

# **CYBER ATTACK DETECTION IN POWER SYSTEM SCADA NETWORKS USING MACHINE LEARNING TECHNIQUES**

A thesis submitted in partial fulfilment of the requirements  
for the award of the degree of

**B. Tech**

**in**

**Electrical and Electronics Engineering**

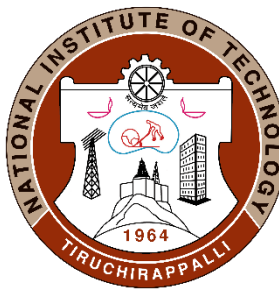
**By**

**Angad Ripudaman Singh Bajwa (107118014)**

**Mandar Burande (107118056)**

**Aditya Pethkar (107118072)**

**Priyansh Joshi (107118076)**



**ELECTRICAL AND ELECTRONICS ENGINEERING  
NATIONAL INSTITUTE OF TECHNOLOGY  
TIRUCHIRAPPALLI – 620015**

**MAY 2022**

## **BONAFIDE CERTIFICATE**

This is to certify that the project titled **CYBER ATTACK DETECTION IN POWER SYSTEM SCADA NETWORKS USING MACHINE LEARNING TECHNIQUES** is a bonafide record of the work done by

**Angad Ripudaman Singh Bajwa (107118014)**

**Mandar Burande (107118056)**

**Aditya Pethkar (107118072)**

**Priyansh Joshi (107118076)**

in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Electrical and Electronics Engineering** of the **NATIONAL INSTITUTE OF TECHNOLOGY, TIRUCHIRAPPALLI**, during the year 2021-22.

**Dr. M. JAYA BHARATHA REDDY**

Project Guide

**Dr. V. SANKARANARAYANAN**

Head of the Department

Project Viva-voce held on \_\_\_\_\_

**Internal Examiner**

**External Examiner**

## ABSTRACT

The immense increase in the demand for electrical energy has become a challenge not only for production but also for its distribution. This growing demand, along with the advancement in technology, has led to the remote control and monitoring of power grids through supervisory control and data acquisition (SCADA) system and automation at substations which decreases the cost of power transmission and increases efficiency. Any interruption in the transmission, like widespread blackouts, will cause irreparable effects on different aspects of society. However, this opportunity also may lead to a threat if cyber security is met with a lack of attention in smart grids. Transmitting data from IEDs leaves them vulnerable to cyber-attacks. Attackers can inject false data into the power system control center and prevent the operators from obtaining the actual operating conditions of the system. Thus, it is highly necessary to separate naturally occurring conditions from cyber-attacks to confirm the grid's reliability. Automatically and accurately detecting cyber-attacks in the power system remains a big challenge within the field of intelligent fault diagnosis. In this paper, we employ machine learning algorithms to differentiate cyber-attacks from naturally occurring conditions in a power system.

*Keywords:* SCADA Networks, Power Grids, Cyber-Attacks, Fault Diagnosis, Machine Learning

## ACKNOWLEDGEMENTS

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without mentioning the people who made it possible, whose constant guidance and encouragement have crowned our efforts with success.

The first and foremost person who deserves our boundless gratitude is our project guide, **Dr. M. JAYA BHARATHA REDDY**, Professor, Department of Electrical and Electronics Engineering, for his whole-hearted support, valuable guidance, discussion, and constant encouragement throughout the tenure of project work.

We would like to thank **Dr. V. SANKARANARAYANAN**, Head of the Department, Department of Electrical and Electronics Engineering, for his constant encouragement and support throughout the tenure of project work.

Our sincere thanks to **Dr. G. AGHILA**, Director, NIT Tiruchirappalli for having provided all facilities to carry out this project work.

Last but not the least; we thank our parents, the real inspiration for helping us to complete this project without much difficulty.

**Angad Ripudaman Singh Bajwa**

**Mandar Burande**

**Aditya Pethkar**

**Priyansh Joshi**

# TABLE OF CONTENTS

Title	Page No.
<b>ABSTRACT .....</b>	<b>i</b>
<b>ACKNOWLEDGEMENTS .....</b>	<b>ii</b>
<b>TABLE OF CONTENTS .....</b>	<b>iii</b>
<b>LIST OF TABLES .....</b>	<b>vi</b>
<b>LIST OF FIGURES .....</b>	<b>vii</b>
<b>ABBREVIATIONS .....</b>	<b>ix</b>
<b>CHAPTER 1            INTRODUCTION .....</b>	<b>1</b>
1.1 General .....	1
1.2 Objectives .....	3
<b>CHAPTER 2            LITRATURE REVIEW .....</b>	<b>4</b>
2.1 Cyber Security In Supervisory Control And Data Acquisition (Scada) Networks .....	4
2.1.1 General .....	4
2.1.2 Scada Network Vulnerabilities And Threats .....	4
2.1.3 Wscc 9 Bus System .....	6
2.1.4 Faults .....	7
2.1.5 Categories Of Fault .....	8
2.1.6 Fault Detection And Classification .....	9
2.2 Modbus Protocol And Cyberattacks .....	10
2.2.1 Modbus Tcp Protocol: An Overview .....	10

2.2.2 Man-In-The-Middle Attack.....	11
2.2.3 Types Of Man-In-The-Middle Attack.....	12
2.2.4 Anatomy Of An Arp Spoofing Attack .....	13
<b>CHAPTER 3            METHODOLOGIES .....</b>	<b>14</b>
3.1 Matlab Simulations Of Normal Conditions And Natural Faults.....	14
3.2 Cyber Attacks Methodology .....	15
3.2.1 Setup Modbus Client - Server Architecture .....	15
3.2.2 Encode Data To Modbus Format .....	15
3.2.3 Setup Cyber-Attack Architecture .....	16
3.2.4 Perform Man-In-The-Middle Attacks .....	16
3.2.5 Gather Cyber-Attack Data.....	16
3.3 Machine Learning Techniques .....	17
<b>CHAPTER 4            RESULTS AND ANALYSIS.....</b>	<b>20</b>
4.1 Matlab Simulation Results .....	20
4.1.1 Single Line To Ground Fault (L-G) .....	20
4.1.2 Line To Line Fault (L-L).....	23
4.1.3 Double Line To Ground Fault (L-L-G).....	25
4.1.4 All Three Phase To Ground Fault (L-L-L-G) .....	27
4.1.5 All Three Phase Short Circuited (L-L-L).....	29
4.2 Cyber-Attack Results .....	32
4.2.1 Modbus Tcp Server .....	32

4.2.2 Modbus Tcp Client.....	32
4.2.3 Encode Data To Modbus Format .....	33
4.2.4 Setup Cyber-Attack Architecture .....	34
4.2.5 Perform Man-In-The-Middle Attacks .....	35
4.2.6 Gather Cyber Attack Data .....	35
4.3 Case Study Of An Arp Poisoning Attack.....	35
4.4 Machine Learning Results.....	38
<b>CHAPTER 5 CONCLUSION .....</b>	<b>40</b>
5.1 Scope For Further Research .....	40
<b>APPENDIX.....</b>	<b>41</b>
<b>REFERENCES.....</b>	<b>44</b>

## LIST OF TABLES

Table 2.1: Types of faults and their probability of occurrence .....	8
Table 3.1: Different Line Notations .....	14
Table 3.2: Fault types and distances in each line .....	14
Table 4.1: Ettercap Filter Syntax .....	37
Table 4.2: Performance Matrix of K-Means Algorithm .....	38
Table 4.3: Performance Matrix of Random Forest Algorithm.....	38
Table 4.4: Performance Matrix of Decision Tree Classifier Algorithm .....	39



## LIST OF FIGURES

Figure 2.1 : SCADA Architecture .....	4
Figure 2.2: Single Line Diagram of a WSCC 9 Bus System .....	6
Figure 2.3: Modbus TCP Headers .....	11
Figure 2.4: Modbus Application Protocol Header (MBAP) and the Protocol Data Unit (PDU).....	11
Figure 2.5: Conceptual diagram of a typical Man-in-the-middle attack.....	12
Figure 3.1 Methodology for machine learning .....	17
Figure 4.2: Phase Current of Line 4-6 .....	21
Figure 4.1: Phase Voltage of Line 4-6 .....	21
Figure 4.3: Phase Voltage of Line 5-4 .....	22
Figure 4.4: Phase Current of line 5-4.....	22
Figure 4.5: Phase Voltage of Line 4-6 .....	23
Figure 4.6: Phase Current of Line 4-6 .....	23
Figure 4.8: Phase Current of Line 5-4 .....	24
Figure 4.7: Phase Voltage of Line 5-4 .....	24
Figure 4.9: Phase Voltage of Line 4-6 .....	25
Figure 4.10: Phase Current of Line 4-6 .....	25
Figure 4.11: Phase Voltage of Line 5-4 .....	26
Figure 4.12: Phase Current of Line 5-4 .....	26
Figure 4.13: Phase Voltage of Line 4-6 .....	27
Figure 4.14: Phase Current of Line 4-6 .....	27

Figure 4.15: Phase Voltage of Line 5-4 .....	28
Figure 4.16: Phase Current of Line 5-4 .....	28
Figure 4.17: Phase Voltage of Line 4-6 .....	29
Figure 4.18: Phase Current of Line 4-6 .....	29
Figure 4.19: Phase Voltage of line 5-4 .....	30
Figure 4.30: Phase Current of Line 5-4 .....	30
Figure 4.21 Consolidated dataset .....	31
Figure 4.22: Modbus TCP Server Code.....	32
Figure 4.23: Modbus TCP Client Code .....	33
Figure 4.24: PyModbus Payload Code .....	33
Figure 4.25: The decoded payload data, as received back by the client device. ....	34
Figure 4.26: Cyber-attack Architecture .....	34
Figure 4.27: Loading a filter in Ettercap.....	35
Figure 4.28: Unified Sniffing.....	35
Figure 4.29: Ettercap sniffing .....	36
Figure 4.30: Ettercap Scan Host .....	36
Figure 4.31: ARP Poisoning .....	36
Figure 4.32: Ettercap Filter .....	38
Figure 4.33: Confusion Matrices for Line C obtained from K-Means, Random Forest and Decision Tree Classifier respectively 0 - Natural Operation, 1 – Fault, 2- Cyber Attack.....	39

## **ABBREVIATIONS**

OL	Overhead Lines
ASAP	As Soon As Possible
RTU	Remote Terminal Unit
PMU	Phasor Measurement Unit
ML	Machine Learning
MLT	Machine Learning Techniques
ANN	Artificial Neural Network
DT	Decision Tree
SCADA	Supervisory Control And Data Acquisition
ICS	Industrial Control Systems
HMI	Human Machine Interface
PLC	Programmable Logic Controllers
CI	Critical Infrastructures
IT	Information Technologies
MTU	Master Terminal Units
ICS-CERT	Industrial Control Systems Cyber Emergency Response Team
TCP	Transmission Control Protocol
IP	Internet Protocol
ADU	Application Data Unit
PDU	Protocol Data Unit
MBAP	Modbus Application Protocol
ASCII	American Standard Code for Information Interchange
MITM	Man-in-the-middle

ARP	Address Resolution Protocol
MAC	Media Access Control
LAN	Local Area Network

# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 GENERAL**

The grids today have evolved from a network of transmission lines into a cyber physical system where bi-directional information flow plays an important part in the decision-making process. Integration of high-speed communication links to the existing power network has made it more dynamic in terms of demand response and other energy management capabilities.

However, increased reliance on information networks has rendered the system vulnerable to threats associated with security breaches in communication networks. The NIST Smart Grid interoperability panel has identified Availability, Integrity, and Confidentiality as the high-level security requirements for the proper functioning of the grid. Availability ensures timely and reliable access to information; Integrity implies protection against illegal and improper modification of information and Confidentiality prevents unauthorized access to proprietary information.

In the paradigm of Smart Grids security threats can come from malicious users violating communication protocols attempting to impersonate, access, and manipulate the system components and data. In a Spoofing attack, the attacker's device takes advantage of open address fields in MAC frame, impersonates a healthy device from the network and sends false data to other devices connected to the network. The other kind of attacks aimed at stealthily modifying data at substations or control centers are referred to as Data Integrity Attacks. Telemetered data such as power injections, line flows, voltage measurements and the status information of breakers and switches are vulnerable to such attacks. An integrity attack on telemetered data can jeopardize the stable operation of the power grid.

The power transmission network is the most vital link in the country's energy system because it carries massive quantities of power from generators to substations at high voltages. The modern power system network is a complex network that needs a high-speed, accurate, and reliable system of protection. Faults in the power system are inevitable, and as the configurations become more complex, the complexity of protecting transmission line configurations increases, predicting faults (type and

location) with considerable accuracy, therefore, improves the system's operational reliability and stability and helps prevent colossal power failure. With 85%-87% of failures occurring on distribution lines, power quality has become a major concern in power system engineering in recent years. Still, it is necessary to predict the line faults in advance to overcome the above-said problems.

Digital technology was introduced with the introduction of a smart grid enabling the installation of sensors along with the transmission lines that can capture live fault data because they present the useful data that can be used to detect abnormalities in transmission lines. A substantial amount of heterogeneous data continuously collected by the growing number of distributed low-cost and high-quality sensors, such as Remote Terminal Units (RTU), Phasor Measurement Units (PMU), and smart meters, along with those generated by other measuring devices is required for the operational control and performance analysis of smart grids. Conventional time-domain techniques are inefficient in computational terms and will not meet real-time application specifications. Application of machine learning algorithms on the transmission line for fault detection, classification and location identification has been explored during this research. We can learn without direct programming from the data and, once exposed to new data, can respond independently.

Most researchers believe that the approach of machine learning (ML) like artificial neural networks (ANN), decision trees (DT), deep learning models, etc. is capable of providing relevant information on safety in power systems. With the introduction of the smart grid, the operation, monitoring, and regulation of the power system network is becoming smarter and supported by machines. In line with the fundamental goals and point of view of the power system network, recognition of line fault patterns and clearance of faults must be done more intelligently, judiciously, and automatically, with less intrusion from the operator. The boundless extent of power systems network and application requires improving suitable fault classification techniques in transmission systems, increasing system efficiency, and avoiding significant damage. The paper analyses the scientific literature and summarizes the most relevant approaches which will be applied in power transmission systems to fault identification methodologies.

## **1.2 OBJECTIVES**

The Objectives of our project are as follows

1. To studying in detail, monitor and analyze real-time data flow in Supervisory Control And Data Acquisition (SCADA) network.
2. To simulate events like line faults, line maintenance and collecting relevant data.
3. To detect and thwart various incoming cyber-attacks such as man-in-the-middle and remote tripping commands, etc.
4. To apply various Machine Learning Techniques (MLTs) on the collected data to generate inferences.

## CHAPTER 2

### LITRATURE REVIEW

#### 2.1 CYBER SECURITY IN SUPERVISORY CONTROL AND DATA ACQUISITION (SCADA) NETWORKS

##### 2.1.1 GENERAL

Supervisory Control and Data Acquisition (SCADA) systems are Industrial Control Systems (ICS) broadly utilized by industries to study and control different processes such as oil and gas pipelines, water distribution, power supply grids, etc. These systems provide automated control and remote monitoring of services being employed in everyday life. For example, states and municipalities use SCADA systems to observe and regulate water levels in reservoirs, pipe pressure, and water distribution.

A typical SCADA system comprises of components like computer workstations, Human Machine Interface (HMI), Programmable Logic Controllers (PLCs), sensors, and actuators. Historically, these systems had private and dedicated networks. However, because of the wide-range deployment of remote management, open IP networks (e.g., Internet) are now used for SCADA systems communication. This exposes SCADA systems to the cyberspace and makes them prone to cyber-attacks using the web.

##### 2.1.2 SCADA NETWORK VULNERABILITIES AND THREATS

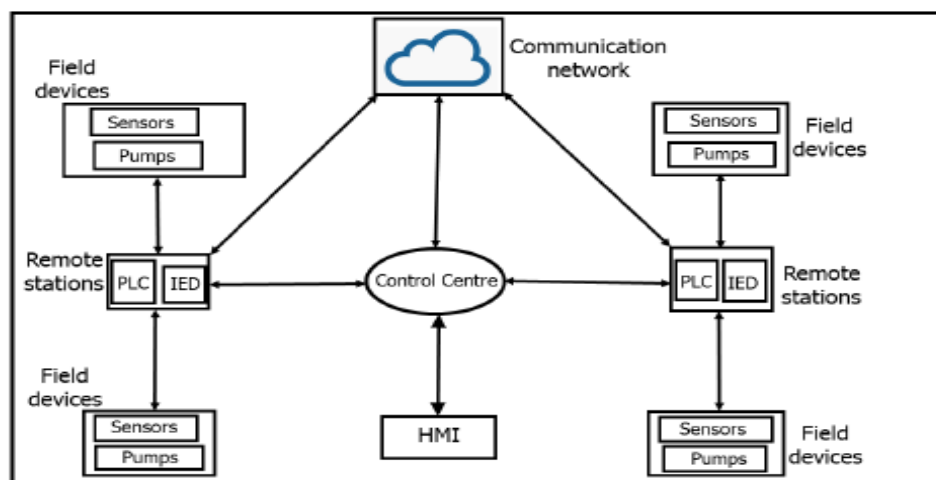


Figure 2.1 : SCADA Architecture



Critical infrastructures (CI) like the power system, oil and gas pipelines, water distribution, etc. are monitored and controlled by SCADA systems which combine the CI together as a network through advanced Information Technologies (IT). As shown in Figure 2.1, the SCADA system architecture for the electricity grid basically comprises of four major operational parts namely:

- The "Field" devices like sensors for sensing the status of SCADA equipment under concern (power level, pressure, etc.) and control them as per the received commands.
- The "Remote Station" devices which include the Remote Terminal Units (RTUs) and Programmable Logic Controllers. They serve the aim of sending and receiving digital data to and from the control centers and also the field devices.
- The "Control Centre" devices consisting of the Master Terminal Units (MTU) that issues command to the remote station devices.
- Human Machine Interface (HMI): devices which present processed data to operators usually via graphic user interface. With the interface, the operators can monitor, control and interact with SCADA processes.

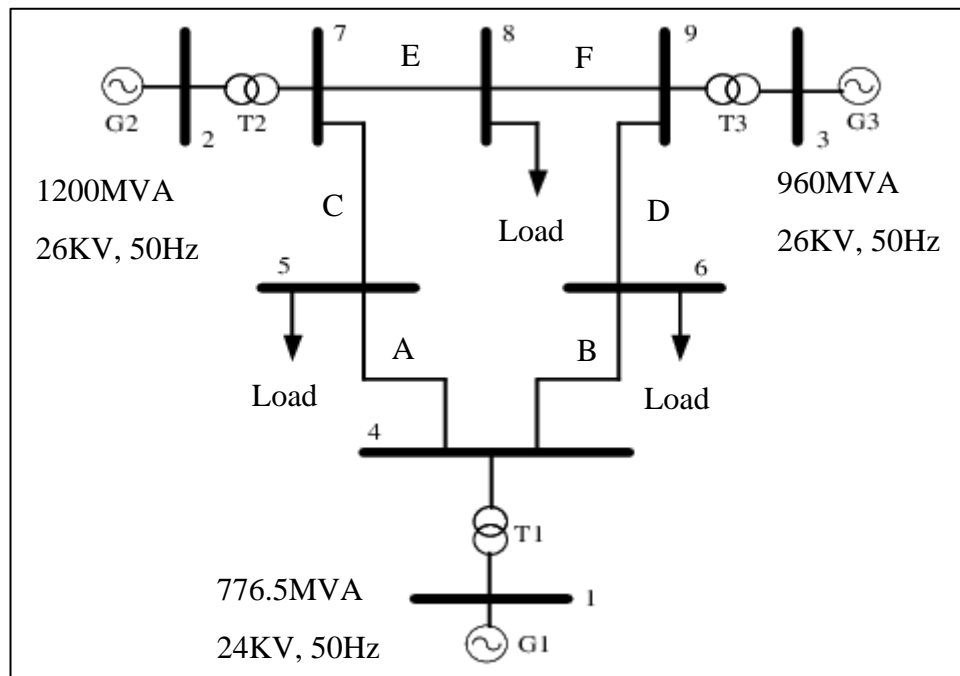
Historically, when SCADA systems were first deployed, the key threat was sabotage through the physical destruction of the utility's hardware because the old SCADA systems had private and dedicated networks that are secured by traditional air gapped separations. However, over the past 20 years, SCADA networks have been equipped with IoT devices that sometimes communicate over open channels which exposes the networks to numerous vulnerabilities and network based cyber-attacks. These threats and attacks are projected to escalate in geometric rates in the upcoming future as intruders/attackers find the energy infrastructures (arguably the most important of all CI) as a lucrative avenue to gain attention. The Industrial Control Systems Cyber Emergency Response Team (ICS-CERT) announced that, out of the 245 recorded cyber incidents on CI in 2014, seventy-nine have been focused on the energy sector.

Based on the motives and the reason of attacks, SCADA threats and attacks can be categorized as:

1. Internal/Malicious - operators, employees, or contractors with intentional motives to cause disasters to the SCADA network. For example, a well-publicized Stuxnet worm attack by a resentful engineer via a removable drive.
2. Internal/Non-malicious - operator making an accidental mistake that causes harm to power system network. For example, In 2003, Ohio Davis-Besse nuclear plant "Slammer" worm infection that led to the plant being disabled for hours.
3. External/Opportunistic - hackers seeking a challenge or fooling around.
4. External/Deliberate - this could be described as an attack by an external organized group that targets vulnerabilities in another nation/state power system like the 2015 cyberattack on Ukrainian power system network whereby the hackers were linked to Russia.

These experiences and several other reported cases have showcased the immeasurable consequences of attacks on SCADA networks.

### 2.1.3 WSCC 9 BUS SYSTEM



**Figure 2.2: Single Line Diagram of a WSCC 9 Bus System**

In Figure 2.2 shows a power system network consists of three generators. It has three load buses—Bus 5, 6 and 8. Voltage and MVA rating of generators 1, 2 and 3 are 776.5

MW and 24 kV; 1200 MVA and 26 kV; 960 MVA and 26 kV respectively. The rating of the load connected to bus 5, 6 and 7 are 115 MW and 50 MVAR; 96 MW and 30 MVAR; 110 MW and 35 MVAR respectively. Work presented here, attempts to identify type of fault, fault location and the bus parameters in pre-fault, during fault and post-fault condition of LG, LL, LLG, LLL, LLLG fault. The faults are made to occur at load buses.

#### **2.1.4 FAULTS**

Basically, the Power System network equipment or appliances are so designed in such a way to perform a non-stop required function except in case of preventive maintenance or because of lack of external sources. The fault is the random character that may appear in the power system network, and because of this inability to perform the desired function, since the fault can occur at any situation and any location within the power system, the fault is random. The balanced three-phase A.C. is the steady-state operating mode of a power system, because of adverse external and internal changes within the system, the above condition is disrupted. In any case, if the insulation of the system fails in the following particular locations like Phase conductors or Phase conductor and earth or any earthed screens surrounding the conductors, the fault will occur.

***Leading Causes of Faults:*** Faults within the power system grid occurs because of the various causes mainly it is categorized in two ways as follows:

- 1 Breakdown or failure at typical voltages due to deterioration of insulation, damage due to unpredictable causes like an unfortunate tree falling across the line, vehicles colliding with electric poles, short-circuiting by birds.
- 2 Breakdown or failure at abnormal voltages due to Lightning or Switching surges, Arcing ground.

In most cases, the chance of failure or breakdown occurs within the overhead lines (OL) due to the greater length of the conductor's exposure to the atmosphere. Transmission network or Transmission lines of the power system network used for the transportation of the majority power from the sending end to receiving end (i.e., from the generating station to the load centre) are due to its characteristics in nature it is always exposed to the all atmospheric condition either the temperature is high or low it is designed as per

recruitment using the sag calculation by the due course this conductor has the highest fault rate when put next to the other equipment within the power systems. The practical study of grid failures could be a critical issue in many power system research, like network scheduling, equipment design, and alignment of protective systems.

### 2.1.5 CATEGORIES OF FAULT

In most cases, the chance of failure or short circuit fault is the most essential and dangerous common fault that probably occurs in the power system as already discussed. These kinds of faults occur due to breakdown or failure in the insulation of current-carrying phase conductors relative to earth or in the insulation between the phases.

The fault that occurred due to a short circuit in the three-phase a.c. Power circuit is

- The Line-to-Line Fault (L-L).
- Single Line to Ground Fault (L-G).
- Double Line to Ground Fault (L-L-G).
- All the three phases to ground Fault (L-L-L-G).
- All three phases short-circuited (L-L-L).

**Table 2.1: Types of faults and their probability of occurrence**

Type of Fault	Percentage %
Single line to ground fault (L-G)	65-70
Line to Line fault (L-L)	10-15
Double line to ground fault (L-L-G)	8-10
All three phase to ground fault (L-L-L-G)	2-3
All three-phase short circuited (L-L-L)	2-3

On the type, as mentioned earlier of a fault, the first three types are said to be unbalanced operating condition because it involves only one or two-phase, and hence, it is termed as unsymmetrical faults. i.e., in short, different currents in the three phases. The last two types of faults occurred in all three phases, and so it is termed as symmetrical faults i.e., equal fault current in the three phases with  $120^\circ$ . Compared to all the above faults, the line to ground(L-G) fault is the most common fault that occurred in the OH lines, whereas the balanced three-phase fault is the rare one, but it is the severe fault which

occurred because of the carelessness of the operating personnel. Fault analysis is essential for secure and high-speed protective relaying supported by digital distance protection. Therefore, a proper evaluation of those methods is required.

#### **2.1.6 FAULT DETECTION AND CLASSIFICATION**

The techniques for detecting a fault and classifying them make use of changes in current and voltage signals just in case of fault. Techniques range from hand-coded expert-defined rules supported certain thresholds to artificial intelligence-based techniques like ANNs, vector supporting machines, and blurred decision systems. The methods vary from hand-coded and expert-defined rules based on certain thresholds to artificial intelligence-based techniques, like support vector machines, fuzzy decision systems, ANNs. Several characteristics and signal transformations were suggested and used for detection purposes, like Fourier and wavelet transformations. While protection of critical lines and system buses is ensured with local protection equipment like relays and circuit breakers, the info made available by PMUs offer the potential to extend understanding and situational awareness during a power management centre as also suggested using the output of a PMU-only state estimator for detection and classification of faults. during this context, the approaches in use decision trees, and employs support vector machines for this purpose. Such methods presume, as discussed above, the entire presence of all the measurements fully synchronization, given the promising results provided in these works. within the scope of this work, we've experimented with two fault detectors for the output of a PMU-only state estimator: one based on ANN and the other based on support vector machines. due to the observed superior performance of ANN and space limitations, we restrict our discussion and findings with ANNs within the following to detect and identify faults. Further work is ongoing for a comparison of various machine learning-based techniques for installation fault detection and classification.

## **2.2 MODBUS PROTOCOL AND CYBERATTACKS**

### **2.2.1 Modbus TCP Protocol: An Overview**

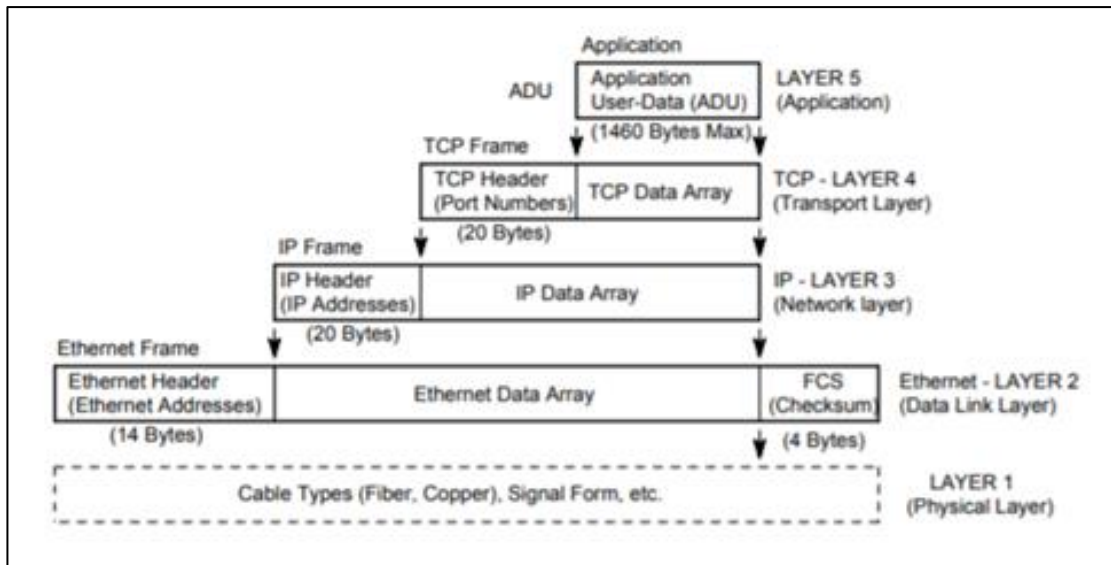
Modbus is a serial communications protocol created by Modicon in 1979 for use with its programmable logic controllers (PLCs) used in factory production, traffic lights, elevators, conveyor belts, substation automation, and other applications.

Modbus TCP is a master-slave communication protocol that works on the application layer of the TCP IP architecture. The slave station and the master station can communicate in a one-to-one or many-to-many manner. The Modbus TCP Application Data Unit (ADU) is made up of a 7-byte protocol header and an N-byte protocol data unit (PDU).

To demonstrate the efficiency of the test bed in carrying out cyber-attacks on a power system protocol, the Modbus TCP protocol was employed as the reference protocol. Modbus TCP was chosen for the following reasons:

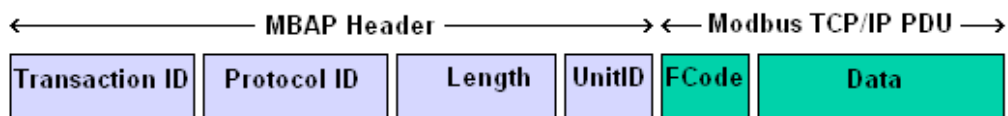
- In power systems, Modbus is extensively used.
- Modbus/TCP is easy to use and set up.
- Modbus protocol libraries for smart grid applications are freely accessible for utilities to leverage.

Modbus is still one of the most widely used protocols for field device communication, notwithstanding its antiquity. As a result, it's no surprise that attackers would try to manipulate an ICS environment via this protocol. Ethernet (Modbus TCP) has become the de facto standard for both corporate enterprise systems and factory networking (Modbus, 2017). The Ethernet header, IP header, TCP header, Modbus TCP, and data are all part of the Modbus TCP protocol. These headers are shown in Figure 2.3.



**Figure 2.3: Modbus TCP Headers**

Ethernet allows the TCP/IP wrapper to introduce the Modbus TCP protocol, historically done via serial lines, to unwrap the layers until it reaches the Modbus TCP header. The Modbus TCP header and its data include the Modbus Application Header (MBAP) and the Protocol Data Unit (PDU) as seen in Figure 2.4.



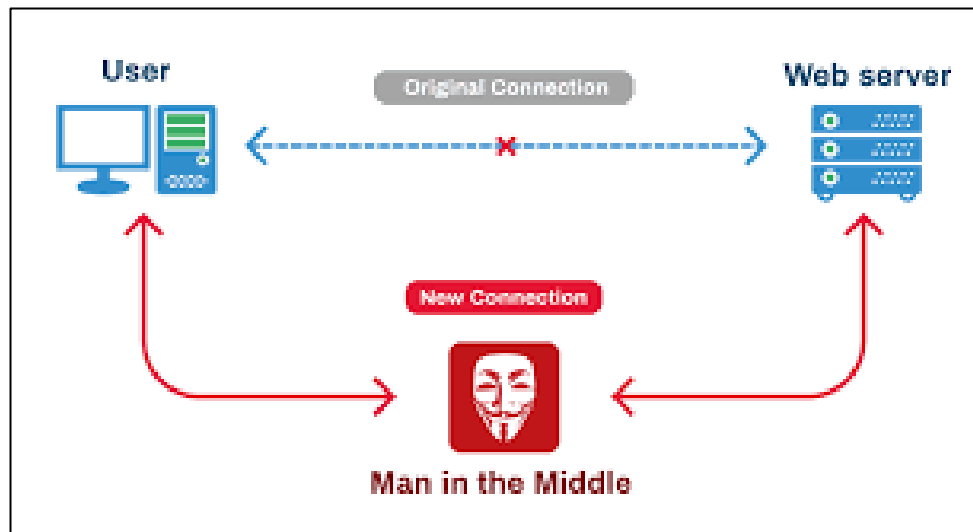
**Figure 2.4: Modbus Application Protocol Header (MBAP) and the Protocol Data Unit (PDU)**

### 2.2.2 Man-in-the-middle attack

A man-in-the-middle (MITM) attack is a type of cyberattack where attackers intercept an existing conversation or data transfer, either by eavesdropping or by pretending to be a legitimate participant. To the victim, it will appear as though a standard exchange of information is underway — but by inserting themselves into the "middle" of the conversation or data transfer, the attacker can quietly hijack information.

It is often used to steal login credentials or personal information, spy on victims, sabotage communications, or corrupt data. MITM attacks are one of the oldest forms of cyberattack. Computer scientists have been looking at ways to prevent threat actors

tampering or eavesdropping on communications since the early 1980s. Because MITM attacks are carried out in real time, they often go undetected until it's too late.



**Figure 2.5: Conceptual diagram of a typical Man-in-the-middle attack**

### **2.2.3 Types of Man-in-the-middle attack**

#### *Active session attack:*

During an active session attack, the attacker stops the original Client from communicating with the Server and then replaces himself within the session. From this point onwards, the attacker will be communicating with the Server, and they will be able to do anything that a normal user can do. He could tamper the data or collect sensitive information that may have a potential impact later.

#### *Passive session attack:*

In a passive session attack, the attacker monitors the data flowing across the network without interrupting the actual communication. The intruder eavesdrops the communication but does not modify the message stream in any way. He could collect all the data passing through the network which may cause an active attack later.

#### *ARP Poisoning:*

It is a technique by which an attacker sends spoofed Address Resolution Protocol (ARP) messages onto a local area network. Generally, the aim is to associate the attacker's MAC address with the IP address of another host, such as the default gateway, causing any traffic meant for that IP address to be sent to the attacker instead.



#### **2.2.4 Anatomy of an ARP Spoofing attack**

The basic anatomy behind an ARP poisoning attack is to exploit the lack of authentication in the ARP protocol by sending spoofed ARP messages onto the network. ARP poisoning attacks can be run from a compromised host on the LAN, or from an attacker's machine that is connected directly to the target LAN.

An attacker using ARP poisoning will disguise as a host to the transmission of data on the network between the users. Then users would not know that the attacker is not the real host on the network.

Generally, the goal of the attack is to associate the attacker's host MAC address with the IP address of a target host, so that any traffic meant for the target host will be sent to the attacker's host. The attacker may choose to inspect the packets (spying), while forwarding the traffic to the actual default destination to avoid discovery, modify the data before forwarding it (man-in-the-middle attack), or launch a denial-of-service attack by causing some or all the packets on the network to be dropped.

## CHAPTER 3

### METHODOLOGIES

#### 3.1 MATLAB SIMULATIONS OF NORMAL CONDITIONS AND NATURAL FAULTS

The WSCC -9 bus system was implemented in MATLAB and serves as the baseline model for all simulations. All naturally occurring events like load changes as well faults were simulated in this. Data was collected from each of the circuit breakers present in the line. This provided instantaneous snapshots of the entire bus system. The data was then extracted to an excel sheet for each line. Five different types of faults- LG, LL, LLG, LLL and LLLG faults were simulated in all the lines. These faults were simulated in each line at varying. The table 3.1 (Refer Fig 2.2) indicates the faults in each line and the distances at which they have occurred. Through this process, data was collected for natural events from the system.

**Table 3.1: Different Line Notations**

Notation	Buses
a	4-5
b	4-6
c	5-7
d	6-9
e	7-8
f	8-9

**Table 3.2: Fault types and distances in each line**

Fault Location	LG	LLG	LL	LLL	LLLG
a	20-130,40-110,60-90	20-130,40-110,60-90	20-130,40-110,60-90	20-130,40-110,60-90	20-130,40-110,60-90
b	20-100,40-80,60-60	20-100,40-80,60-60	20-100,40-80,60-60	20-100,40-80,60-60	20-100,40-80,60-60

<b>c</b>	20-100,40-80,60-60	20-100,40-80,60-60	20-100,40-80,60-60	20-100,40-80,60-60	20-100,40-80,60-60
<b>d</b>	20-120,40-100,60-80	20-120,40-100,60-80	20-120,40-100,60-80	20-120,40-100,60-80	20-120,40-100,60-80
<b>e</b>	20-90,40-70,60-50	20-90,40-70,60-50	20-90,40-70,60-50	20-90,40-70,60-50	20-90,40-70,60-50
<b>f</b>	20-90,40-70,60-50	20-90,40-70,60-50	20-90,40-70,60-50	20-90,40-70,60-50	20-90,40-70,60-50

## 3.2 CYBER ATTACKS METHODOLOGY

### 3.2.1 Setup Modbus Client - Server architecture

The Modbus setup consists of a two-tier architecture in which a presentation layer or interface runs on a client, and a data layer or data structure gets stored on a server. In our case, the PLCs or RTUs are the Client which send data or make requests to the Server which is the Control Centre consisting of Master Terminal Units (MTUs).

The library used for this is the PyModbus v3.0.0, a full Modbus protocol implementation using twisted/tornado/asyncio for its asynchronous communications core. Since the library is written in python, it allows for easy scripting and/or integration into their existing solutions.

### 3.2.2 Encode data to Modbus format

Data encoded in the Modbus format contains various headers and protocol syntax requirements to enable it to be transmitted over the internet in the form of Modbus Payload. This can be achieved using the PyModbus library and its built-in functions to encode and decode the data and its corresponding payload.

This encoded payload is sent from the Client to the Server over the internet and stored in the holding registers. The Client then sends a GET or Fetch request to the Server and receives the encoded data from the database. This is payload is then finally decoded back to raw data using PyModbus.

### **3.2.3 Setup cyber-attack architecture**

The cyber-attack architecture consists of three machines, real or virtual. A communication is established between two of the devices over the internet via the Modbus protocol. A third machine will perform the cyber-attack by inserting itself in between the other two in such a way that all the communication is routed through it.

### **3.2.4 Perform Man-in-the-middle attacks**

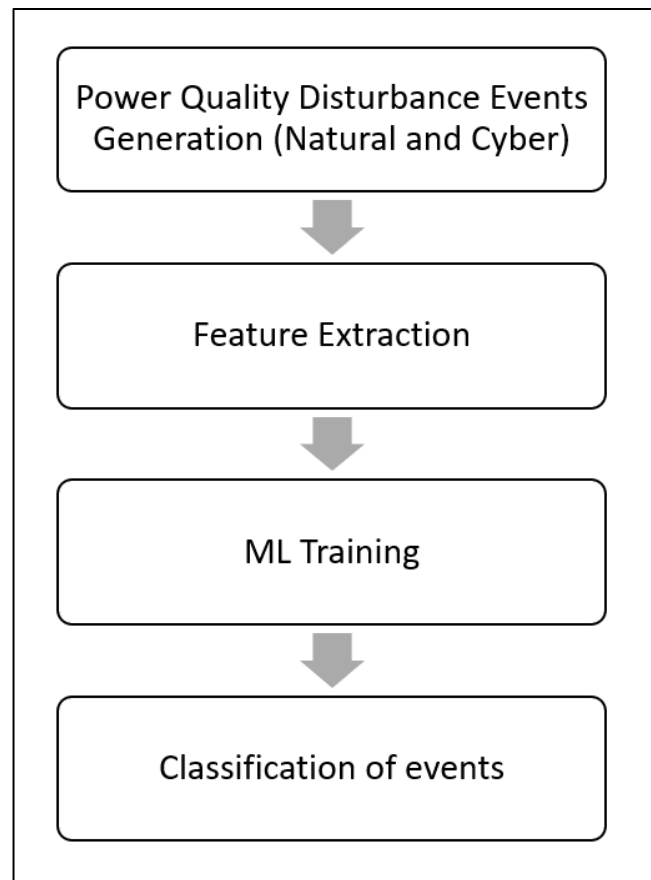
A MITM attack was performed with an Ettercap tool that maliciously modified the Modbus TCP commands between the Master and PLC workstations. Weaknesses of the Modbus TCP that were targeted by the Ettercap tool involved sending commands in clear text and lack of authentication within the Modbus TCP protocol. The Attacker workstation then uses the MAC addresses provided by the Ettercap scan for a MITM with ARP poisoning (ARP spoofing) to send falsified ARP messages. The ARP spoofing results in the linking of the attacker's MAC address with the IP address of a legitimate computer or Server on the network.

### **3.2.5 Gather cyber-attack data**

Generate and gather the data produced from conducting cyber-attacks on the network. Perform data wrangling, feature extraction, etc on the gathered data to prepare it for the machine learning model.

### 3.3 MACHINE LEARNING TECHNIQUES

The data collected from the natural events and cyber-attacks were first combined and consolidated. This was followed by performing feature extraction on this data. Pre-processing is the initial step for solving problems related to machine learning. It is a method of converting raw data into a format suitable for training the machine learning model. Raw data is noisy, incomplete, and inconsistent. Figure 3.1 indicates the steps followed from data generation to classification of events.



**Figure 3.1 Methodology for machine learning**

Pre-processing steps:

- *Handling of missing or NaN values*

Identifying and coping up with missing values is essential for effective cyberattack identification research. There are several methods of handling missing values like average value, majority value or replacing with zero. Out of

all these methods, replacing with the NaN values with zero tends to produce the best results.

- *Data Standardization*

Data standardization is essential before implementing ML algorithms, as data standardization can significantly impact the outcome of the ML training model. Therefore, it is very crucial to have all the data on the same scale. Standard scaler approach was chosen to standardize the data by keeping mean 0 and variance 1.

This pre-processed data was then split into training and test set using a split ratio of 80:20. This is served as input to the machine learning algorithms.

There are two types of learning algorithms - supervised and unsupervised

- *Supervised learning* is where the dataset contains both input and output, and the algorithm aims to learn the mapping function during the training phase.
- *Unsupervised learning* has only input data, and the algorithm learns the underlying structure or distribution of the data.

### **3.3.1 K-mean clustering**

The K-means algorithm divides M points in N dimensions into K clusters with the goal of minimizing the sum of squares within each cluster. Except when M, N are small and  $K = 2$ , it is impractical to demand that the solution have a minimum sum of squares against all partitions. Instead, we're looking for "local" optima, or solutions where moving a point from one cluster to another has no effect on the sum of squares within that cluster.

### **3.3.2 Decision Tree Classifier**

By creating a decision tree, the decision tree classifier (creates the classification model). Each node in the tree represents a test on an attribute, and each branch descending from that node represents one of the property's possible values. Each leaf represents one of the instance's class labels. The training set's instances are categorized by navigating them from the root of the tree to a leaf, based on the results of the tests along the way. Each node in the tree splits the instance space into two or more sub-spaces based on an

attribute test condition, starting with the root node. A new node is produced by moving down the tree branch matching to the value of the attribute. This process is then repeated for the subtree rooted at the new node, until all records in the training set have been classified.

### 3.3.3 Random Forest

Random Forest is a machine learning technique that mixes the output of numerous decision trees to produce a single outcome. Its popularity is due to its ease of use and adaptability, since it can handle both classification and regression problems. Because it uses both bagging and feature randomness to produce an uncorrelated forest of decision trees, the random forest technique is an extension of the bagging method. Feature bagging, or "the random subspace approach," is another name for feature randomness.

For the proposed model and training, evaluation criteria were set on the accuracy, recall, precision, and F1 score, which are given as

$$Accuracy = \frac{TN + TP}{TN + FP + TP + FN} \quad (3.1)$$

$$Precision = \frac{TP}{TP + FP} \quad (3.2)$$

$$Recall = \frac{TP}{TP + FN} \quad (3.3)$$

$$F1\ score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (3.4)$$

## **CHAPTER 4**

### **RESULTS AND ANALYSIS**

#### **4.1 MATLAB SIMULATION RESULTS**

A WSCC 9 bus system is simulated for all types of faults i.e., LG, LL, LLG, LLL, LLLG, and at all the 6 transmission lines for different distance combination. The fault is made to occur at  $t = 0.5$  sec and the circuit breaker clear the after 0.2 sec from occurrence of the fault i.e.,  $t = 0.7$  sec.

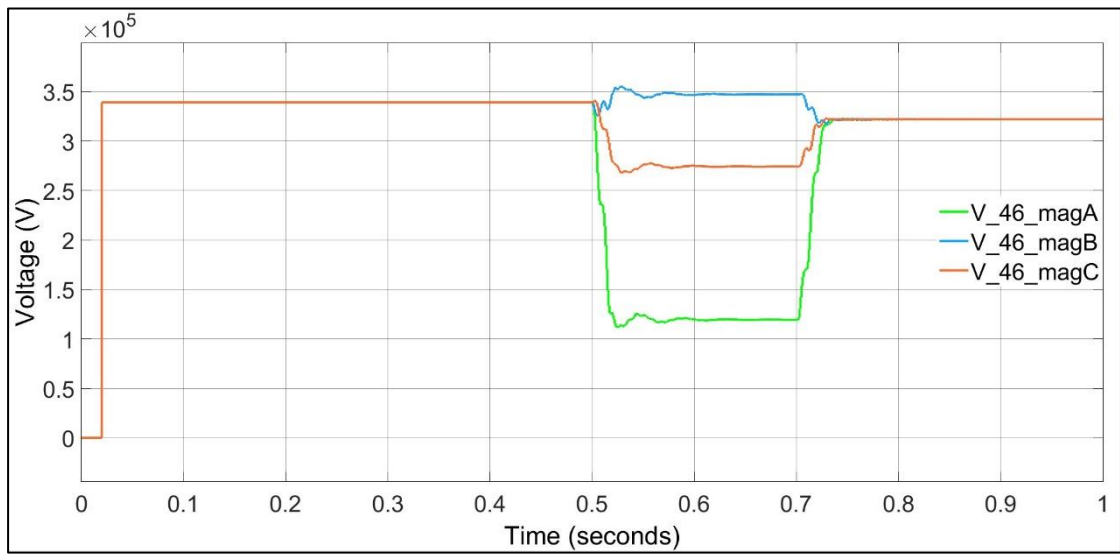
- The region from  $t = 0$  to  $t = 0.5$  sec represents the pre fault state of the system.
- The region from  $t = 0.5$  sec to  $t = 0.7$  sec represents the fault state of the system.
- The region from  $t = 0.7$  sec to  $t = 1$  sec represent the post fault state of the system.

##### **4.1.1 SINGLE LINE TO GROUND FAULT (L-G)**

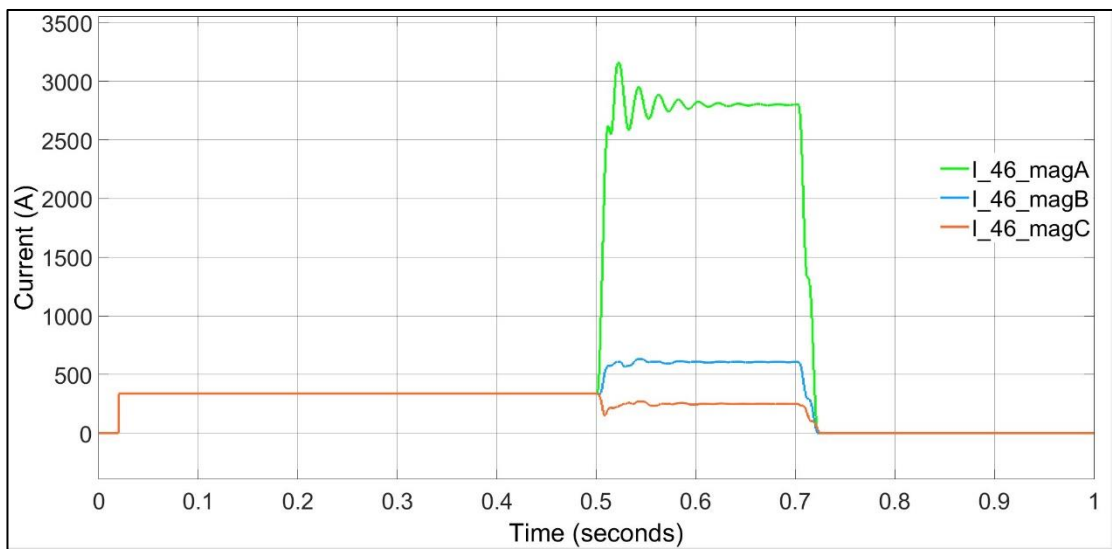
A single line-to-ground fault on a transmission line occurs when one conductor drops to the ground or meets the neutral conductor. Such types of failures may occur in power system due to many reasons like high-speed wind, falling off a tree, lightning, etc. Fig 4.1 and Fig 4.2 shows an L-G fault occurred on phase A on line 4-6. As a result, the phase A current shoot up and voltage dip has been observed. Fig 4.1 represents phase voltage of bus 4 and Fig 4.2 represents the phase current through bus 4. Fig 4.3 and Fig 4.4 represents the phase voltage and phase current of bus 5 respectively.



**Line Containing Fault (Line 4-6, B)**

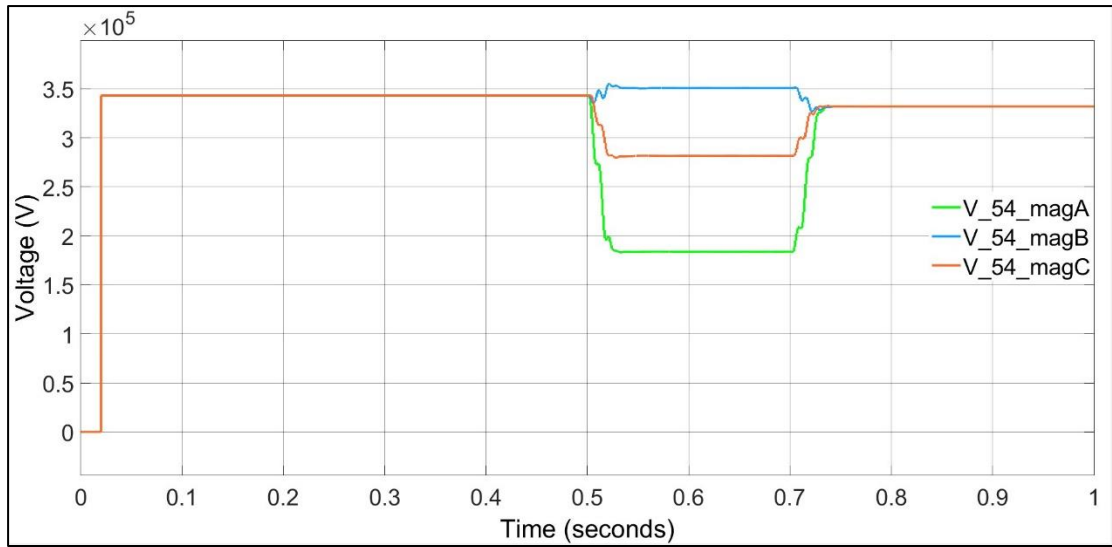


**Figure 4.1: Phase Voltage of Line 4-6**

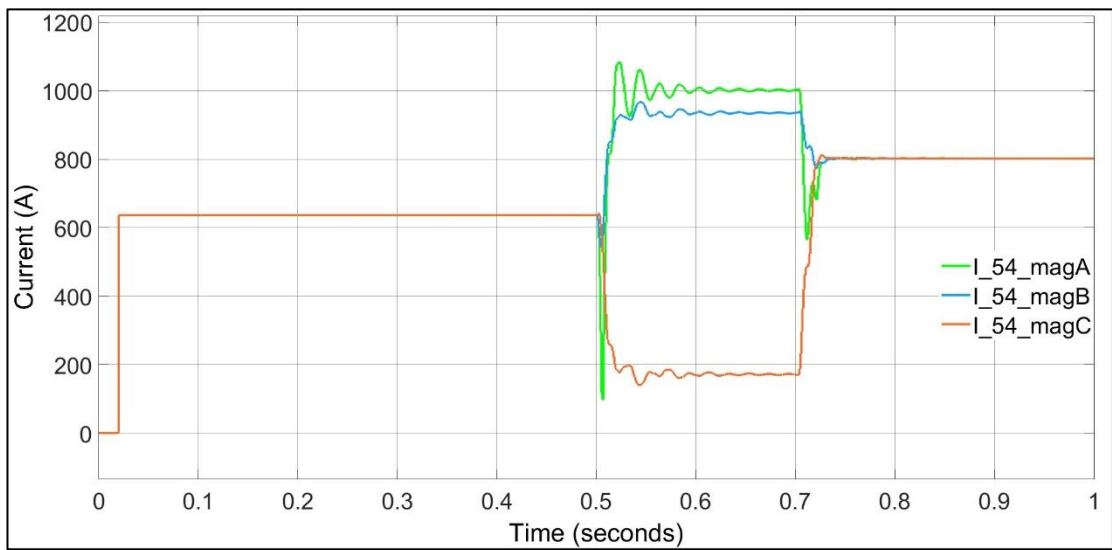


**Figure 4.2: Phase Current of Line 4-6**

***Line Without Fault (Line 5-4, A)***



**Figure 4.3: Phase Voltage of Line 5-4**

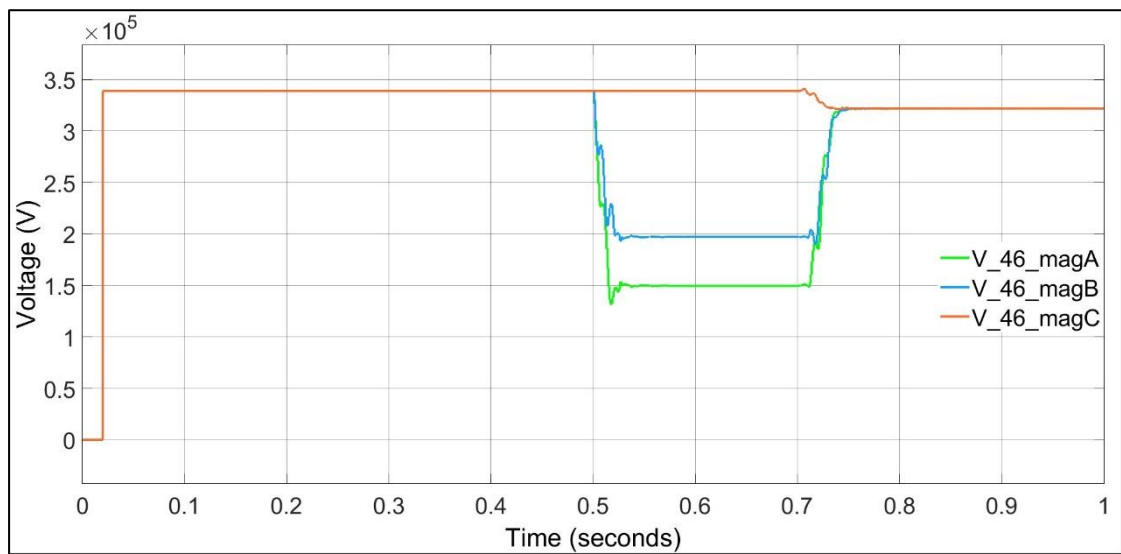


**Figure 4.4: Phase Current of line 5-4**

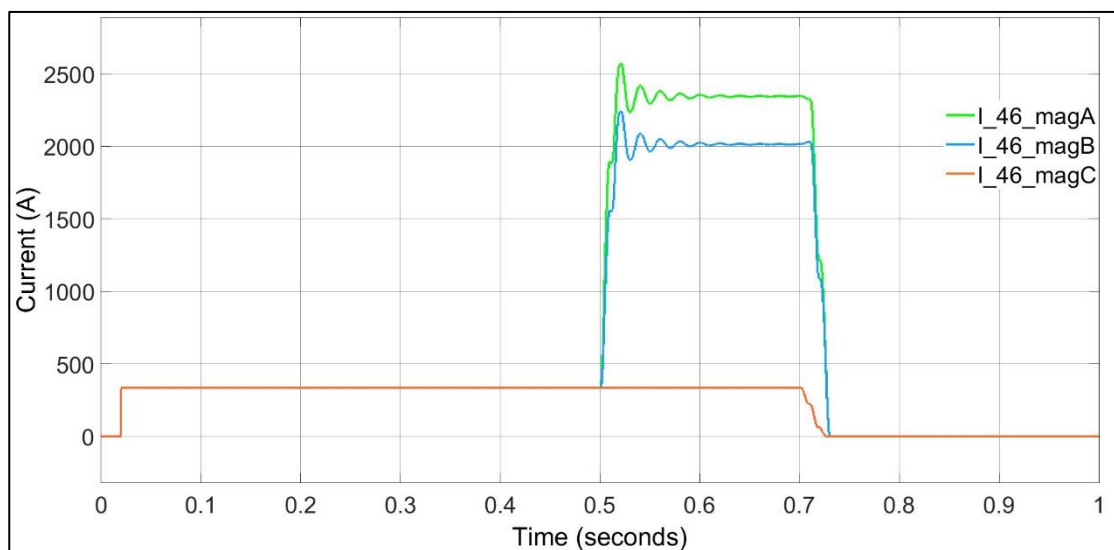
#### 4.1.2 LINE TO LINE FAULT (L-L)

A line-to-line fault or unsymmetrical fault occurs when two conductors are short circuited. Fig 4.5 and Fig 4.6 shows an L-L fault occurred on phase A and phase B on line 4-6. As a result, the phase current of A and B shoot up and voltage dip is observed. Fig 4.5 represents phase voltage of bus 4 and Fig 4.6 represents the phase current through bus 4. Fig 4.7 and Fig 4.8 represents the phase voltage and phase current of bus 5 respectively.

##### *Line Containing Fault (Line 4-6, B)*

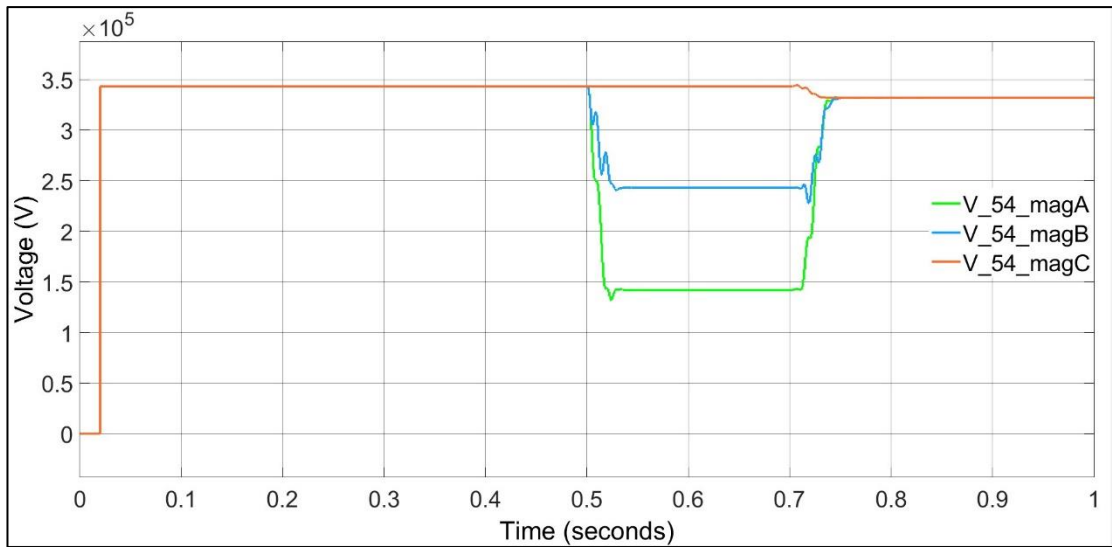


**Figure 4.5: Phase Voltage of Line 4-6**

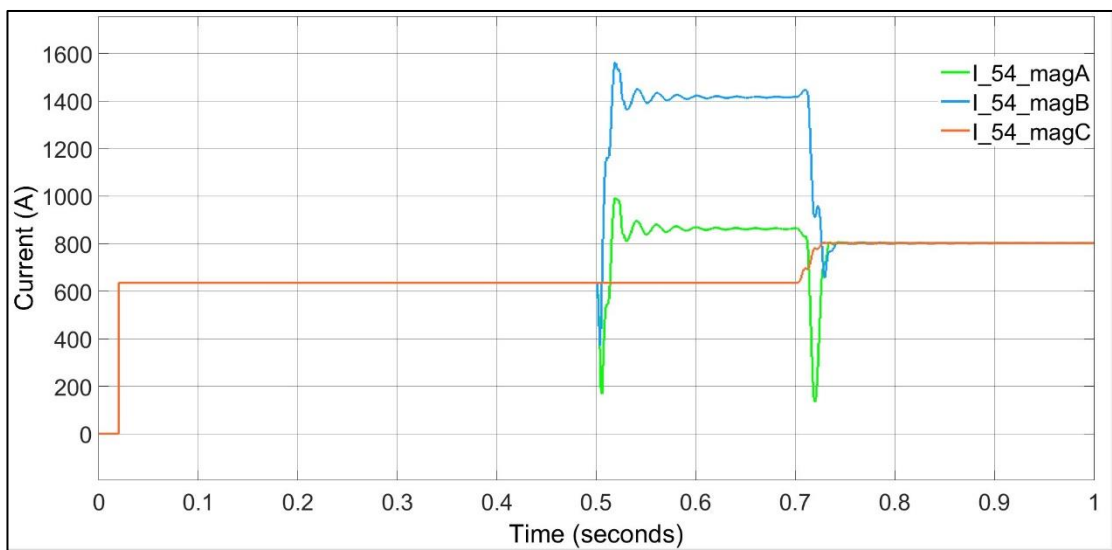


**Figure 4.6: Phase Current of Line 4-6**

***Line Without Fault (Line 5-4, A)***



**Figure 4.7: Phase Voltage of Line 5-4**



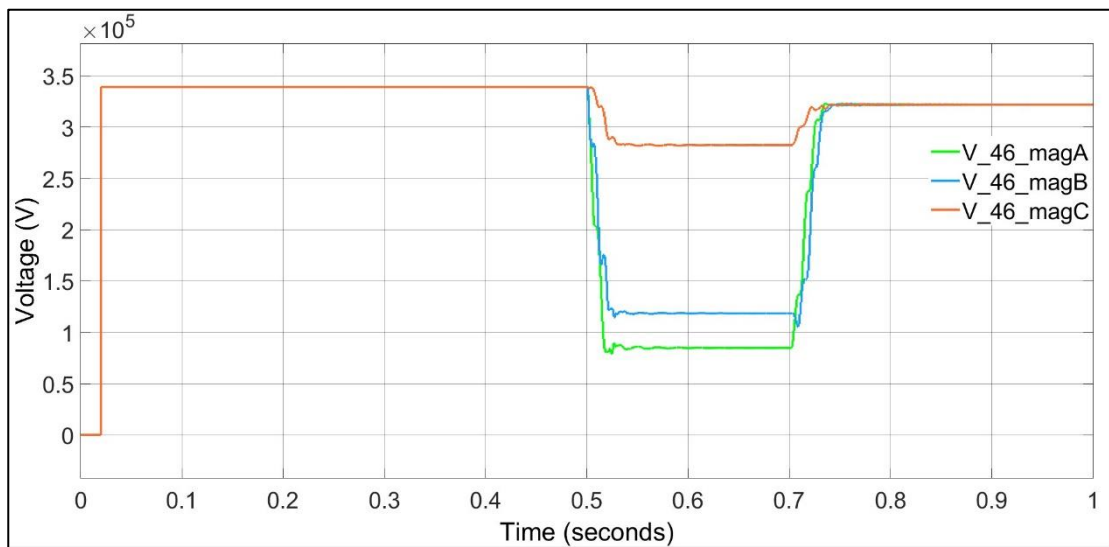
**Figure 4.8: Phase Current of Line 5-4**

### 4.1.3 DOUBLE LINE TO GROUND FAULT (L-L-G)

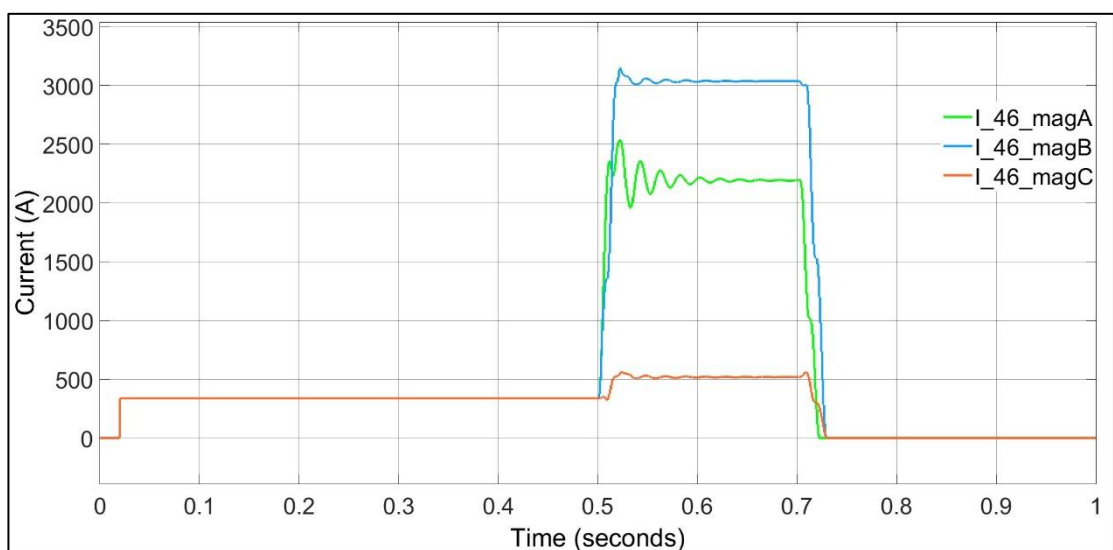
Fig 4.9 and Fig 4.10 shows an L-L-G fault occurred on phase A and phase B on line 4-6. As a result, the phase current of A and B shoot up and voltage dip is observed.

Fig 4.9 represents phase voltage of bus 4 and Fig 4.10 represents the phase current through bus 4. Fig 4.11 and Fig 4.12 represents the phase voltage and phase current of bus 5 respectively.

**Line Containing Fault (Line 4-6, B)**

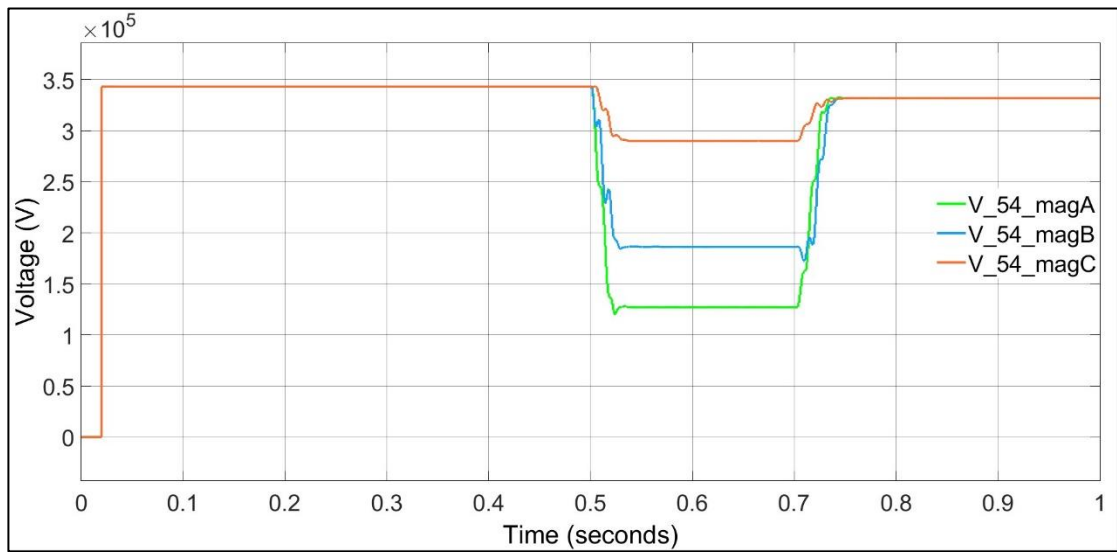


**Figure 4.9: Phase Voltage of Line 4-6**

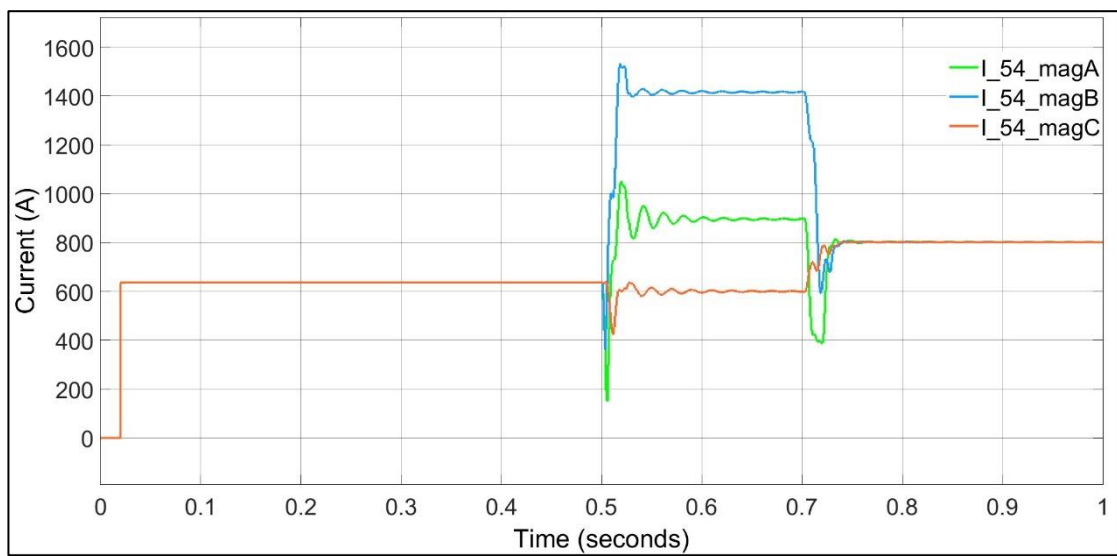


**Figure 4.10: Phase Current of Line 4-6**

***Line Without Fault (Line 5-4, A)***



**Figure 4.11: Phase Voltage of Line 5-4**

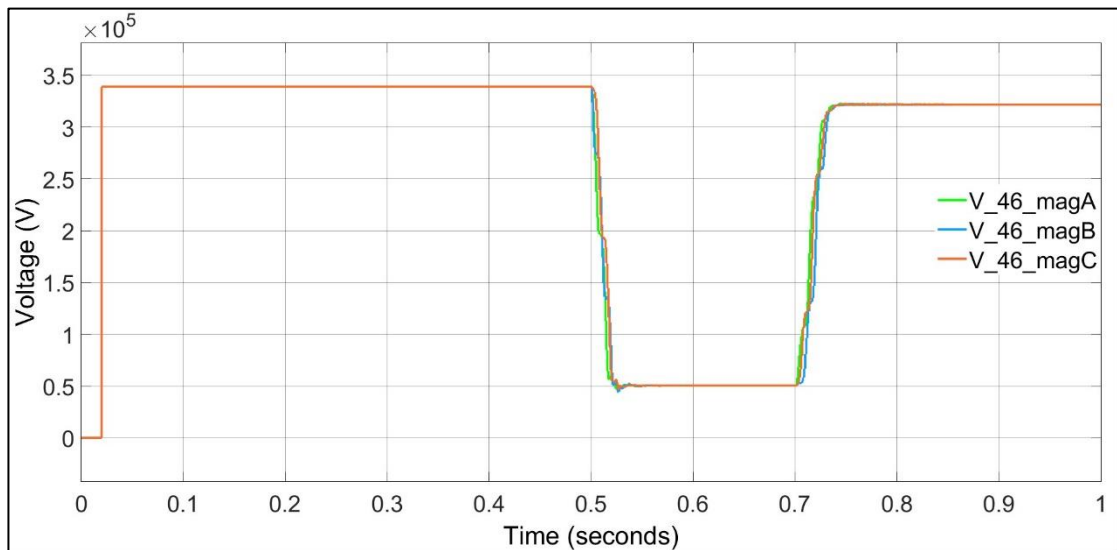


**Figure 4.12: Phase Current of Line 5-4**

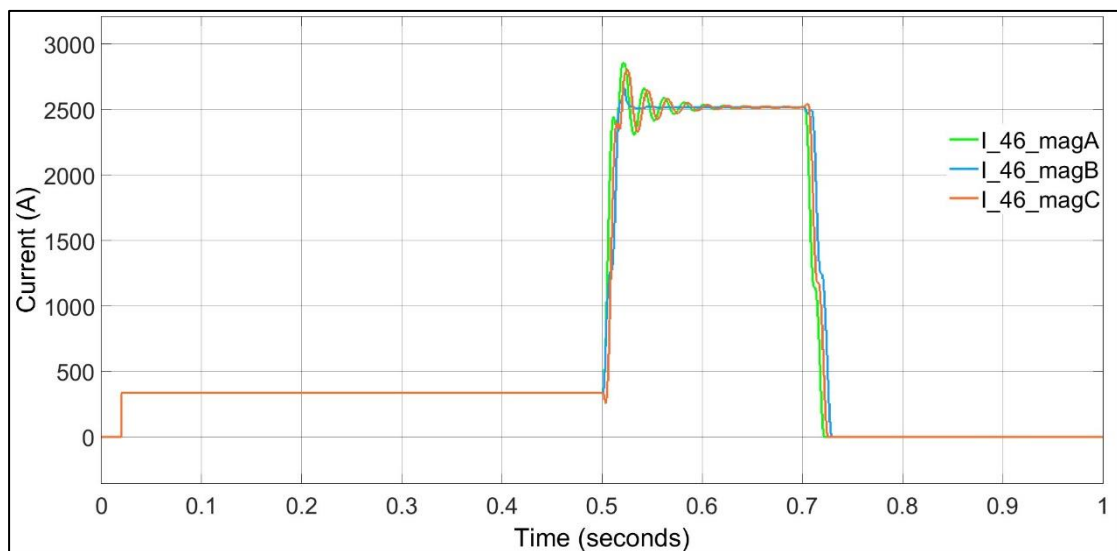
#### 4.1.4 ALL THREE PHASE TO GROUND FAULT (L-L-L-G)

Fig 4.13 and Fig 4.14 shows an L-L-L-G fault occurred on line 4-6. As a result, the voltage dip and current shoot is observed. Fig 4.13 represents phase voltage of bus 4 and Fig 4.14 represents the phase current through bus 4. Fig 4.15 and Fig 4.16 represents the phase voltage and phase current of bus 5 respectively.

***Line Containing Fault (Line 4-6, B)***

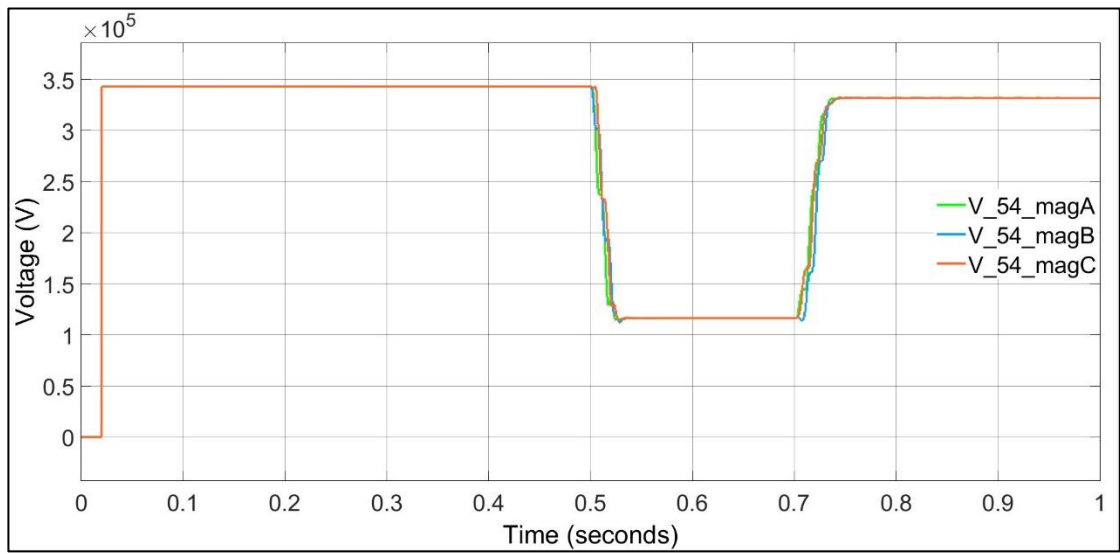


**Figure 4.13: Phase Voltage of Line 4-6**

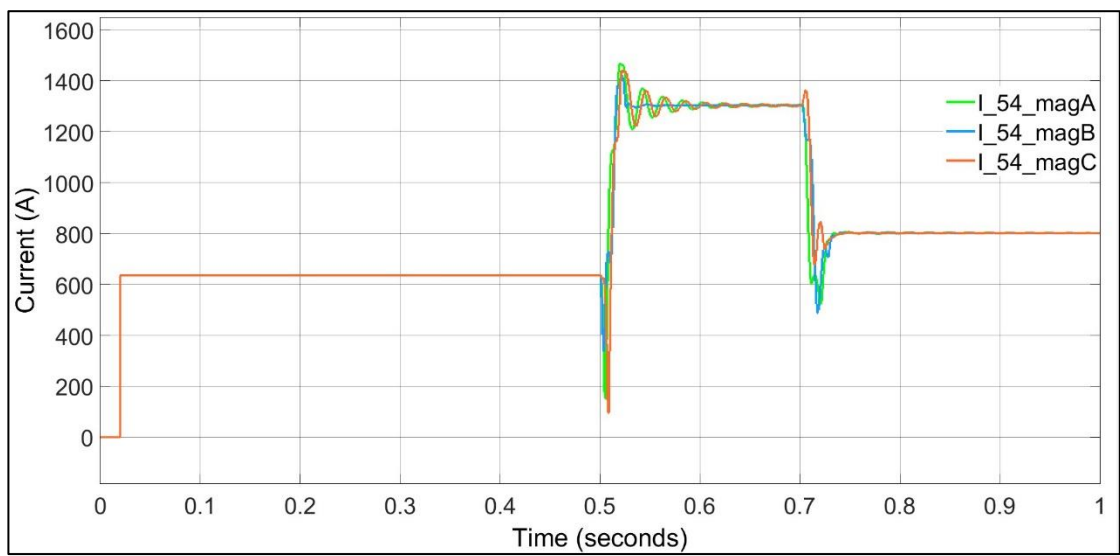


**Figure 4.14: Phase Current of Line 4-6**

***Line Without Fault (Line 5-4, A)***



**Figure 4.15: Phase Voltage of Line 5-4**



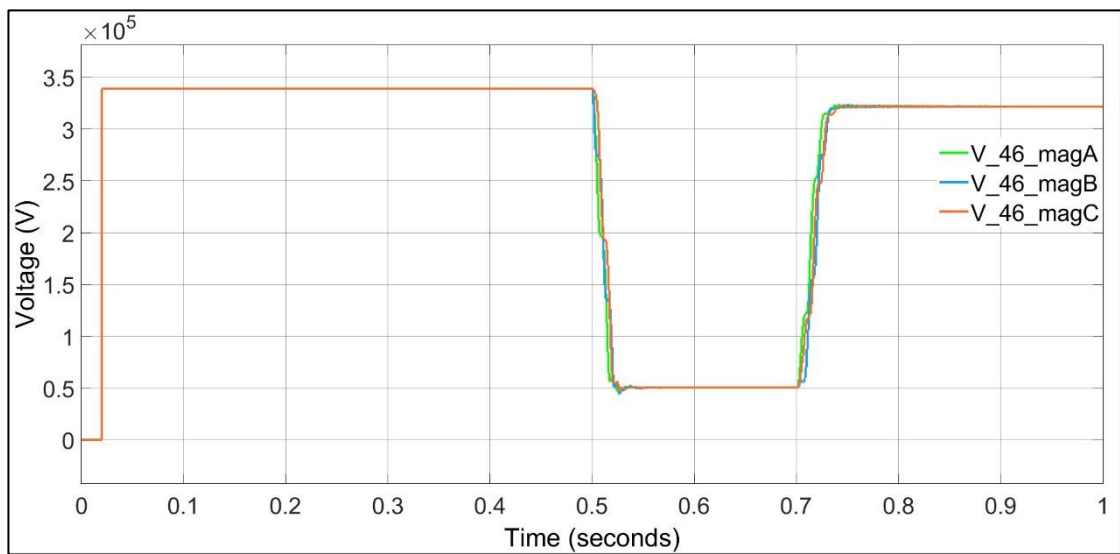
**Figure 4.16: Phase Current of Line 5-4**



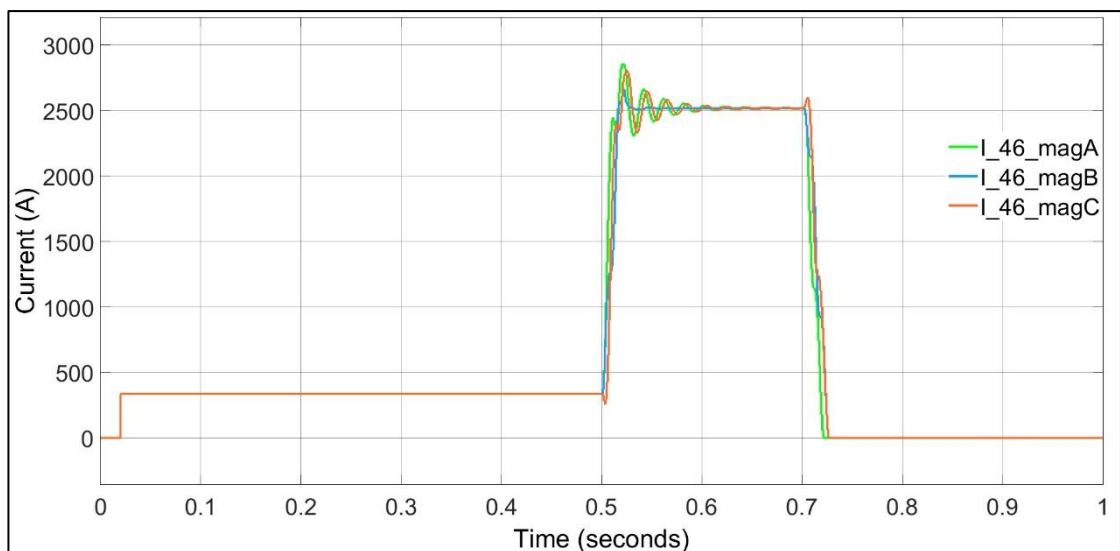
#### 4.1.5 ALL THREE PHASE SHORT CIRCUITED (L-L-L)

Fig 4.17 and Fig 4.18 shows an L-L-L fault occurred on line 4-6. As a result, the voltage dip and current shoot is observed. Fig 4.17 represents phase voltage of bus 4 and Fig 4.18 represents the phase current through bus 4. Fig 4.19 and Fig 4.20 represents the phase voltage and phase current of bus 5 respectively.

##### *Line Containing Fault (Line 4-6, B)*

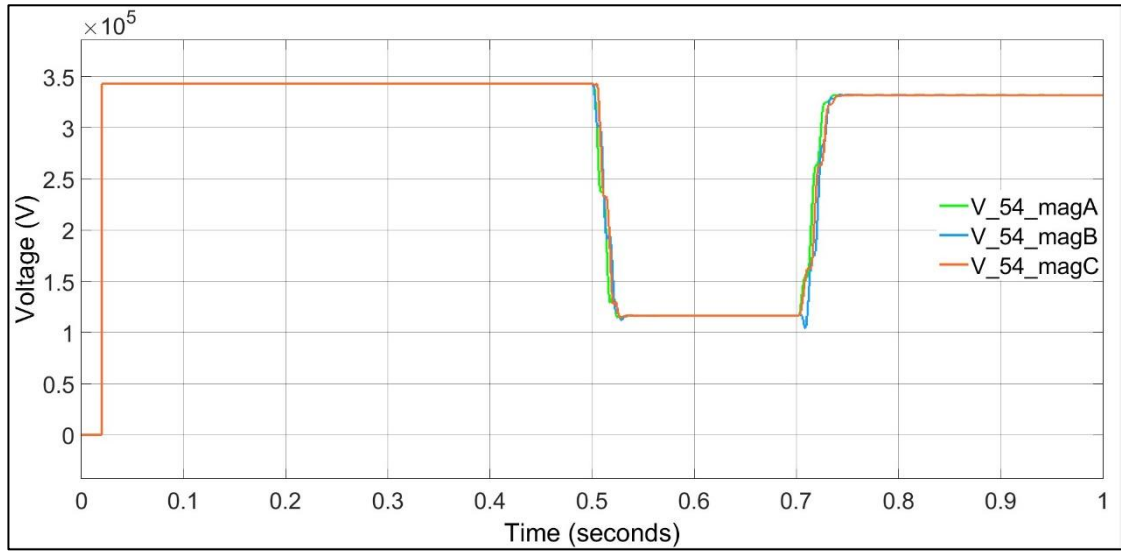


**Figure 4.17: Phase Voltage of Line 4-6**

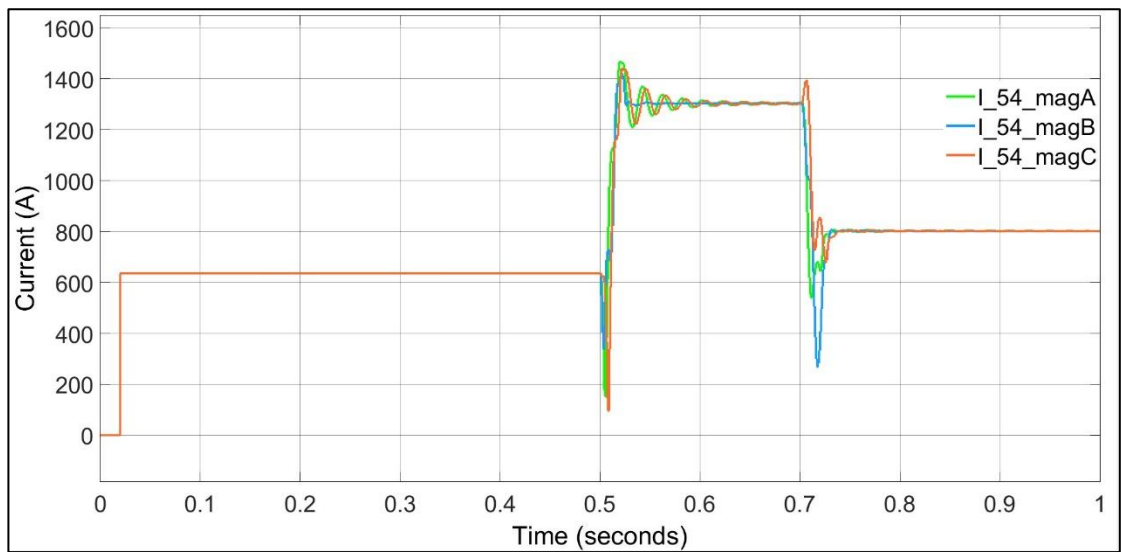


**Figure 4.18: Phase Current of Line 4-6**

***Line Without Fault (Line 5-4, A)***



**Figure 4.19: Phase Voltage of line 5-4**



**Figure 4.30: Phase Current of Line 5-4**

	E	F	G	H	I	J	K	L	M	N
1	V_magA	V_magB	V_magC	I_angleA	I_angleB	I_angleC	I_magA	I_magB	I_magC	Condition
2	0	0	0	0	0	0	0	0	0	0
3	339129.1274	339174.2646	339120.4549	42.32377724	-77.67682107	162.3177591	673.3249848	673.4025995	673.357707	0
4	339109.0626	339109.2964	339109.6045	42.31852025	-77.6811343	162.3179937	673.3296228	673.3391181	673.3378873	0
5	339109.3018	339109.0065	339109.3242	42.31833781	-77.68144481	162.3182603	673.330996	673.333528	673.3344754	0
6	339109.3693	339109.3052	339109.3813	42.31822754	-77.68168937	162.3182982	673.3319044	673.3315148	673.3325562	0
7	339109.3945	339109.42	339109.4322	42.31819114	-77.68181242	162.3182564	673.332436	673.331532	673.3319488	0
8	339109.4008	339109.4389	339109.4463	42.31818592	-77.68185389	162.3182292	673.3326731	673.3318221	673.3318434	0
9	339109.4029	339109.4316	339109.444	42.31818894	-77.68185876	162.3182216	673.3327207	673.33196	673.3318559	0
10	339109.4057	339109.4264	339109.4398	42.31819329	-77.68185197	162.3182219	673.3326857	673.3319965	673.3318815	0
11	339109.4089	339109.4255	339109.4373	42.31819777	-77.68184263	162.3182235	673.3326272	673.3320104	673.3319086	0
12	339109.4118	339109.4261	339109.4359	42.31820214	-77.68183327	162.3182247	673.3325673	673.3320273	673.3319379	0
13	339109.4144	339109.4268	339109.4349	42.31820621	-77.68182452	162.3182256	673.3325123	673.3320471	673.3319679	0
14	339109.4165	339109.4273	339109.434	42.31820985	-77.68181661	162.3182263	673.3324633	673.3320659	673.3319963	0
15	339109.4183	339109.4276	339109.4333	42.31821303	-77.68180963	162.3182271	673.3324204	673.3320822	673.3320217	0
16	339109.4198	339109.4279	339109.4326	42.31821577	-77.68180357	162.3182277	673.3323833	673.332096	673.3320438	0
17	339109.4211	339109.4281	339109.432	42.31821811	-77.68179836	162.3182283	673.3323515	673.3321078	673.332063	0
18	339109.4222	339109.4283	339109.4316	42.31822012	-77.6817939	162.3182288	673.3323243	673.3321179	673.3320794	0
19	339109.4232	339109.4284	339109.4312	42.31822182	-77.68179009	162.3182292	673.332301	673.3321264	673.3320934	0
20	339109.424	339109.4286	339109.4308	42.31822328	-77.68178685	162.3182295	673.3322813	673.3321337	673.3321054	0
21	339109.4247	339109.4287	339109.4305	42.31822452	-77.68178408	162.3182298	673.3322644	673.3321399	673.3321157	0
22	339109.4253	339109.4288	339109.4303	42.31822557	-77.68178172	162.3182301	673.33225	673.3321452	673.3321244	0
23	339109.4258	339109.4289	339109.4301	42.31822647	-77.68177972	162.3182303	673.3322378	673.3321498	673.3321318	0
24	339109.4262	339109.429	339109.4299	42.31822723	-77.68177801	162.3182305	673.3322274	673.3321536	673.3321382	0
25	339109.4266	339109.429	339109.4297	42.31822788	-77.68177655	162.3182307	673.3322186	673.3321569	673.3321436	0
26	339109.4269	339109.4291	339109.4296	42.31822844	-77.68177531	162.3182308	673.3322111	673.3321597	673.3321482	0
27	339109.4271	339109.4291	339109.4295	42.31822891	-77.68177426	162.3182309	673.3322047	673.332162	673.3321521	0
28	173866.5744	345305.6814	282939.1511	-26.09966641	-73.93301491	171.2511233	2165.008953	807.4377816	568.3175833	1
29	165256.8021	342706.8161	283007.2674	-27.27530904	-72.89221831	169.9198067	2040.051416	799.993996	558.4583754	1
13821	339110.3205	339110.3213	162173.9982	42.31885301	-77.68114691	162.3188531	1571.915109	4126.549346	3546.617807	2
13822	339110.3205	339110.3213	339110.3207	42.31885302	-77.68114689	162.3188531	673.3298191	673.3298246	673.3298238	2
<div> <div></div> <div></div> <div>B_75</div> <div>B_78</div> <div>B_87</div> <div>B_46</div> <div>B_64</div> <div>B_96</div> <div>B_69</div> <div>B_89</div> <div>B_98</div> <div>B_57</div> <div>B_54</div> <div>B_45</div> <div>+</div> </div>										

**Figure 4.21 Consolidated dataset**

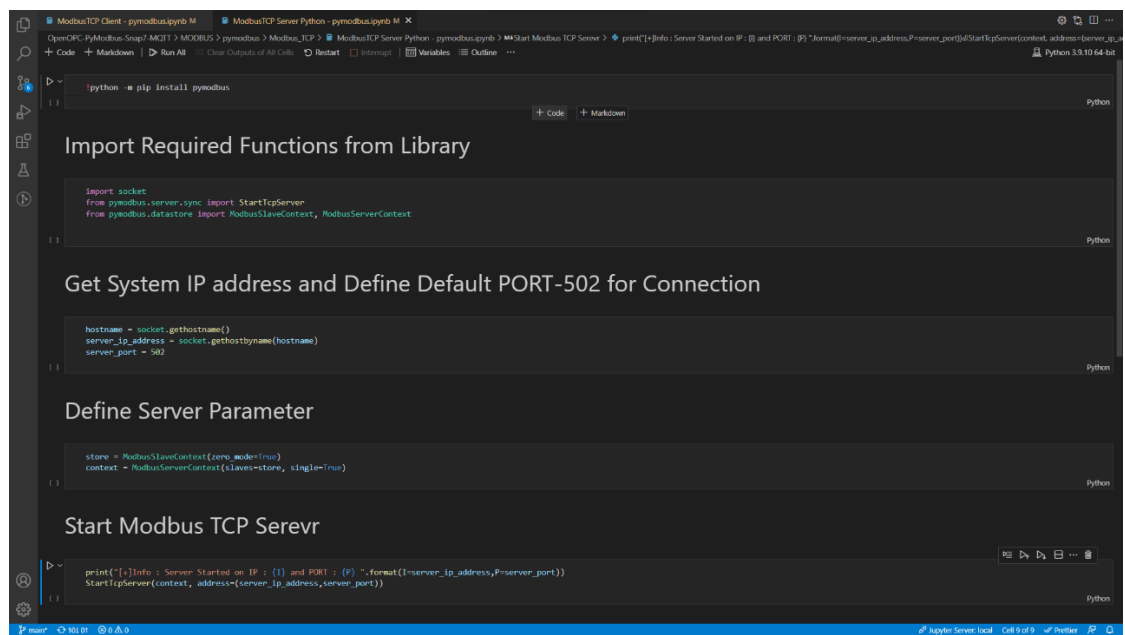
Figure 2.21 provides an example of the consolidated dataset. Three types of condition are indicated as – Normal Operation (0), Faults (1) and Cyber Attacks (2). The following features were sent from across from the circuit breaker to the main control substation, indicated below:

- Phase A Voltage magnitude
- Phase A Voltage Angle
- Phase B Voltage magnitude
- Phase B Voltage Angle
- Phase C Voltage magnitude
- Phase C Voltage Angle
- Phase A Current magnitude
- Phase A Current Angle
- Phase B Current magnitude
- Phase B Current Angle
- Phase C Current magnitude
- Phase C Current Angle

## 4.2 CYBER-ATTACK RESULTS

### 4.2.1 Modbus TCP Server

The server code will be running at the Control Centre and will be responsible for listening and responding to requests as well interacting with the database to store data values. A screenshot of the server code is attached below, and the same source code can be found in the appendix



```
python -m pip install pymodbus

import socket
from pymodbus.server.sync import StartTcpServer
from pymodbus.datastore import ModbusSlaveContext, ModbusServerContext

hostname = socket.gethostname()
server_ip_address = socket.gethostname(hostname)
server_port = 502

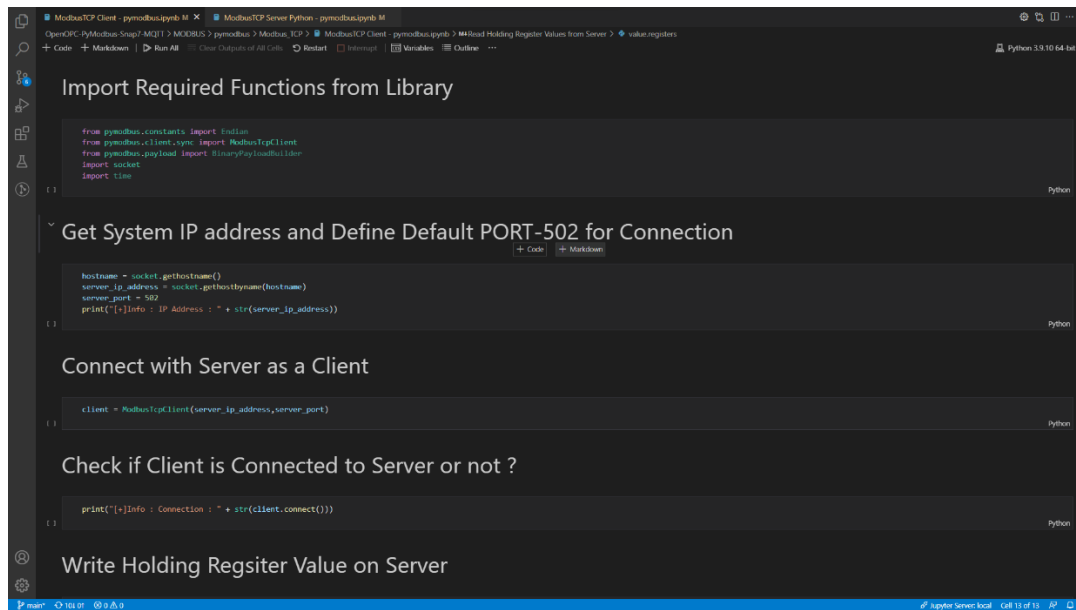
store = ModbusSlaveContext(zero_mode=True)
context = ModbusServerContext(slaves=store, single=True)

print("[*]Info : Server Started on IP : {} and PORT : {}".format(server_ip_address,server_port))
StartTcpServer(context, address=(server_ip_address,server_port))
```

**Figure 4.22: Modbus TCP Server Code**

### 4.2.2 Modbus TCP Client

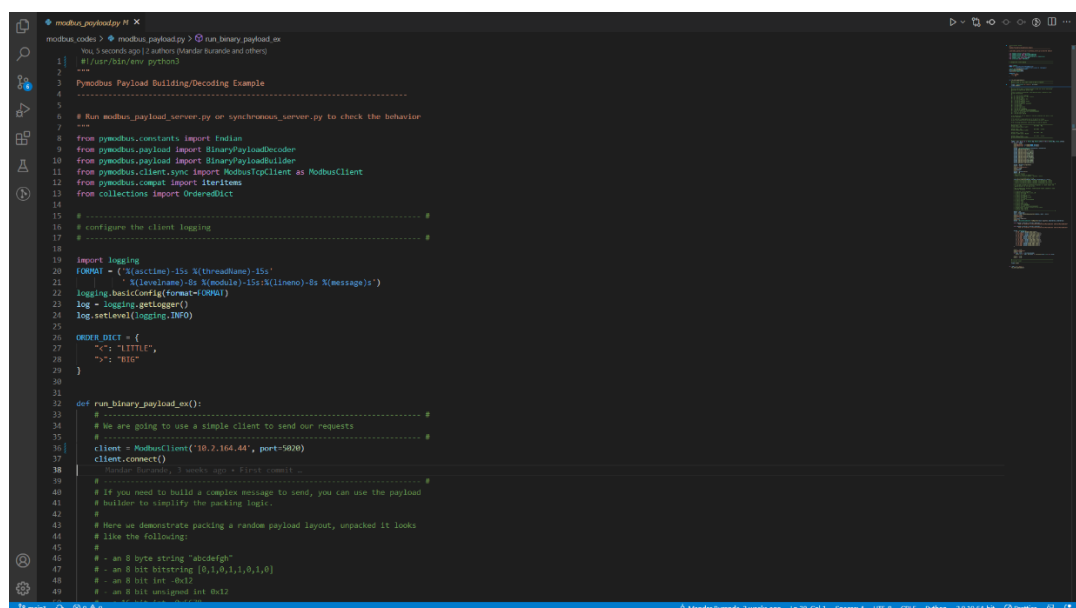
The client code will be running in the PLCs and RTUs. The client code allows the devices to send data and requests to the Server. A screenshot of the client code is attached below, and the same source code can be found in the appendix.



**Figure 4.23: Modbus TCP Client Code**

### 4.2.3 Encode data to Modbus format

The raw data is converted to Modbus format and is sent to the Server. When the payload is fetched it is then finally decoded back to raw data using PyModbus. A screenshot of the payload encoder - decoder code is attached below, and the same source code can be found in the appendix.



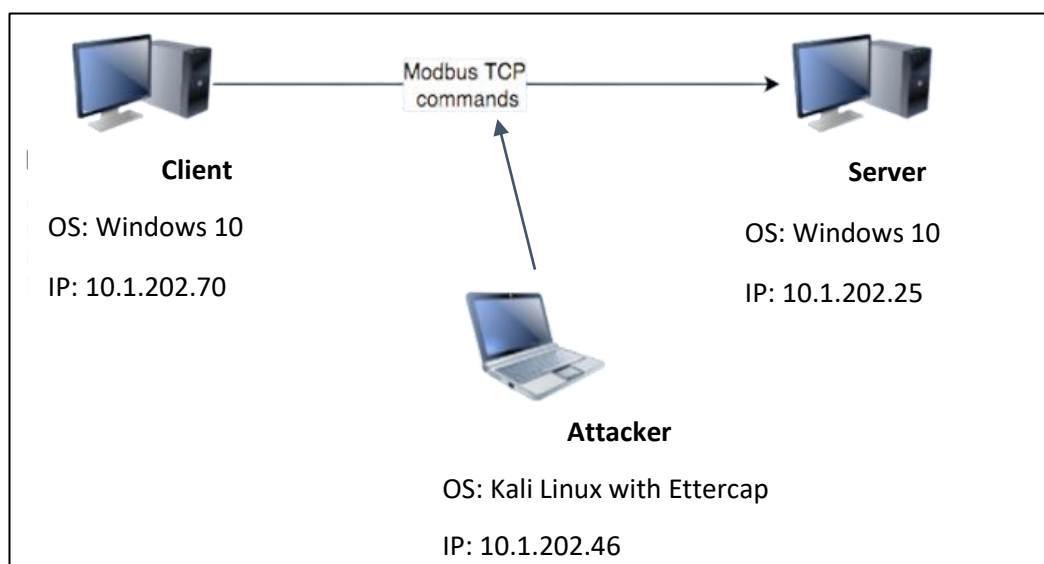
**Figure 4.24: PyModbus Payload Code**

Decoded Data	
V_75_angleA	39.98264694213867
V_75_angleB	-80.01714324951172
V_75_angleC	159.98382568359375
V_75_magA	328342.40625
V_75_magB	328336.90625
V_75_magC	328340.09375
I_75_angleA	37.61441421508789
I_75_angleB	96.51009368896484
I_75_angleC	160.12152099609375
I_75_magA	8.639863047221752e-09
I_75_magB	3.810341375753978e-09
I_75_magC	2.925536923825689e-09

**Figure 4.25:** The decoded payload data, as received back by the client device.

#### 4.2.4 Setup cyber-attack architecture

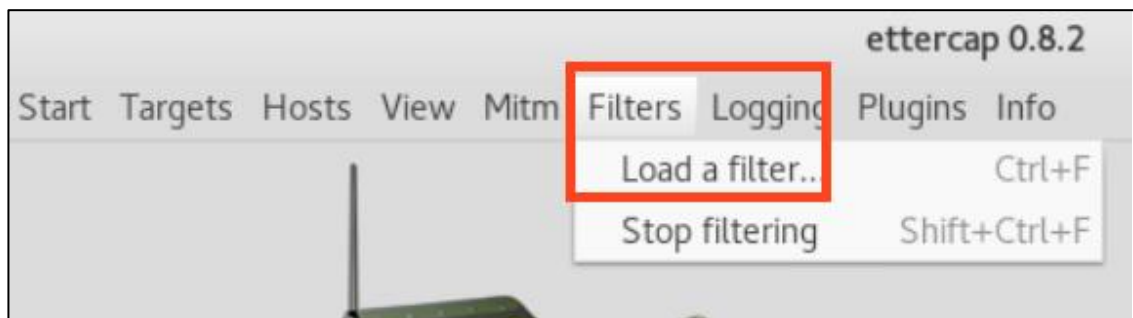
As shown in Figure 4.26, we started with two Windows devices which served as the Client and Server for Modbus communication. This was accompanied by a third device running Kali Linux on it. It utilised Ettercap v0.8.2 for sniffing and launching ARP Poisoning attacks.



**Figure 4.26:** Cyber-attack Architecture

#### 4.2.5 Perform man-in-the-middle attacks

A MITM attack was performed with an Ettercap tool that maliciously modified the Modbus TCP commands between the Master and PLC workstations. An Ettercap filter within the Ettercap tool was then created to modify Modbus TCP communications.



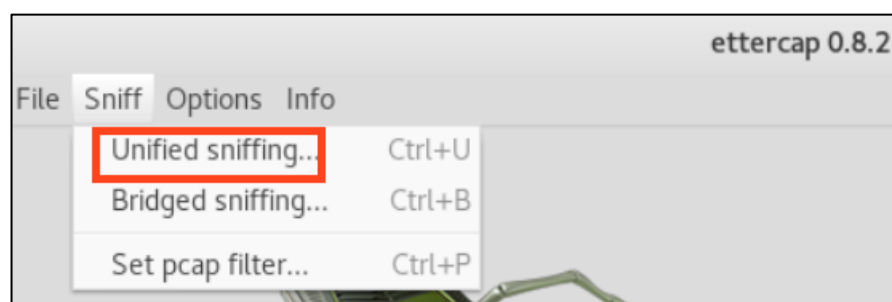
**Figure 4.27: Loading a filter in Ettercap**

#### 4.2.6 Gather Cyber Attack data

Generate and gather the data produced from conducting cyber-attacks on the network. Perform data wrangling, feature extraction, etc on the gathered data to prepare it for the machine learning model.

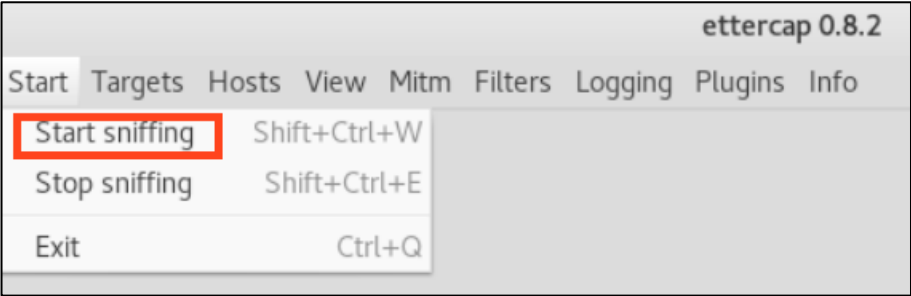
### 4.3 CASE STUDY OF AN ARP POISONING ATTACK

A MITM attack was performed with an Ettercap tool that maliciously modified the Modbus TCP commands between the Master and PLC workstations. Weaknesses of the Modbus TCP that were targeted by the Ettercap tool involved sending commands in clear text and lack of authentication within the Modbus TCP protocol. The Kali Linux workstation first used the Ettercap tool to put the network interface into unified sniffing mode. This is shown in Figure 4.28



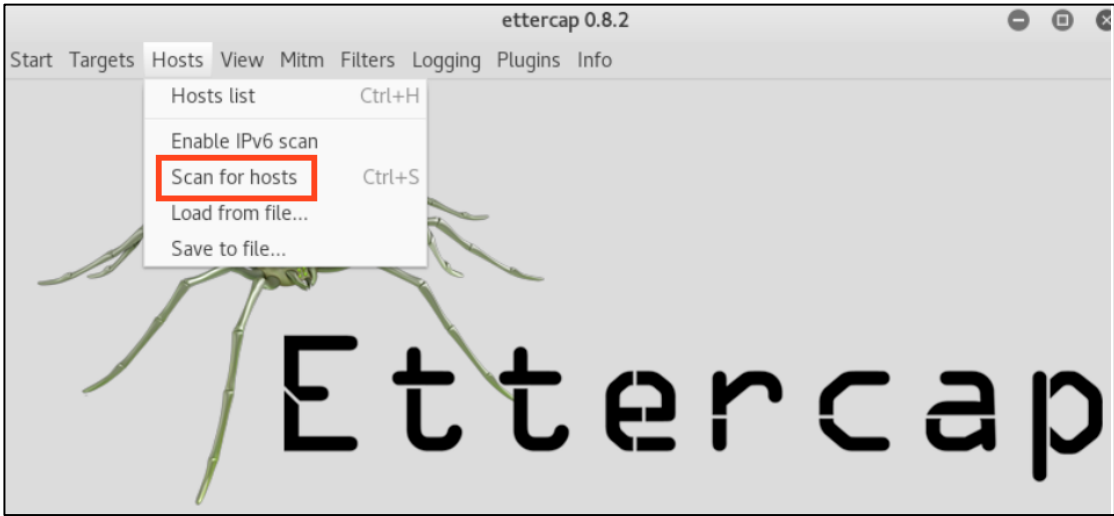
**Figure 4.28: Unified Sniffing**

The Ettercap tool then sniffed packets on the network. This is shown in Figure 4.29



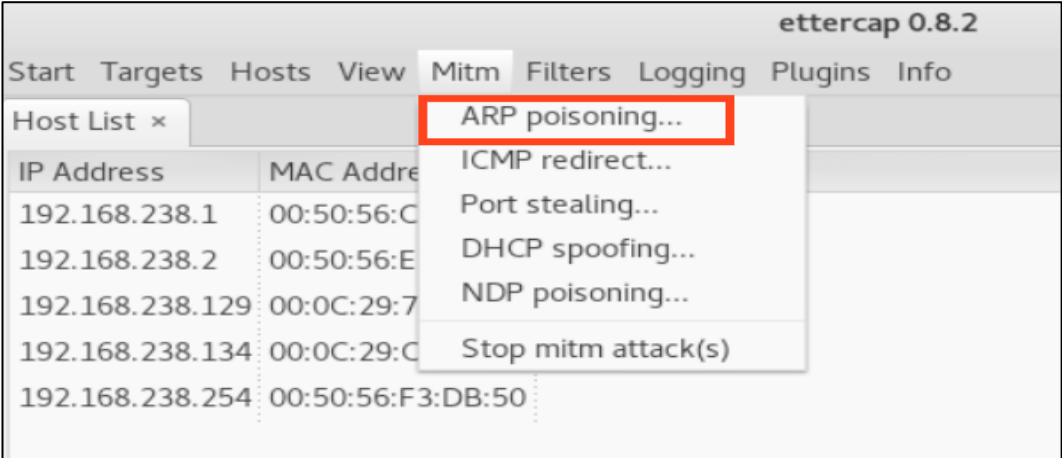
**Figure 4.29: Ettercap sniffing**

The Attacker workstation then scanned for hosts to find the Server and Client workstations to attack. This is shown in Figure 4.30.



**Figure 4.30: Ettercap Scan Host**

The Attacker workstation then used the MAC addresses provided by the Ettercap scan for a MITM with ARP poisoning (ARP spoofing) to send falsified ARP messages over a LAN. This is depicted in Figure 4.31.



**Figure 4.31: ARP Poisoning**



The ARP spoofing resulted in the linking of the attacker's MAC address with the IP address of a legitimate computer or Server on the network.

An Ettercap filter within the Ettercap tool was then created to modify Modbus TCP communications coming from the Master workstation with a destination to the PLC workstation. This resulted in the normal Modbus TCP communications from the master workstation sending a read/write to the PLC workstation with a Modbus TCP command of one for the PLC's first address. The Ettercap MITM attack, combined with the proper filter, modified any Modbus TCP command with a hex value of ff00 (which resulted in the coil being on) to a hex value 0000 (which meant that the coil was off). The filter called Modbus. Filter was used to compile text into a binary filter that could be interpreted by the Ettercap tool. To find the proper syntax with the etterfilter utility, the requirements for the lab were first mapped with syntax available from the etterfilter utility.

**Table 4.1: Ettercap Filter Syntax**

Actions for Lab	Etterfilter Syntax
Isolate TCP protocol with destination port 502. This port is associated with Modbus TCP.	Commands: ip.proto and tcp.dst
Search payload for Modbus TCP command of hex value ff00.	<p>search (where, what)</p> <p>This function searches the string 'what' in the buffer 'where'. The buffer can be either DATA.data or DECODED.data.</p> <p>The former is the payload at the layer DATA (on top of TCP or UDP) as it is</p>
Modify payload for Modbus TCP command ff00 to hex value 0000.	<p>replace(what, with)</p> <p>This function replaces the string 'what' with the string 'with'. They can be a binary string and must be escaped. The replacement is always performed in DATA.data since it is</p>

	<p>the only payload that gets forwarded.</p> <p>The 'DECODED.data' buffer is only used internally and never reaches the wire.</p>
--	---

The final syntax, based on Table 4.1, for the Modbus.filter, is shown in Figure 4.32

```

if (ip.proto == TCP && tcp.dst == 502) {
  if (search(DATA.data, "\xff\x00")) {
    msa( "Found Modbus On Switch of ff 00" );
    replace("\xff\x00", "\x00\x00");
  }
}

```

**Figure 4.32: Ettercap Filter**

With Modbus TCP's inability to combat against clear text messages and the lack of Modbus TCP with built-in authentication, the Ettercap filter successfully modified the packet from the Server to the Client to change the coil to a hex value of 0000.

#### 4.4 MACHINE LEARNING RESULTS

Both supervised, and unsupervised machine learning techniques were applied for the classification of events. K-Means clustering (unsupervised), Random Forest (supervised), and Decision Tree Classifier (supervised) were the three machine learning algorithms used. Results for each are shown in Tables 4.1, 4.2, and 4.3.

**Table 4.2: Performance Matrix of K-Means Algorithm**

Fault Line	Accuracy	Precision	Recall	F1 Score
a	86.9	82	88	87
b	87.5	83	87	89
c	86.5	84	86	85
d	86.9	83	87	86
e	87.3	87	88	85
f	87.2	89	82	85

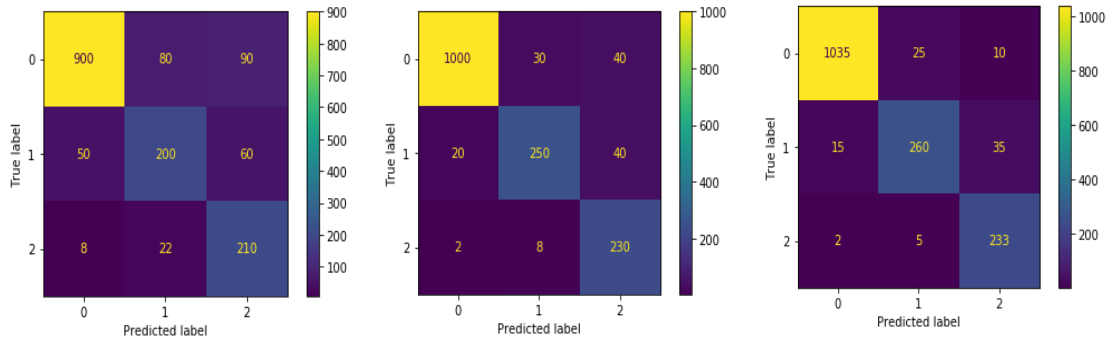
**Table 4.3: Performance Matrix of Random Forest Algorithm**

Fault Line	Accuracy	Precision	Recall	F1 Score
a	95.1	95	96	94
b	95.6	96	95	95
c	96.3	96	97	94
d	95.8	96	95	92

<b>e</b>	95.3	96	94	95
<b>f</b>	95.9	94	96	95

**Table 4.4: Performance Matrix of Decision Tree Classifier Algorithm**

<b>Fault Line</b>	<b>Accuracy</b>	<b>Precision</b>	<b>Recall</b>	<b>F1 Score</b>
<b>a</b>	96.1	96	93	95
<b>b</b>	96.6	98	94	96
<b>c</b>	97.3	98	96	96
<b>d</b>	96.8	95	95	95
<b>e</b>	96.3	96	94	94
<b>f</b>	96.9	97	96	95



**Figure 4.33: Confusion Matrices for Line C obtained from K-Means, Random Forest, and Decision Tree Classifier, respectively**  
**0 - Natural Operation, 1 – Fault, 2- Cyber Attack**

From tables 4.1, 4.2, and 4.3, the performance of each machine learning algorithm was seen on each line in the system. The Decision Tree algorithm performs the best out of the three, with an average accuracy of 96%. In general, supervised learning algorithms perform slightly better than unsupervised algorithms in this case.

The confusion matrices for each of the algorithms on line C are shown in Figure 4.27. Despite the high accuracy of the algorithms, there are few cases where cyber-attacks mimic natural events and are misclassified.

## **CHAPTER 5**

### **CONCLUSION**

This thesis presents a detailed discussion about Cyber Attack Detection in Power System Scada Networks Using Machine Learning Techniques. During the process of this work, we simulated multiple types of power system faults – natural as well as cyber-attacks, collected relevant data from the generated data samples and analyzed them using multiple machine learning techniques. Industry-standard tools like MATLAB, Simulink, Linux, Ettercap, Wireshark, Jupyter Notebooks, sklearn, pandas, numpy, and more were used during this project. This further elevates the relevance of the work done.

Furthermore, a methodology was devised and proposed to simulate and investigate the occurrence as well as the impact of the different obstacles and sabotages on the system. Based on the observed results, inferences were gleaned and suggestions to counter as well as thwart the problems were proposed.

#### **5.1 SCOPE FOR FURTHER RESEARCH**

This thesis has a wide-ranging scope for further research as it delves into the domains of electrical engineering as well as software engineering. Smart-grid technologies are seeing an increasing number of applications, opening the distribution system to multiple types of malicious threats.

The scope for further research development in this field can be identified as follows:

- Expanding the project's scope to incorporate multiple bus systems with different sizes containing diverse types of buses and loads. This will lead to an even more, representative dataset and help train the machine learning model so that it can be applied with lesser modifications.
- Incorporating a larger variety of cyber-attacks that different leverage media of attack allows the machine learning model to become more robust in detecting external interference.
- Performing the project upon a Real-Time Digital System (RTDS) with relevant hardware like PLCs, RTUs, servers, and attack machines will lead to more fruitful results and learnings.

## APPENDIX

```
1 def run_binary_payload_ex():
2
3     client = ModbusClient('10.2.160.1', port=5020)
4     client.connect()
5
6     combos = [(wo, bo) for wo in [Endian.Big, Endian.Little] for bo in [Endian.Big, Endian.Little]]
7     for wo, bo in combos:
8         print("-" * 60)
9         print("Word Order: {}".format(ORDER_DICT[wo]))
10        print("Byte Order: {}".format(ORDER_DICT[bo]))
11        print()
12        builder = BinaryPayloadBuilder(byteorder=bo, wordorder=wo)
13        builder.add_32bit_float(0.02)
14
15        payload = builder.to_registers()
16        print("-" * 60)
17        print("Writing Registers")
18        print("-" * 60)
19        print(payload)
20        print("\n")
21        payload = builder.build()
22        address = 0
23
24        client.write_registers(address, payload, skip_encode=True, unit=1)
25
26        address = 0x0
27        count = len(payload)
28        result = client.read_holding_registers(address, count, unit=1)
29        print("-" * 60)
30        print("Registers")
31        print("-" * 60)
32        print(result.registers)
33        print("\n")
34        decoder = BinaryPayloadDecoder.fromRegisters(result.registers, byteorder=bo, wordorder=wo)
35
36        assert decoder._byteorder == builder._byteorder, \
37            "Make sure byteorder is consistent between BinaryPayloadBuilder and BinaryPayloadDecoder"
38        assert decoder._wordorder == builder._wordorder, \
39            "Make sure wordorder is consistent between BinaryPayloadBuilder and BinaryPayloadDecoder"
40
41
42        decoded = OrderedDict([
43            ('Time', decoder.decode_32bit_float()),
44        ])
45
46
47        print("-" * 60)
48        print("Decoded Data")
49        print("-" * 60)
50        for name, value in iteritems(decoded):
51            print("%s\t" % name, hex(value) if isinstance(value, int) else value)
52        print("-" * 60)
53        print("-" * 60)
54
55    client.close()
```

### Modbus Payload Code

```

1  def run_payload_server():
2      # ----- #
3      # build your payload
4      # ----- #
5      builder = BinaryPayloadBuilder(byteorder=Endian.Little,
6                                     wordorder=Endian.Little)
7      builder.add_string('abcdefgh')
8
9      block = ModbusSequentialDataBlock(1, builder.to_registers())
10     store = ModbusSlaveContext(di=block, co=block, hr=block, ir=block)
11     context = ModbusServerContext(slaves=store, single=True)
12
13     identity = ModbusDeviceIdentification()
14     identity.VendorName = 'Pymodbus'
15     identity.ProductCode = 'PM'
16     identity.VendorUrl = 'http://github.com/riptideio/pymodbus/'
17     identity.ProductName = 'Pymodbus Server'
18     identity.ModelName = 'Pymodbus Server'
19     identity.MajorMinorRevision = version.short()
20     # ----- #
21     # run the server
22     # ----- #
23     StartTcpServer(context, identity=identity, address=("0.0.0.0", 5020))

```

## Modbus Payload Server Code

```

1  df_trial = df[5].append(df[6], ignore_index=True)
2  y_trial = y[5].append(y[6], ignore_index=True)
3  X_train, X_test, y_train, y_test = train_test_split(df_trial, y_trial, stratify=y_trial, test_size=0.2)
4  print(y_train.value_counts()), print(y_test.value_counts())
5  model = KMeans(n_clusters = 3)
6  model.fit(shuffle(X_train))
7  predicted_label = model.predict(X_test)
8  y_pred=predicted_label
9  # Model Accuracy: how often is the classifier correct?
10 print("Accuracy:", metrics.accuracy_score(y_test, y_pred))
11 target_names = ['class 0', 'class 1', 'class 2']
12 print(classification_report(y_test, y_pred, target_names=target_names))
13 cm=confusion_matrix(y_test, y_pred)
14
15 disp = ConfusionMatrixDisplay(confusion_matrix=cm)
16 disp.plot()
17 plt.show()

```

## K-Means

```

1 from sklearn.tree import DecisionTreeClassifier
2 clf = DecisionTreeClassifier()
3
4 df_trial = df[10].append(df[11], ignore_index=True)
5 y_trial = y[10].append(y[11], ignore_index=True)
6 X_train, X_test, y_train, y_test = train_test_split(df_trial, y_trial, stratify=y_trial, test_size=0.2)
7 print(y_train.value_counts()),print(y_test.value_counts())
8
9 # Train Decision Tree Classifier
10 clf = clf.fit(X_train,y_train)
11 predicted_label = clf.predict(X_test)
12 y_pred=predicted_label
13 # Model Accuracy: how often is the classifier correct?
14 print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
15 target_names = ['class 0', 'class 1', 'class 2']
16 print(classification_report(y_test, y_pred, target_names=target_names))
17 cm=confusion_matrix(y_test, y_pred)
18
19 disp = ConfusionMatrixDisplay(confusion_matrix=cm)
20 disp.plot()
21 plt.show()

```

## Decision Tree Classifier

```

1 from sklearn.ensemble import RandomForestClassifier
2 df_trial = df[10].append(df[11], ignore_index=True)
3 y_trial = y[10].append(y[11], ignore_index=True)
4 X_train, X_test, y_train, y_test = train_test_split(df_trial, y_trial, stratify=y_trial, test_size=0.2)
5 print(y_train.value_counts()),print(y_test.value_counts())
6
7 clf = RandomForestClassifier(max_depth=30, random_state=0)
8 clf.fit(X_train, y_train)
9
10 predicted_label = clf.predict(X_test)
11 y_pred=predicted_label
12 # Model Accuracy: how often is the classifier correct?
13 print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
14 target_names = ['class 0', 'class 1', 'class 2']
15 print(classification_report(y_test, y_pred, target_names=target_names))
16 cm=confusion_matrix(y_test, y_pred)
17
18 disp = ConfusionMatrixDisplay(confusion_matrix=cm)
19 disp.plot()
20 plt.show()

```

## Random Forest

## REFERENCES

1. **O. A. Alimi, K. Ouahada, and A. M. Abu-Mahfouz** (2020) A Review of Machine Learning Approaches to Power System Security and Stability. IEEE Access, vol. 8, pp. 113512-113531, 2020
2. **Lemay, Antoine, and José M. Fernandez** (2016) Providing SCADA Network Data Sets for Intrusion Detection Research. CSET @ USENIX Security Symposium.
3. **R. C. Borges Hink, J. M. Beaver, M. A. Buckner, T. Morris, U. Adhikari and S. Pan** (2014) Machine learning for power system disturbance and cyber-attack discrimination. 7th International Symposium on Resilient Control Systems (ISRCS)
4. **W. Rahman, M. Ali, A. Ullah, H. Rahman, M. Iqbal, H. Ahmad, A. Zeb, Z. Ali, M. Shahzad, and B. Taj** (2012) Advancement in Wide Area Monitoring Protection and Control Using PMU's Model in MATLAB/SIMULINK. Smart Grid and Renewable Energy, Vol. 3 No. 4
5. **CENTRAL ELECTRICITY AUTHORITY NEW DELHI** (2014) Transmission Planning Criteria. Northern Regional Power Grid (NRPG) Data, Power Grid Corporation of India Limited (PGCIL)
6. **Y. A. Farrukh, Z. Ahmad, I. Khan, and R. M. Elavarasan** (2021) A Sequential Supervised Machine Learning Approach for Cyber Attack Detection in a Smart Grid System. 2021 North American Power Symposium (NAPS)
7. **M. Ferragut, J. Laska, M. M. Olama and O. Ozmen** (2017) Real-Time Cyber-Physical False Data Attack Detection in Smart Grids Using Neural Networks. 2017 International Conference on Computational Science and Computational Intelligence (CSCI), pp. 1-6, doi: 10.1109/CSCI.2017.1.
8. **M. Mallouhi, Y. Al-Nashif, D. Cox, T. Chadaga and S. Hariri** (2011) "A testbed for analyzing security of SCADA control systems (TASSCS)," ISGT 2011, pp. 1-7, doi: 10.1109/ISGT.2011.5759169.
9. **B. Chen, N. Pattanaik, A. Goulart, K. L. Butler-purphy and D. Kundur** (2015) Implementing attacks for modbus/TCP protocol in a real-time cyber physical system test bed. 2015 IEEE International Workshop Technical Committee on Communications Quality and Reliability (CQR), pp. 1-6, doi: 10.1109/CQR.2015.7129084



10. **M. Kalech** (2019) Cyber-attack detection in SCADA systems using temporal pattern recognition techniques. *Comput. Secur.*, vol. 84, pp. 225-238
11. **G. Liang, J. Zhao, F. Luo, S. R. Weller, and Z. Y. Dong** (2017) A review of false data injection attacks against modern power systems. *IEEE Trans. Smart Grid*, vol. 8, no. 4, pp. 1630-1638