detection Using Cuckoo Search Algorithm-Trained Radial Basis Function," in *IEEE Access*, vol. 10, pp. 63844-63854, 2022, doi:10.1109/ACCESS.2022.3182818

- [21] Grandini M, Bagli E, Visani G. Metrics for multi-class classification: an overview. arXiv preprint arXiv:2008.05756. 2020 Aug 13
- [22] https://www.saedsayad.com/artificial_neural_network_rbf.htm
- [23] Mehta, Ruchi, and M. M. Parmar. "Trust based mechanism for securing iot routing protocol rpl against wormhole & grayhole attacks." In 2018 3rd International Conference for Convergence in Technology (I2CT), pp. 1-6. IEEE, 2018.
- [24] Abdan, M. and Seno, S.A.H., 2022. Machine learning methods for intrusive detection of wormhole attack in mobile ad hoc network (MANET). Wireless Communications and Mobile Computing, 2022, pp.1-12.
- [25] Kumar, Ananth. "An adaptive technique for wormhole attack detection in MANET using quantized ad-hoc on-demand multipath distance vector routing protocol." In International Conference on Mathematical and Statistical Physics, Computational Science, Education, and Communication (ICMSCE 2022), vol. 12616, pp. 34-55. SPIE, 2023.
- [26] Ezhilarasi, M., Gnanaprasanambikai, L., Kousalya, A. and Shanmugapriya, M., 2023. A novel implementation of routing attack detection scheme by using fuzzy and feed-forward neural networks. Soft Computing, 27(7), pp.4157-4168.
- [27] Tahboush, M. and Agoyi, M., 2021. A hybrid wormhole attack detection in mobile ad-hoc network (MANET). IEEE Access, 9, pp.11872-11883.
- [28] Ali, S., Nand, P. and Tiwari, S., 2022. Detection of wormhole attack in vehicular ad-hoc network over real map using machine learning approach with preventive scheme. Journal of Information Technology Management, 14(Security and

Resource Management challenges for Internet of Things), pp.159-179.

- [29] Zahra, F., Jhanjhi, N.Z., Brohi, S.N., Khan, N.A., Masud, M. and AlZain, M.A., 2022. Rank and wormhole attack detection model for RPL-based internet of things using machine learning. *Sensors*, 22(18), p.6765.

