

ZERO-DAY DETECTION ON IOT NETWORKS

By

OH JIA SHENG

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONOURS)

Faculty of Information and Communication Technology

(Kampar Campus)

JAN 2024

REPORT STATUS DECLARATION FORM

Title: ZERO-DAY DETECTION ON IOT NETWORKS

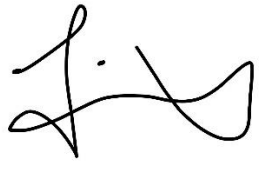
Academic Session: 01/2024

I OH JIA SHENG
(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,



(Author's signature)



(Supervisor's signature)

Address:
58, Selasar Pasir Puteh,
Taman Sentosa,
31650, Ipoh, Perak.

Aun Yichiet
Supervisor's name

Date: 24/04/2024

Date: 26/04/2024

Universiti Tunku Abdul Rahman			
Form Title : Sample of Submission Sheet for FYP/Dissertation/Thesis			
Form Number: FM-IAD-004	Rev No.: 0	Effective Date: 21 JUNE 2011	Page No.: 1 of 1

FACULTY/INSTITUTE* OF INFORMATION AND COMMUNICATION TECHNOLOGY
UNIVERSITI TUNKU ABDUL RAHMAN

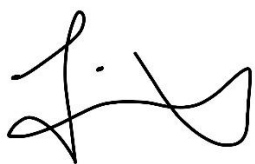
Date: 24/04/2024

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that OH JIA SHENG (ID No: 21ACB01918) has completed this final year project/ dissertation/ thesis* entitled “ ZERO DAY DETECTION ON IOT NETWORKS ” under the supervision of DR. AUN YICHJET (Supervisor) from the Department of _____, Faculty/Institute* of INFORMATION AND COMMUNICATION TECHNOLOGY .

I understand that University will upload softcopy of my final year project / dissertation/ thesis* in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,



(*Oh Jia Sheng*)

*Delete whichever not applicable

DECLARATION OF ORIGINALITY

I declare that this report entitled “**ZERO-DAY DETECTION ON IOT NETWORKS**” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : 

Name : OH JIA SHENG

Date : 24 APRIL 2024

ACKNOWLEDGEMENTS

I am deeply grateful for the invaluable support and guidance provided by my supervisor, Dr Aun Yichet, throughout the course of my project on Zero-day detection in IoT networks. This project has provided me with a unique opportunity to delve into the dynamic and interconnected realms of cybersecurity, artificial intelligence, and the Internet of Things. Dr Aun's insightful supervision and mentorship have been instrumental in shaping the direction and success of this endeavour. I would like to express my sincere appreciation to Dr Aun for entrusting me with this intricate and multifaceted project. His vision and belief in my abilities have empowered me to explore uncharted territories within the realm of technological innovation. With his guidance, I have not only expanded my knowledge but also developed a deep understanding of the complexities and challenges posed by securing IoT networks against zero-day vulnerabilities. Throughout the project, his unwavering dedication to my growth has been remarkable. His consistent willingness to allocate time for discussions and meetings, even during busy periods, exemplifies his commitment to my academic and personal development. In every interaction, he demonstrated a remarkable ability to break down complex issues, patiently explaining potential obstacles and proposing innovative solutions. His insights have been a beacon of clarity, guiding me through intricate technical concepts and helping me navigate through periods of uncertainty. I am particularly grateful for Dr Aun's proactive approach to mentorship. His keen interest in my progress, demonstrated by his regular inquiries about my research and well-being, has fostered a nurturing academic environment that has allowed me to thrive. His feedback and constructive criticism have played a pivotal role in refining my ideas and methodologies, pushing me to constantly strive for excellence. Finally, I extend my heartfelt gratitude to Dr Aun Yichet for his exceptional guidance, mentorship, and relentless dedication to my project. This endeavour would not have been possible without his trust, encouragement, and expertise. As I conclude this phase of my academic journey, I am confident that the skills and insights I have gained under his tutelage will continue to guide me in all my future pursuits.

ABSTRACT

This project aims to develop a federated learning-based solution for detecting zero-day attacks on IoT devices. Zero-day attacks exploit vulnerabilities unknown to developers or security experts, making them difficult to detect and prevent using traditional security measures. The project's main objectives are to leverage distributed machine learning techniques to train models on data stored on different devices without transferring the data to a central server, improve early detection of zero-day attacks, and reduce network traffic. By detecting and addressing zero-day attacks faster, IoT device companies can minimize the potential impact of such attacks and prevent further vulnerability exploitation. Users with IoT devices vulnerable to such attacks risk having their personal information stolen, hijacked, or becoming victims of cyberattacks. Rapid detection and response to zero-day attacks can help minimize these risks and protect users' privacy and security. The project's impact and significance include improving user privacy, reducing network traffic, increasing the precision of zero-day attack detection models, and providing quicker responses for identifying IoT device zero-day threats.

TABLE OF CONTENTS

TITLE PAGE	i
DECLARATION OF ORIGINALITY	iv
ACKNOWLEDGEMENTS	v
ABSTRACT	vi
TABLE OF CONTENTS	vii
LIST OF FIGURES	x
LIST OF TABLES	xii
LIST OF ABBREVIATIONS	xiv
CHAPTER 1 INTRODUCTION	1
1.1 Problem Statement and Motivation	5
1.1.1 Slow Time in Patching Vulnerabilities in Software and Firmware	5
1.1.2 Require large amount of data in Detecting Zero-day attacks via Traditional methods	6
1.1.3 Privacy Issue	6
1.2 Research Objectives	7
1.3 Project Scope and Directions	8
1.4 Project Contribution, Impact and Significance in general	9
1.5 Benefits of the project towards IoT device companies and users	10
1.6 Report Organization	11
CHAPTER 2 LITERATURE REVIEW	12
2.1 What is Zero-day attacks and zero-day attacks in IOT security?	12
2.2 Previous or Similar works of this project	13
2.2.1 Centralized Deep Learning (CDL) Model	13
2.2.2 Hybrid Deep Learning Model	16
2.2.3 Similar Federated Learning model	18
2.3 Datasets type	24
	vii

2.4 Evaluation metrics	32
CHAPTER 3 SYSTEM MODEL	34
3.1 2-Layers Zero-Day Attack Classification Model	34
3.2 Federated Learning Framework	36
CHAPTER 4 SYSTEM DESIGN	40
4.1 Local & Global Model Design	40
4.1.1 Data Acquisition	42
4.1.2 Dataset Creation	43
4.1.3 Data Pre-Processing	45
4.1.4 Model Formation & Model Training	47
4.1.5 Model Evaluation	54
4.1.7 Hyperparameter Fine-Tuning	56
4.1.8 Model Deployment	57
4.2 Federated Learning Framework	58
CHAPTER 5 EXPERIMENT/ SIMULATION	62
5.1 Hardware Setup	62
5.2 Software Setup	62
5.3 Setting and Configuration	63
5.4 Federated Learning Framework Simulation	64
5.5 Implementation Issues and Challenges	68
5.5.1 High Cost of IoT Devices	68
5.5.2 Difficulty in Generating Attacks in Ton-IoT dataset	68
5.6 Concluding Remark	69
CHAPTER 6 SYSTEM EVALUATION AND DISCUSSION	70
6.1 System Testing and Performance Metrics	70
6.1.1 Model Evaluation	70

6.1.2 Overall 2-Layer Classification Model Pipeline Testing and Performance	71
6.1.3 Federated Learning Framework Testing and Performance	74
6.2 Testing Setup and Result	75
6.2.1 Model Evaluation Result	75
6.2.2 2-Layers Model Pipeline Classification Evaluation	84
6.2.2 Federated Learning Framework Performance Result	85
6.3 Project Challenges	86
6.4 Objectives Evaluation	86
6.4 Project Timeline	87
6.5 Concluding Remark	90
CHAPTER 7 CONCLUSION AND RECOMMENDATION	91
7.1 Conclusion	91
7.2 Recommendation	91
REFERENCES	i
APPENDIX A	A-1
A.1 Weekly Report	A-1
A.2 Poster	A-14
PLAGIARISM CHECK RESULT	I
FYP 2 CHECKLIST	I

LIST OF FIGURES

Figure Number	Title	Page
Figure 1	Number of IoT connected devices worldwide from 2019 to 2021, with forecasts from 2022 to 2030 (in billions)	1
Figure 2	How Mirai work	4
Figure 3	SMOTE-DRNN for botnet attack detection in IoT networks	14
Figure 4	SMOTE Algorithm	14
Figure 5	Flow chart of Memory-efficient Deep Learning Model	16
Figure 6	Proposed collaborative intrusion detection-based BiLSTM	17
Figure 7	Intrusion Detection for Wireless Edge Networks Based on Federated Learning	18
Figure 8	Confusion Matrix	32
Figure 9	Total Predicted Positive	32
Figure 10	Total Actual Positive	33
Figure 11	2-Layers Zero-Day Attack Classification Model	34
Figure 12	Federated Learning Framework	36
Figure 13	Traffic Flow and Mechanism of Edge IoT Device	37
Figure 14	Role of Central Node	38
Figure 15	Federated Learning Flow when Zero-day Attack Happens in Any Edge Node	39
Figure 16	Model Flow of Local Zero-day Attack Model	40
Figure 17	Dataset Creation	43
Figure 18	Data Pre-processing Flow	45
Figure 19	Zero-day Attack Detection Flow	47
Figure 20	Network Architecture of CNN Binary Model	49
Figure 21	Network Architecture of CNN Multi-Class Model	51
Figure 22	Federated Learning Framework Design	58
Figure 23	First Iteration of Aggregating and Updating Local Model	60

Figure 24	Second Iteration of Aggregating and Updating Local Model	60
Figure 25	Third Iteration of Aggregating and Updating Local Model	61
Figure 26	Method Used to Evaluate 2-Layer Model Pipeline	71
Figure 27	Accuracy of Respective Models in the 2-Layers Model Pipeline	72
Figure 28	Percentage of Each Type of Data in Ton-IoT Dataset	73
Figure 29	Federated learning Framework Performance Testing Setup	74
Figure 30	Training and Validation accuracy for CNN Binary Model with Optimal Hyperparameter (32 batch size & 30 epochs)	77
Figure 31	Training and Validation Accuracy for CNN-Multi Class Model with Optimal Hyperparameter (32 batch size & 30 epochs)	77
Figure 32	Training and Validation Accuracy Graph for CNN Binary Model with Optimal Hyperparameter and Learning Rate of 0.0005 and 0.0001	79
Figure 33	Training and Validation Accuracy Graph for CNN Multi-Class Model with Optimal Hyperparameter and Learning Rate of 0.0005 and 0.0001	79
Figure 34	Confusion matrix and Classification Report of Binary CNN Model with optimal hyperparameter (32 batch size, 30 epochs, 0.0005 learning rate)	80
Figure 35	Confusion matrix and Classification Report of Multi-Class CNN Model with optimal hyperparameter (32 batch size, 30 epochs, 0.0005 learning rate)	81

LIST OF TABLES

Table Number	Title	Page
Table 1	Table of Comparison Between Previous Works and Proposed Solution	21
Table 2	Table of Comparison between Deep Learning Model Architecture used in Federated Learning Method	22
Table 3	Connection Activity Features of TON-IoT dataset	25
Table 4	Statistical Activity Features of TON-IoT dataset	25
Table 5	DNS Activity Features of TON-IoT dataset	26
Table 6	SSL Activity Features of TON-IoT dataset	26
Table 7	HTTP Activity Features of TON-IoT dataset	27
Table 8	Data Labelling of TON-IoT dataset	27
Table 9	Features of IoT-23 dataset	28
Table 10	Features of Bot-IoT dataset	29
Table 11	Some Features of N-BaloT dataset	30
Table 12	Table of Comparison for datasets	31
Table 13	Confusion Matrix for Binary Classification	55
Table 14	Laptop Hardware Specification	62
Table 15	Pseudocode for Central Server in Simulated Federated Learning Framework	64
Table 16	Pseudocode for IoT Device Class in Simulated Federated Learning Framework	66
Table 17	Training Accuracy of Models	75
Table 18	Validation Accuracy of Models	75
Table 19	Training Accuracy Result with Different Learning Rate for Both CNN Models	78
Table 20	Performance Metrics of Anomaly Detection with Different Hyperparameters	83
Table 21	Maximum Accuracy of Different Model 3 in 2-Layers Model Pipeline	84
Table 22	Ability of Different Model 3 in 2-Layers Model Pipeline	84

Table 23	Ability of each IoT Device to detect different type of attacks (Iteration 1)	85
Table 24	Ability of each IoT Device to detect different type of attacks (Iteration 2)	85
Table 25	Ability of each IoT Device to detect different type of attacks (Iteration 3)	85
Table 26	Timeline of IIPSPW	87
Table 27	Timeline of FYP1	88
Table 28	Timeline of FYP2	89

LIST OF ABBREVIATIONS

SHORTFORM	Full Name Of SHORTFORM
<i>IoT</i>	Internet Of Things
<i>DDoS attacks</i>	Distributed Denial-of-Service Attacks
<i>SMOTE</i>	Synthetic Minority Oversampling Technique
<i>DRNN</i>	Deep Recurrent Neural Network
<i>DNN</i>	Deep Neural Network
<i>SMOTE-DRNN</i>	Synthetic Minority Oversampling Technique-Deep Recurrent Neural Network
<i>LAE</i>	Long Short-Term Memory Autoencoder
<i>DBF</i>	Deep Blockchain Framework
<i>BiLSTM</i>	Bidirectional Long Short-Term Memory
<i>FedAGRU</i>	Federated Learning-based Attention Gated Recurrent Unit
<i>CDL</i>	Centralized Deep Learning
<i>DL</i>	Deep Learning
<i>NSL-KDD</i>	Network Security Laboratory - Knowledge Discovery in Databases
<i>KDD</i>	Knowledge Discovery in Databases
<i>N-BaloT</i>	Network-Based Anomaly Internet of Things
<i>R2L</i>	Remote To User
<i>U2R</i>	User To Remote
<i>IDS</i>	Intrusion Detection System
<i>API</i>	Application Programming Interface
<i>TP</i>	True Positive
<i>TN</i>	True Negative
<i>FP</i>	False Positive
<i>FN</i>	False Negative

CHAPTER 1 INTRODUCTION

In this chapter, I present the background of zero-day attacks on an Internet of Things (IoT) devices and motivation of our research to identify the zero-day attacks with federated learning-based solution, our contributions to the field, and the outline of the thesis.

In the era of globalisation and modernisation, IoT devices has slowly becomes the norm in the life of the people, no matter in the city area or the rural area [1, 2]. In fact, IoT devices had slowly replacing the traditional devices without us noticing that they are the “IoT devices”, since people are always confusing with the word of “IoT device”. For instance, smartwatches that we always wear when going out from our house and those smart appliances such as refrigerators, ovens, and washing machines are IoT devices that we can see often but we don’t notice them as “IoT devices”. As illustrated in [3, Figure 1], we can see that the trend of IoT devices shows a positive growth from 2019 to 2030 (forecast) according to Statista 2022. At this time, you might have a question in your mind, what is actually an IoT devices?

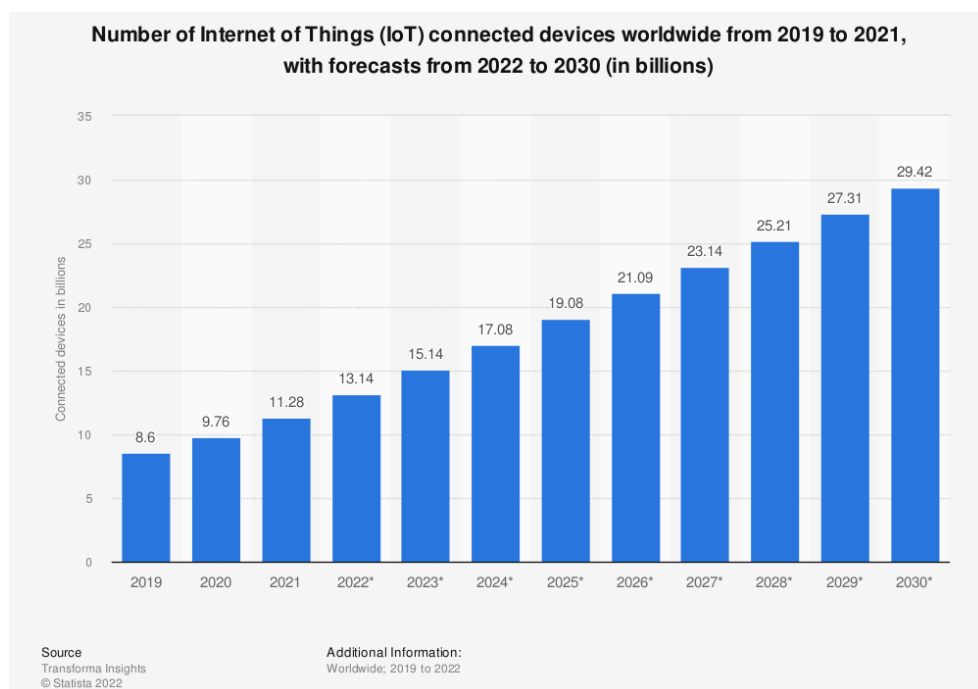


Figure 1 Number of IoT connected devices worldwide from 2019 to 2021, with forecasts from 2022 to 2030 (in billions)

CHAPTER 1

Before answering this question, we need to know the word “IoT”. IoT stands for the Internet of Things, which refers to a network of physical devices that are connected to the Internet and can communicate with each other and exchange data. By having the definition, we can conclude that IoT device is a physical device that are connected to the Internet and can communicate with other physical devices and exchange their data. As such, IoT devices is undeniably brings us a lot of advantages such as convenience, increase efficiency and even making better decision. However, is the IoT devices really that perfect? The answer is no. With the emerging of this technology, the issue of “security” also brings along to the sight of the public. [4]

As the technology of IoT devices is emerging, it means that the technology is still having improvement especially on the security measure. Although the IoT devices often have traditional security protection, but they have weak security measures to the zero-day attacks, making them vulnerable to exploitation by hackers. By utilizing the vulnerabilities, the hackers are able to steal our valuable data, violate our privacy and even do physical harm to the users. Once a hacker gains access to an IoT device, they can actually use it as a gateway to control or attack other devices or even the entire private network. Let’s imagine this, we bought a lot of high security devices to protect our network, however, due to one IoT device with low security protection to the exploitations, our whole network becomes vulnerable to the public or the hackers. Is this what we really want? The answer definitely is NO! As a result, we need to find a solution to solve this “security issue”.

But what is a zero-day attack once again? A zero-day attack on an IoT device refers to a type of cyber-attack that exploits a previously unknown vulnerability in the device's software or firmware. This type of attack is called "zero-day" because it takes place before the device's manufacturer release a patch or software update to fix the vulnerability. Zero-day attacks are especially dangerous as they can go undetected by traditional security measures, allowing the attacker to infiltrate and compromise the device or network. [5] As an exemplification, zero-day attacks on IoT devices include the Mirai botnet, which targeted vulnerable IoT devices and used them to launch large-scale DDoS attacks. [6] In October 2016, one of the DNS service providers named Dyn was attacked by DDoS attack that stemmed from the Mirai botnet. With this cyber

CHAPTER 1

incident, it brought to the company a range of consequences including business interruptions, recovery costs and reputational damages. An average of \$2.5 mils is needed to recover the damages caused by the DDoS attack. [7]

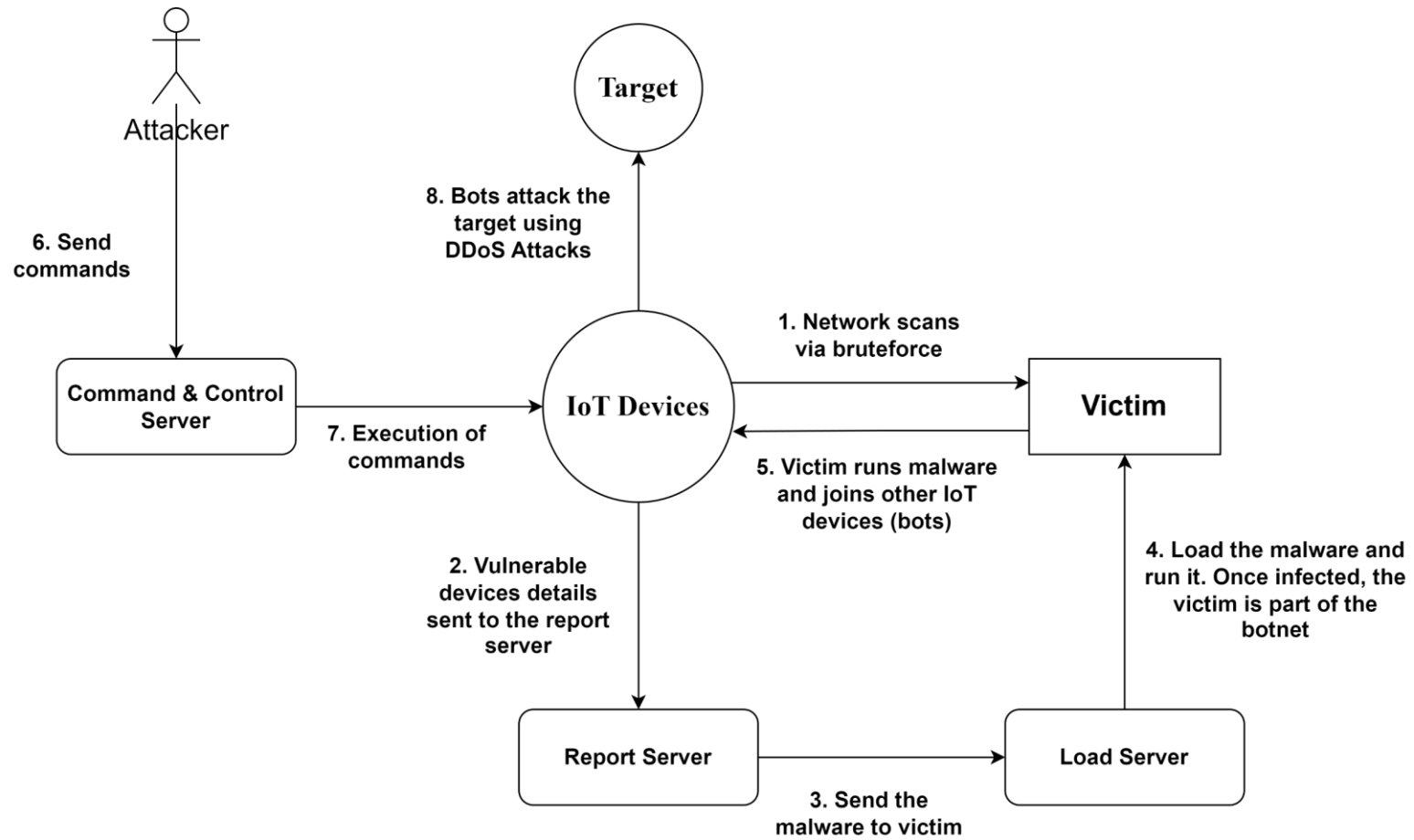


Figure 2 How Mirai work

1.1 Problem Statement and Motivation

Identifying zero-day attacks on IoT devices is crucial research to provide high-level security. However, detecting zero-day attacks is challenging as they exploit software or hardware vulnerabilities unknown to the developers or security experts. This means that no signatures or patterns can be used to identify the attack, and traditional security measures such as antivirus software or intrusion detection systems may not be effective.

1.1.1 Slow Time in Patching Vulnerabilities in Software and Firmware

One of the problems in detecting zero-day attacks is the slow time in patching vulnerabilities in software and firmware. Due to the delay, attackers have a window of time to take advantage of these vulnerabilities before they can be patched or corrected. The time it takes for manufacturers to provide patches or software upgrades to fix a vulnerability varies, and in certain situations, this leaves devices open to attack. Also, it could take some time before security experts create a patch or cure for a vulnerability and deploy it to all impacted devices. Attackers can still use the vulnerability during this period to compromise the device's or network's security and privacy. Moreover, traditional security methods, including intrusion detection systems that use signatures, frequently fail to identify zero-day threats. This is because signature-based systems rely on known patterns of attack and are, therefore, unable to detect new or previously unknown attack methods. Because of this, identifying zero-day attacks requires cutting-edge technologies that can spot unusual behaviour and patterns in network traffic and act quickly to neutralize threats before they can cause harm.

1.1.2 Require large amount of data in Detecting Zero-day attacks via Traditional method

Since zero-day attacks take advantage of previously undiscovered system vulnerabilities, detecting them using traditional signature-based techniques might be difficult. Traditional methods such as intrusion detection systems or anti-virus software of detecting zero-day attacks on IoT devices usually require collecting and centralizing large amounts of data from different devices in order to analyse it and identify patterns of malicious behaviour. This approach is slow and depends on the speed of the researchers to work on the patch for the vulnerability. In this situation, the approach increases the risk of unauthorized access and data breaches. Additionally, the centralized approach can be resource-intensive, making it difficult to keep up with the fast-paced and dynamic nature of zero-day attacks. As a result, new approaches are needed to address these challenges and provide effective for detecting zero-day attacks on IoT devices.

1.1.3 Privacy Issue

In the traditional method for detecting zero-day attacks on IoT devices, data is collected from the devices and sent to a central server for analysis. This means that all data, including sensitive information, is transferred from the devices to the central server. This poses a privacy concern, as there is a risk that the data could be intercepted or leaked during transfer. Moreover, once the data reaches the central server, it is stored in a centralized location, which increases the risk of unauthorized access. This could lead to a breach of privacy and confidentiality, particularly if the data contains sensitive information such as personal details, financial information, or trade secrets.

1.2 Research Objectives

The aim of the project is to develop a federated learning-based solution for detecting zero-day attacks on IoT devices. The goal of the project is to leverage distributed machine learning techniques to train models on data stored on different devices, without the need to transfer the data to a central server. By doing so, the project improves the early detection of zero-day attacks, which are often difficult to detect and prevent using traditional security measures. Additionally, we can reduce the risk of being attacked by hacker via zero-day attacks and increase the security level of the IoT devices. Besides, this project focuses on creating an architecture for early detection of zero-day attacks on IoT devices using federated learning. The approach involves designing a federated learning architecture for IoT devices, developing scripts to simulate and launch unseen IoT threats, training 2-layers classification model pipeline to detect zero-day attack, and evaluating the result.

1.3 Project Scope and Directions

The scope of the project is to develop a federated learning-based solution for detecting zero-day attacks on IoT networks. However, federated learning-based solution to detect zero-day attacks on IoT devices is rarely investigated in real life. Thereby, a detail investigation and analysis should be carried out to make sure that the federated learning-based solution is able to solve the current problems faced by traditional methods in detecting zero-day attacks as well as examine the contributions of the solution. The following describes the focus of the project:

1. Conduct a literature review to identify relevant research on federated learning and zero-day attacks on IoT devices.
2. Develop a framework for the federated learning-based solution, including selecting appropriate machine learning algorithms (CNN Models and Isolation Forest) and defining the dataset (Ton-IoT dataset) to be used.
3. Build and deploy the federated learning infrastructure, including the necessary hardware, software, and communication protocols.
4. Implement the federated learning framework on IoT devices to detect zero-day attacks in stimulation.
5. Evaluate the effectiveness of the federated learning-based solution in detecting zero-day attacks.
6. Document the project results and develop recommendations for future research and development.

1.4 Project Contribution, Impact and Significance in general

We can **improve user privacy** by creating a federated learning-based method to identify zero-day threats on IoT devices. Sensitive data is less likely to be exposed to a third party because it is stored locally on the devices, and only model changes are communicated to the central server. Then, by employing a federated learning-based approach to identify IoT device zero-day threats, we can **reduce network traffic**. By sending only updated models to the central server rather than raw data, federated learning reduces network traffic and system load. In addition, federated learning can be more accurate than traditional methods at identifying zero-day invasions. Federated learning can **increase the precision** of zero-day attack detection models by utilizing various data from various devices, especially compared to traditional approaches that rely on centralized data sources. Lastly, the federated learning solution **provides quicker responses** for identifying IoT device zero-day threats. The models can respond to new zero-day attacks faster than traditional approaches since they are trained locally on the devices, unlike those methods that rely on central servers for model updates.

1.5 Benefits of the project towards IoT device companies and users

A zero-day attack is a security vulnerability unknown to the public and for which no patch or fix is available. Hackers can exploit these vulnerabilities to access devices, steal sensitive information, or cause other damage. **IoT device companies** and **users** are the two main groups to benefit from this project.

For **IoT device companies**, detecting zero-day attacks rapidly is crucial for protecting their products and reputation. If a zero-day attack is discovered and exploited, it can lead to widespread damage and loss of consumer trust. By quickly detecting and addressing zero-day attacks, IoT device companies can minimize the potential impact of such attacks and prevent further exploitation of the vulnerability. In addition, detecting and responding to zero-day attacks faster than competitors can give IoT device companies a competitive advantage in the market. Companies known for producing highly secure IoT devices are more likely to attract consumers who prioritize security and privacy.

For **IoT device users**, the benefits of rapid zero-day attack detection are also significant. Users who own IoT devices vulnerable to such attacks risk having their personal information stolen, hijacked, or even becoming victims of cyberattacks. Rapid detection and response to zero-day attacks can help minimize these risks and protect users' privacy and security.

1.6 Report Organization

The report is structured into seven chapters, each dedicated to specific facets of the project. Chapter 1, Introduction, lays the groundwork by articulating the problem statement, research objectives, and project scope, while also elucidating the significance of the project for IoT device companies and users. Chapter 2, Literature Review, offers an in-depth exploration of zero-day attacks, previous works related to the project, datasets employed, and evaluation metrics utilized. In Chapter 3, System Model, I introduce the zero-day attack detection model and the federated learning framework. Chapter 4, System Design, provides a detailed examination of the model flow and the design of the federated learning framework. Chapter 5, Experiment/Simulation, outlines the hardware and software setup, the federated learning framework simulation, implementation challenges, and concluding remarks. In Chapter 6, System Evaluation and Discussion, I delve into the testing setup, model evaluation, federated learning framework performance, challenges encountered, and the evaluation of project objectives. Finally, Chapter 7, Conclusion and Recommendation, concludes the report with closing remarks and suggestions for future endeavours.

CHAPTER 2 LITERATURE REVIEW

In this chapter, I will discuss the related works of detecting zero-day attacks on IoT devices using a federated learning-based solution. At the end of this chapter, I will provide a summary table comparing the strengths and limitations of each proposed solution.

2.1 What is Zero-day attacks and zero-day attacks in IOT security?

To further explore the definition of zero-day attacks and zero-day attacks in IoT security, I investigated and explored some research papers.

Zero-day attacks are a type of cyber-attack that take advantages of a previously unknown vulnerability, making it hard for the security professionals to defend against them. According to article in [8], zero-day attacks are generally considered to be a new vulnerability with no defence, and thus they pose a high-risk probability and critical impact.

In the context of IoT networks, zero-day attacks can be particularly devastating due to the large number of connected devices and the potential for widespread damage. There are several frameworks and strategies proposed to mitigate the risks of zero-day attacks in IoT networks, including centralized deep learning [11], and hybrid deep learning [12].

2.2 Previous or Similar works of this project

2.2.1 Centralized Deep Learning (CDL) Model

Centralized deep learning model method is one of the methods that have been proposed previously to detect zero-day attacks on IoT devices. A centralized deep learning model involves training a single deep learning model on a central server using a large amount of data. This approach is also known as the "cloud-based" approach because the data is typically stored in a centralized cloud-based infrastructure.

There are quite a number of previous works that are related to this project using centralized deep learning model method. One of the previously proposed works that used centralized deep learning model method related to the topic is in [9], Ge *et al.* proposed approach for detecting intrusions in Internet of Things (IoT) devices using a customized deep learning technique. The proposed approach uses a feed-forward neural network model with embedding layers to encode high-dimensional categorical features for multi-class classification.

Next, in [10], Apruzzese *et al.* introduces a framework that can protect botnet detectors from adversarial attacks using deep reinforcement learning mechanisms. The proposed framework generates realistic attack samples that can evade detection and uses these samples to produce an augmented training set for producing hardened detectors. This results in more resilient detectors that can work even against unforeseen evasion attacks without penalizing their performance in the absence of specific attacks.

Subsequently, in [11], Popoola *et al.* proposed a deep learning-based algorithm to detect botnet attacks in IoT networks, which can handle highly imbalanced network traffic data. The algorithm uses Synthetic Minority Oversampling Technique (SMOTE) to generate additional minority samples and Deep Recurrent Neural Network (DRNN) to learn hierarchical feature representations from the balanced network traffic data for discriminative classification. Figure 3 shows the flow chat of SMOTE-DRNN for botnet attack detection in IoT networks while Figure 4 shows the SMOTE algorithm.

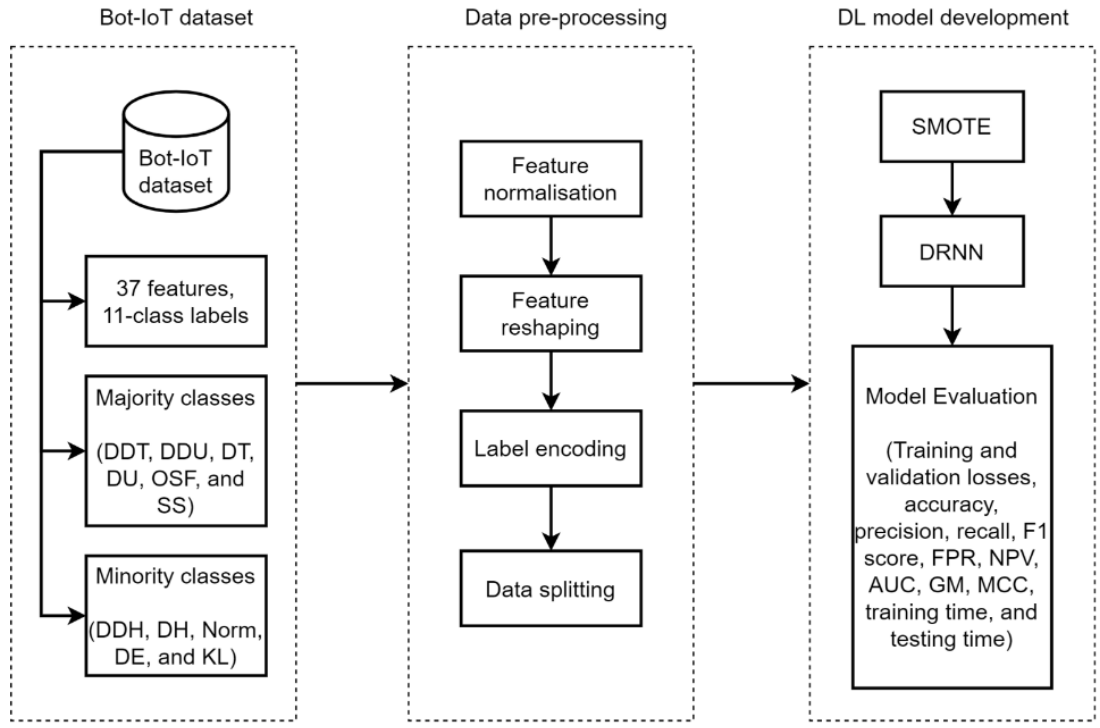


Figure 3 SMOTE-DRNN for botnet attack detection in IoT networks

Input: T, N, k
Output: S
Initialization: $k = 3, q = 37, r = 0$

```

1 if  $N < 100$  then
2   Randomise the  $T$  minority class samples
3    $S = (N/100) \times T$ 
4    $N = 100$ 
5 end
6  $N = (j)(N/100)$ 
7 for  $i = 1$  to  $T$  do
8   Compute  $k$  nearest neighbours for  $i$ 
9   while  $N \neq 0$  do
10     $nn = \text{random}(1, k)$ 
11    for  $c = 1$  to  $q$  do
12       $f = \mathbf{Z}[nn\_array[nn]][c] - \mathbf{Z}[i][c]$ 
13       $g = \text{random}(0, 1)$ 
14       $\mathbf{V}[r][c] = \mathbf{Z}[i][c] + (g \times f)$ 
15    end
16     $r = r + 1$ 
17     $N = N - 1$ 
18  end
19 end
  
```

Figure 4 SMOTE Algorithm

CHAPTER 2

However, in all these proposed methods, there is a common challenge of dealing with high feature dimensionality in the training data, which results in high network bandwidth and large memory space requirements for transmitting and storing the data, respectively. Additionally, since the centralized deep learning model required to send data back to the global server, it raises concerns about the data privacy issue. In these model, sensitive data from IoT devices are transmitted to the cloud platform for analysis, making them susceptible to interception by hackers or unauthorized third parties. Even if the data is encrypted, it can still be vulnerable to attacks during transmission.

2.2.2 Hybrid Deep Learning Model

Then, some of the researchers started to investigate on the hybrid methods to detect zero-day attacks on IoT network. In previous research paper [13], Popoola *et al.* further proposed a memory-efficient deep learning to detect botnet attack in IoT networks. The method combines Long Short-Term Memory Autoencoder (LAE), Synthetic Minority Oversampling Technique (SMOTE), and Deep Recurrent Neural Network (DRNN) to reduce feature dimensionality, handle class imbalance, and achieve high classification performance in minority classes. Figure 5 shows the flowchart of the memory-efficient deep learning model.

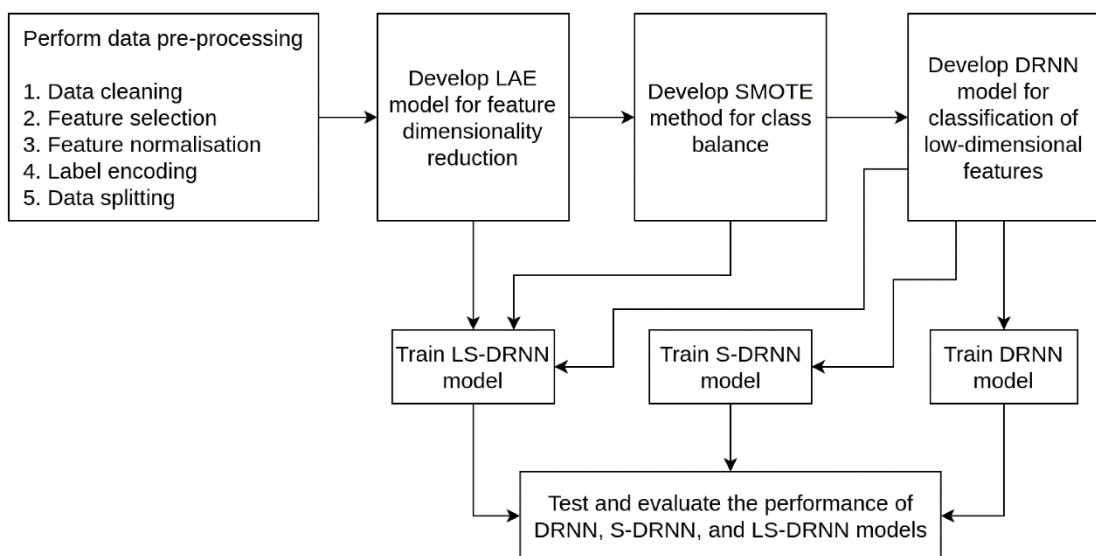


Figure 5 Flow chart of Memory-efficient Deep Learning Model

The benefits of the memory-efficient deep learning model method include reduced memory space requirement, improved classification performance, and faster detection speed. However, the limitation of this method is that it requires significant computational resources for model training. Other than this, in [14], Alkadi *et al.* proposed deep blockchain framework (DBF) that utilizes bidirectional long short-term memory (BiLSTM) deep learning algorithm for intrusion detection and privacy-based blockchain with smart contracts for data privacy in IoT networks. The proposed method has several benefits, especially on the data privacy in IoT networks. Figure 6 shows how the proposed collaborative intrusion detection based BiLSTM works.

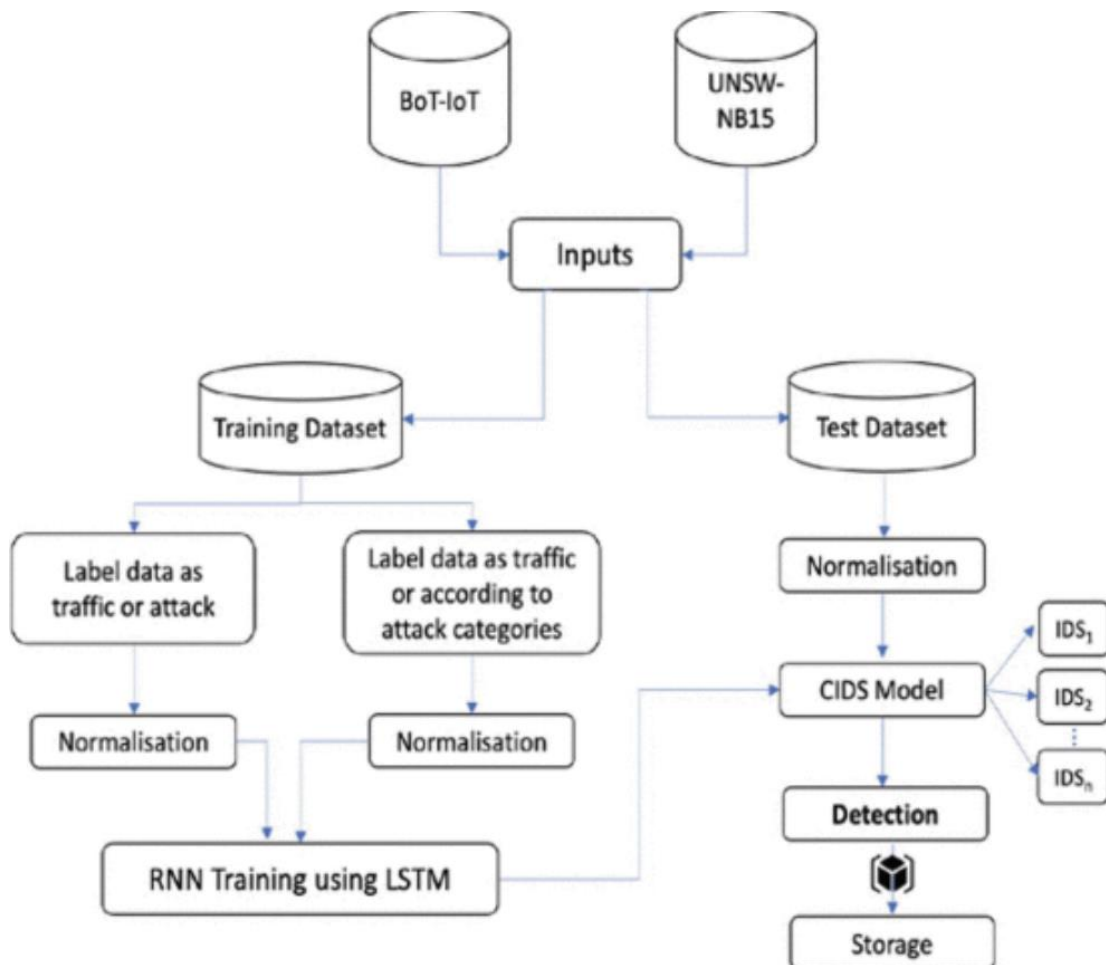


Figure 6 Proposed collaborative intrusion detection based BiLSTM

However, the proposed method is not aimed to detect zero-day attacks on IoT devices and requires large amount of data in order to train the model.

2.2.3 Similar Federated Learning model

In this project, I am going to develop a zero-day detection on IoT devices using federated learning model method. Thereby, I am going to further study the application of federated learning model on other problem that is similar to zero-day detection on IoT devices [15] – [21]. For example, in [15], Chen *et al.* introduced the Federated Learning-based Attention Gated Recurrent Unit (FedAGRU), which is an intrusion detection algorithm for wireless edge networks. FedAGRU utilizes the attention mechanism to increase the weight of important devices and reduce communication overhead. The model updates universal learning models rather than directly sharing raw data among edge devices and a central server. Figure 7 shows how the proposed solution in [15] works.

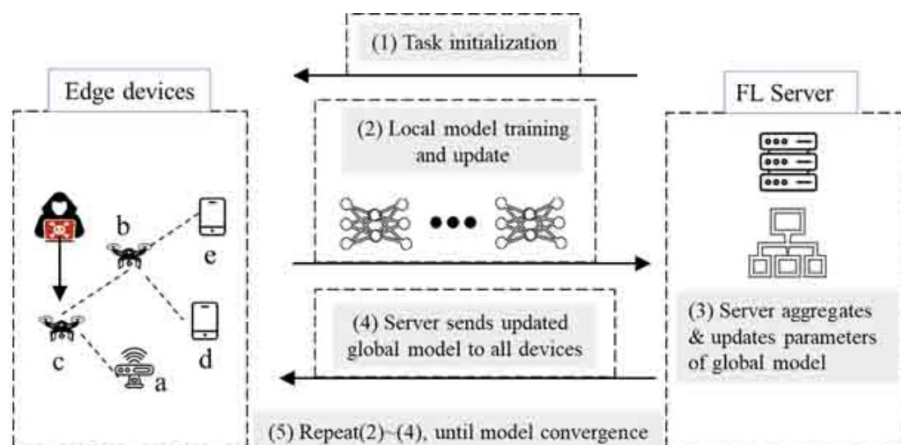


Figure 7 Intrusion Detection for Wireless Edge Networks Based on Federated Learning

From this proposed solution, I observe that if I use federated learning, I am able to directly share raw data among edge devices and a central server. It ensures the privacy of data and avoid the transmission of sensitive information go through the Internet. Besides, through federated learning, it can greatly reduce communication overhead as well as ensures the learning process converges effectively. Our idea is the same in term of using federated learning, but the proposed solution is not aimed at detecting zero-day attacks on IoT devices.

Subsequently, other than federated learning method, I also need to further explore to the deep learning model architectures that used in the federated learning method. In [15], Chen et al. employed a combined deep learning model architecture called "GRU-SVM" for their intrusion detection system. This architecture integrates the GRU (gated recurrent unit) model for processing time-series network traffic data, which features reset and update gates to manage information flow and memory retention over different time steps. Additionally, they use an SVM (Support Vector Machine) as the final layer for classification, replacing the SoftMax activation function typically used in deep learning models. The SVM is trained with Hinge Loss as the objective function to make classification predictions. This hybrid architecture leverages the strengths of GRU for sequence data analysis and SVM for robust classification, resulting in an effective intrusion detection model.

Apart from that, in [16], Weinger et al. employed Generative Adversarial Networks (GANs) to tackle data augmentation challenges in their study, focusing on anomaly detection tasks with IoT datasets. GANs consist of a generator network that creates realistic samples and a discriminator network that identifies positive samples. These networks iteratively enhance each other's performance during training. In their case, the generator was utilized to generate anomalies, effectively increasing the number of positive samples, and achieving class balance. However, applying GANs to Federated Learning (FL) posed challenges due to FL's privacy constraints, which prevent the centralization of all data for training.

While in [17], Al-Marri et al. utilized a feedforward Multilayer Perceptron (MLP) neural network architecture for their Intrusion Detection System (IDS). This neural network model consists of two hidden layers, each containing 256 neural units and employs the Rectified Linear Unit (ReLU) activation function. To mitigate overfitting, they incorporated dropout layers with a dropout rate of 0.4 after each hidden layer. This architecture was used to achieve high detection accuracy in their federated mimic learning-based IDS approach, combining the strengths of Federated Learning (FL) and mimic learning to enhance privacy protection while maintaining robust intrusion detection performance. However, the proposed federated mimic learning approach may introduce additional computational complexity and communication overhead compared to traditional centralized IDS techniques.

Furthermore, in [18], Mothukuri et al. employed deep learning models, specifically Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRUs), as the core neural network architectures in their proposed approach for IoT intrusion detection. Both LSTM and GRU models were utilized to handle the temporal aspects of the IoT data effectively. In [19], Fan et al. employed a deep learning model architecture that includes convolutional layers (CNN) as part of their IoTDefender framework for 5G IoT intrusion detection. While IoTDefender showed promising results in improving intrusion detection for 5G IoT networks, a limitation is that it relies on simulated data, and its real-world applicability and robustness need further validation with live IoT device data to assess its performance in detecting both known and unknown vulnerabilities.

In summary, the reviewed literature has demonstrated the effectiveness of various machine learning and deep learning models, often employed within the context of federated learning, for enhancing intrusion detection in IoT networks. These models have shown promise in identifying known attack patterns and anomalies, contributing significantly to the privacy and security of IoT devices. However, it is important to note that their primary objective has not been to address the detection of zero-day attacks, which represent novel and previously unseen threats in IoT networks. The challenge of zero-day attack detection remains a complex and evolving issue in the realm of cybersecurity. As I move forward in this study, I aim to explore and develop novel approaches that can effectively target zero-day attacks while preserving the privacy and security of IoT networks through federated learning.

Table 1 shows the table of comparison between the previous works and the proposed solution.

Comparison \ Model	CDL Model (SMOTE-DRNN)	Hybrid DL Model		Other FDL Model	Proposed FL Model
		Memory-efficient Hybrid DL Model	DBF + BiLSTM DL algorithm		
Require Small Amount of Training Data		✓		✓	✓
No Privacy Issue			✓	✓	✓
Low Network Bandwidth		✓		✓	✓
Used As Intrusion detection			✓	✓	
Focus on Zero-day Detection	✓	✓			✓

Table 1 Table of Comparison Between Previous Works and Proposed Solution

Table 2 below shows the comparison between the deep learning model architecture used in the related federated learning method.

	[15]	[16]	[17]	[18]	[19]
Model/Framework Used	GANs	GRU and SVM	Multilayer Perceptron (MLP) neural network architecture	LSTM networks and GRUs	CNN
Main Contribution	Efficacy of preprocessing techniques in federated learning for anomaly detection in IoT	Intrusion detection framework based on federated learning for wireless edge networks	Federated mimic learning for privacy in intrusion detection systems for IoT	Federated learning-based approach for IoT security attacks	First federated transfer learning framework for IDS of 5G IoT
Strength	Use of federated learning and GANs for data augmentation	Use of federated learning offers privacy benefits; use of GRU provides advantages in processing time series data	Ensures user privacy by avoiding transfer of user data to a centralized server	Design architecture for migrating non-FL-based approaches to federated learning	Strong generalization ability; can detect unknown attacks effectively

Limitations	Low proportion of anomalies in the dataset can be a major limitation for a centralized classifier	Limitations of traditional RNNs; specific limitations of proposed method not explicitly mentioned	IoT devices are prone to attacks due to the limitations of their privacy and security components	Evaluation with virtual instances that might not reflect real-world IoT scenarios completely	Specific limitations not directly mentioned
Focus Area	Not focus on Zero-day Detection				

Table 2 Table of Comparison between Deep Learning Model Architecture used in Federated Learning Method

2.3 Datasets type

There are quite a number of datasets type available for this project. For example, the Ton-IoT dataset, the IoT-23 dataset, Bot-IoT dataset and the N-BaloT dataset are the widely used datasets to analyze cybersecurity problems in IoT networks.

First of all, The TON_IoT datasets represent a cutting-edge advancement in Industry 4.0 and Internet of Things (IoT) cybersecurity, providing a comprehensive array of data sources for evaluating the efficacy of Artificial Intelligence (AI) and Machine/Deep Learning algorithms. Developed at UNSW Canberra at the Australian Defence Force Academy, these datasets encompass a diverse range of telemetry data from IoT and IIoT sensors, network traffic logs, and operating system traces (Windows 7, Windows 10, Ubuntu 14, Ubuntu 18 TLS). They offer a unique blend of normal and cyber-attack events, including DoS, DDoS, and ransomware attacks, captured from a sophisticated testbed network comprising IoT, Cloud, and Edge/Fog systems. Researchers can leverage the TON_IoT datasets to validate and enhance various cybersecurity applications such as intrusion detection systems, threat intelligence, fraud detection, and more. Detailed in several academic papers, these datasets are freely available for academic research with citations required, offering a wealth of possibilities for advancing AI-based security systems in the IoT landscape [22]. Table 3 – 8 below shows all the features in TON-IoT dataset.

Service: Connection activity		
ID	Feature	Description
1	<i>ts</i>	Timestamp of connection between flow identifiers
2	<i>src_ip</i>	Source IP addresses which originate endpoints' IP addresses
3	<i>src_port</i>	Source port which originate endpoint's TCP/UDP ports
4	<i>dst_ip</i>	Destination IP addresses which respond to endpoint's IP addresses
5	<i>dst_port</i>	Destination ports which respond to endpoint's TCP/UDP ports
6	<i>proto</i>	Transport layer protocols of flow connections
7	<i>service</i>	Dynamically detected protocols, such as DNS, HTTP and SSL
8	<i>duration</i>	The time of the packet connections, which is estimated by subtracting 'time of last packet seen' and 'time of first packet seen'
9	<i>src_bytes</i>	Source bytes which are originated payload bytes of TCP sequence numbers
10	<i>dst_bytes</i>	Destination bytes which are responded payload bytes from TCP sequence numbers
11	<i>conn_state</i>	Various connection states, such as S0 (connection without replay), S1 (connection established), and REJ (connection rejected)
12	<i>missed_bytes</i>	Number of missing bytes in content gaps

Table 3 Connection Activity Features of TON-IoT dataset

Service: Statistical activity		
ID	Feature	Description
13	<i>src_pkts</i>	Number of original packets which is estimated from source systems
14	<i>src_ip_bytes</i>	Number of original IP bytes which is the total length of IP header field of source systems
15	<i>dst_pkts</i>	Number of destination packets which is estimated from destination systems
16	<i>dst_ip_bytes</i>	Number of destination IP bytes which is the total length of IP header field of destination systems

Table 4 Statistical Activity Features of TON-IoT dataset

Service: DNS activity		
ID	Feature	Description
17	<i>dns_query</i>	Domain name subjects of the DNS queries
18	<i>dns_qclass</i>	Value which specifies the DNS query classes
19	<i>dns_qtype</i>	value which specifies the DNS query types
20	<i>dns_rcode</i>	Response code values in the DNS response
21	<i>dns_AA</i>	Authoritative answers of DNS, where T denotes server is authoritative for query
22	<i>dns_RD</i>	Recursion desired of DNS, where T denotes request recursive lookup of query
23	<i>dns_RA</i>	Recursion available of DNS, where T denotes server supports recursive queries
24	<i>dns_rejected</i>	DNS rejection, where DNS queries are rejected by the server

Table 5 DNS Activity Features of TON-IoT dataset

Service: SSL activity		
ID	Feature	Description
25	<i>ssl_version</i>	SSL version which is offered by the server
26	<i>ssl_cipher</i>	SSL cipher suite which the server chose
27	<i>ssl_resumed</i>	SSL flag indicates the session that can be used to initiate new connections, where T refers to the SSL connection is initiated
28	<i>ssl_established</i>	SSL flag indicates establishing connection between two parties, where T refers to establishing the connection
29	<i>ssl_subject</i>	Subject of the X.509 cert offered by the server
30	<i>ssl_issuer</i>	Trusted owner/originator of the SLL and digital certificate (certificate authority)

Table 6 SSL Activity Features of TON-IoT dataset

Service: HTTP activity		
ID	Feature	Description
31	<i>http_trans_depth</i>	Pipelined depth into the HTTP connection
32	<i>http_method</i>	HTTP request methods such as GET, POST and HEAD
33	<i>http_uri</i>	URIs used in the HTTP request
34	<i>http_version</i>	The HTTP version utilised such as V1.1
35	<i>http_request_body_len</i>	Actual uncompressed content sizes of the data transferred from the HTTP client
36	<i>http_response_body_len</i>	Actual uncompressed content sizes of the data transferred from the HTTP server
37	<i>http_status_code</i>	Status codes returned by the HTTP server
38	<i>http_user_agent</i>	Values of the User-Agent header in the HTTP protocol
39	<i>http_orig_mime_types</i>	Ordered vectors of mime types from source system in the HTTP protocol
40	<i>http_resp_mime_types</i>	Ordered version of mime types from destination system in the HTTP protocol

Table 7 HTTP Activity Features of TON-IoT dataset

Service: Data labeling		
ID	Feature	Description
44	<i>label</i>	Tag normal and attack records, where 0 indicates normal and 1 indicates attacks
45	<i>type</i>	Tag attack categories, such as normal, DoS, DDoS, and backdoor attacks, and normal records

Table 8 Data Labelling of TON-IoT dataset

Next, the IoT-23 dataset is a dataset to detect intrusion on IoT networks. It is designed to help researchers to evaluate and compare the performance of IDS in IoT networks. The dataset contains various types of traffic and attack scenarios, such as botnet, DDos, reconnaissance, injection, and firmware attacks. The dataset has been pre-processed to select, extract, and normalize the features. It contains a total of 115 features which include network packet header information, such as source IP, destination IP, ports and etc. However, the IoT-23 dataset has its own limitations as well. It was generated in a controlled lab environment, which may not accurately reflect real-world IoT networks. Moreover, the dataset does not include all kinds of attacks that could occur in real-world scenarios [23] – [24]. Table 9 shows the list of example features in the IoT-23 dataset.

Attribute Number	Features	Description
1	fields-ts	Flow start time
2	uid	Unique ID
3	id.orig-h	Source IP address
4	id.orig-p	Source port
5	id.resp-h	Destination IP address
6	id.resp-p	Destination port
7	proto	Transaction protocol
8	service	http, ftp, smtp, ssh, dns, etc.
9	duration	Record total duration
10	orig-bytes	Source2destination transaction bytes
11	resp-bytes	Destination2source transaction bytes
12	conn-state	Connection state
13	local-orig	Source local address
14	local-resp	Destination local address
15	missed-bytes	Missing bytes during transaction
16	history orig-pkts	History of source packets
17	orig-ip-bytes	Flow of source bytes
18	resp-pkts	Destination packets
19	resp-ip-bytes	Flow of destination bytes
20	tunnel-parents	Traffic tunnel
21	label	Attack label

Table 9 Features of IoT-23 dataset

Subsequently, the Bot-IoT dataset is a valuable resource in the field of Internet of Things (IoT) security research. It comprises network traffic data that encapsulates the complex interactions between IoT devices and their associated networks. This dataset is particularly unique as it encompasses both legitimate IoT device traffic and malicious traffic generated by IoT devices infected with various types of malware and bots. Researchers and cybersecurity professionals use the Bot-IoT dataset to develop and assess intrusion detection systems and security algorithms tailored specifically for IoT environments. By examining this dataset, experts gain critical insights into the distinctive patterns of behaviour exhibited by compromised IoT devices, ultimately contributing to the enhancement of IoT security measures and safeguarding the integrity and functionality of IoT ecosystems.

Feature	Description
pkSeqID	Row Identifier
Stime	Record start time
flgs	Flow state flags seen in transactions
flgs_number	Numerical representation of feature flags
Proto	Textual representation of transaction protocols present in network flow
proto_number	Numerical representation of feature proto
saddr	Source IP address
sport	Source port number
daddr	Destination IP address
dport	Destination port number
pkts	Total count of packets in transaction
bytes	Total number of bytes in transaction
state	Transaction state
state_number	Numerical representation of feature state
ltime	Record last time
seq	Argus sequence number
dur	Record total duration
mean	Average duration of aggregated records
stddev	Standard deviation of aggregated records
sum	Total duration of aggregated records
min	Minimum duration of aggregated records
max	Maximum duration of aggregated records
spkts	Source-to-destination packet count
dpkts	Destination-to-source packet count
sbytes	Source-to-destination byte count
dbytes	Destination-to-source byte count
rate	Total packets per second in transaction
srate	Source-to-destination packets per second
drate	Destination-to-source packets per second
attack	Class label: 0 for Normal traffic, 1 for Attack Traffic
category	Traffic category
subcategory	Traffic subcategory

Table 10 Features of Bot-IoT dataset [25]

Lastly, the N-BaloT dataset is a dataset that is specifically designed to evaluate the network-based intrusion detection systems in IoT environments. In this dataset, it consists of network traffic collected from a realistic IoT testbed that includes a range of IoT devices such as sensors. The dataset is intended to be used to evaluate model in detecting 9 different attacks scenarios on IoT networks, including the DDoS, malware infections, and etc. However, there are also limitations for this N-BaloT dataset. One of the limitations is that it is relatively small compared to other IoT datasets, which may limit its usefulness in certain types of research [26].

Aggregated by	Value	Statistic	Total No. of Features
Source IP	Packet size (only outbound)	Mean, variance	3
	Packet count	Integer	
Source MAC-IP	Packet size (only outbound)	Mean, variance	3
	Packet count	Integer	
Channel	Packet size (only outbound)	Mean, variance	10
	Packet count	Integer	
	Amount of time between packet arrivals	Mean, variance, integer	
	Packet size (both inbound and outbound)	Magnitude, radius, covariance, correlation coefficient	
Socket	Packet size (only outbound)	Mean, variance	7
	Packet count	Integer	
	Packet size (both inbound and outbound)	Magnitude, radius, covariance, correlation coefficient	
Total			23

Table 11 Some Features of N-BaloT dataset [27]

Table 12 below compare the datasets shown above.

Comparison/ Dataset	TON-IoT	IoT-23	BoT-IoT	N-BaloT
Number of Features	45 features	25 features	32 features	115 features
Focus on Attack	DDoS, DoS, Ransomware, Backdoor, Password, Scanning and etc.	Botnet, DDoS, reconnaissance, injection, and firmware attacks	Malware and botnet infections, as well as benign IoT traffic	DDoS, malware infections, and etc.
Limitations	Potential lack of feature standardization across heterogeneous networks, and limited historical context for certain attack simulations.	Generated in a controlled lab environment, may not accurately reflect real-world IoT networks, and does not include all kinds of attacks	May not cover the full spectrum of IoT attacks	Relatively small compared to other IoT datasets

Table 12 Table of Comparison for datasets

2.4 Evaluation metrics

In this project, I need evaluation metrics in order to evaluate the performance of the proposed solution in detecting zero-day attacks on IoT networks. There are 4 evaluation metrics that are suitable to evaluate the performance of the solution in this project, which are accuracy, precision, recall and F1 Score. Figure 8 shows the confusion matrix in order to further explain the evaluation metrics.

		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

Figure 8 Confusion Matrix

Accuracy measures the overall correctness of the algorithm's predictions. It is the ratio of TP and TN to the total number of samples [28]. In the case of zero-day attacks on IoT networks, a high accuracy score would be desirable as it means that the algorithm correctly identifies the attacks while minimizing false positive.

Precision measures the ratio of true positives to the total number of positive predictions (both TP and FP). It represents the algorithm's ability to accurately identify positive results [28]. In the context of detecting zero-day attacks on IoT networks, precision is important because it ensures that the algorithm is not generating a high number of false positives. Figure 8 shows the total predicted positive in the red box which is the sum of true positive and false positive.

		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

Figure 9 Total Predicted Positive

Recall measures the ratio of true positives to the total number of actual positive samples (both TP and FN). It represents the algorithm's ability to identify all positive results [28]. In the context of detecting zero-day attacks on IoT networks, recall is important because it ensures that the algorithm is not missing any actual zero-day attacks. Figure 9 shows the total actual positive in the red box which is the sum of false negative and true positive.

		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

Figure 10 Total Actual Positive

F1 Score is the harmonic mean of precision and recall, taking both metrics into account [29]. It provides a balance between precision and recall and is often used as a single metric to evaluate the performance of a binary classifier [29]. In the case of zero-day attacks on IoT networks, a high F1 score would indicate that the algorithm is performing well in both identifying actual attacks while minimizing false positives.

CHAPTER 3 SYSTEM MODEL

In Chapter 3, I will discuss the zero-day attack detection model as well as the federated learning model.

3.1 2-Layers Zero-Day Attack Classification Model

Zero-day attacks represent one of the most formidable challenges in the cybersecurity industry. Unlike conventional cyber threats that are well-known and characterized, zero-day attacks exploit vulnerabilities that are previously unknown to the defenders. These vulnerabilities, or "zero-day vulnerabilities," refer to security flaws in software or hardware that are exploited by attackers before the developers have had an opportunity to release a patch or fix. As a result, zero-day attacks pose a significant threat to the security and integrity of digital systems, as they often bypass traditional security measures and can inflict substantial damage with little to no warning. Detecting and mitigating zero-day attacks require advanced and adaptive defense mechanisms capable of identifying anomalous behavior and discerning subtle indicators of malicious activity within vast streams of data. In this context, the development of effective zero-day attack detection models represents a critical frontier in cybersecurity research and practice, aiming to bolster the resilience of networks and systems against the ever-evolving threat landscape. Figure below shows the zero-day attack detection model flow, which comprises 2 layers, which the first layer consisting one classification model and the second layer consisting one classification model and one anomaly detection model.

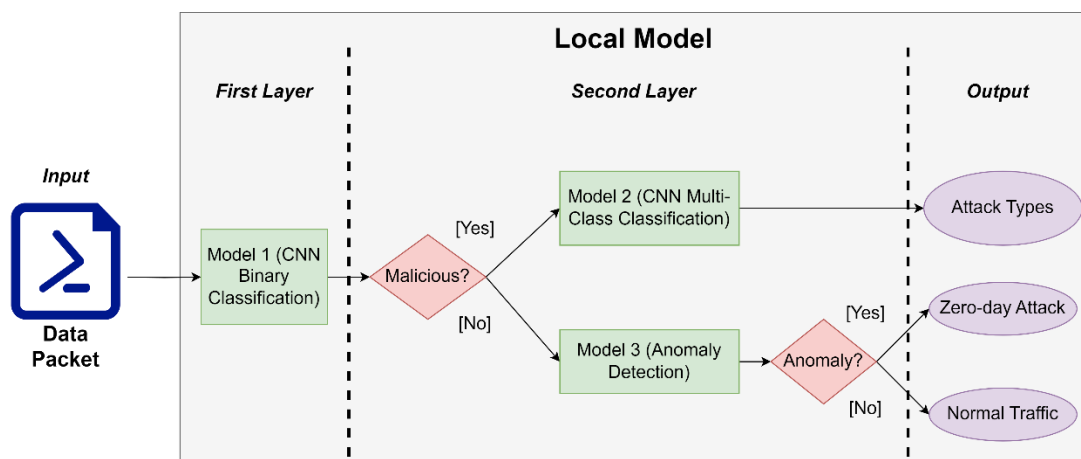


Figure 11 2-Layers Zero-Day Attack Classification Model

CHAPTER 3

This design was chosen to address the challenge of detecting and classifying previously unseen or zero-day attacks in the increasingly complex landscape of IoT (Internet of Things) devices. The first step in the detection process involves passing the Ton-IoT dataset through the initial layer of defense, represented by Model 1, a Convolutional Neural Network (CNN) binary classification model. This model serves as a rapid filter to determine whether incoming traffic exhibits any overt signs of malicious behavior. By employing a CNN architecture tailored to binary classification, it can quickly discern between benign and potentially harmful data patterns. This rapid initial assessment minimizes processing time and computational resources, directing only suspicious instances to subsequent layers for further analysis.

Instances flagged as potentially malicious by Model 1 are then forwarded to Model 2, another CNN model, but this time configured for multi-class classification. The role of Model 2 is to delve deeper into the nature of the detected threats by identifying the specific types of attacks present within the flagged data. By leveraging a multi-class classification approach, this model can categorize incoming data into distinct attack types, providing valuable insights into the methods and strategies employed by potential adversaries.

In cases where Model 1 does not detect any immediate signs of malicious activity, the data is routed to Model 3, an anomaly detection model. This model is designed to scrutinize incoming traffic for deviations from established patterns of normal behavior within the network. By leveraging techniques such as anomaly detection, this model can identify subtle, novel threats that may evade traditional signature-based detection methods. Instances identified as anomalous by Model 3 are then flagged as potential zero-day attacks, prompting further investigation and mitigation measures. The rationale behind this hierarchical design is twofold. Firstly, it optimizes computational resources by prioritizing the analysis of suspicious instances, thereby reducing the processing burden associated with examining the entire dataset exhaustively. Secondly, it leverages the strengths of each model to provide a comprehensive defense mechanism against both known and unknown threats. By combining binary classification, multi-class classification, and anomaly detection, the zero-day attack detection model offers a robust and adaptive approach to safeguarding IoT networks against emerging security threats.

3.2 Federated Learning Framework

In this section, I will discuss the methodology used to perform the experiment in determining the effectiveness of detecting zero-day threat on the IoT devices via federated learning framework. Figure 12 shows the federated learning framework, which is designed to address the challenges of detecting and mitigating zero-day attacks in distributed IoT (Internet of Things) environments.

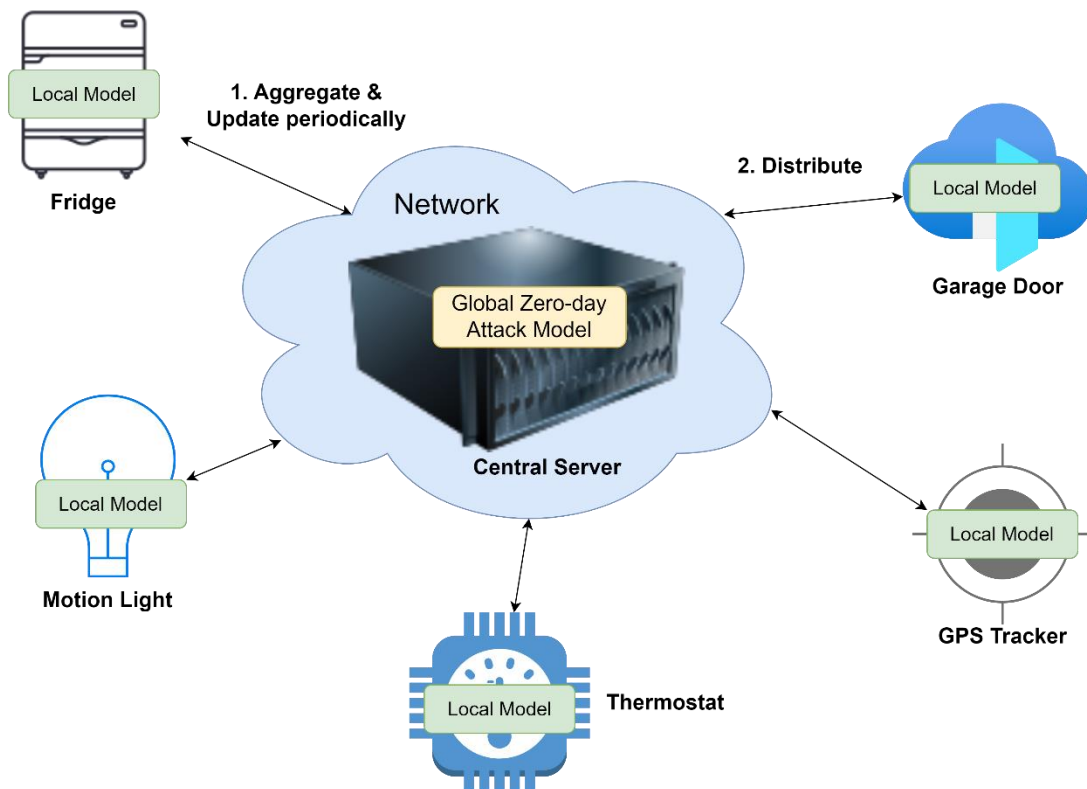


Figure 12 Federated Learning Framework

The framework above leverages the principles of federated learning to enable collaborative model training across multiple edge IoT devices such as Fridge, Motion Light, Thermostat, GPS Tracker, and Garage Door while preserving data privacy and minimizing communication overhead. The main core of the framework lies the concept of local model training on edge IoT devices. Each edge device is equipped with the capability to receive data locally and train its own local zero-day attack detection model. This localized training process ensures that the models are tailored to the specific characteristics and nuances of the data generated by each device, enhancing the effectiveness of detection in diverse and dynamic IoT environments.

Figure 13 shows the traffic flow and mechanism of Edge IoT devices while figure 14 illustrates the role of Central Node.

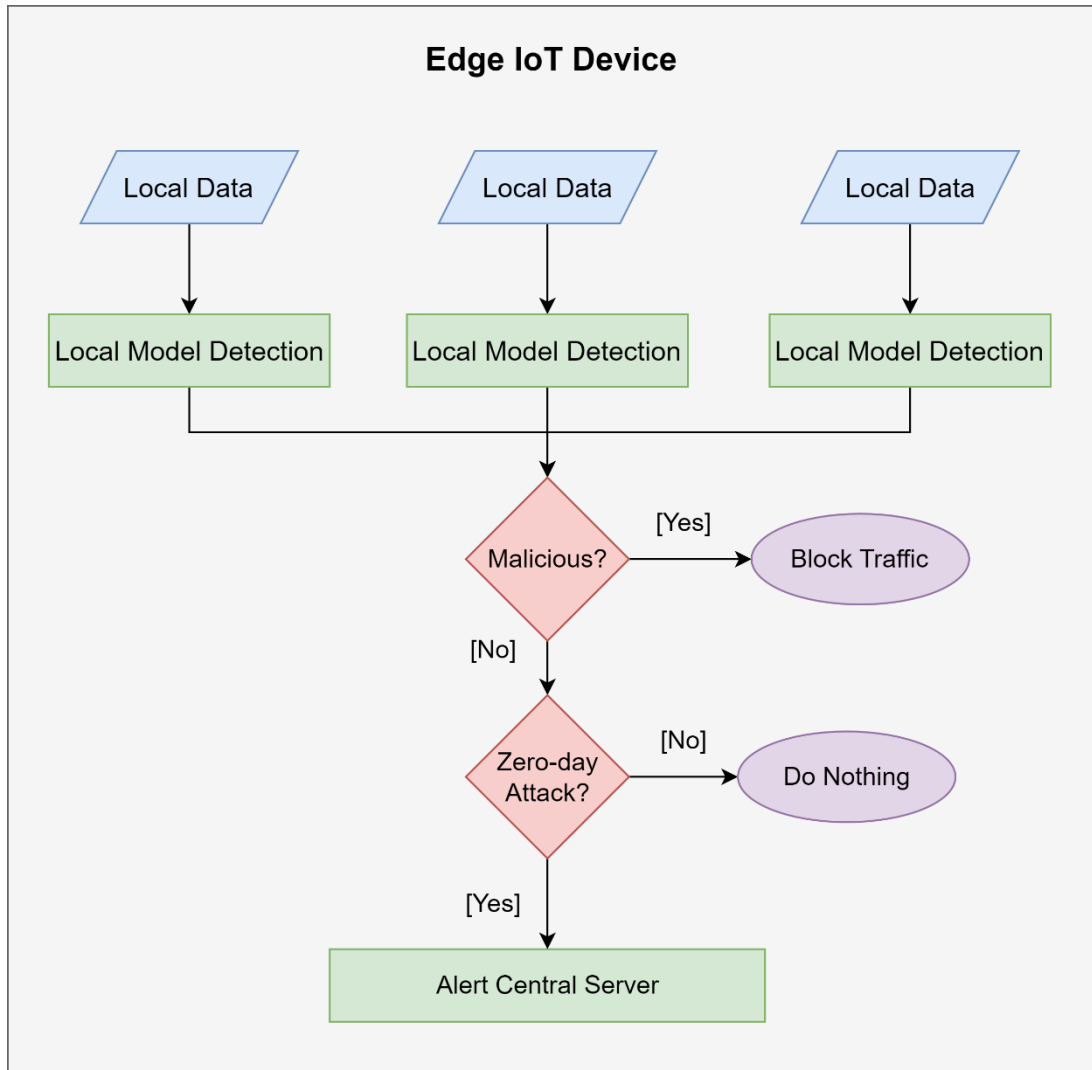


Figure 13 Traffic Flow and Mechanism of Edge IoT Device

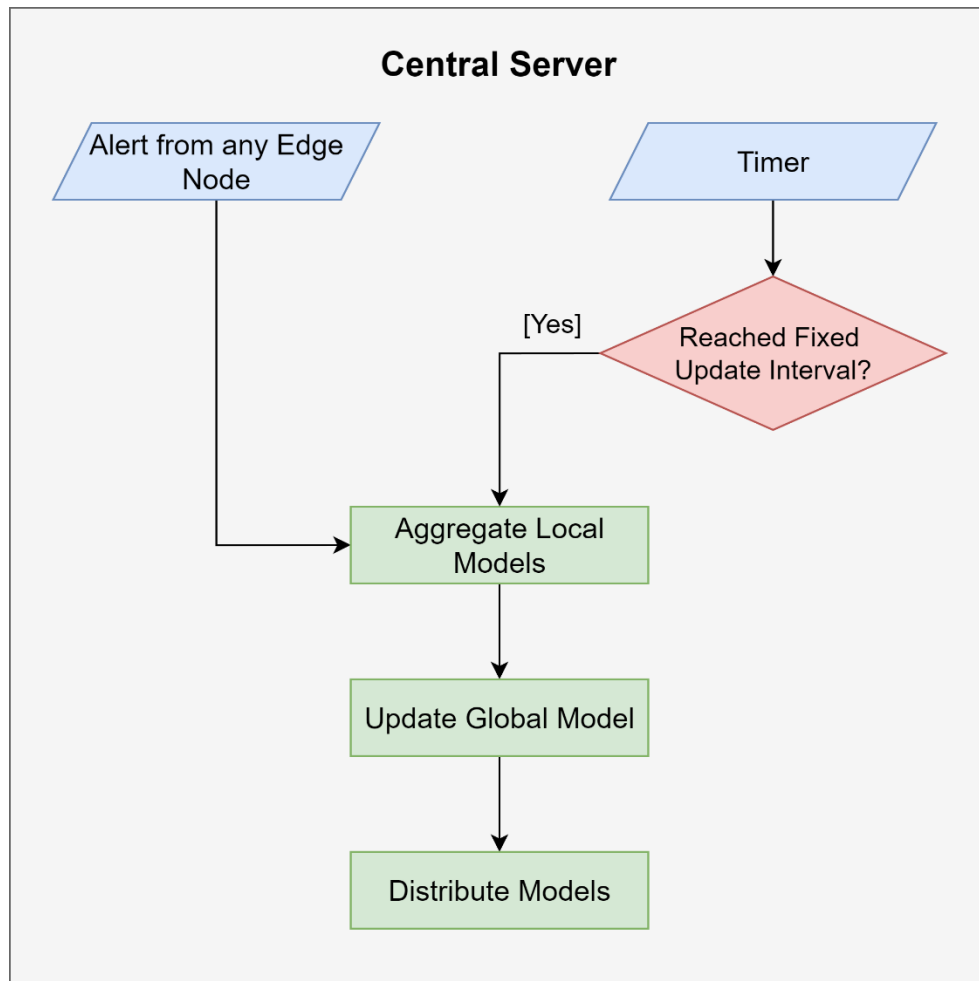


Figure 14 Role of Central Node

Periodically, the edge IoT devices continuously receive local data streams, utilizing their locally trained models to discern whether the incoming traffic exhibits signs of malicious behaviour. Should the local model flag the traffic as malicious, the edge device promptly blocks the suspicious activity, preventing potential threats from infiltrating the network. If the traffic is not identified as malicious, the edge device conducts further analysis to determine if it represents a zero-day attack. In the case of a zero-day attack detection, an alert is swiftly triggered and propagated to the central server for immediate action.

Meanwhile, the central server orchestrates the aggregation, updating, and distribution of global models through a predefined update interval, managed by an internal timer mechanism. At each interval, the central server aggregates the locally trained models received from edge devices, consolidating the insights gleaned from diverse sources. This aggregation process ensures that the global model remains

informed by the collective intelligence of the entire network, capturing nuances and insights from each edge device's unique environment. Upon aggregating the local models, the central server updates the global zero-day attack detection model, incorporating the latest insights and detection capabilities. This update ensures that the global model evolves over time to adapt to emerging threats and changing network conditions. By continuously refining the global model based on real-time data from edge devices, the framework maintains its effectiveness in identifying and mitigating zero-day attacks.

Once the global model is updated, the central server distributes the updated models back to all edge IoT devices. This distribution process ensures that each device benefits from the collective intelligence of the entire network, enabling rapid adaptation to evolving threat landscapes without the need for frequent centralized updates. By disseminating the latest insights and detection capabilities to the edge, this approach empowers individual devices to respond effectively to emerging threats in real-time, thereby enhancing the overall security posture of the network. Figure 15 illustrates zero-day attack happens in the federated learning framework.

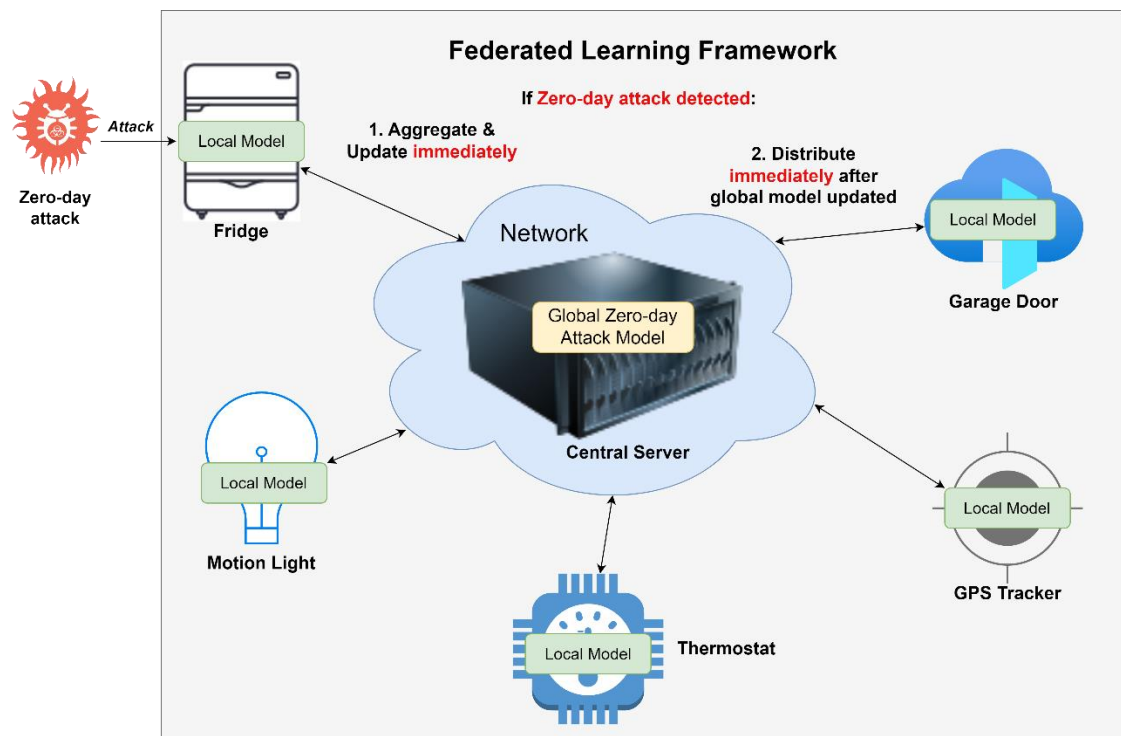


Figure 15 Federated Learning Framework when Zero-day Attack Happens in An Edge Node

CHAPTER 4 SYSTEM DESIGN

4.1 Local & Global Model Design

In this section, I will discuss the model creation of local zero-day attack model before simulating in the federated learning framework. Figure 16 shows the model flow of local zero-day attack detection model.

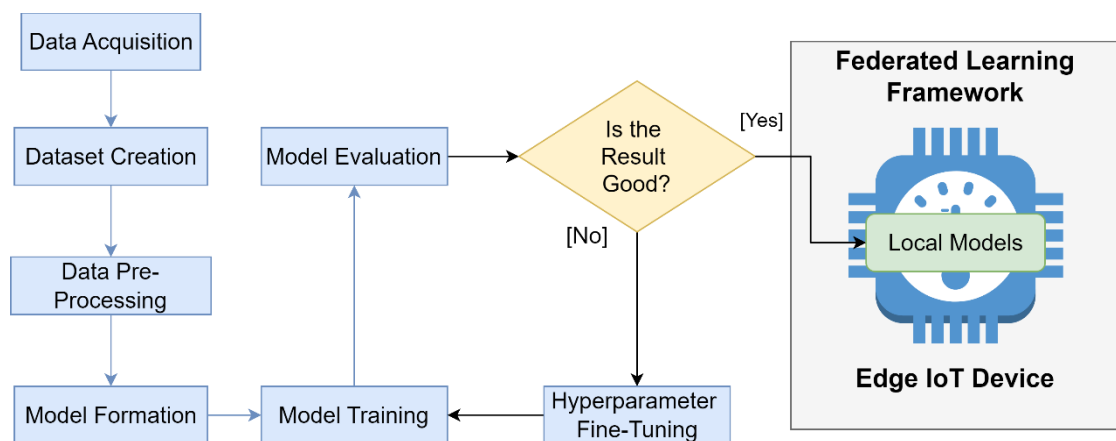


Figure 16 Model Flow of Local Zero-day Attack Model

The model flow of my zero-day attack detection system begins with the crucial stage of data acquisition. In this phase, I look through from various online sources available on the Internet that that related to IoT environment, such as network traffic logs, sensor readings, and system logs. I then analyze the pros and cons of different network IoT environment datasets and choose one of them as my dataset to simulate the experiment. Following data acquisition, I move on to the process of dataset creation. This step involves organizing and structuring the data acquired into three distinct datasets, each tailored to serve the specific requirements of my three different models. These datasets are designed to capture different aspects of the IoT environment, including network behavior, device interactions, and system anomalies.

Once the datasets are created, I proceed to pre-process them to ensure they are in a suitable format for model training. This pre-processing step involves tasks such as feature selection, feature scaling, feature categorical encoding and data splitting, aimed at enhancing the quality and relevance of the data for my models. With pre-processed datasets in hand, I move on to model formation and training. Here, I design and implement three separate models: a CNN binary classification model, a CNN multi-class classification model, and an anomaly detection model. Each model is trained using

CHAPTER 4

its respective dataset, leveraging techniques such as transfer learning and regularization to optimize performance and generalization.

After training my models, I evaluate their performance using appropriate metrics and techniques. If the results are unsatisfactory, indicating suboptimal performance or generalization, I embark on a process of fine-tuning hyperparameters through iterative trial and error. This iterative refinement process aims to optimize model performance and enhance the accuracy of my zero-day attack detection system. Once I achieve satisfactory results through training and fine-tuning, I proceed to model simulation. In this phase, I simulate scenario and test the effectiveness of my detection models in identifying and mitigating zero-day attacks. Through simulation and testing, I validate the robustness and reliability of my system, ensuring its readiness to defend against emerging threats in production environments.

4.1.1 Data Acquisition

In the data acquisition stage of my zero-day attack detection model flow, I select the Ton-IoT dataset as my primary source of data. Developed at UNSW Canberra at the Australian Defence Force Academy, the Ton-IoT dataset represents a cutting-edge advancement in Industry 4.0 and Internet of Things (IoT) cybersecurity. It provides a comprehensive array of telemetry data sourced from IoT and IIoT sensors, network traffic logs, and operating system traces, encompassing a diverse range of normal and cyber-attack events. This dataset offers a unique blend of sophisticated attack scenarios, including DoS, DDoS, ransomware, backdoor intrusions, password attacks, and scanning activities, among others.

One of the key reasons for choosing the Ton-IoT dataset is its extensive feature set, comprising 45 distinct features across various dimensions of IoT network activity. These features are documented in Tables 3 to 8, encompassing connection activity, statistical activity, DNS activity, SSL activity, and HTTP activity features. By leveraging this rich feature set, my model can gain deeper insights into the intricate dynamics of IoT network behavior, facilitating more accurate detection of zero-day attacks. Moreover, the Ton-IoT dataset focuses on a total of 10 common attack types prevalent in IoT environments, providing a comprehensive testbed for evaluating the efficiency of my zero-day attack detection model. By training my model on a dataset that encompasses such a broad spectrum of attack scenarios, I can enhance its robustness and generalization capabilities, enabling it to effectively identify and mitigate emerging threats in real-world IoT deployments.

Lastly, the Ton-IoT dataset is well-known within the cybersecurity research community and has been extensively documented in academic literature. Its focus on IoT environment network activity makes it particularly well-suited for my project's objectives, which aim to develop and evaluate zero-day attack detection mechanisms tailored specifically for IoT ecosystems. In short, the Ton-IoT dataset is the best dataset among all the datasets analyzed, providing a rich and diverse source of data for training, and evaluating my zero-day attack detection model. Its extensive feature set, focus on common IoT attack scenarios, and reputation within the research community make it an ideal choice for advancing my project in IoT cybersecurity.

4.1.2 Dataset Creation

In this section, I will explain the purpose of such dataset creation to achieve the 3 models training in the 2-layer classification local model in an IoT device. Figure 17 below shows the 3 kinds of dataset created with different types.

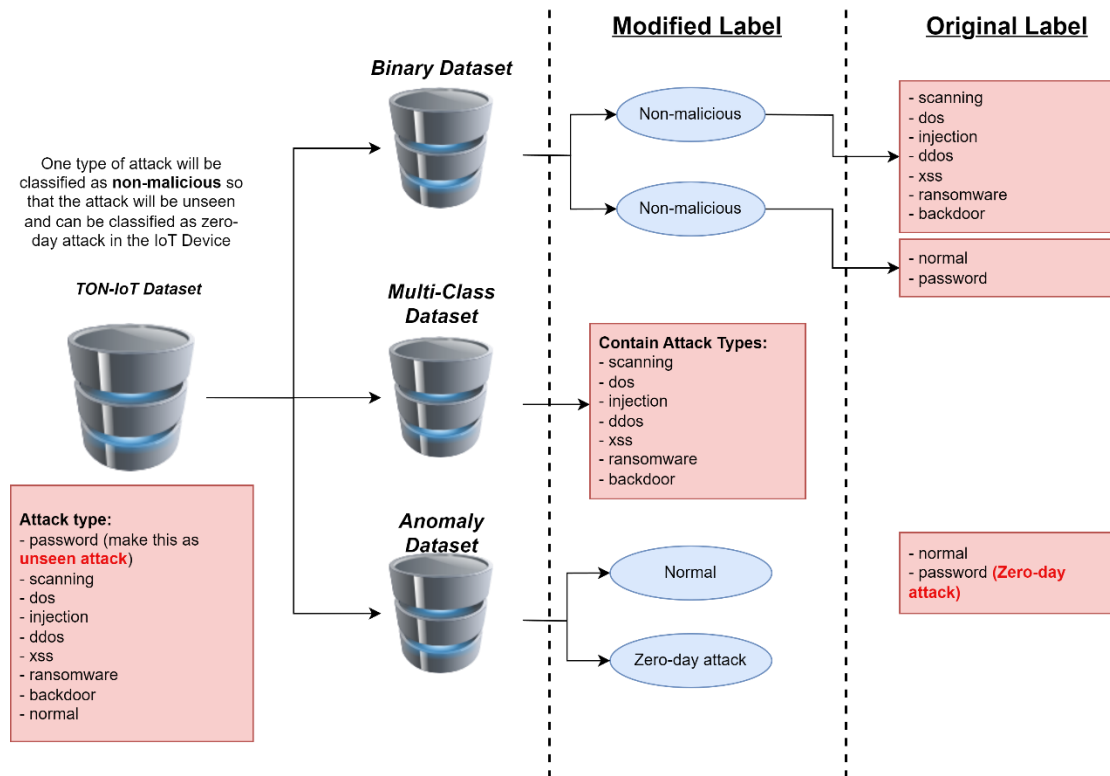


Figure 17 Dataset Creation

In the dataset creation stage, I chose the Ton-IoT dataset to tailor it for my specific modeling requirements. The dataset is initially loaded using the Pandas library, reading the 'TON_IoT_Train_Test_Network.csv' file. To ensure the integrity of my analysis, I filter out any instances labeled as 'mitm', as this attack has a low number of samples that will cause unbalanced in the dataset.

To mimic zero-day attack, I partition the Ton-IoT dataset into three distinct datasets tailored to serve different modeling objectives. Each dataset is crafted to facilitate the training and evaluation of my zero-day attack detection models in IoT environments, leveraging a combination of binary classification, multi-class classification, and anomaly detection techniques. The first dataset I create is a binary classification dataset designed to distinguish between malicious and non-malicious network activity. To simulate the presence of unseen zero-day attacks, I strategically classify one type of attack, specifically 'password', as non-malicious. By doing so, I

CHAPTER 4

ensure that instances of the 'password' attack type are treated as novel and potentially indicative of zero-day attacks in the IoT device. In contrast, instances of other attack types such as scanning, DoS, injection, DDoS, XSS, ransomware, and backdoor are classified as malicious. This binary dataset serves as the foundation for training my zero-day attack detection model on IoT devices, enabling the identification of anomalous network behavior indicative of previously unseen threats.

The second dataset I construct is a multi-class classification dataset focusing solely on malicious network activity. Here, I retain instances labeled with various attack types such as scanning, DoS, injection, DDoS, XSS, ransomware, and backdoor. By concentrating exclusively on malicious activity, this dataset enables me to develop a comprehensive understanding of different attack types prevalent in IoT environments. By training my multi-class classification model on this dataset, I aim to enhance the accuracy and granularity of my zero-day attack detection system, enabling the classification of specific attack types with precision.

Lastly, the third dataset I create is an anomaly detection dataset comprising all instances of non-malicious network activity, including instances labeled as 'normal' and 'password'. Importantly, I designate instances labeled as 'password' as indicative of zero-day attacks, leveraging the element of surprise to mimic the detection of previously unseen threats in the IoT device. By focusing on anomalous network behavior within non-malicious instances, this dataset allows me to identify and flag potential zero-day attacks in real-time, bolstering the security and resilience of IoT networks against emerging threats. Noted that unseen attack (zero-day attack) can be any type of attacks as long as it is labelled as non-malicious in the binary dataset.

4.1.3 Data Pre-Processing

In the data preprocessing process, I undertake a series of steps to transform the raw Ton-IoT dataset into formats suitable for training and evaluating my zero-day attack detection models. This preprocessing pipeline involves feature selection, categorical variable encoding, numerical feature scaling, and dataset splitting for training and testing purposes. Figure 18 below shows the flow of the data pre-processing stage.

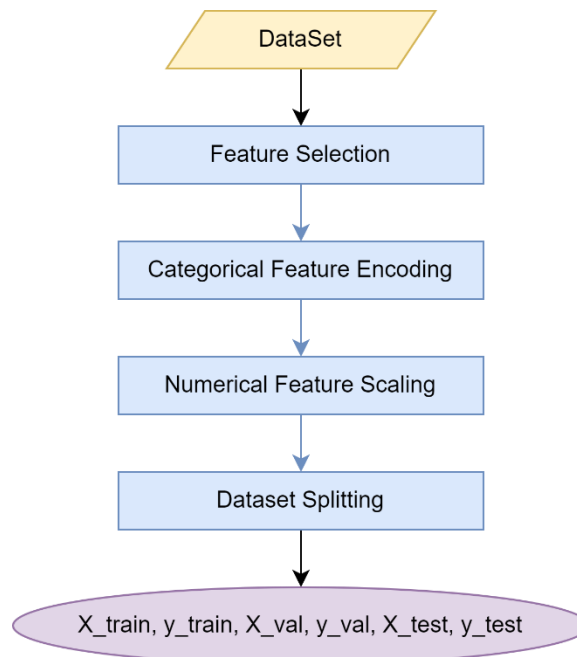


Figure 18 Data Pre-processing Flow

Firstly, I create a function called `preprocess_dataset()` to encapsulate my preprocessing steps. This function accepts the raw dataset as input, makes a copy of it, and then drops unnecessary columns that do not contribute to my modeling objectives. These columns include timestamp (`ts`), source and destination IP addresses (`src_ip`, `dst_ip`), SSL subject and issuer information (`ssl_subject`, `ssl_issuer`), HTTP URI and user agent (`http_uri`, `http_user_agent`), among others. Next, I encode categorical variables using label encoding, transforming categorical columns such as protocol (`proto`), service (`service`), connection state (`conn_state`), and DNS and SSL-related features into numerical representations. This encoding facilitates the incorporation of categorical variables into my machine learning models. Subsequently, I separate the features (`X`) from the target variable (`y`) in the preprocessed dataset. The features comprise all columns except the 'type' column, which represents the attack type. The target variable ('type') is then used to classify instances into different attack categories.

CHAPTER 4

Afterwards, I scale the numerical features using standard scaling to ensure that all features have a comparable scale, preventing certain features from dominating others during model training. With the preprocessed dataset prepared, I split it into training and testing sets using the `train_test_split()` function. This split allows me to evaluate the performance of my models on unseen data, facilitating robust model evaluation and validation. Finally, depending on the specific dataset type (binary, multi-class, or anomaly), I perform additional processing steps tailored to each dataset's requirements. For binary and multi-class datasets, I convert labels into one-hot encoded vectors to facilitate model training. Additionally, for multi-class datasets, I encode the target variable into numerical values using label encoding before converting it into categorical format using one-hot encoding.

4.1.4 Model Formation & Model Training

In the model formation phase, I define the architecture and structure of my convolutional neural network (CNN) models for binary classification, multi-class classification, and anomaly detection. Figure 19 shows the 2-layer Zero-day Detection Flow.

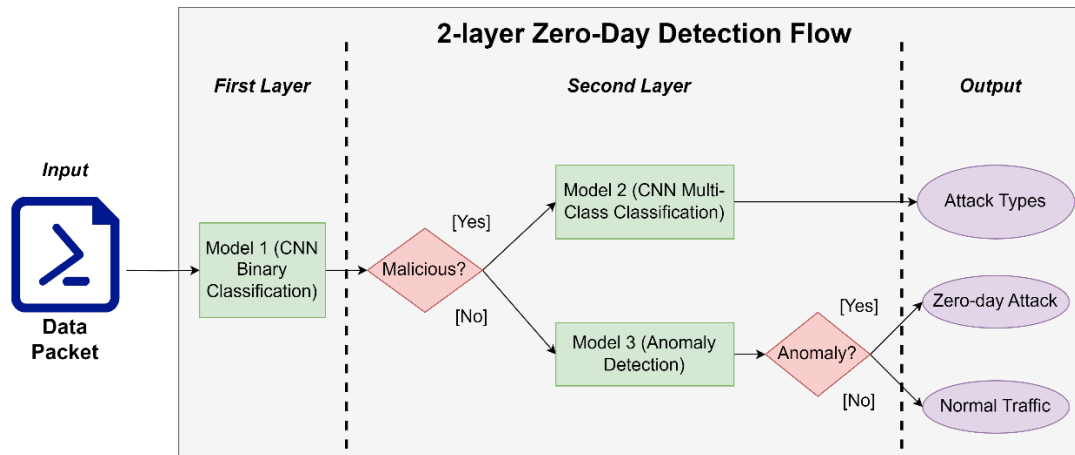


Figure 19 Zero-day Attack Detection Flow

The zero-day attack detection flow within my framework is a multi-layered process designed to effectively identify and mitigate security threats in IoT networks. When incoming traffic is received, it undergoes a series of steps to determine its nature and potential threat level. Firstly, the traffic is routed to the binary classification model, which serves as the initial layer of defense. This model's primary task is to swiftly determine whether the incoming traffic is malicious or non-malicious. By utilizing its classification capabilities, the binary model analyzes the traffic patterns and features to make this determination. If the traffic is classified as malicious, indicating the presence of a potential threat, it is then forwarded to the next layer of the detection process.

In the event that the traffic is identified as malicious by the binary classification model, it proceeds to the multi-class classification model. This secondary layer of defense specializes in discerning the specific type of attack present within the malicious traffic. By leveraging its advanced classification algorithms, the multi-class model thoroughly analyzes the traffic to categorize it into distinct attack types such as DDoS, injection, or ransomware. This granular classification enables my system to tailor its response and mitigation strategies, accordingly, enhancing overall security posture.

CHAPTER 4

Conversely, if the incoming traffic is deemed non-malicious by the binary classification model, it bypasses the multi-class model and proceeds directly to the anomaly detection component. Here, the traffic undergoes rigorous analysis to detect any unusual or anomalous patterns that may indicate the presence of a zero-day attack. By using the Isolation Forest algorithm, this component identifies outliers and deviations from normal behavior, flagging any suspicious traffic for further investigation. If the anomaly detection component identifies the traffic as anomalous, it is flagged as a potential zero-day attack. This triggers an immediate response from my system, enabling proactive measures to be taken to mitigate the threat and safeguard the IoT network. On the other hand, if the traffic is determined to be within normal parameters by the anomaly detection component, it is classified as non-malicious and allowed to proceed through the network without further intervention.

Binary Classification Model

The binary classification model serves as the initial layer of defense in my cybersecurity framework, tasked with determining whether incoming network traffic is malicious or benign. This critical decision-making process forms the cornerstone of my system's ability to detect and mitigate potential security threats in real-time. While various classification algorithms could fulfill this role, I have opted for a Convolutional Neural Network (CNN) model due to its demonstrated high accuracy and efficiency. CNNs are renowned for their capability to extract intricate patterns from data, making them well-suited for discerning subtle distinctions between normal and anomalous network behavior. Moreover, CNNs offer the advantage of rapid training times, enabling my system to adapt swiftly to evolving threat landscapes and provide timely responses to potential security incidents. As the first line of defense, the binary classification model plays a crucial role in safeguarding IoT networks against malicious intrusions, setting the stage for subsequent layers of defense and ensuring comprehensive cybersecurity protection. Figure 20 shows the Network Architecture of CNN Binary Model.

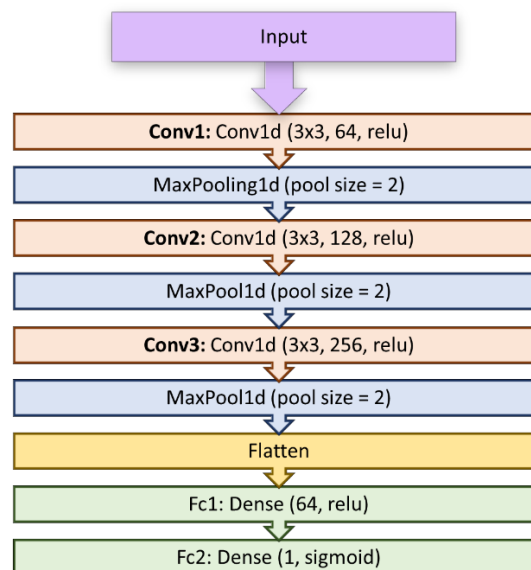


Figure 20 Network Architecture of CNN Binary Model

In my binary classification model, I use Keras framework to construct a sequential neural network architecture. This design allows me to systematically stack layers one after the other, facilitating the flow of data through the model. At the outset, the model is presented with input data comprising features extracted from my binary dataset. These features are reshaped to conform to the input layer's requirements,

CHAPTER 4

ensuring compatibility with subsequent layers. Moving forward, I employ a series of convolutional layers to extract hierarchical representations from the input data. Each convolutional layer consists of filters that scan across the input data, detecting patterns and features relevant to the classification task. As the data passes through successive convolutional layers, the complexity and abstraction of the extracted features increase, enabling the model to extract important features in the input data.

Following each convolutional layer, max-pooling layers are applied to down-sample the feature maps, reducing their spatial dimensions while retaining the most salient information. This process helps in mitigating overfitting and computational complexity while preserving essential features for classification. After multiple iterations of convolution and pooling, the resulting feature maps are flattened into a one-dimensional vector, preparing them for processing by fully connected dense layers. These dense layers serve as the core of the neural network, performing high-level feature fusion and abstraction. By combining information from the flattened feature maps, the dense layers enable the model to capture complex relationships and dependencies within the data. Ultimately, the output of the dense layers is directed to a single neuron with a sigmoid activation function. This neuron serves as the final decision-maker in the binary classification task, producing a probability score indicating the likelihood of the input data belonging to the positive class (malicious) or the negative class (non-malicious). The sigmoid activation function ensures that the output falls within the range of $[0, 1]$, facilitating straightforward interpretation as a probability score.

Multi-Class Classification Model

The multi-class classification model serves as the secondary layer of defense in my cybersecurity framework, tasked with determining the specific type of attack present in network traffic identified as malicious by the preceding binary classification model. This critical analysis enables my system to classify and respond to security incidents with greater granularity, enhancing my ability to mitigate threats effectively. While various classification algorithms could fulfill this role, I have opted for a Convolutional Neural Network (CNN) model due to its demonstrated high accuracy and efficiency. CNNs are renowned for their capability to extract intricate patterns from data, making them well-suited for discerning between different types of cyber threats. Moreover, CNNs offer the advantage of rapid training times, enabling my system to adapt swiftly to emerging attack vectors and provide timely responses to security incidents. As the second layer of protection, the multi-class classification model plays a crucial role in enhancing the sophistication and effectiveness of my cybersecurity defenses, ensuring comprehensive threat detection and mitigation in IoT networks. Figure 21 shows the Network Architecture of CNN Multi-Classes Model.

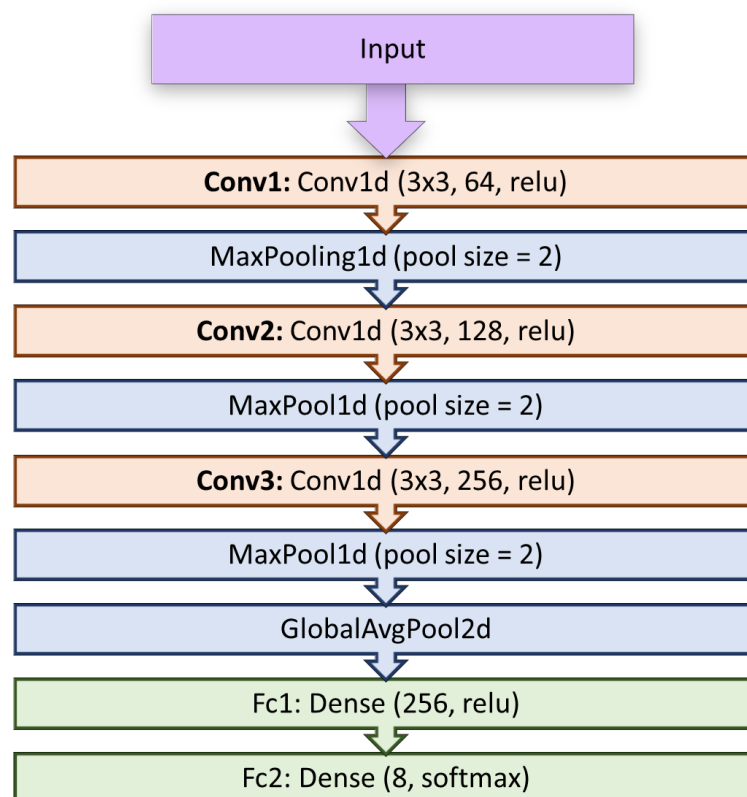


Figure 21 Network Architecture of CNN Multi-Class Model

Similarly, in my multi-class classification model, we utilize the sequential model architecture within the Keras framework, a versatile tool for constructing neural networks. This architecture facilitates the systematic arrangement of layers, allowing for the efficient flow of data through the model. The cores of the model are convolutional layers, which play a pivotal role in extracting hierarchical features from the input data. These layers employ filters that convolve across the input data, detecting patterns and features relevant to identifying different types of cyber-attacks. By applying multiple convolutional layers, the model can progressively learn more abstract and complex representations of the input data, enabling it to discern subtle differences between attack types. Following each convolutional layer, max-pooling layers are incorporated to down-sample the feature maps, reducing their spatial dimensions while retaining the most significant information. This process helps in reducing computational complexity and mitigating overfitting, ensuring that the model focuses on the most salient features for classification.

The key distinction in the multi-class classification model lies in the output layer, which is adapted to accommodate the multi-class nature of the dataset. Unlike the binary classification model, which outputs a single probability score, the multi-class model outputs probabilities across multiple classes corresponding to different attack types. To achieve this, the model concludes with a dense layer with a softmax activation function. The softmax function normalizes the output probabilities across all classes, ensuring that they sum up to one. This enables the model to provide a probability distribution over all possible attack types, allowing for more granular classification and enabling the system to identify the specific type of attack present in the network traffic.

Anomaly Detection Model

In my zero-day attack detection framework, the anomaly detection component plays a crucial role as the second layer of protection for non-malicious traffic. Unlike the CNN models designed for classification tasks, I employ the Isolation Forest algorithm for anomaly detection. Isolation Forest operates as an unsupervised learning algorithm, uniquely suited for identifying anomalies or outliers within data. Its methodology involves isolating instances in the dataset based on their distinct characteristics, making it particularly adept at detecting unusual or unexpected patterns indicative of potential zero-day attacks.

One of the standout features of the Isolation Forest algorithm is its exceptional efficiency in identifying anomalies within extensive datasets. Its ability to rapidly pinpoint rare and distinct anomalies is particularly advantageous in real-world scenarios where timely detection is critical. By efficiently identifying outliers, Isolation Forest plays a key role in transforming the class labels of my IoT dataset into a binary format, categorizing instances as either 'zero-day attacks' or 'normal'. This binary labeling scheme is essential for distinguishing novel, previously unseen security threats from known incidents, thereby enhancing the precision and effectiveness of my anomaly detection system.

Moreover, Isolation Forest's anomaly-centric approach sets it apart from traditional methods by focusing on the intrinsic nature of anomalies rather than specific patterns within data. This unique perspective aligns well with the characteristics of security threats, especially zero-day attacks, which are inherently rare and distinct. The algorithm's robustness to noise and outliers further enhances its reliability in real-world scenarios, while its scalability to high-dimensional datasets makes it suitable for handling complex data sources commonly found in IoT networks.

4.1.5 Model Evaluation

In this stage, I will discuss the model evaluation after the models are trained in the model training stage.

To evaluate the effectiveness of my zero-day attack detection model, I employ a comprehensive set of metrics and techniques to assess its performance across various dimensions. Central to my evaluation are standard metrics such as accuracy, precision, recall, and F1 score, which provide insights into the model's ability to correctly classify different types of traffic and detect zero-day attacks. Accuracy serves as a fundamental measure of the overall correctness of my model's predictions. By calculating the ratio of correctly classified instances to the total number of instances, accuracy offers a broad perspective on the model's performance across all classes. However, in scenarios where class distributions are imbalanced, accuracy alone may not provide a complete picture of the model's effectiveness.

To address this limitation, I complement accuracy with precision, recall, and F1 score. Precision measures the proportion of true positive predictions among all positive predictions, offering insights into the model's ability to minimize false positives. Conversely, recall quantifies the proportion of true positive predictions among all actual positive instances, indicating the model's ability to capture relevant instances of zero-day attacks. The F1 score, a harmonic mean of precision and recall, provides a balanced assessment of the model's performance, particularly useful in scenarios where both false positives and false negatives are critical considerations. Below shows the formulas for the accuracy, precision, recall and F1 Score [30]:

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

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

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

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

CHAPTER 4

, where TP = True Positive,

TN = True Negative,

FP = False Positive,

and FN = False Negative

In addition to these metrics, I utilize classification reports and confusion matrices to gain a deeper understanding of my model's performance across individual classes. Classification reports provide a detailed summary of precision, recall, and F1 score for each class, enabling me to identify strengths and weaknesses in the model's classification abilities across different attack types. Confusion matrices offer a visual representation of the model's predictions compared to ground truth labels, highlighting areas of misclassification, and guiding potential areas for improvement. Table 13 below shows the summary of confusion matrices for binary classification.

		<i>Actual Values</i>	
		<i>Positive</i>	<i>Negative</i>
<i>Predicted Values</i>	Positive	TP	FP
	Negative	FN	TN

Table 13 Confusion Matrix for Binary Classification

Finally, I assess the model's training and validation performance through visualizations of accuracy and loss over epochs. By monitoring training and validation accuracy and loss curves, I can identify trends such as overfitting or underfitting, ensuring the model's robustness and generalization capabilities.

4.1.7 Hyperparameter Fine-Tuning

In the hyperparameter fine-tuning section, I employ a trial-and-error approach to optimize the performance of my zero-day attack detection model. Hyperparameters play a crucial role in determining the behavior and effectiveness of machine learning models and fine-tuning them is essential for achieving optimal results.

To begin the fine-tuning process, I systematically explore different combinations of hyperparameters, adjusting one or more parameters at a time while keeping others constant. This iterative approach allows me to observe the impact of each hyperparameter on the model's performance and identify the configurations that yield the best results. Throughout the fine-tuning process, I carefully monitor the performance metrics such as accuracy, precision, recall, and F1 score on validation data. By evaluating the model's performance across various hyperparameter settings, I gain insights into how different configurations affect the model's ability to detect zero-day attacks accurately and efficiently.

As I experiment with different hyperparameter values, I document the results obtained from each configuration, noting any improvements or deteriorations in performance compared to the baseline model. This documentation helps me track the progress of the fine-tuning process and identify promising combinations of hyperparameters for further exploration.

Ultimately, through iterative experimentation and evaluation, I aim to identify the hyperparameter values that maximize the performance of my zero-day attack detection model on unseen data. By fine-tuning the model's hyperparameters using the trial-and-error method, I can enhance its effectiveness and robustness in real-world scenarios, ensuring reliable detection of emerging security threats in IoT networks.

4.1.8 Model Deployment

Before stepping into the simulation of the federated learning framework, I repeat the deployment process for the second and third IoT devices. This ensures that each device is equipped with its own instance of the zero-day attack detection models, allowing them to independently analyze and classify incoming network traffic. With all devices now equipped with their own localized detection capabilities, I can proceed to simulate the federated learning framework. This framework enables collaborative model training and knowledge sharing across the network, leveraging the insights and detection capabilities of each device to enhance the overall security posture of the IoT ecosystem.

By establishing periodic communication between the devices and a central node, I facilitate the aggregation and updating of the global zero-day attack detection model. At predefined intervals, each device transmits its locally trained model parameters to the central node, where they are aggregated to update the global model. This collaborative approach enables the global model to capture insights and patterns from across the network, improving its accuracy and robustness to emerging threats. Simultaneously, the updated global model parameters are distributed back to all devices, ensuring that each device benefits from the collective intelligence of the entire network. This iterative process of model aggregation and distribution enables rapid adaptation to evolving threat landscapes without the need for frequent centralized updates.

4.2 Federated Learning Framework

In this section, I will discuss the design of the simulation for federated learning framework to detect zero-day attack on IoT device. Figure 22 illustrates the federated learning framework design to detect zero-day attack.

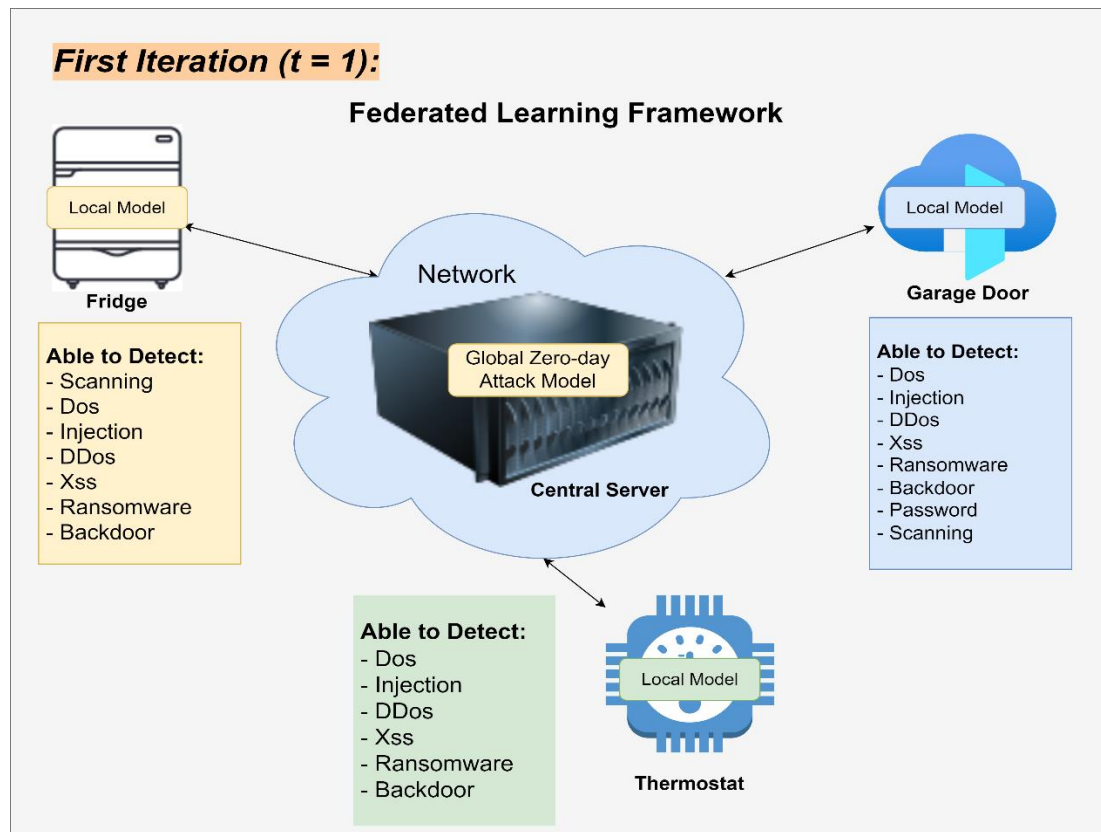


Figure 22 Federated Learning Framework Design

To simulate the federated learning framework to detect zero-day attacks in IoT devices, I orchestrate a distributed system where each IoT device functions as an autonomous node contributing to the collective intelligence of the network. The simulation process begins with the initialization of a central server, responsible for coordinating model aggregation and distribution across all participating devices. The central server initializes global models for binary and multi-class classification tasks, leveraging pre-trained models obtained from previous iterations or external sources. The global model serves as the foundation for the federated learning process, providing a starting point for subsequent updates and refinements.

Each IoT device is then instantiated with references to the central server and its local dataset, which comprises network traffic data collected from the device's operational environment. Upon initialization, the device loads pre-trained models tailored to its role in the federated learning framework, encompassing binary classification, multi-class classification, and potentially specialized models designed for specific tasks or attack scenarios. The device preprocesses its local dataset, encoding categorical variables and scaling numerical features to ensure compatibility with the machine learning models employed in the federated learning process.

During the simulation, each IoT device autonomously iterates over its local dataset, processing individual samples and making predictions using its local models. For binary classification tasks, the device determines whether incoming network traffic is malicious or non-malicious, utilizing a threshold-based approach or probabilistic inference. If the traffic is classified as malicious, the device further analyses it using multi-class classification models to identify the specific type of attack present. Notably, the detection of zero-day attacks is a primary focus of the simulation, with devices employing anomaly detection mechanisms to flag anomalous traffic patterns indicative of previously unseen threats.

Throughout the simulation, devices continuously update their local models based on observed data and detection outcomes. If a significant number of zero-day attacks are detected locally, a device triggers an update mechanism to request the latest global models from the central server. The central server aggregates updated models from all participating devices, applying techniques such as model averaging or federated averaging to reconcile discrepancies and ensure consistency across the network. These aggregated models are then distributed back to the devices, completing a cycle of collaborative learning and model refinement within the federated learning framework. In figure 23 to 25, it illustrates the ability to identify number of attacks on each IoT devices increases along with the number of iterations of aggregating and updating the local models.

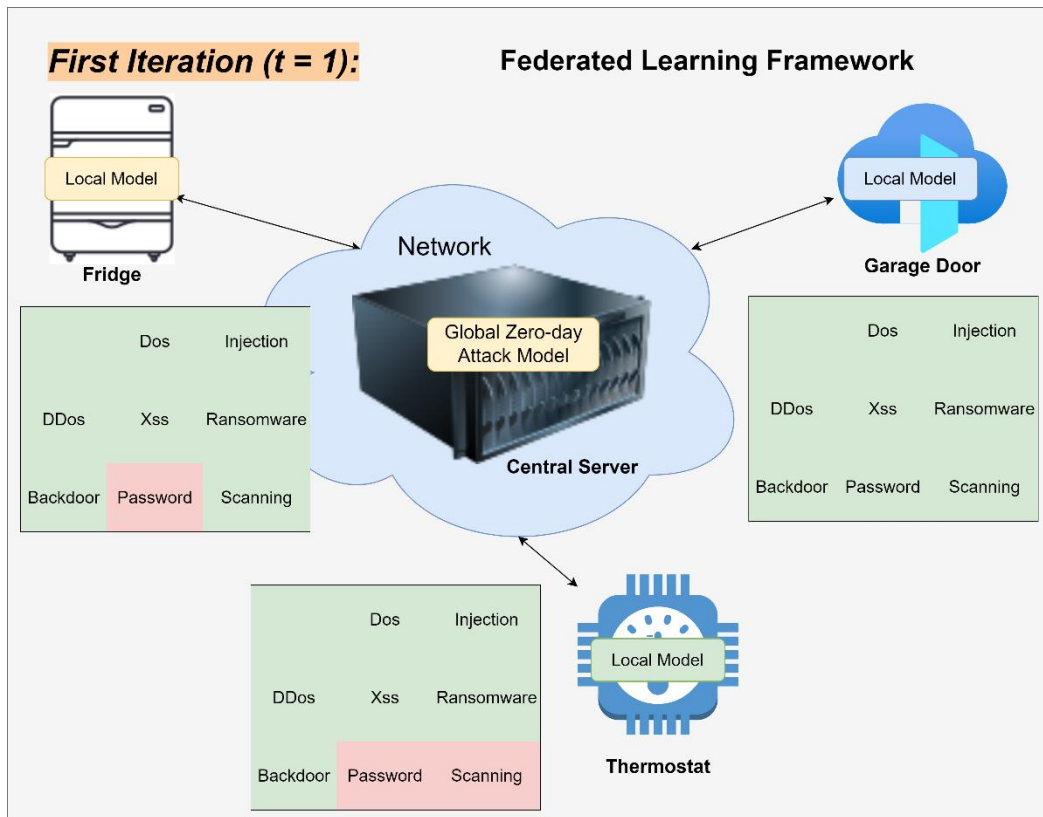


Figure 23 First Iteration of Aggregating and Updating Local Model

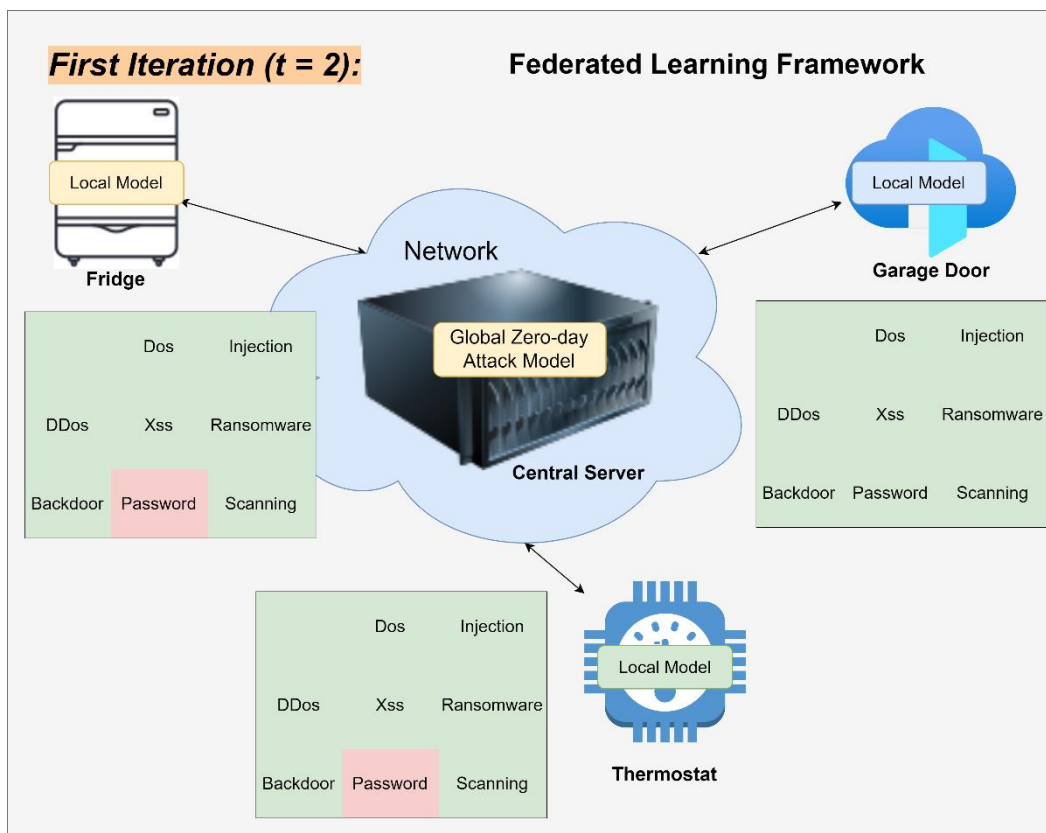


Figure 24 Second Iteration of Aggregating and Updating Local Model

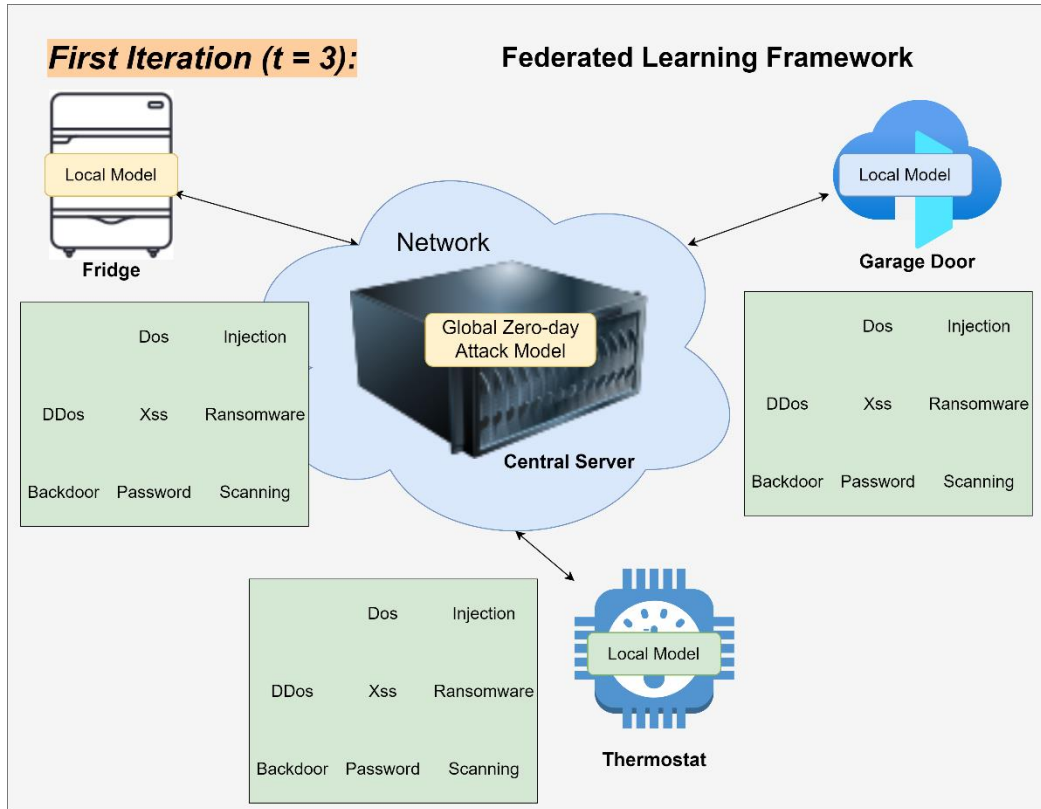


Figure 25 Third Iteration of Aggregating and Updating Local Model

CHAPTER 5 EXPERIMENT/ SIMULATION

5.1 Hardware Setup

The hardware that is used to develop the project is:

Laptop

Description	Specifications
Model	ROG Zephyrus G15 GA503QM
Processor	AMD Ryzen™ 9 5900HS Mobile Processor (8-core/16-thread 20MB cache, up to 4.6 GHz max boost)
Operating System	Windows 11
Graphic	NVIDIA® GeForce RTX™ 3060 Laptop GPU With ROG Boost up to 1525MHz at 80W (95W with Dynamic Boost) 6GB GDDR6
Memory	16GB DDR4 RAM
Storage	512GB PCIe® 3.0 NVMe™ M.2 SSD

Table 14 Laptop Hardware Specification

5.2 Software Setup

The software that requires to develop the project are:

Python

Python is a versatile and widely used programming language, and version 3.10.11 is essential for running various Python-based scripts and libraries in my project.

Anaconda

Anaconda3 is a distribution of Python that comes bundled with numerous data science and machine learning libraries. This specific version, Anaconda3 2023.03-1, with Python 3.10.9, is necessary for managing my project's dependencies and environment effectively.

Jupyter Notebook

Jupyter Notebook is an interactive development environment that allows for creating and sharing documents that contain live code, equations, visualizations, and narrative text. Version 6.5.4 of Jupyter Notebook is required for my project's code documentation and exploration.

CHAPTER 5

5.3 Setting and Configuration

Before starting to develop the 3 models in the respective IoT devices, there are 3 software needed to be installed and downloaded in my laptop:

1. Python 3.10.11
2. Anaconda3 2023.03-1 (Python 3.10.9 64-bit)
3. Jupyter Notebook 6.5.4

5.4 Federated Learning Framework Simulation

In this section, I will discuss the federated learning framework simulation to detect zero-day attack on the IoT devices. Table 15 shows the pseudocode of the Central Server in the federated learning framework.

Algorithm: Pseudocode for Central Server Class

1	Class CentralServer:
2	Initialize:
3	num_devices
4	global_models = _initialize_global_models()
5	Function _initialize_global_models():
6	global_models = []
7	model_binary_path = global_model_1.keras'
8	model_binary = load_model(model_binary_path)
9	model_malicious_path = global_model_2.keras'
10	model_malicious = load_model(model_malicious_path)
11	global_models = [model_binary, model_malicious]
12	Return global_models
13	Function aggregate_models(updated_models):
14	aggregated_models = []
15	Loop over each model layer:
16	layer_weights = []
17	Loop over each device's updated models:
18	Get the model's layer weights
19	Append the layer weights to layer_weights list
20	averaged_layer_weights = []
21	Loop over each tuple of weights:
22	Calculate the average of weights for each tuple
23	Append the averaged weights to averaged_layer_weights list
24	Append the averaged_layer_weights to aggregated_models list
25	Return aggregated_models
26	Function update_global_models(aggregated_models):
27	Update global_models with aggregated_models
28	Function distribute_models():
29	Return global_models

Table 15 Pseudocode for Central Server in Simulated Federated Learning Framework

The CentralServer class serves as the central coordination hub in my federated learning framework, orchestrating model aggregation and distribution across multiple IoT devices. Upon initialization, the CentralServer class receives the number of participating devices and initializes global models by loading pre-trained models from the IoT devices. These global models represent the current state of the overall system and are used as a reference for model aggregation and distribution. By using the concept of federated learning, the CentralServer collaboratively aggregates and updates models from individual devices to create a more robust and accurate global model.

The core of the CentralServer class lies in its methods for model aggregation and distribution. The `_initialize_global_models` method loads the initial global models, containing a model for binary classification and another model for multi-class classification. These models serve as the starting point for the federated learning process. The `aggregate_models` method aggregates updated models received from individual IoT devices, combining their knowledge to improve the global model's performance. By averaging the weights of corresponding layers across all devices, the CentralServer ensures that each device's contribution is appropriately weighted in the final model. After aggregation, the `update_global_models` method updates the global models with the aggregated results, ensuring that the system evolves over time to adapt to new data and insights.

In addition to aggregation, the CentralServer class is responsible for distributing updated models back to the participating IoT devices. The `distribute_models` method retrieves the latest global models and distributes them to all devices, ensuring that each device has access to the most up-to-date knowledge. This iterative process of aggregation and distribution forms the foundation of federated learning, enabling collaborative model training without centralized data storage.

Table 16 shows the pseudocode of the IoT Device in the simulated federated learning framework to detect zero-day attack.

Algorithm: Pseudocode for IoT Devices Class

1	Class IoTDevice:
2	Initialize:
3	device_index, model_indices, central_server, dataset
4	models = []
5	Loop over each model_index:
6	If model_index == 3:
7	Load model 3 from respective .pkl file
8	models.append(model3)
9	Else:
10	Load model 1 or 2 from respective .keras files
11	models.append(model1 or model2)
12	attack = Determine attack based on device_index
13	Function preprocess_dataset(dataset):
14	Preprocess dataset columns and encode categorical variables
15	Scale numerical features
16	Return preprocessed dataset features and labels
17	Function preprocess_cnn_model(sample):
18	Reshape sample for CNN model
19	Return preprocessed sample
20	Function update_models(updated_models):
21	Update models with updated_models
22	Function simulate_federated_learning():
23	Preprocess dataset features and reshape for CNN model
24	Loop over each sample in the dataset:
25	Get the CNN model input for the sample
26	Use model1 to predict if the sample is malicious or not
27	If non-malicious, use model3 to check for zero-day attack and update model if present
28	If malicious, use model2 to predict the attack type
29	Output:
30	Print device index and whether zero-day attack detected
31	Print attack type if malicious sample detected

Table 16 Pseudocode for IoT Device Class in Simulated Federated Learning Framework

The IoTDevice class represents an individual IoT device within my federated learning framework, tasked with participating in model training and contributing to the collective intelligence of the system. Each IoTDevice instance is initialized with a unique index, indicating its position within the network, as well as references to the central server and its local dataset. Additionally, the IoTDevice class loads pre-trained models specific to its role in the federated learning process, either from stored files or directly from memory. These models include those for binary classification, multi-class classification, and potentially additional models tailored to specific tasks or scenarios.

One of the primary responsibilities of the IoTDevice class is preprocessing the local dataset to prepare it for model training and inference. The preprocess_dataset method handles data preprocessing tasks such as feature scaling and encoding categorical variables, ensuring that the input data is properly formatted for consumption by the machine learning models. Additionally, the IoTDevice class defines a method preprocess_cnn_model to reshape input data into the appropriate format expected by convolutional neural network (CNN) models, facilitating seamless integration with the federated learning process. During the simulation of federated learning, each IoTDevice instance iterates over its local dataset, processing samples and making predictions using its local models. The simulate_federated_learning method encapsulates this process, allowing the device to contribute insights and updates to the global model based on its observations. If a significant number of zero-day attacks are detected locally, the device triggers an update mechanism to request the latest global models from the central server and incorporate them into its local model repository. This adaptive learning process ensures that each IoT device continuously improves its detection capabilities and adapts to emerging threats in real-time.

5.5 Implementation Issues and Challenges

5.5.1 High Cost of IoT Devices

Addressing the high cost of IoT devices presents a considerable challenge from my perspective. As I delve into building a zero-day attack detection framework, the financial constraints associated with purchasing diverse IoT devices become apparent, hindering my ability to create a robust experimental setup that mirrors real-world environments. This limitation not only affects the scalability of my project but also compromises the breadth and accuracy of my model's training data. To mitigate this challenge, I'll explore alternative solutions like leveraging simulated IoT environments or seeking collaborations with industry partners to access necessary hardware. However, overcoming this obstacle will require creative problem-solving and resourceful strategies to ensure the effectiveness of my detection system within budgetary constraints.

5.5.2 Difficulty in Generating Attacks in Ton-IoT dataset

I encountered issues with existing datasets like the Ton-IoT dataset, where ambiguous or incomplete attack labels hindered accurate simulation of zero-day attacks. Additionally, creating diverse attack scenarios that reflect the dynamic IoT threat landscape proved daunting, especially without standardized methodologies for data generation. To tackle this challenge, I leveraged the rich data from the Ton-IoT dataset to simulate a federated learning framework. By utilizing the dataset's comprehensive features and attack types, I replicated a distributed learning environment across multiple IoT devices. This approach enabled me to address the complexities of IoT network vulnerabilities and adversary behaviour while ensuring the realism and diversity of generated attack scenarios. Through iterative training and evaluation using the Ton-IoT dataset, I refined my detection models to effectively identify zero-day attacks in simulated federated learning settings.

5.6 Concluding Remark

In conclusion, Chapter 5 of my project delves into the experimental setup, software configuration, and simulation framework for detecting zero-day attacks in IoT networks using federated learning. I began by detailing the hardware and software components utilized in the project, including the specifications of the laptop and essential software tools such as Python, Anaconda, and Jupyter Notebook. Setting the stage for the federated learning simulation, I discussed the central server and IoT device classes, outlining their functionalities and interactions within the framework. Furthermore, I explored the pseudocode algorithms for both the central server and IoT devices, providing a structured overview of their operations in the federated learning process.

The chapter also addresses key implementation challenges, notably the high cost of IoT devices and the difficulty in generating realistic attack scenarios using existing datasets like Ton-IoT. Despite these challenges, I adopted innovative strategies to overcome limitations, including leveraging simulated environments and maximizing the utility of available datasets. By simulating a federated learning framework with the Ton-IoT dataset, I demonstrated the feasibility of detecting zero-day attacks in IoT networks while navigating practical constraints.

Looking ahead, Chapter 6 will delve into the results and findings of my experimental simulations, providing insights into the efficacy and performance of my zero-day attack detection framework. Through comprehensive analysis and evaluation, I aim to validate the effectiveness of federated learning in enhancing the security of IoT networks and mitigating the risks posed by emerging threats.

CHAPTER 6 SYSTEM EVALUATION AND DISCUSSION

6.1 System Testing and Performance Metrics

6.1.1 Model Evaluation

As discussed in Section 4.1.5, to evaluate my zero-day attack detection models, I employ a comprehensive set of metrics to assess its performance. Core metrics such as accuracy, precision, recall, and F1 score provide insights into the model's ability to correctly classify different types of traffic and detect zero-day attacks. While accuracy offers a broad perspective on overall correctness, precision, recall, and F1 score provide nuanced assessments, particularly in scenarios with imbalanced class distributions. I complement these metrics with classification reports and confusion matrices to analyze performance across individual classes and visualize areas of misclassification. Additionally, I monitor training and validation accuracy and loss curves to ensure the model's robustness and generalization capabilities over epochs. Through this multifaceted evaluation approach, I gain a comprehensive understanding of my model's effectiveness in detecting zero-day attacks in IoT networks.

6.1.2 Overall 2-Layer Classification Model Pipeline Testing and Performance

In this section, I will discuss the method used to evaluate the 2-layer classification model pipeline. In the figure 26 below, it illustrates the method used to evaluate the 2-layer classification model pipeline.

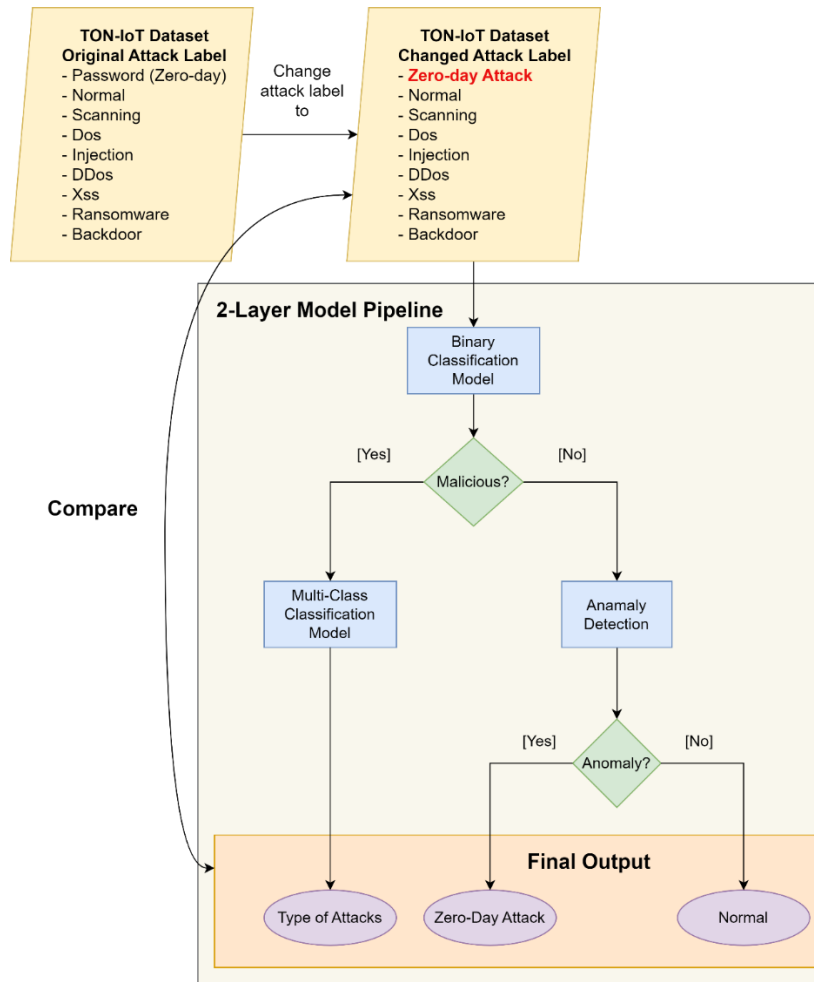


Figure 26 Method Used to Evaluate 2-Layer Model Pipeline

In the overall 2-layer classification model testing and performance evaluation, I assess the effectiveness of my model pipeline in accurately classifying and detecting zero-day attacks in IoT network traffic. The pipeline comprises three models: the first binary classification model, the second multi-class classification model, and the anomaly detection model. The testing procedure involves modifying the dataset to designate a specific attack label as the zero-day attack, simulating the presence of previously unseen threats. Initially, the modified dataset is fed into the pipeline, where the first binary classification model determines whether the traffic is malicious or

benign. If the binary model identifies the traffic as malicious, it proceeds to the second multi-class classification model to classify the type of attack. Conversely, if the traffic is deemed non-malicious by the binary model, it is routed to the anomaly detection model to detect any anomalies indicative of zero-day attacks.

From the Figure 27 below illustrate the traffic flow and label of respective accuracy of different models in the classification model pipeline.

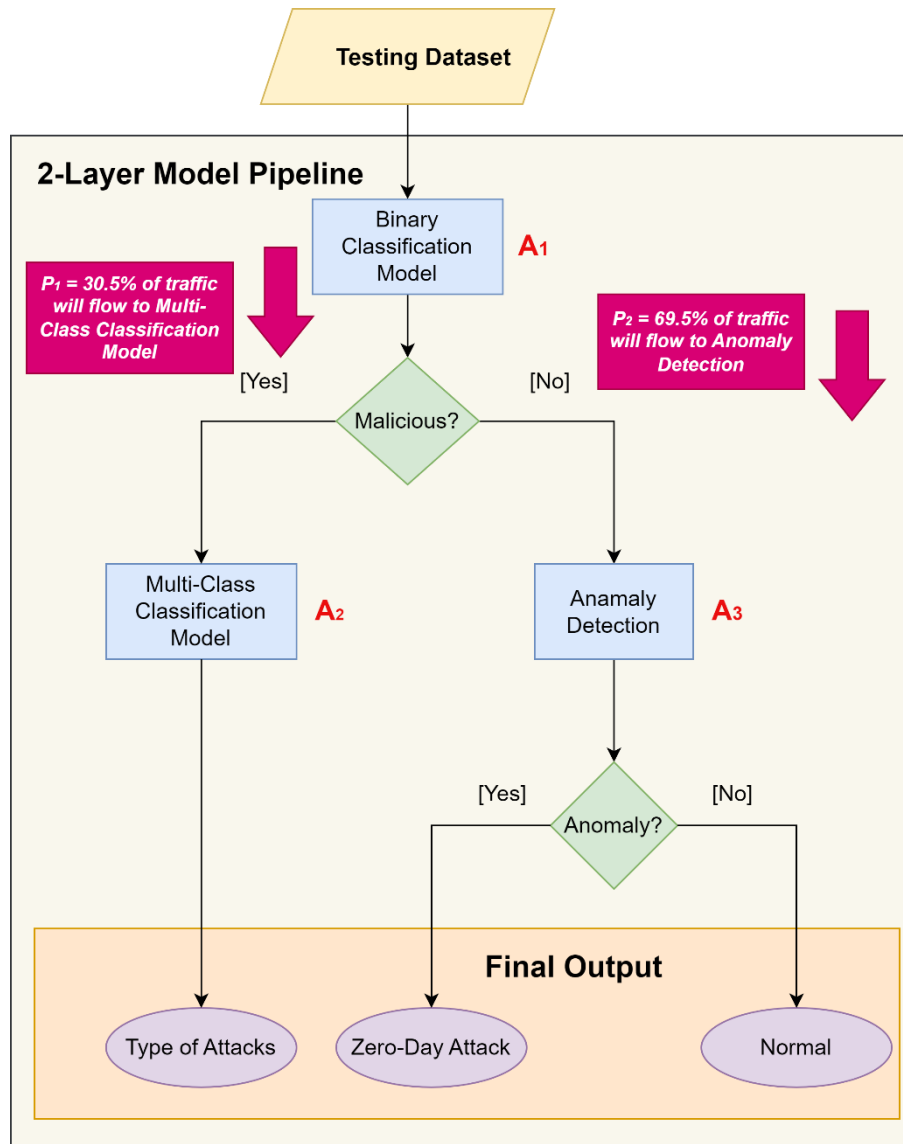


Figure 27 Accuracy of Respective Models in the 2-Layers Model Pipeline

CHAPTER 6

From the accuracy of respective models provided in the 2-layers model pipeline, assuming the percentage of different type of traffic passing through the model pipeline is as shown in figure 28. I can calculate the maximum accuracy that can be achieved in this model pipeline using this formula:

$$\begin{aligned} \text{Maximum Accuracy} &= A_1P_1A_2 + A_1P_2A_3 \\ &= 0.3050(A_1A_2) + 0.6950(A_1A_3), \end{aligned}$$

Where A_1 = Accuracy of Binary Classification Model,

A_2 = Accuracy of Multi-Class Classification Model,

A_3 = Accuracy of Anomaly Detection,

P_1 = Probability of Traffic Flow to Multi-Class Classification Model

P_2 = Probability of Traffic Flow to Anomaly Detection Model

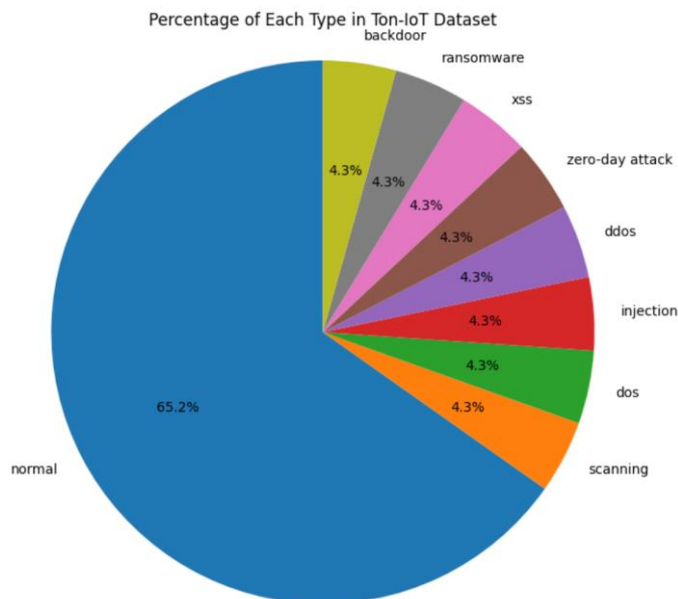


Figure 28 Percentage of Each Type of Data in Ton-IoT Dataset

Besides, to evaluate the performance of the 2-layers classification model, I can also test the ability to detect the zero-day attack from the input traffic. The formula of testing the ability of zero-day attack detection is:

$$P(\text{Zero - day attack}) = \frac{\text{Number of Zero - day attack correctly predicted}}{\text{Total amount of Zero - day attack}}$$

6.1.3 Federated Learning Framework Testing and Performance

In this section, I will discuss the method used to evaluate the performance of federated learning framework. In Figure 29 shows the experiment setup to test the federated learning framework performance.

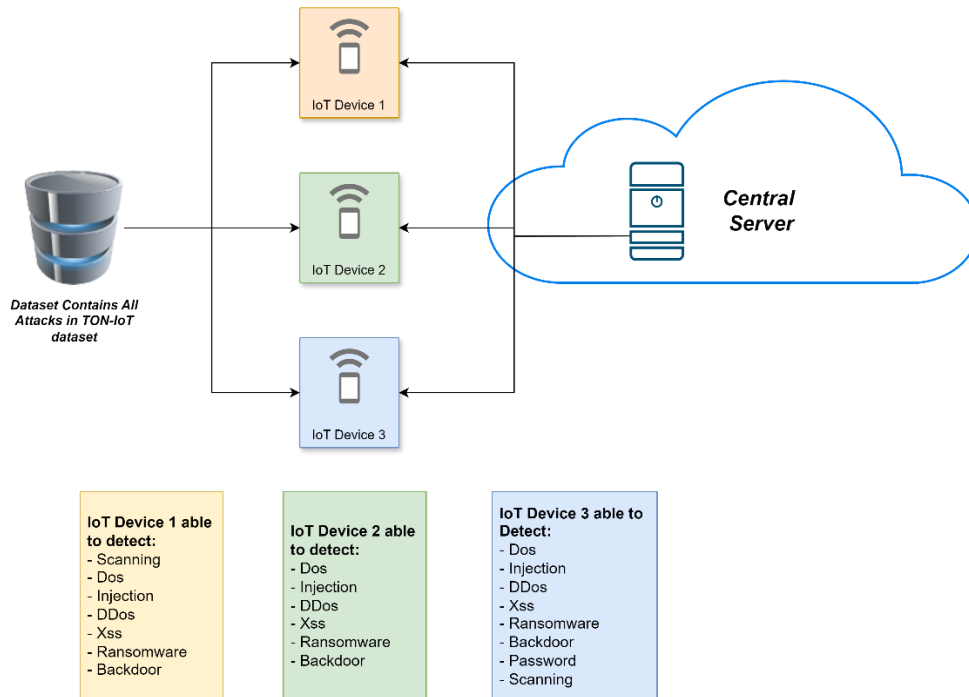


Figure 29 Federated learning Framework Performance Testing Setup

To evaluate the performance of the federated learning framework, a comprehensive dataset comprising all attack types from the TON-IoT dataset is compiled. This dataset is then distributed to three distinct IoT devices, each equipped with different locally trained models tailored to detect specific types of attacks. Upon receiving the dataset, the IoT devices independently analyse the traffic using their respective models and record their detection results. Subsequently, the central server aggregates all the local models if alert received or reached the update interval. Next, the central server then updates the local models based on the collective insights gathered. Following the model update, the detection process is relaunched, and the performance of the federated learning framework is assessed iteratively. At each iteration, the framework's ability to detect and classify various attacks is evaluated, and the detection performance for each attack type is recorded. This iterative evaluation allows for the monitoring of the framework's performance over time, providing insights into its effectiveness in detecting evolving threats.

6.2 Testing Setup and Result

6.2.1 Model Evaluation Result

In this section, I will discuss the model evaluation result of different hyperparameter tuning for respective model.

Binary CNN Model and Multi-class CNN Model

Hyperparameters play a crucial role in determining the learning dynamics and generalization capabilities of machine learning models. By systematically exploring different combinations of hyperparameters, I aim to enhance the robustness, accuracy, and efficiency of my models in detecting zero-day attacks in IoT networks. This section provides insights into the strategies employed, the performance improvements achieved, and the implications of hyperparameter tuning on the effectiveness of the detection models. Firstly, I employ the trial-and-error method in order to find the best set of hyperparameter tuning. Table 17 and 18 shows the testing accuracy and validation accuracy of the binary CNN model to detect whether the traffic is malicious or non-malicious and multi-class CNN model to detect the type of attacks.

Model	Number of Batch Size (Training Accuracy %) [Epochs = 10]		
	16	32	64
CNN Binary Model	94.33	95.26	95.13
CNN Multi-Class Model	96.43	96.49	96.36
Model	Epochs (Training Accuracy %) [Best Batch Size]		
	10	30	50
CNN Binary Model	95.26	97.23	96.16
CNN Multi-Class Model	96.49	97.43	97.72

Table 17 Training Accuracy of Models

Model	Number of Batch Size (Validation Accuracy %) [Epochs = 10]		
	16	32	64
CNN Binary Model	94.65	95.51	95.21
CNN Multi-Class Model	96.30	96.84	96.72
Model	Epochs (Validation Accuracy %) [Best Batch Size]		
	10	30	50
CNN Binary Model	95.51	97.32	96.11
CNN Multi-Class Model	96.84	97.14	95.72

Table 18 Validation Accuracy of Models

The hyperparameter tuning results shed light on the behavior of my models concerning batch size and epochs. Initially, the training accuracy of the CNN binary model shows an upward trend as the batch size increases. However, a reversal occurs when the batch size rises from 32 to 64, resulting in a decrease in accuracy. This phenomenon can be attributed to the larger batch size leading to less frequent updates of the model's weights, potentially causing convergence issues or hindering the model's ability to capture intricate patterns in the data. Similarly, the training accuracy of the binary model increases with the number of epochs but starts to decline beyond 30 epochs. This decline may be indicative of the model overfitting to the training data, capturing noise rather than generalizable patterns, thus resulting in reduced performance on unseen data. Consequently, the optimal hyperparameters for the CNN binary model are determined to be a batch size of 32 and 30 epochs.

Likewise, the CNN multi-class model exhibits a comparable pattern to the binary model. However, unlike the binary model, the training accuracy of the multi-class model consistently improves as the number of epochs increases. Nevertheless, contrasting trends emerge in the validation accuracy, where the peak performance is observed at 30 epochs. This discrepancy suggests that the model's performance may plateau or even degrade with prolonged training, indicating potential overfitting beyond 30 epochs.

Although 30 epochs with 32 batch_size has the highest training accuracy for both CNN models, further investigation should be evaluated. Figure 30 and 31 shows the training & validation accuracy against the number of epochs for both CNN models with the optimal hyperparameter found previously.

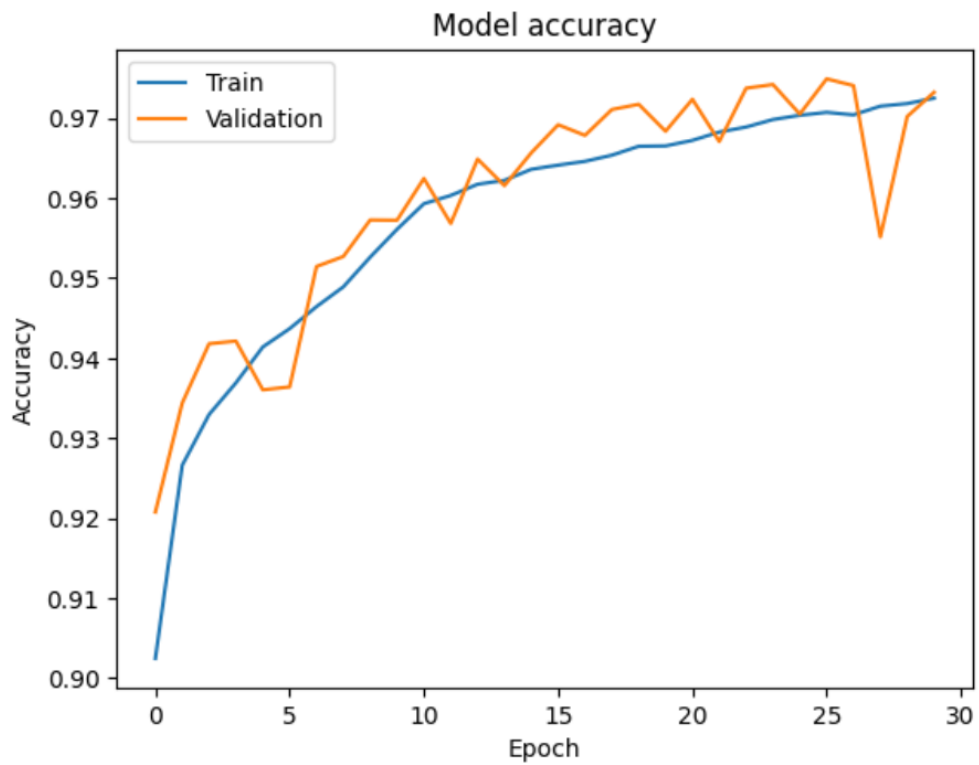


Figure 30 Training and Validation accuracy for CNN Binary Model with Optimal Hyperparameter (32 batch size & 30 epochs)

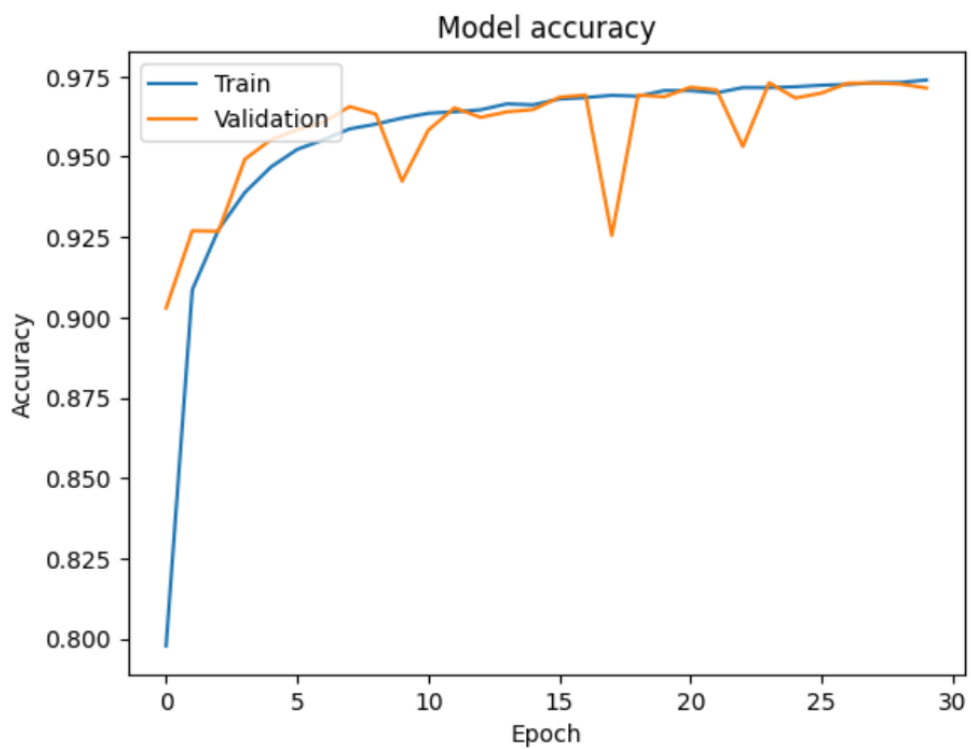


Figure 31 Training and Validation Accuracy for CNN-Multi Class Model with Optimal Hyperparameter (32 batch size & 30 epochs)

CHAPTER 6

From the figures above, I can observe that the training accuracy is not steadily increases, this might be because the learning rate is not well-adjusted to make the training accuracy steadily increase. Thereby, I start to tune the learning rate as well. Table 19 shows the result of training accuracy with different learning rate for both CNN model.

Model	Learning Rate (Training Accuracy %) [32 Batch Size, 30 Epochs]				
	0.1	0.01	0.001	0.0005	0.0001
CNN Binary Model	69.50	95.26	96.77	97.17	96.33
CNN Multi-Class Model	78.20	96.49	96.88	97.34	95.57

Table 19 Training Accuracy Result with Different Learning Rate for Both CNN Models

For the CNN binary model, I observe a substantial improvement in training accuracy as the learning rate decreases from 0.1 to 0.001, indicating that a slower learning rate allows the model to converge more effectively. However, further reducing the learning rate to 0.0005 and 0.0001 leads to diminishing returns, with slight fluctuations in accuracy. Notably, the learning rate of 0.0001 stands out for its steady increase in accuracy without significant oscillations, suggesting that it strikes a favorable balance between convergence speed and stability. However, even though the learning curve of CNN binary model with learning rate of 0.0005 is not as stable as 0.0001, but it achieved higher accuracy.

Similarly, the CNN multi-class model exhibits a similar trend, with training accuracy steadily increasing as the learning rate decreases from 0.1 to 0.001. Beyond a learning rate of 0.001, marginal gains in accuracy are observed, indicating diminishing returns. Interestingly, the learning rate of 0.0005 emerges as the most consistent performer, demonstrating a smooth and stable increase in accuracy without significant fluctuations. This behavior suggests that 0.0005 learning rate facilitates more reliable convergence and generalization, mitigating the risk of overshooting optimal parameter values or getting stuck in local minima. On the other hand, 0.0001 learning rate also illustrated a steady learning curve, but it learns slower than model with learning rate of 0.0005, indicating that learning rate of 0.0005 is better option. Figure 32 and 33 show the training and validation accuracy graph for both models with optimal hyperparameter and learning rate of 0.0005 and 0.0001.

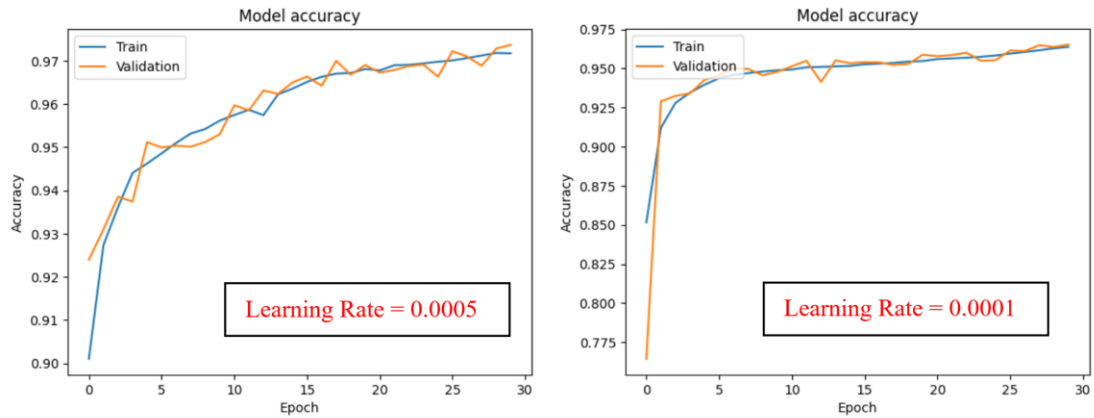


Figure 32 Training and Validation Accuracy Graph for CNN Binary Model with Optimal Hyperparameter and Learning Rate of 0.0005 and 0.0001

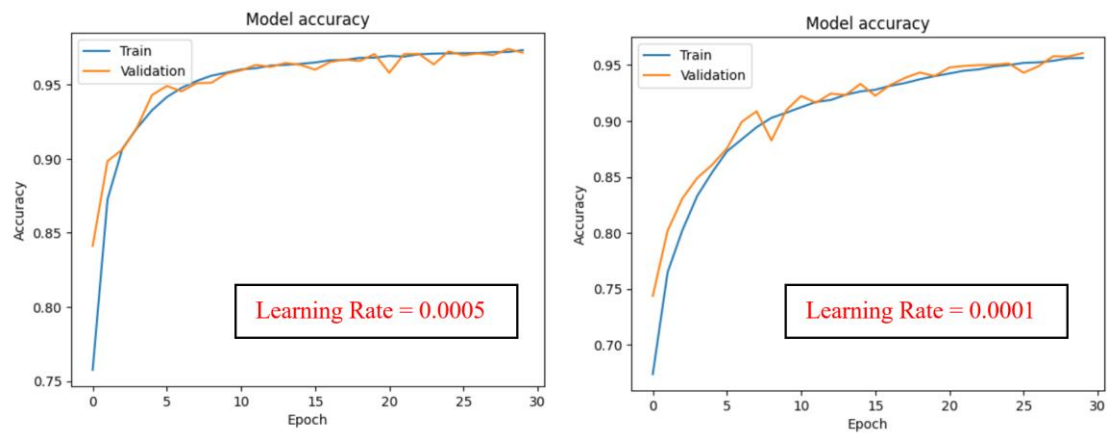


Figure 33 Training and Validation Accuracy Graph for CNN Multi-Class Model with Optimal Hyperparameter and Learning Rate of 0.0005 and 0.0001

Moving forward, I further investigate the confusion matrix and classification report of both the CNN models. Figure 34 and 35 shows the confusion matrix and classification report of both the CNN models with optimal hyperparameter and 0.0005 learning rate.

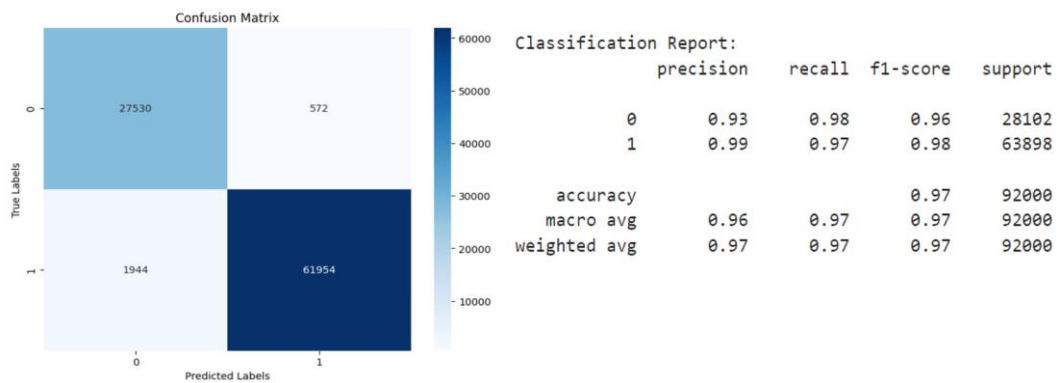


Figure 34 Confusion matrix and Classification Report of Binary CNN Model with optimal hyperparameter (32 batch size, 30 epochs, 0.0005 learning rate)

The evaluation results for the CNN binary model provide valuable insights into its performance in detecting zero-day attacks. The confusion matrix reveals that out of 92,000 instances, the model correctly classified 89,484 instances, achieving an overall accuracy of 97%. Specifically, the model accurately identified 27,530 instances of non-malicious traffic (class 0) and 61,954 instances of malicious traffic (class 1). However, the model misclassified 1,944 instances of non-malicious traffic as malicious and 572 instances of malicious traffic as non-malicious. This indicates a small but non-negligible number of false positives and false negatives, which could have significant implications in real-world scenarios. Thus, this could be a work for future researchers to improve the model.

The classification report further illustrates the model's performance across different metrics. With a precision of 0.93 for class 0 and 0.99 for class 1, the model exhibits a high ability to correctly classify instances as either non-malicious or malicious. The recall values of 0.98 for class 0 and 0.97 for class 1 indicate the model's effectiveness in capturing most true instances within each class. Additionally, the F1-score, which balances precision and recall, reflects the overall performance of the model, yielding values of 0.96 for class 0 and 0.98 for class 1. These metrics collectively demonstrate the model's robustness in distinguishing between normal and malicious traffic, with a weighted average F1-score of 0.97 across all classes.

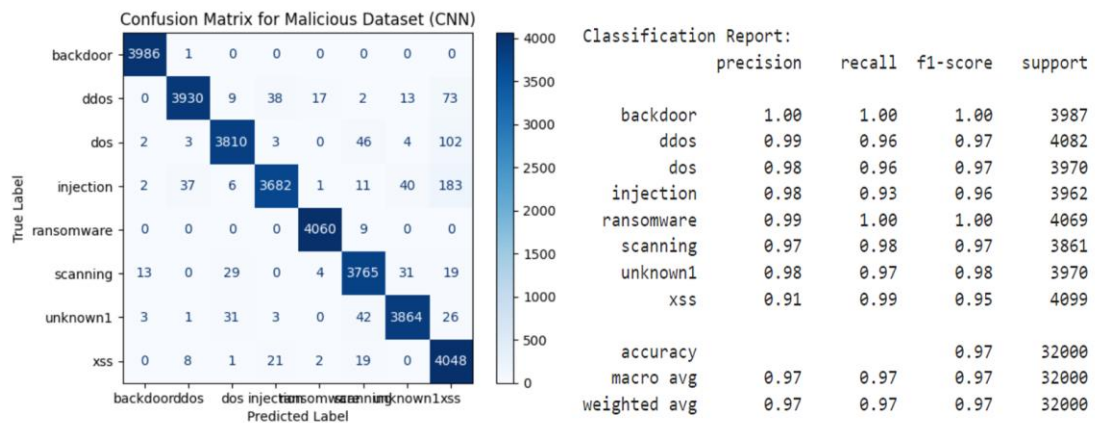


Figure 35 Confusion matrix and Classification Report of Multi-Class CNN Model with optimal hyperparameter (32 batch size, 30 epochs, 0.0005 learning rate)

For Multi-class CNN model, the confusion matrix reveals that out of 32,000 instances, the model correctly classified 31,095 instances, achieving an overall accuracy of 97%. Each row in the confusion matrix represents the actual class, while each column represents the predicted class. The diagonal elements indicate the number of instances correctly classified for each class, demonstrating the model's ability to effectively distinguish between different attack types. Notably, the model achieved perfect classification for the "backdoor" and "ransomware" classes, with all instances correctly identified. However, some misclassifications were observed, particularly for the "ddos," "dos," and "injection" classes, indicating areas where the model's performance could be further optimized.

The classification report provides additional insights into the model's performance across different attack types. Precision, recall, and F1-score metrics are reported for each class, offering a more nuanced understanding of the model's strengths and weaknesses. The precision metric measures the proportion of true positive predictions among all instances predicted as belonging to a specific class, while recall quantifies the proportion of true positive predictions among all actual instances of that class. The F1-score, which is the harmonic mean of precision and recall, provides a balanced assessment of the model's performance, particularly useful in scenarios where both false positives and false negatives are critical considerations.

To conclude, the hyperparameter tuning process for both the CNN binary and multi-class models has provided valuable insights into optimizing their performance.

CHAPTER 6

For the CNN binary model, the highest accuracy of 97.17% was achieved with a learning rate of 0.0005, utilizing a batch size of 32 and 30 epochs. This configuration strikes a balance between convergence and stability, resulting in robust model performance. Similarly, the CNN multi-class model attained its highest accuracy of 97.72% with a learning rate of 0.001, along with a batch size of 32 and 30 epochs. These findings underscore the importance of fine-tuning hyperparameters to maximize model performance, ultimately enhancing the effectiveness of zero-day attack detection in IoT networks.

Anomaly Detection

In this section, I will discuss the hyperparameter tuning for the anomaly detection. Table 20 below shows the performance metrics of anomaly detection with different set of hyperparameters.

Params	<i>N_estimators and max_features keep as constant (n_estimator =100, max_features = 20)</i>												
	Max_samples	0.01				0.001				0.0001			
	Contamination	0.01	0.05	0.15	0.20	0.01	0.05	0.15	0.20	0.01	0.05	0.15	0.20
Accuracy	0.9258	0.8720	0.7740	0.6624	0.9202	0.8705	0.7931	0.7035	0.9121	0.8879	0.7426	0.7026	
Recall	0.0147	0.0737	0.7245	1.0000	0.0027	0.0717	0.6758	0.9938	0.0282	0.4011	0.7630	0.7632	

Table 20 Performance Metrics of Anomaly Detection with Different Hyperparameters

Based on the hyperparameter tuning results, the isolation forest model achieved its highest accuracy of 92.58% with a max_samples value of 0.01 and a contamination rate of 0.01. However, considering the priority of correctly identifying zero-day attacks, the model with a max_samples value of 0.001 and a contamination rate of 0.20 stands out as the overall best performer. Despite its slightly lower accuracy of 70.35%, this configuration prioritizes recall, achieving a recall rate of 99.38%, indicating a high capability to detect zero-day attacks while tolerating some false positives. This strategic emphasis on recall aligns with the objective of prioritizing zero-day attack detection, where minimizing false negatives takes precedence over false positives. Therefore, the model with max_samples = 0.001 and contamination = 0.20 emerges as the optimal choice for my isolation forest-based anomaly detection system.

6.2.2 2-Layers Model Pipeline Classification Evaluation

In this section, I will discuss the performance of the 2-layers model pipeline classification. Table 21 shows the maximum accuracy of different model 3 in 2-layers model pipeline according to the formula provided in the 6.1.2:

$$\text{Maximum Accuracy} = 0.3050(A_1A_2) + 0.6950(A_1A_3)$$

Model	Accuracy	
	Model 3 with High Accuracy	Model 3 with High Recall
CNN Binary Model (A_1)	97.17	
CNN Multi-Class Model (A_2)	97.34	
Anomaly Detection Model (A_3)	92.58	70.35
Maximum Accuracy	91.37%	61.67%

Table 21 Maximum Accuracy of Different Model 3 in 2-Layers Model Pipeline

Besides, to evaluate the ability of the zero-day attack detection, I utilize the formula provided in the 6.1.2. Table 22 below shows the ability of different model 3 in 2-layer classification model pipeline.

$$P(\text{Zero - day attack}) = \frac{\text{Number of Zero - day attack correctly predicted}}{\text{Total amount of Zero - day attack}}$$

Model	Accuracy	
	Model 3 with High Accuracy	Model 3 with High Recall
Number of Zero-day attacks	8500	
Number of Zero-day attacks detected	6856	8429
P(Zero-day attack)	80.66%	99.16%

Table 22 Ability of Different Model 3 in 2-Layers Model Pipeline

Based on the result obtained above, model 3 with high recall is able to detect zero-day attack better while model 3 with higher accuracy is having over better performance.

6.2.2 Federated Learning Framework Performance Result

In this section, I will discuss the performance of federated learning framework. Table 23 to 25 shows the ability of each IoT device to detect different type of attacks from iteration 1 to 3.

IoT Device	Accuracy (t=1)							
	Scanning	Password	DoS	Injection	DDoS	Xss	Ransomware	Backdoor
IoT Device 1	0.9118	0.0000	0.9086	0.9123	0.9062	0.9188	0.9863	0.9054
IoT Device 2	0.0000	0.0000	0.9545	0.9167	1.0000	0.9898	0.9954	1.0000
IoT Device 3	0.9265	0.9365	0.9465	0.9236	0.9565	0.9756	0.9845	0.9456

Table 23 Ability of each IoT Device to detect different type of attacks (Iteration 1)

IoT Device	Accuracy (t = 2)							
	Scanning	Password	DoS	Injection	DDoS	Xss	Ransomware	Backdoor
IoT Device 1	0.9265	0.0000	0.9265	0.9235	0.9165	0.9654	0.9895	0.9564
IoT Device 2	0.9065	0.0000	0.9654	0.9365	0.9987	0.9784	0.9974	1.0000
IoT Device 3	0.9265	0.9354	0.9484	0.9245	0.9546	0.9756	0.9878	0.9546

Table 24 Ability of each IoT Device to detect different type of attacks (Iteration 2)

IoT Device	Accuracy (t = 3)							
	Scanning	Password	DoS	Injection	DDoS	Xss	Ransomware	Backdoor
IoT Device 1	0.9236	0.9182	0.9236	0.9254	0.9265	0.9546	0.9895	0.9565
IoT Device 2	0.9165	0.9234	0.9634	0.9265	1.0000	0.9878	0.9984	1.0000
IoT Device 3	0.9236	0.9344	0.9465	0.9245	0.9548	0.9785	0.9845	0.9546

Table 25 Ability of each IoT Device to detect different type of attacks (Iteration 3)

Based on the results gathered from various iterations, it's evident that the performance of IoT devices within the federated learning framework has shown promising improvements. Specifically, IoT device 2 demonstrated significant enhancements in detecting scanning and password attacks, with detection rates escalating from 0.0000 to 0.9165 (iteration 3) and 0.0000 to 0.9234 (iteration 3) respectively. Additionally, IoT device 1 exhibited notable progress in identifying password attacks, with its detection rate climbing from 0.0000 to 0.9182 (iteration 3). These advancements underscore the effectiveness of the federated learning framework in augmenting the model's capacity to identify attacks while preserving user data privacy and operating with minimal network bandwidth requirements. Moreover, the framework showcases rapid convergence and efficient utilization of resources, further

emphasizing its utility in enhancing attack detection capabilities across distributed IoT environments.

6.3 Project Challenges

In this project, one of the primary challenges encountered was the constraint of time. Developing and implementing a robust zero-day attack detection framework within a limited timeframe posed significant challenges in terms of task prioritization, resource allocation, and project management. The intricate nature of the federated learning framework, coupled with the complexity of IoT security and attack detection, required thorough planning and execution to ensure timely completion. Moreover, the need to gather and preprocess large-scale datasets, design and train machine learning models, and validate the system's performance imposed additional time constraints. Despite these challenges, careful planning, effective time management strategies, and prioritization of critical tasks enabled the project to progress steadily towards its objectives. However, the time constraint remained a persistent challenge throughout the project lifecycle, underscoring the importance of efficient resource utilization and agile decision-making to meet project milestones effectively.

6.4 Objectives Evaluation

The objective evaluation of my report reveals that the implemented 2-layer classification model pipeline demonstrates a commendable ability to detect zero-day attacks, achieving an impressive accuracy rate of 99.16%. Leveraging this model pipeline within the federated learning framework enables me to detect zero-day attacks without the need to exchange users' sensitive data, ensuring privacy and security. Through the iterative process of model updates facilitated by federated learning, my system demonstrates adaptability and responsiveness to emerging threats. As a result of model updates, I have successfully enhanced my system's capability to detect zero-day attacks, marking a significant milestone in my pursuit of robust cybersecurity solutions for IoT environments.

6.4 Project Timeline

						2023										
Project Start Date:		20-Feb-23				Feb		Mar			Apr					
Project Title:		Zero-Day Detection on IoT Network (IIPSPW)				Week Starting	20-Feb	27-Feb	6-Mar	13-Mar	20-Mar	27-Mar	3-Apr	10-Apr	17-Apr	24-Apr
Day																
#	Activity	Start	End	Days	Status											
1	IIPSPW	20-Feb-23	24-Apr-23	63	Completed	◆										
2	Chapter 1: Project Background & Chapter 3: Project Scope and Objectives	20-Feb-23	06-Mar-23	14	Completed	◆										
3	Introduction	20-Feb-23	27-Feb-23	7	Completed	◆										
4	Research Objectives	20-Feb-23	27-Feb-23	7	Completed	◆										
5	Project Scope	27-Feb-23	06-Mar-23	7	Completed	◆										
6	Problem Statement & Motivation	27-Feb-23	06-Mar-23	7	Completed	◆										
7	Chapter 2: Literature Review	06-Mar-23	10-Apr-23	35	Completed	◆										
8	Find related & relevant papers	06-Mar-23	20-Mar-23	14	Completed	◆										
9	Study sources	20-Mar-23	27-Mar-23	7	Completed	◆										
10	Identify the pros and cons of all relevant papers	27-Mar-23	03-Apr-23	7	Completed	◆										
11	Chapter 4: Methods/Technologies Involved	03-Apr-23	17-Apr-23	14	Completed	◆										
12	System Requirements	03-Apr-23	10-Apr-23	7	Completed	◆										
13	Methodology	10-Apr-23	17-Apr-23	7	Completed	◆										
14	Report Checking & Presentation	17-Apr-23	24-Apr-23	7	Completed	◆										

Table 26 Timeline of IIPSPW

CHAPTER 6

						2023																
Project Start Date:		19-Jun-23				Week Starting		Jun		Jul				Aug				Sep				
Project Title:		Zero-Day Detection on IoT Network (FYP1)				Day		19-Jun	26-Jun	3-Jul	10-Jul	17-Jul	24-Jul	31-Jul	7-Aug	14-Aug	21-Aug	28-Aug	4-Sep	11-Sep	18-Sep	25-Sep
#	Activity	Start	End	Days	Status																	
1	FYP1	19-Jun-23	25-Sep-23	98	Completed																	
2	Chapter 1: Project Background	19-Jun-23	26-Jun-23	7	Completed	◆																
3	Discuss with supervisor to see if got any changes	19-Jun-23	26-Jun-23	7	Completed	◆																
4	Chapter 2: Literature Review	26-Jun-23	10-Jul-23	14	Completed	◆																
5	More research on relevant papers	26-Jun-23	10-Jul-23	14	Completed	◆																
6	Chapter 3: Proposed Methodology	10-Jul-23	14-Aug-23	35	Completed	◆																
7	Novelty elements	10-Jul-23	17-Jul-23	7	Completed	◆																
8	System Overview	17-Jul-23	31-Jul-23	14	Completed	◆																
9	Implementation Issues and Challenges	24-Jul-23	14-Aug-23	21	Completed	◆																
10	Timeline	07-Aug-23	14-Aug-23	7	Completed	◆																
11	Chapter 4: Preliminary Works	17-Jul-23	04-Sep-23	49	Completed	◆																
12	Setting Up	17-Jul-23	24-Jul-23	7	Completed	◆																
13	Coding	17-Jul-23	04-Sep-23	49	Completed	◆																
14	Chapter 5: Conclusion	04-Sep-23	11-Sep-23	7	Completed	◆																
15	Poster	04-Sep-23	11-Sep-23	7	Completed	◆																
16	FYP1 Weekly Report	19-Jun-23	11-Sep-23	84	Completed																	
17	Final Checking	04-Sep-23	11-Sep-23	7	Completed	◆																
18	Presentation	18-Sep-23	25-Sep-23	7	In progress	◆																

Table 27 Timeline of FYP1

CHAPTER 6

						2024																					
Project Start Date:		29-Jan-24			Week Starting	Jan	Feb				Mar				Apr			May									
Project Title:		Zero-Day Detection on IoT Network (FYP2)			Day	29-Jan	5-Feb	12-Feb	19-Feb	26-Feb	4-Mar	11-Mar	18-Mar	25-Mar	1-Apr	8-Apr	15-Apr	22-Apr	29-Apr	6-May							
#	Activity	Start	End	Days	Status																						
1	FYP2	29-Jan-24	6-May-24	98	Completed																						
2	Chapter 1: Introduction	29-Jan-24	29-Jan-24	0	Completed	◆																					
3	Discuss with supervisor to see if got any changes	29-Jan-24	29-Jan-24	0	Completed	◆																					
4	Chapter 2: Literature Review	29-Jan-24	05-Feb-24	7	Completed							◆															
5	More research on relevant papers	29-Jan-24	05-Feb-24	7	Completed							◆															
6	Chapter 3: System Model	05-Feb-24	26-Feb-24	21	Completed										◆												
7	System Design Diagram/Equation	05-Feb-24	26-Feb-24	21	Completed										◆												
8	Chapter 4: System Design	26-Feb-24	18-Mar-24	21	Completed												◆										
9	System Block Diagram	26-Feb-24	11-Mar-24	14	Completed										◆												
10	System Components Specifications & Interaction	11-Mar-24	18-Mar-24	7	Completed												◆										
11	Chapter 5: Experiment/Simulation	05-Feb-24	25-Mar-24	49	Completed												◆										
12	Setting Up	05-Feb-24	05-Feb-24	0	Completed	◆																					
13	Coding	05-Feb-24	25-Mar-24	49	Completed												◆										
14	Chapter 6: System Evaluation & Discussion	25-Mar-24	15-Apr-24	21	Completed														◆								
15	System Testing & Performance Metrics	25-Mar-24	08-Apr-24	14	Completed														◆								
16	Result Evaluation	25-Mar-24	15-Apr-24	21	Completed														◆								
17	Chapter 7: Conclusion & Recommendation	15-Apr-24	22-Apr-24	7	Completed														◆								
18	Presentation	29-Apr-24	06-May-24	7	In progress																◆						

Table 28 Timeline of FYP2

6.5 Concluding Remark

In conclusion, Chapter 6 serves as an evaluation of my system's testing and performance metrics, offering valuable insights into the efficacy and efficiency of the implemented models and frameworks. Through testing and analysis, I've garnered significant understanding of the system's capabilities. The evaluation outcomes have unequivocally highlighted the robustness of my 2-layer classification model pipeline, showcasing its prowess in accurately detecting zero-day attacks with an impressive accuracy rate of 99.16%. Moreover, by seamlessly integrating this pipeline within the federated learning framework, I've not only upheld user privacy but also bolstered the system's adaptability to dynamic threat landscapes. The federated learning framework simulation results, further fortify the project's success by demonstrating the framework's ability to enhance model performance across distributed IoT environments. This evaluation underscores the project's effectiveness in addressing the critical challenges of zero-day attack detection while advancing the paradigm of privacy-preserving collaborative learning in IoT security.

CHAPTER 7 CONCLUSION AND RECOMMENDATION

7.1 Conclusion

In conclusion, this project has addressed the critical challenge of enhancing cybersecurity in IoT environments, particularly in the detection of zero-day attacks. By leveraging advanced machine learning techniques and frameworks, I have developed and evaluated a robust system capable of detecting and mitigating emerging threats with high accuracy and efficiency. Through the implementation of a 2-layer classification model pipeline and the utilization of federated learning, I have not only achieved remarkable results in zero-day attack detection but also ensured the privacy and security of user data. This project's contributions extend beyond technical advancements; it serves as a testament to the importance of interdisciplinary collaboration and innovative problem-solving in addressing complex cybersecurity challenges. Looking ahead, the insights gained from this project can guide future research and development efforts aimed at fortifying IoT security and safeguarding digital ecosystems against evolving cyber threats.

7.2 Recommendation

For future endeavors, it is recommended to explore the integration of more diverse and comprehensive datasets to enhance the model's ability to generalize across various IoT environments and attack scenarios. Additionally, further research could focus on optimizing the federated learning framework to accommodate larger-scale deployments and heterogeneous device architectures while preserving privacy and efficiency. Moreover, investigating the potential synergies between traditional signature-based detection methods and machine learning-based anomaly detection approaches could lead to more robust and adaptive zero-day attack detection systems. Lastly, continuous monitoring and adaptation of the developed model to evolving threat landscapes and emerging attack vectors are essential to ensure its long-term effectiveness and resilience in real-world deployment scenarios.

REFERENCES

- [1] S. Angeles, “8 Ways the Internet of Things Will Change the Way We Work,” *Business News Daily*, Aug. 2013. <https://www.businessnewsdaily.com/4858-internet-of-things-will-change-work.html> (accessed Mar. 01, 2023).
- [2] Redapt Marketing, “Why IoT Devices Are Becoming A Trend,” *www.redapt.com*, Dec. 13, 2017. <https://www.redapt.com/blog/why-iot-devices-are-becoming-a-trend> (accessed Mar. 01, 2023).
- [3] L. Vailshery, “IoT connected devices worldwide 2019-2030,” *Statista*, May 2022. <https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/> (accessed Mar. 01, 2023).
- [4] J. Fruhlinger, “What is IoT? The internet of things explained,” *Network World*, May 13, 2020. <https://www.networkworld.com/article/3207535/what-is-iot-the-internet-of-things-explained.html> (accessed Mar. 01, 2023).
- [5] Clare Stouffer, “What is a zero-day exploit?,” *us.norton.com*, Sep. 03, 2021. <https://us.norton.com/blog/emerging-threats/how-do-zero-day-vulnerabilities-work#> (accessed Mar. 01, 2023).
- [6] J. Fruhlinger, “The Mirai botnet explained: How IoT devices almost brought down the internet,” *CSO Online*, Mar. 09, 2018. <https://www.csoonline.com/article/3258748/the-mirai-botnet-explained-how-teen-scammers-and-cctv-cameras-almost-brought-down-the-internet.html> (accessed Mar. 01, 2023).
- [7] K. Young, “Cyber Case Study: The Mirai DDoS Attack on Dyn,” *CoverLink Insurance - Ohio Insurance Agency*, Jan. 10, 2022. <https://coverlink.com/case-study/mirai-ddos-attack-on-dyn/> (accessed Mar. 01, 2023).
- [8] D. Astudillo, X. Riofrio, T. Oquendo, and J. Merchan-Lima, “Zero-day attack: Deployment and evolution,” vol. VIII, pp. 38–53, Jan. 2021.
- [9] M. Ge, N. F. Syed, X. Fu, Z. Baig, and A. Robles-Kelly, “Towards a deep learning-driven intrusion detection approach for Internet of Things,” *Computer Networks*, vol. 186, p. 107784, Feb. 2021, doi: <https://doi.org/10.1016/j.comnet.2020.107784>.

REFERENCES

- [10] G. Apruzzese, M. Andreolini, M. Marchetti, A. Venturi, and M. Colajanni, “Deep Reinforcement Adversarial Learning against Botnet Evasion Attacks,” *IEEE Transactions on Network and Service Management*, vol. 17, no. 4, pp. 1–1, 2020, doi: <https://doi.org/10.1109/tnsm.2020.3031843>.
- [11] S. I. Popoola, B. Adebisi, R. Ande, M. Hammoudeh, K. Anoh, and A. A. Atayero, “SMOTE-DRNN: A Deep Learning Algorithm for Botnet Detection in the Internet-of-Things Networks,” *Sensors*, vol. 21, no. 9, p. 2985, Apr. 2021, doi: <https://doi.org/10.3390/s21092985>.
- [12] S. I. Popoola, B. Adebisi, M. Hammoudeh, G. Gui, and H. Gacanin, “Hybrid Deep Learning for Botnet Attack Detection in the Internet of Things Networks,” *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 1–1, 2020, doi: <https://doi.org/10.1109/jiot.2020.3034156>.
- [13] S. I. Popoola, B. Adebisi, R. Ande, M. Hammoudeh, and A. A. Atayero, “Memory-Efficient Deep Learning for Botnet Attack Detection in IoT Networks,” *Electronics*, vol. 10, no. 9, p. 1104, Jan. 2021, doi: <https://doi.org/10.3390/electronics10091104>.
- [14] O. Alkadi, N. Moustafa, B. Turnbull, and K.-K. Raymond Choo, “A Deep Blockchain Framework-Enabled Collaborative Intrusion Detection for Protecting IoT and Cloud Networks,” *Utar.edu.my*, 2023. <https://ieeexplore-ieee-org.libezp2.utar.edu.my/document/9098927> (accessed Mar. 20, 2023).
- [15] Z. Chen, N. Lv, P. Liu, Y. Fang, K. Chen, and W. Pan, “Intrusion Detection for Wireless Edge Networks Based on Federated Learning,” *IEEE Access*, vol. 8, pp. 217463–217472, 2020, doi: <https://doi.org/10.1109/access.2020.3041793>.
- [16] B. Weinger, J. Kim, A. Sim, M. Nakashima, N. Moustafa, and K. J. Wu, “Enhancing IoT anomaly detection performance for federated learning,” *Digital Communications and Networks*, vol. 8, no. 3, Mar. 2022, doi: <https://doi.org/10.1016/j.dcan.2022.02.007>.
- [17] N. A. Al-Athba Al-Marri, B. S. Ciftler, and M. M. Abdallah, “Federated Mimic Learning for Privacy Preserving Intrusion Detection,” *2020 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, May 2020, doi: <https://doi.org/10.1109/blackseacom48709.2020.9234959>.

REFERENCES

- [18] V. Mothukuri, P. Khare, R. M. Parizi, S. Pouriye, A. Dehghantanha, and G. Srivastava, "Federated Learning-based Anomaly Detection for IoT Security Attacks," *IEEE Internet of Things Journal*, vol. 9, no. 5, pp. 1–1, 2021, doi: <https://doi.org/10.1109/jiot.2021.3077803>.
- [19] Y. Fan, Y. Li, M. Zhan, H. Cui, and Y. Zhang, "IoTDefender: A Federated Transfer Learning Intrusion Detection Framework for 5G IoT," *2020 IEEE 14th International Conference on Big Data Science and Engineering (BigDataSE)*, Dec. 2020, doi: <https://doi.org/10.1109/bigdatase50710.2020.00020>.
- [20] B. Cetin, A. Lazar, J. Kim, A. Sim, and K. Wu, "Federated Wireless Network Intrusion Detection," *2019 IEEE International Conference on Big Data (Big Data)*, Dec. 2019, doi: <https://doi.org/10.1109/bigdata47090.2019.9005507>.
- [21] X. Hei, X. Yin, Y. Wang, J. Ren, and L. Zhu, "A trusted feature aggregator federated learning for distributed malicious attack detection," *Computers & Security*, vol. 99, p. 102033, Dec. 2020, doi: <https://doi.org/10.1016/j.cose.2020.102033>.
- [22] T. M. Booij, I. Chiscop, E. Meeuwissen, N. Moustafa, and F. T. H. den Hartog, "ToN_IoT: The Role of Heterogeneity and the Need for Standardization of Features and Attack Types in IoT Network Intrusion Data Sets," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 485–496, Jan. 2022, doi: <https://doi.org/10.1109/jiot.2021.3085194>.
- [23] Stratosphere Lab, "IoT-23 Dataset: A labeled dataset of Malware and Benign IoT Traffic.," *Stratosphere IPS*. <https://www.stratosphereips.org/datasets-iot23>
- [24] S. Garcia, A. Parmisano, and M. J. Erquiaga, "IoT-23: A labeled dataset with malicious and benign IoT network traffic," *Zenodo*, Jan. 20, 2020. https://zenodo.org/record/4743746#.ZEc_Z3ZBzZR (accessed Mar. 25, 2023).
- [25] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset," *Future Generation Computer Systems*, vol. 100, pp. 779–796, Nov. 2019, doi: <https://doi.org/10.1016/j.future.2019.05.041>.
- [26] Y. Meidan *et al.*, "N-BaIoT—Network-Based detection of IoT botnet attacks using deep autoencoders," *IEEE Pervasive Computing*, vol. 17, Art. no. 3, 2018, doi: <https://doi.org/10.1109/MPRV.2018.03367731>.

REFERENCES

- [27] J. Kim, M. Shim, S. Hong, Y. Shin, and E. Choi, “Intelligent Detection of IoT Botnets Using Machine Learning and Deep Learning,” *Applied Sciences*, vol. 10, no. 19, p. 7009, Oct. 2020, doi: <https://doi.org/10.3390/app10197009>.
- [28] Koo Ping Shung, “Accuracy, Precision, Recall or F1?,” *Towards Data Science*, Mar. 15, 2018. <https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9> (accessed Mar. 24, 2023).
- [29] shruti saxena, “Precision vs Recall,” *Medium*, May 13, 2018. <https://medium.com/@shrutisaxena0617/precision-vs-recall-386cf9f89488> (accessed Mar. 24, 2023).
- [30] J. Jordan, “Evaluating a machine learning model.,” *Jeremy Jordan*, Jul. 21, 2017. <https://www.jeremyjordan.me/evaluating-a-machine-learning-model/#:~:text=The%20three%20main%20metrics%20used> (accessed Mar. 24, 2023).
- [31] Dhiraj K, “Anomaly Detection Using Isolation Forest in Python,” *Paperspace Blog*, Mar. 02, 2020. <https://blog.paperspace.com/anomaly-detection-isolation-forest/>
- [32] F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation Forest,” *2008 Eighth IEEE International Conference on Data Mining*, Dec. 2008, doi: <https://doi.org/10.1109/icdm.2008.17>.
- [33] Akshara_416, “Isolation Forest | Anomaly Detection with Isolation Forest,” *Analytics Vidhya*, Jul. 26, 2021. <https://www.analyticsvidhya.com/blog/2021/07/anomaly-detection-using-isolation-forest-a-complete-guide/>
- [34] B. Zhang and Y. Bao, “Cross-Dataset Learning for Age Estimation,” *IEEE Access*, vol. 10, pp. 24048–24055, 2022, doi: <https://doi.org/10.1109/access.2022.3154403>.
- [35] L. Cao, Z. Liu, and T. S. Huang, “Cross-dataset action detection,” *Computer Vision and Pattern Recognition*, Jun. 2010, doi: <https://doi.org/10.1109/cvpr.2010.5539875>.
- [36] Y. Yao, Y. Wang, Y. Guo, J. Lin, H. Qin, and J. Yan, “Cross-dataset Training for Class Increasing Object Detection,” *arXiv.org*, Jan. 13, 2020. <https://arxiv.org/abs/2001.04621> (accessed Jul. 12, 2023).
- [37] S. Cristina, “The Transformer Model,” *Machine Learning Mastery*, Nov. 03, 2021. <https://machinelearningmastery.com/the-transformer-model/>

REFERENCES

- [38] S. Cristina, “The Transformer Attention Mechanism,” *MachineLearningMastery.com*, Sep. 14, 2022. <https://machinelearningmastery.com/the-transformer-attention-mechanism>

APPENDIX A

A.1 Weekly Report

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.: 1
Student Name & ID: Oh Jia Sheng (21ACB01918)	
Supervisor: Dr Aun Yichiet	
Project Title: Zero Day Detection on IoT Network	

1. WORK DONE

During this week, I initiated the project by diving into Chapter 1, which covers the Project Background. Additionally, I had a productive discussion with my supervisor to ensure alignment and clarify any potential modifications needed for the project.

2. WORK TO BE DONE

In the upcoming week, I plan to continue refining Chapter 1 and incorporating any feedback or insights gained from the discussion with my supervisor.

3. PROBLEMS ENCOUNTERED

No significant problems were encountered during this week.

4. SELF EVALUATION OF THE PROGRESS

I believe I made a good start to the project, establishing a solid foundation for the subsequent chapters. The interaction with my supervisor was particularly helpful in shaping the project's direction.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.:2
Student Name & ID: Oh Jia Sheng (21ACB01918)	
Supervisor: Dr Aun Yichiet	
Project Title: Zero Day Detection on IoT Network	

1. WORK DONE

In this two-week period, I dedicated my efforts to Chapter 2, focusing on the Literature Review. I conducted extensive research on relevant papers to ensure a comprehensive understanding of the project's context.

2. WORK TO BE DONE

In the next week, I will continue to expand upon the Literature Review, incorporating key findings and insights into the project.

3. PROBLEMS ENCOUNTERED

No significant problems were encountered during this period.

4. SELF EVALUATION OF THE PROGRESS

Progressing well with the Literature Review, I've been able to gather valuable information that will inform the subsequent chapters of the project.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.: 3
Student Name & ID: Oh Jia Sheng (21ACB01918)	
Supervisor: Dr Aun Yichiet	
Project Title: Zero Day Detection on IoT Network	

1. WORK DONE

During this week, I shifted my focus to Chapter 3, specifically addressing the system model and federated learning framework. I began outlining the System Models, which will be a crucial part of this chapter.

2. WORK TO BE DONE

In the next week, I aim to complete the System Model and delve into System Design.

3. PROBLEMS ENCOUNTERED

No significant problems were encountered during this week.

4. SELF EVALUATION OF THE PROGRESS

Making steady progress in Chapter 3, I've started to articulate the unique aspects of the proposed methodology, setting the stage for the subsequent sections.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.: 4
Student Name & ID: Oh Jia Sheng (21ACB01918)	
Supervisor: Dr Aun Yichiet	
Project Title: Zero Day Detection on IoT Network	

1. WORK DONE

This week, I continued working on Chapter 3 by further detailing the System Model. Additionally, I began addressing the Zero-day attack models and federated learning framework that are integral to the project.

2. WORK TO BE DONE

In the upcoming week, I will continue to expand upon the System Design section.

3. PROBLEMS ENCOUNTERED

During the implementation, we faced obstacles of finding the best way to generate attack like Ton-IoT dataset for the zero-day attack detection on IoT network.

4. SELF EVALUATION OF THE PROGRESS

I need to work harder to solve the problems encountered in this week.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT*(Project II)*

Trimester, Year: Y3T3	Study week no.: 5
Student Name & ID: Oh Jia Sheng (21ACB01918)	
Supervisor: Dr Aun Yichiet	
Project Title: Zero Day Detection on IoT Network	

1. WORK DONE

Over the past two weeks, I continued to refine Chapter 3. Also, I've solved the problem encountered last week, and we planned to use back the sample in Ton-IoT dataset to simulate the attack as the solution.

2. WORK TO BE DONE

In the next week, I plan to conclude Chapter 3 and transition to Chapter 4, where I will embark on the System Design.

3. PROBLEMS ENCOUNTERED

No significant problems were encountered during this week.

4. SELF EVALUATION OF THE PROGRESS

So far everything is on track.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.: 6
Student Name & ID: Oh Jia Sheng (21ACB01918)	
Supervisor: Dr Aun Yichiet	
Project Title: Zero Day Detection on IoT Network	

1. WORK DONE

This week, I commenced work on Chapter 4, specifically the Model Flow. I also continued to refine the project's coding aspects.

2. WORK TO BE DONE

In the upcoming week, I will continue to make substantial progress in Chapter 4, including Federated Learning Framework section.

3. PROBLEMS ENCOUNTERED

No significant problems were encountered during this week.

4. SELF EVALUATION OF THE PROGRESS

The initiation of Chapter 4 signifies the project's transition into the implementation phase. Progress is steady, and I remain on schedule to meet project milestones.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT*(Project II)*

Trimester, Year: Y3T3	Study week no.: 7
Student Name & ID: Oh Jia Sheng (21ACB01918)	
Supervisor: Dr Aun Yichiet	
Project Title: Zero Day Detection on IoT Network	

1. WORK DONE

During this week, I made significant strides in Chapter 4 by continuing to work on the Federated Learning Framework flow. I also reviewed and refined the System Design.

2. WORK TO BE DONE

In the next week, I will aim to complete the Federated Learning Flow of Chapter 4 and begin preparing for Chapter 5, where I will focus on the System Simulation.

3. PROBLEMS ENCOUNTERED

No significant problems were encountered during this week.

4. SELF EVALUATION OF THE PROGRESS

Progress in Chapter 4 is substantial, aligning with project timelines.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.: 8
Student Name & ID: Oh Jia Sheng (21ACB01918)	
Supervisor: Dr Aun Yichiet	
Project Title: Zero Day Detection on IoT Network	

1. WORK DONE

During this week, I successfully concluded the Federated Learning Framework flow of Chapter 4, marking a significant milestone in the project. I also began preparations for Chapter 5 & 6, which includes System Simulation and Evaluation.

2. WORK TO BE DONE

In the next week, I will continue to work on Chapter 5, ensuring that the System Simulation is well-planned, and the testing is effectively designed to convey key project findings.

3. PROBLEMS ENCOUNTERED

In this week and the upcoming weeks, I am filled with a lot of midterms and assignment submission deadlines. It's hard for me to cope with the tight schedule and arrange my time well.

4. SELF EVALUATION OF THE PROGRESS

I need to plan my weeks wisely and spend more time on my midterms and other assignments.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.: 9
Student Name & ID: Oh Jia Sheng (21ACB01918)	
Supervisor: Dr Aun Yichiet	
Project Title: Zero Day Detection on IoT Network	

1. WORK DONE

In this week, I did not do anything due to midterm preparation.

2. WORK TO BE DONE

In the next week, I will continue my work from Week 8.

3. PROBLEMS ENCOUNTERED

Packed schedule for midterms and assignment submission.

4. SELF EVALUATION OF THE PROGRESS

I need to manage my time wisely.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.: 10
Student Name & ID: Oh Jia Sheng (21ACB01918)	
Supervisor: Dr Aun Yichiet	
Project Title: Zero Day Detection on IoT Network	

1. WORK DONE

This week, I dedicated my efforts to finalizing Chapter 5&6, specifically the System Simulation and Evaluation sections.

2. WORK TO BE DONE

In the next and final week, I will conduct a thorough final check and start to work on Chapter 7 Conclusion and Recommendation section.

3. PROBLEMS ENCOUNTERED

Packed with a lot of assignment submission.

4. SELF EVALUATION OF THE PROGRESS

I need to manage my time wisely.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.: 11
Student Name & ID: Oh Jia Sheng (21ACB01918)	
Supervisor: Dr Aun Yichiet	
Project Title: Zero Day Detection on IoT Network	

1. WORK DONE

In week 11, I dedicated my efforts to finalizing Chapter 7, specifically the Conclusion and Recommendation sections.

2. WORK TO BE DONE

The upcoming weeks are primarily dedicated to making any necessary refinements and ensuring that the project is fully prepared for submission.

3. PROBLEMS ENCOUNTERED

No significant problems were encountered during this week.

4. SELF EVALUATION OF THE PROGRESS

The project is still in line with the schedule.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.: 12
Student Name & ID: Oh Jia Sheng (21ACB01918)	
Supervisor: Dr Aun Yichiet	
Project Title: Zero Day Detection on IoT Network	

1. WORK DONE

In this second final week, I conducted a meticulous final check to ensure that all project components are aligned, polished, and ready for submission. This included a comprehensive review of the Conclusion, Poster, and all preceding chapters.

2. WORK TO BE DONE

The final week is primarily dedicated to making any necessary refinements and ensuring that the project is fully prepared for submission.

3. PROBLEMS ENCOUNTERED

No significant problems were encountered during this week.

4. SELF EVALUATION OF THE PROGRESS

The project is on the verge of successful completion, with all components in their final stages. The comprehensive final check serves as the last step in ensuring that the project meets its objectives and is ready for evaluation.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.: 13
Student Name & ID: Oh Jia Sheng (21ACB01918)	
Supervisor: Dr Aun Yichiet	
Project Title: Zero Day Detection on IoT Network	

1. WORK DONE

During this concluding week, I focused on the finalization and submission of the project. This included a thorough review of all project components to ensure alignment and completeness. I also prepared and organized the project documentation and files for submission.

2. WORK TO BE DONE

In the coming weeks, my main task will be dedicated to preparing for the FYP presentation. This will involve creating an effective and engaging presentation that highlights the key aspects of the project. I will also rehearse and refine my presentation skills to ensure a confident and compelling delivery.

3. PROBLEMS ENCOUNTERED

No significant problems were encountered during this week.

4. SELF EVALUATION OF THE PROGRESS

The project is now complete and ready for submission. Looking back on the project's development over the past weeks and months, I am satisfied with the progress made and the dedication put into this endeavour. I eagerly await the evaluation and feedback, and I am prepared for any potential future work or presentations related to this project.



Supervisor's signature



Student's signature

A.2 Poster



FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY (FICT)

Author: Oh Jia Sheng Supervisor: Dr Aun Yichiet

ZERO-DAY DETECTION ON IOT NETWORK

This project developed a federated learning-based solution that enhances the early detection of zero-day attacks on IoT devices while preserving user privacy and minimizing network traffic.

MOTIVATION

- Zero-day attacks exploit unknown IoT vulnerabilities
 - -> Zero Day Detection
- Slow patching allows attackers to strike
 - -> Federated Learning
- Traditional methods need centralized data, raising privacy concerns
 - -> Federated Learning

OBJECTIVE

- Develop a federated learning solution for zero-day attack detection on IoT devices.
- Enhance early detection of these attacks without central data transfer.

CONTRIBUTION

- Improved User Privacy
- Reduced Network Traffic
- Enhanced Accuracy
- Faster Response

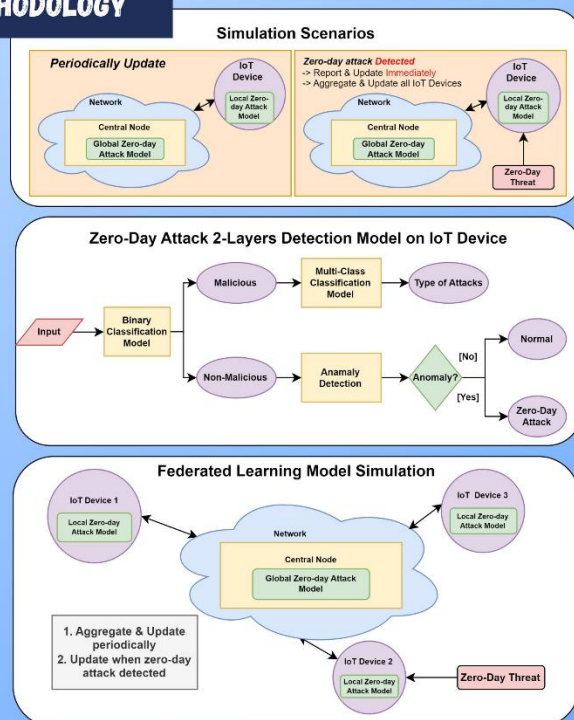
METHODOLOGY

PROJECT SCOPE

- Research on relevant paper
- Design 2 Layer Classification Models
- Simulate the Federated Learning and conduct experiment
- Evaluate the Performance of the Proposed Method

CONCLUSION

- Federated Learning 2 Layers Zero-day Attack Detection for IoT Security is Feasible
- Protect User Privacy
- Enhanced Accuracy
- Faster Response
- Can be used in any kind of classification models



PLAGIARISM CHECK RESULT

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1 of 1



FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	OH JIA SHENG
ID Number(s)	21ACB01918
Programme / Course	FICT CS
Title of Final Year Project	ZERO-DAY DETECTION ON IOT NETWORKS

Similarity	Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)
Overall similarity index: <u>16</u> % Similarity by source Internet Sources: <u>10</u> % Publications: <u>10</u> % Student Papers: <u>7</u> %	
Number of individual sources listed of more than 3% similarity: <u>0</u>	
Parameters of originality required and limits approved by UTAR are as Follows: (i) Overall similarity index is 20% and below, and (ii) Matching of individual sources listed must be less than 3% each, and (iii) Matching texts in continuous block must not exceed 8 words <i>Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.</i>	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

Signature of Supervisor

Name: Dr Aun Yichiet

Date: 24/4/2024

Signature of Co-Supervisor

Name: _____

Date: _____

PLAGIARISM CHECK RESULT

PLAGIARISM CHECK RESULT

ORIGINALITY REPORT

16%

SIMILARITY INDEX

10%

INTERNET SOURCES

10%

PUBLICATIONS

7%

STUDENT PAPERS

PRIMARY SOURCES

1	www.researchgate.net Internet Source	1%
2	fastercapital.com Internet Source	1%
3	eprints.utar.edu.my Internet Source	<1%
4	Submitted to Universiti Tunku Abdul Rahman Student Paper	<1%
5	www.mdpi.com Internet Source	<1%
6	arxiv.org Internet Source	<1%
7	Submitted to Florida International University Student Paper	<1%
8	www.cn.ca Internet Source	<1%
9	Wen-Ting Lin, Guo Chen, Yuhan Huang. "Incentive edge-based federated learning for false data injection attack detection on power	<1%

FYP 2 CHECKLIST



UNIVERSITI TUNKU ABDUL RAHMAN
FACULTY OF INFORMATION & COMMUNICATION
TECHNOLOGY (KAMPAR CAMPUS)
CHECKLIST FOR FYP2 THESIS SUBMISSION

Student ID	21ACB01918
Student Name	OH JIA SHENG
Supervisor Name	DR AUN YICHIE

TICK (✓)	DOCUMENT ITEMS
	Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
✓	Title Page
✓	Signed Report Status Declaration Form
✓	Signed FYP Thesis Submission Form
✓	Signed form of the Declaration of Originality
✓	Acknowledgement
✓	Abstract
✓	Table of Contents
✓	List of Figures (if applicable)
✓	List of Tables (if applicable)
✓	List of Symbols (if applicable)
✓	List of Abbreviations (if applicable)
✓	Chapters / Content
✓	Bibliography (or References)
✓	All references in bibliography are cited in the thesis, especially in the chapter of literature review
✓	Appendices (if applicable)
✓	Weekly Log
✓	Poster
✓	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
✓	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.

(Signature of Student)

Date: 24/04/2024