DETECTION AND PREVENTION OF AMORPHOUS CYBER-ATTACKS IN PROCESS CONTROL NETWORKS OF OIL AND GAS INSTALLATIONS

By

OBONNA UGOCHUKWU ONYEKACHI
B.Eng., M.Sc. (Eng.)
Reg. NO.: 20154989648

A PhD DISSERTATION
SUBMITTED TO THE POSTGRADUATE SCHOOL,
FEDERAL UNIVERSITY OF TECHNOLOGY, OWERRI.

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF DOCTOR OF PHILOSOPHY (Ph.D) DEGREE IN CONTROL ENGINEERING IN THE DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING, POST GRADUATE SCHOOL FEDERAL UNIVERSITY OF TECHNOLOGY, OWERRI.

SEPTEMBER, 2023.

CERTIFICATION

This is to certify that this work titled "Detection and Prevention of Amorphous Cyber-attacks in Process Control Networks of Oil and Gas Installations" was carried out by Obonna, Ugochukwu Onyekachi with registration number: 20154989648 in partial fulfillment of the requirements for the award of Doctor of Philosophy (Ph.D) Degree in Electrical and Electronic Engineering (Control Engineering option).

ENGR. PROF. F. K. OPARA (Principal Supervisor)	1/11/23 DATE
ENGR. DR. C. C. MBAOCHA (Co-Supervisor)	1/11/23 DATE
ENGR. DR. J. K. OBICHERE (Co-Supervisor)	2/11/23 DATE
ENGR. DR. N. CHUKWUCHEKWA (Head of Department)	02/11/2023 DATE
ENGR. PROF. M. C. NDINECHI (Dean, SESET)	4/11/23 DATE
PROF. B. O. ESONU (Dean, School of Postgraduate Studies)	..... DATE
Engr. Prof. O. I. Okoro EXTERNAL EXAMINER	02-11-2023 DATE

ACKNOWLEDGEMENTS

I wish to use this opportunity to acknowledge all the Professors, Senior Lecturers, Lecturers and staff of the Department of Electrical/Electronic Engineering who made so much impact on me while undergoing my studies. My special appreciation to my principal supervisor Prof. F.K. Opara and his co-supervisors Dr. C. C. Mbaocha and Dr. J. K. Obichere for their guidance and corrections. I also wish to appreciate specially, the Head of Department, Dr. N. Chukwuchekwa and the Dean of SESET Prof. M.C. Ndinechi. I will also like to mention Prof. Mrs G. N. Eze, Prof. Damian Dike, Prof. E.N.C. Okafor, Prof. G. A. Chukwudebe, Prof. Lazarus O. Uzoechina, and the other lecturers who supported me in guidance and direction. I am grateful to the Dean, Lecturers and staff of the Post Graduate School FUTO.

I will also like to thank the following for their contributions towards the successful completion of this research, Dr. Cosmos Ifeanyi Nwakamma and Love Ahakonye both of Kumoh National Institute of Technology, South Korea for their collaborations while carrying out the research. I will also like to mention Dr. Maryleen Amaizu for introducing me to the Foundational machine learning course on Cousera.org by Prof. Andrew Ng. Also Prof. Don Adjeroh of West Virginia University USA for granting me the privilege to participate in the annual WVAR-CRESH Summer School on AI and Smart Health, also the organizers of Machine Learning Summer Schools both in Taipei 2021 and Online Asian Machine Learning School 2021 and 2022 editions, the opportunities helped to sharpen my understanding of machine learning and AI, Miss Mmesoma Amaefule helped with data analysis. I am grateful to my employer Chevron Nigeria Limited.

Lastly, I am indebted to my loving wife Mrs. Nma Linda Obonna; my children Ebubechukwu, Munachimso, Chiziterem for their trust, encouragement and sacrifices; my mother Mrs. Margareth Ogbonna; my siblings Chikezie, Ijeoma, Onyinyechi; my late father Christopher Ohiaeri Ogbonna who believed so much in education and our late brother Udoka. May God bless and reward you all.

DEDICATION

I wish to dedicate this work firstly to the Almighty God for granting me the privileges and opportunities. Secondly, I dedicate this work to my late father Christopher Ohiaeri Ogbonna and my late elder brother Udoka Obonna, I am sure both will be happy with me for this great achievement. Lastly, I dedicate this work to my loving wife Mrs. Nma Linda Obonna and children Ebubechukwu, Munachimso and Chiziterem.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACK	Acknowledged
AES	Advanced Encryption Standard
AI	Artificial Intelligence
ANN	Artificial Neural Network
API	American Petroleum Institute
APTs	Advanced Persistent Threats
AUC	Area Under Curve
BDV	Blowdown Valve
BST	Boosted Tree
BT	Bagged Tree
CBC	Cipher Block Chaining
CCPSA	Calculus of Cyber-Physical systems and Attacks
CCTV	Closed Circuit Television
CEO	Chief Executive Officer
CGSVM	Support Vector Machine
CI	Critical Infrastructure
CIA	Confidentiality, Integrity and Availability
CM	Confusion Matrix
CPS	Cyber Physical System
CPU	Central Processing Unit
CR	Classification Rate
CSVM	Cubic Support Vector Machine

CT	Coarse Tree
DC	Direct Current
DCS	Distributed Control Systems
DDoS	Distributed Denial of Service
DMZ	Demilitarized Zone
DNP3	Distributed Network Protocol 3
DNS	Domain Name Server
DoS	Denial of Service
DT	Decision Tree
ECI	European Critical Infrastructure
ELM	Extreme Learning Machine
ESS	Emergency Shutdown Systems
EWS	Engineering Workstations
FAR	False Alarm Rate
FDI	Fault Detection and Isolation
FDIR	Fault Detection, Isolation and reconfiguration
FGSVM	Support Vector Machine
FN	False Negative
FP	False Positive
FT	Fine Tree
GPU	Graphics Processing Unit
HAL	High Alarm Condition
HH	High-High Trip Condition

HMI	Human Machine Interface
HSD	High-High Shutdown
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IACS	Industrial Automation and Control Systems
ICMP	Internet Control Message Protocol
ICS	Industrial Control System
IDS	Intrusion Detection Systems
IEC	International Electrotechnical Commission
IIoT	Industrial Internet of Things
IoT	Internet of Things
IP	Internet Protocol
IPS	Intrusion Protection Systems
IPv4	Internet Protocol version 4
IQR	Inter Quartile Range
ISO	International Organization for Standardization
IT	Information Technology
ITU	International Telecommunication Union
KNB	Kernel Naive Bayes
KNN	k-Nearest Neighbor
LAL	Low Alarm Condition
LAN	Local Area Network
LDR	Linear Discriminant

LL	Low-Low
LOF	Local Outlier Factor
LR	Logistic Regression
LSD	Low-Low Shutdown Trip Condition
LSTM	Long Short-Term Memory
MATLAB	Matrix Laboratory
MCE	Minimum Classification Error
MGSVM	Support Vector Machine
MitM	Man-in-the-Middle
MT	Medium Tree
MTU	Master Terminal Units
NCS	Network Controlled Systems
OPC	Open Platform Communication
OPC DA	Open Platform Communication data Access
ORNL	Oak Ridge National Laboratories
OSI	Open System Interconnection
OT	Operational Technology
PAL	Pressure Alarm Low
PAH	Pressure Alarm High
PCN	Process Control Network
PII	Personally Identifiable Information
PLC	Programmable Logic Controllers
PM	Performance Metrics

PSV	Pressure Safety Valve
PyOD	Python Outlier Detection
QDR	Quadratic Discriminant
RAM	Random Access Memory
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
ROC	Receiver Operator Characteristics
RTU	Remote Terminal Unit
RUBT	RSUBoosted Tree
SAE	Stacked Autoencoder
SCADA	Supervisory Control And Data Acquisition
SD	Subspace Discriminant
SDV	Shutdown Valve
SIS	Safety Instrumented Systems
SKNN	Subspace KNN
SQL	Structured Query Language
SQLi	Structured Query Language injection
SVM	Support Vector Machine
SYN	Synchronized
TCF	Tons Cubic Feet
TCP	Transmission Control Protocol
TN	True Negative
TP	True Positive

UDP	User Data Protocol
URL	Uniform Resource Locator
USB	Universal Serial Bus
WUSTL	Washington University in St. Louis

ABSTRACT

Amorphous cyber-attacks in process control networks (PCN) of oil and gas installations have posed a major cyber security challenge to the industry, due to the consistent deployment of unpredictable dynamic attack strategies by the attackers, which has made it difficult to predict their next attack modes. The aim of this dissertation is to monitor, detect, prevent and mitigate the effect of these malicious attacks on PCN. To achieve this, standard engineering structured methods, tools and techniques were applied by developing and analyzing mathematical models of the attacks, designing secured, centralized and distributed process control network architecture, designing a defense system capable of detecting false data injection attacks. The engineering materials, principles and concepts applied in the simulation, testing and validation of the developed models include: top-down structural approach, block diagrams, Machine learning algorithms, Deep learning toolkits, structured programming languages and simulation packages, such as: Python 3.0 Libraries, MATLAB, Allen Bradley PLC RSLogix 5000 emulator software, flowcharts and algorithmic representations of normal as well as compromised plant operations. Modelling and simulation of a 3-phase separator under attack was used to showcase attacks on PCN, predictions and forecasting using different machine learning algorithms. Real-time 68,722 SCADA dataset used in this research helped to overcome some of the shortfalls of previous researchers who used identical and repeated datasets that affected the learning ability of their algorithms and their final outcome. Several other machine learning algorithms and analytical tools were explored using the same dataset, but the Coarse Tree algorithm produced the best results with 100% accuracy, zero False Alarm Rate, one million observations per second prediction speed and 0.45488 seconds computation time. The results obtained showed the quality and precision of attack detections, hence the model's robust performance in detecting network intrusions. The various units and system's tests conducted showed highly improved results of about 95% comparatively with previous industry research results, thereby confirming the probability of useful contributions made in ameliorating theft in Oil and Gas sector of the economy. The successful integration of the developed models to the developed PCN architecture, shows that the cyber-attacks vulnerabilities on Oil and Gas infrastructures could be detected, prevented and reduced to barest minimum, thereby preventing production downtime, with adverse impact on the economy of the country, in general.

Keywords: Amorphous cyber-attacks, process control networks, machine learning, anomaly detection. distributed control systems, SCADA.

CHAPTER ONE

INTRODUCTION

1.1 Background of Study

Amorphous cyber-attacks on the process control networks (PCN) of oil and gas industry have continuously exposed the PCN to failures, there is need to proactively monitor, detect and prevent these malicious cyber-attacks on the critical infrastructure of the world major energy supply, thereby preventing catastrophic failures which results in fatalities, oil spills, fire and explosion with huge impact on the people, assets, environment and the economy. To ensure safety of personnel, environment and assets as well as maintain the stability of the produced crude oil, ensure effective 3-phase separation of the produced crude oil into crude oil, gas and water there is need to monitor pressure, crude oil level, water level, interface level, temperature, flow and so on, of the processes. These process variables in the Process Control Network PCN serves as inputs to the logic solvers or controllers which make real-time decisions on the final control elements to ensure a continuous and safe operation of the plant. A real-time adjustment or modification of the input variables results to the controller effecting the change on the operating conditions of the logic solvers which eventually results to outputs to the final control elements. There is need to ensure a secured communication between the field sensors, the controllers and the final control elements (Irmak & Erkek, 2018).

The integration of a real-time monitoring systems which has the capability to detect and prevent cyber-attacks to the PCN will go a long way in verifying data integrity and communication between the sensors, the controllers and the final control elements through the network devices. Integrating the real-time anomaly detection system into the existing PCN architecture will improve the security of the PCN with a system which monitors the communication between the sensors, the controllers and the final control elements with the capability of detecting outliers in the data stream, annunciate alarms based on the alarm criticality, then the console operator will take action either to resolve

the issues locally or escalate for further investigation (Malviya, 2019)(Singh et al., 2019)(Peerlkamp & Nieuwenhuis, 2010).

The petroleum industry is a worldwide energy industry that is highly dependent on technology for its processes, communication and operations. It is widely understood that information security is important for office computing systems such as desktop computers, laptops, servers, software programs, and so on. what is less recognized is that computer technology has become pervasive throughout the entire organization, including network access to plant equipment to allow vendors to maintain systems remotely, and remote access connections to process control systems (Supervisory Control And Data Acquisition - SCADA) to allow engineers to troubleshoot problems. In all of these environments, improper controls could allow unauthorized individuals to accidentally or intentionally harm the information assets of the petroleum industry (American Petroleum Institute, 2005).

The intercommunication between sensors, transducers, controllers, final control element on a SCADA network and other network devices which include computers, network adapters, switches and routers, in a networked controlled system (NCS) has created several points of vulnerabilities. Researches have shown that apart from unforeseen failure of a network component in a Process Control Systems, exposure to malicious attacks may also impact the network availability which may end up in system downtime and its resultant effect on the throughput. Malicious attacks on a Process Control Network may be in the form of Denial of Service (DoS) and Distributed Denial-of-Service (DDoS), Man-in-the-middle (MitM) attack which could lead to delay in communication among network devices, false data injection which may result to equipment or system shutdown or malfunction but can also result to a major catastrophic failure which may lead to outright loss of control of the entire network (Johnson, 1999).

A PCN is a network of real-time industrial control systems ICS which manage, monitor, and control industrial infrastructure. PCNs make use of software, hardware, networks

and their connectivity for accessing, controlling and transferring data with each other. PCN is also known as Distributed Control System DCS or SCADA (Malviya, 2019). SCADA system controls many Critical Infrastructures such as Power Grids, Water Treatment Systems, Nuclear Reactors, Sewage Treatment systems, Oil and Gas Installations. SCADA network could be each of combination for the following systems: Distributed Control Systems (DCS), Safety Instrumented Systems (SIS), Programmable Logic Controllers (PLC), Remote Terminal Unit (RTU).

Oil and gas installations in Nigeria are classified as critical infrastructure due to their direct positive influence on the economy of the nation. The oil and gas industry in Nigeria is faced with myriad of challenges ranging from theft, pipeline vandalism, illegal bunkering and now intrusion attacks (G. Wilson, 2014)(Hunga & Adishi, 2017). As stated in American Petroleum Institute (API) Security Guidance "Safe and reliable energy is a vital link in the nation's critical infrastructure. Petroleum products play an important role in our national economy, national security which are integral to American way of life" (American Petroleum Institute, 2005). Cyber threats or attacks are one of the major setbacks the Industrial Control System (ICS) is presently facing in our world today. A malicious attack on an oil and gas Process Control Network (PCN) could have severe consequences which range from economic loss to the host nation and stakeholders, catastrophic loss of multi-billion-dollar assets, impact to the environment (environmental pollution) and eventual loss of lives.

In the last decade, the highly sophisticated Advanced Persistent Threats (APTs) which focuses majorly on critical infrastructures with devastating impacts has been proven to be evolving with derivatives and not much attention has been given to the detection and prevention by researchers thereby leaving the oil and gas industry (critical infrastructure) vulnerable to attacks. For continual safe operation of an oil and gas facility, there is need to ensure that all the network components of a Process Control System are protected from intruders or attackers through effective monitoring and

surveillance which may help in early detection of attacks and proper mitigation of such attacks (Kaspersky, 2019, 2022)(Marchetti et al., 2016).

1.2 Problem Statement

Process Control Network systems are becoming increasingly open due to network expansion, use of ready-made network components, system integration and connection to the internet which has increased system complexity and more exposure, with increased vulnerability to cyber-attacks (Okada, 2021)(Alanazi et al., 2023)(Beaver et al., 2013).

Also, Reliance on standard network technologies, their associated communication protocols and their preference over proprietary dedicated systems has exposed SCADA Systems to insecure communication protocols such as Modbus, DNP3 and OPC DA, which has exposed the Control Systems to Malicious cyber-attacks. Example of these standard protocols are Modbus/TCP over Vnet/IP (Mohammed, Abubakar Sadiq and Saxena, Neetesh and Rana, 2022)(Smurthwaite & Bhattacharya, 2020)(Parian et al., 2020).

Moreso, as technology is evolving Advanced Persistent Threats are becoming resilient and degenerating into newer derivatives with modified APTs characteristics and are beating Antivirus software as well as Intrusion Detection Systems, this exposes Industrial Control Systems to series of devastating amorphous cyber-attacks (Al-Rabiaah, 2018)(Marchetti et al., 2016)(Virvilis & Gritzalis, 2013a) .

1.3 Objective of Study

The main objective of this research is the detection and prevention of amorphous cyber-attacks in Process Control Networks of oil and gas installations.

The specific objectives are:

i. to develop a mathematical model and analysis of Process Control Networks for attacks and monitoring respectively,
ii. to design a centralized and distributed Process Control Network Architecture for attack detection and identification monitoring,
iii. to design a defense system capable of detecting false data injection attack by developing a test bed that prevents Advanced Persistent Threats and its derivatives in Process Control Networks,
iv. to simulate the various system models that is designed, and
v. to validate the results obtained from the models by comparing them with the results achieved by applying already existing datasets on the developed models.

1.4 Significance of Study

The benefits of this research include, but not limited to;

i. further investigation on the impact of false data injection (deception attack) from an intruder thereby compromising the data integrity from the controllers to the remote Final Control Element. The false data injection attacks which may not be easily detected ab initio, is an area that is yet to be thoroughly investigated,
ii. the development of a better robust model that detects zero-day cyber-attacks as well as the evolving Advanced Persistent Threats (APTs) and its derivatives which has target as the Industrial Control Systems (ICS) of Process Control Networks in the Oil and Gas Production Facilities,
iii. the use of unsupervised Machine Learning algorithm to develop a system that could predict and prevent a cyber-attack of an oil and gas installation is an added advantage in this research because it will ensure real time data supply and immediate or quick response that can prevent devastating disaster.

1.5 Scope of Study

The scope of this research includes to identify the different points of cyber vulnerabilities in the oil and gas industry assets with special focus on false data injection attack in a Process Control Network and effectively manage identified security risks as well as proffer preventive measures to avoid impact to life, assets and environment. Again, the evolution of technology yielded a highly sophisticated threats called Advanced Persistent Threats (APTs) which focuses majorly on the critical industrial sector. APTs when they gain unauthorized access to a computer network, hence, have the capability to remain undetected for an extended period and usually do not need the internet for spreading. Recently, a number of different APTs derivatives have emerged which needs special attention to analyze them in order to find solutions to protect against existing and future APTs. More importantly is to unravel and provide the amorphous solutions towards the various dynamic nature off attacks in recent times. Hence, it has become imperative to use Machine Learning as a tool which has the capability of learning the behavior of a typical Process Control Network data packets in order to detect anomalies when they eventually occur as well as predict future attacks.

CHAPTER TWO

LITERATURE REVIEW

2.1 Cyber Attacks and Cyber Security

Cyberattack is defined as "an attempt to gain illegal access to a computer or computer system for the purpose of causing damage or harm" (Merriam-Webster Dictionary, 2018). Oxford dictionary defines cyberattack as "an attempt by hackers to damage or destroy a computer or system (Oxford Dictionary, 2018). In computers and computer networks, an attack as defined in ISO/IEC 27000:2018 is "any attempt to destroy, expose, alter, disable, steal, or gain unauthorized access to or make unauthorized use of an asset (International Standard Organization, 2018). Cisco Systems a networking hardware manufacturing company defines cyberattack as a malicious and deliberate attempt by an individual or organization to breach the information system of another individual or organization. Usually, the attacker seeks some type of benefit from disrupting the victims network (Cisco.com worldwide, 2019a).

According to Hill, "Cyber security can be defined as the collection of tools and procedures that ensure availability, integrity, authenticity and confidentiality of information and communications. Both computer systems and networks can be attacked to prevent their use (denial of service DoS), to compromise and alter the data they store or transport, to compromise ("spoof") the identification of the originator, and to read data without authorization (eavesdropping) (Hill, 2015). The Hill's definition is a simplified definition of Cyber security found in the ITU-T X.1205 recommendation which states that "Cybersecurity is the collection of tools, policies, security concepts, security safeguards, guideline, risk management approaches, actions, training, best practices, assurance and technologies that can be used to protect the cyber environment and organization and user's assets. Organization and user's assets include connected computing devices, personnel, infrastructure, applications, services, telecommunication systems, and the totality of transmitted and/or stored information in the cyber environment. Cybersecurity strives to ensure the attainment and maintenance

of the security properties of the organization and user's assets against relevant security risks in the cyber environment. The general security objectives comprise the following: Confidentiality, Integrity and Availability (CIA) (International Telecommunication Union, 2008).

2.1.1 Amorphous Cyber Attacks

Merriam-Webster Dictionary definition of the term amorphous is

i. "Having no definite form: shapeless"
ii. "Being without definite character or nature: unclassifiable"
iii. "Lacking organization or unity"

Synonyms of the word amorphous include: shapeless, formless, unformed, unshaped, structureless, unstructured, indeterminate, indefinite, vague, unclassifiable and so on. (Merriam-Webster Dictionary, 2019). Statistics has shown that cyber-attacks can occur in different forms and could take different shapes. The term amorphous is used to define cyber-attacks because there is no clearly defined form, shape or classification in which cyber-attack could be carried out.

Julian Jang-Jaccard and Surya Nepal in their survey of emerging threats in cybersecurity explained that "the exponential growth of the internet interconnections has led to a significant growth of cyber-attack incidents often with disastrous and grievous consequences. Our society, economy, and critical infrastructures have become largely dependent on computer networks and information technology solutions. Cyber-attacks become more attractive and potentially more disastrous as our dependence on information technology increases. They also stated that the reason why cyber-attacks flourish is because cyber-attacks are cheaper, convenient and less risky than physical attacks. Cyber criminals only require a few expenses beyond a computer and an internet connection. They are unconstrained by geography and distance. They are difficult to identify and prosecute due to anonymous nature of the internet. Given that attacks

against information technology systems are very attractive, it is expected that the number and sophistication of cyber-attacks will keep growing (Jang-jaccard & Nepal, 2014).

Amin, Litrico, Sastry and Bayen in their work on "Cyber Security of Water SCADA Systems" explained that network-induced vulnerabilities arise in Network Controlled Systems (NCS) because of four factors. The first was due to wider deployment of off-the-shelf IT services thereby inheriting the vulnerabilities of the devices which include their associated failures. Secondly, the proprietary network protocols of traditional SCADA systems are currently being upgraded to open protocols, thereby making it easier for attackers to learn about the system operations. The third factor is that sensor-control data is being made accessible to authorized remote users including vendors through the business network and internet. This increases the risk of insider attacks. The last is, the existence of organized cybercrime groups enhances attackers capabilities to conduct intrusion into NCS (Amin et al., 2013b, 2013a).

2.1.2 The Common Forms of Cyber-Attacks

There are different forms and classification of cyber-attacks which are:

i. Denial-of-Service attacks (DoS)
ii. Structured Query Language (SQL) Injection attacks (SQLi)
iii. Malware attacks
iv. Man-in-the-Middle (MITM) attacks
v. Phishing attacks
vi. Zero-day exploit
vii. Drive-by attack
viii. Password attack
ix. Insider attack

2.1.2.1 Denial-of-Service Attacks (DoS)

This form of attack floods the systems, servers or networks with unsolicited traffic to exhaust resources and bandwidth. When DoS occurs, the system is kept busy perpetually and it will no longer be able to complete the legitimate requests. In this case, the attacker does not gain increased access to the system or server but derives satisfaction when there is denial of requested services. The intent of this DoS attack may be to ensure that the system is offline so that a different form of attack can be launched on the network (Jeff Melnick, 2018)(Mohammed, Abubakar Sadiq and Saxena, Neetesh and Rana, 2022).

When attackers use multiple compromised devices to launch this form of attack it is known as Distributed Denial-of-Service attacks (DDoS). A distributed-denial-of-service, attack is the bombardment of simultaneous data request to a central server. The attacker generates these requests from multiple compromised systems and hopes to exhaust the internet bandwidth of the target and the system RAM. The ultimate goal of the attacker is to ensure the target's system is crashed thereby disrupting the business of the target (Alhaidari & Al-Dahasi, 2019)(Cisco.com worldwide, 2019b).

There are three general types of DDoS namely:

a. Volume-based attacks - This can be of two types User Data Protocol (UDP) flood and Internet Control Message Protocol (ICMP) (ping) flood.

i. In the UDP floods attack, a remote server has its random ports flooded with UDP packets requests. The host runs verification checks of the ports for the appropriate applications, if no application is found, the system responds to all request with a "destination unreachable" packet. The resulting flood of traffic can overwhelm the service (Shetty, 2021).

ii. In the ICMP (ping) flood attack, ICMP echo request packets (pings) are sent to the host. Pings are common request used to measure the connectivity of two network devices. When a ping request is sent to the server, it quickly responds. However, in a ping flood, the attacker uses an extensive series of

pings to exhaust the incoming and outgoing bandwidth of the targeted server (Cisco.com worldwide, 2019b)(Tufan & Tezcan, 2021).

b. Application attacks – This can be of two types HTTP flood and Slowloris.

i. HTTP flood – This is an OSI Layer 7 application attack that uses botnets, often referred to as a "zombie army". In this type of attack which is a standard get-and-post requests, the web server or application server is flooded with request which may cause it to shut down. These unplanned attacks can be particularly difficult to detect because they appear as valid network traffics.

ii. Slowloris – Named after the Asian primate which moves slowly. The attacker sends small portions of an HTTP request to a server in timed intervals, so the request does not time out, and the server waits for it to be completed. These unfinished requests exhaust bandwidth and affect the server's ability to handle legitimate requests (Cisco.com worldwide, 2019b).

c. Protocol attack – This can be of two types SYN flood and Ping of Death.

i. In SYN flood attack, the attacker sends seemingly normal requests termed SYN to a server, and the server responds with a SYN-ACK (synchronized-acknowledgement) request. Normally, a client typically sends back an ACK then a connection is established. In a SYN flood attack, the attacker's device floods the target server with connection requests, the server's reply of SYN-ACK will be received but the attacker will not respond with a final ACK. The server is left with a large number of unfinished SYN-ACK requests that burdens the entire system and will eventually time out.

ii. In Ping of Death attack, the attacker sends a normal ping request that is either fragmented or oversized which ends up crashing or freezing the server. Typically, the standard size of an IPv4 header is 65,535 bytes, but when a larger ping is sent, the file will be fragmented by the targeted server. Later, when the server formulates a response, the reassembly of this larger file can cause a buffer overload and causes a crash (Cisco.com worldwide, 2019b).

2.1.2.2 Structured Query Language (SQL) Injection Attacks (SQLi)

This type of attack occurs when a malicious code is inserted by an attacker into a server that uses SQL and forces the server to reveal information it normally would not. An SQL injection could be carried out by an attacker, simply by submitting malicious code into a vulnerable website search box.

2.1.2.3 Malware Attacks

These are malicious software which includes spyware, ransomware, viruses and worms. Malware breaches a network through a vulnerability, typically when a user clicks a dangerous link or email attachment which then installs the risky software. Once installed in a system, a malware can do the following:

i. Block access to key components of the network (Ransomware)
ii. Installs malware or additional harmful software.
iii. Covertly obtains information by transmitting data from the hard drive (Spyware)
iv. Disrupts certain components and renders the system inoperable (Cisco.com worldwide, 2019a).

2.1.2.4 Man-in-the-Middle (MitM) Attacks

This is the same thing as eavesdropping attacks and occurs when attackers deliberately insert themselves into a two-party communication. Once the traffic is interrupted by the attacker, they can filter and steal data. The common types of MitM attacks include:

a. **Session Hijacking** – In which the attacker inserts himself between the communications of a trusted client and a network server thereby hijacking the session. The attacking device substitutes its IP address for the trusted client while the server continues the session, believing it is communicating with the trusted client. Meanwhile the attacker's device would have disconnected the client from the

server and replace the clients IP address with its own IP address and spoof the clients sequence numbers. The dialog will then continue between the attacking device and the server (Jeff Melnick, 2018)(Khraisat & Alazab, 2021).

Figure 2.1(a) and (b) shows a man in the middle attack by session hijacking. The victim's device with IP address 192.168.1.25 with an established trusted session with the server will be substituted by an attacker device see Figure 2.1(b) while the server continues to trust the already established session as shown in Figure 2.1(a). Thereby, denying the victims device from services running on the server while the attacker's devices take over the session and may exploit the network.

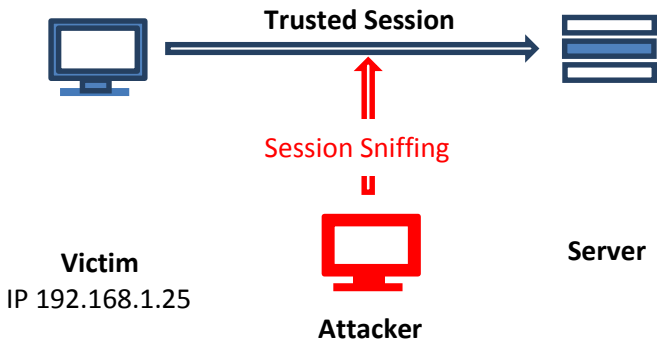


Figure 2.1(a) Session Sniffing (Jeff Melnick, 2018)

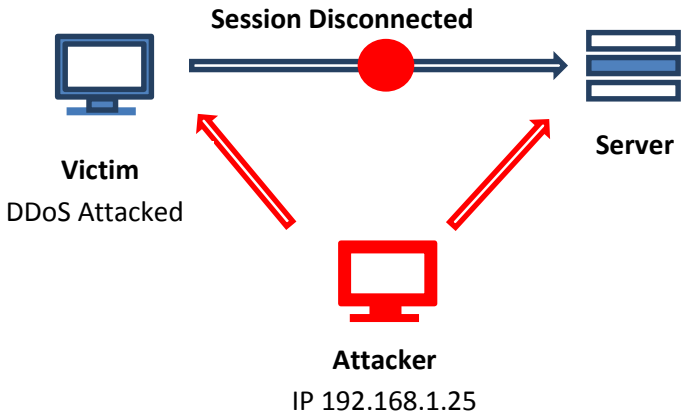


Figure 2.2(b) Session Hijacking (Jeff Melnick, 2018)

b. **IP Spoofing** – Attackers use IP spoofing to convince a system that it is communicating with a known and trusted device on the network thereby granting the attacker access to the network resources. In this case the attacker sends a packet with the IP source address of a trusted device on the network which makes the target host to act upon it once received.
c. **Replay Attack** – This occurs when an attacker intercepts and saves messages from old sessions and tries to send them at a later date or time thereby impersonating one of the participants.

   Two common points of entry for MitM attacks are:

   i. Unsecured public Wi-Fi: attackers insert themselves between the network and a visitor's device, the visitor ignorantly passes all information through the attacker.
   ii. Once a device has been breached by a malware, the attacker can install software to process and extract all of the victim's information.

2.1.2.5 Phishing Attacks

This is an act of sending fraudulent communications that appear to come from a legitimate and reputable source, usually through email. The aim is to compromise and steal sensitive data like credit card details and login information or to install malware on the victim's machine.

There are different types of phishing:

a. **Deceptive Phishing** – This is the most common type of phishing in which an attacker attempts to obtain confidential information from the victims. Attackers use the information to steal money or to launch other attacks. A fake email from a bank asking for a click on a link and verify your account details is an example of deceptive phishing.
b. **Spear Phishing** – This type targets specific individuals instead of a wide group of people. Attackers often research their victims on social media and other sites and

can customize their mode of communications which make them appear more authentic. Spear phishing is often the first step used to penetrate a company's defenses and carry out a targeted attack. Email spoofing is one of the simplest ways a hacker can use in spear phishing, which is when email is made to appear as if it is coming from someone you know or trust such as business associates. Websites can also be cloned by attackers thereby deceiving unsuspecting victims into entering their personally identifiable information (PII) or login credentials.

c. **Whaling** – When attackers go after a "big fish" like a CEO, it is called whaling. These attackers often spend considerable time profiling the target to find the opportune moment and means of stealing login credentials. Whaling is of particular concern because high-level executives are able to access a great deal of company information (Cisco.com Worldwide, 2019).
d. **Pharming** – Similar to phishing, pharming sends users to a fraudulent website that appears to be legitimate. However, in this case, victims do not even have to click a malicious link to be taken to the bogus site. Attackers can infect either the user's computer on the website's DNS server and redirect the user to a fake site even if the correct URL is typed in (Cisco.com Worldwide, 2019).

2.1.2.6 Zero-day Exploit

This type of attack hits after a network vulnerability is announced but before a patch or solution is implemented. Attackers target the disclosed vulnerability during this window of time. Zero-day vulnerability threat detection requires constant awareness (Cisco.com Worldwide, 2019).

2.1.2.7 Drive-by Attack

This is a download attack which is a common way of spreading malware. Attacker's plant malicious scripts on websites that are insecure. The scripts can install malware directly on the computer of unsuspecting victims whenever they visit the website or it

can redirect the victim to a cloned website managed by the attackers. In this form of attack, the victim may not have to click on a download button or open a malicious email attachment to become infected. The attacker may exploit applications, web browsers or operating systems that are not updated.

2.1.2.8 Password Attack

Users are authenticated to the network when the correct passwords are provided. A user's password can be compromised when he fails to protect it, example by writing out and keeping in an unsafe location or by using passwords that can be easily discerned example 1234 and so on. Sniffing a network connection to acquire unencrypted simple passwords using social engineering, gaining access to a password database or outright guesswork (Jeff Melnick, 2018).

a. **Brute force attack** – This is a random password guessing approach in which the combination of the person's name, date of birth, hobbies, places of interest, job titles or similar items are used.
b. **Dictionary attack** – An encrypted file that contains the password is copied and applied to a dictionary of commonly used passwords in an attempt to retrieve the password.

2.1.2.9 Insider Attack

These are internal users within the organization, such as employees, former employees, business associates and contractors who have malicious intents, with correct authentication to the network and knowledge of the company information. Insider attacks has been identified as one of the most dangerous cyber threats for critical infrastructures (CIs), as the attacker continues sending real and legitimate control commands to other network devices, it can lead to catastrophic damages to CIs. Insider attacks usually have huge impact and greater success rate since it is difficult to predict

when they want to attach and how they want to attack thereby making it unpreventable. (Nasr & Varjani, 2014b)(Nasr & Varjani, 2014a)(Tang et al., 2020)(Gönen et al., 2020).

2.1.2.10 Solutions to Some of the Outlined Forms of Attacks

a. DDoS can be controlled by effective network scalability and ensuring the edge routers are configured to manage unsolicited network traffics (Cisco.com worldwide, 2019b).
b. For a TCP SYN flood attack the following countermeasures will help:
    i. Install the servers with the protection of a firewall which has a configuration to stop inbound SYN packets
    ii. Increase the size of the configured connection queue and decrease the configured timeout on open connections
c. To protect the network devices from ICMP attacks, all the IP-directed broadcast must be disabled at the router. Also, the configuration of end systems to keep them from responding to ICMP packets from broadcast addresses.
d. A firewall can be used to block Ping of death attacks which will check fragmented IP packets for maximum size.
e. Replay attack can easily be countered with session timestamps.
f. MitM attackers can be prevented using encryption and digital certificates
g. The risk of Phishing can be reduced by adhering to the following guidelines:
    i. Analyzing emails to ascertain the source before opening.
    ii. Point the mouse but do not click to the “from” email address to ascertain the source of the email
    iii. Using the mouse to hover over the links in the email to see the actual URL
h. Drive-by attacks can be prevented by ensuring applications, browsers and operating systems are updated. Verification of website authenticity before use goes a long way to protect unsuspecting victims, look out for secured websites HTTPS instead of HTTP. Do not keep too many out-of-date applications on your system as this creates vulnerabilities.

i. Insider attacks can be prevented when policies and control measures are enforced based on the CIA, typical example is the enforcement of account and password policies. The thorough implementation of physical security in the organizations work environment will aid to ensure appropriate logs of personnel movement and access. The use of security devices like CCTV cameras, motion sensors as well as security software.

Nisioti et al, in their work identified that in the first quarter of 2017, DDoS attacks increased by 30% when compared to the previous year. Twelve of these attacks exceeded a bandwidth of 100 Gigabits per second (Gbps), while two exceeded 300 Gbps against the media and entertainment sectors. These attacks may not only disrupt the normal operation of organizations and governments, but may also have social consequences which may affect and impact critical infrastructures (Nisioti et al., 2018).

2.1.3 Critical Infrastructure Overview

As defined in the International Critical Information Infrastructure Protection handbook "Critical Infrastructure (CI) is often identified as that infrastructure whose incorrect functioning, even for a limited time period, may negatively affect the economy of individual subjects or groups, involving economic losses and/or even expose people and things to a safety and security risk (Elgin M. Brunner & Suter, 2008)(Montanari & Querzoni, 2014).

Within the European Union a Critical Infrastructure is defined as an asset, system or part thereof located in member states which is essential for the maintenance of vital social functions, health, safety, security, economic or social well-being of people, and the disruption or destruction of which would have a significant impact in a member state as a result of the failure to maintain those functions. The European Critical Infrastructure (ECI) means critical infrastructure located in Member Sates the disruption or destruction of which would have a significant impact on at least two Member States (Council Directive 2008/114/Ec on the Indentification and Designation

of European Critical Infrastructures and the Assessment of the Need to Improve Their Protection, 2008).

The designation of a critical infrastructure as a European Critical Infrastructure (ECI) is the result of a technical-political process, which arises from the potential impact that can be caused by a failure/destruction of an infrastructure in terms of sectoral and inter-sectoral relevance.

The inter-sectoral evaluation criteria relate to:

a. Potential victims, in terms of number of fatalities or injuries
b. Potential economic effects, in terms of financial losses, deterioration of products or services, and environmental effects/damages
c. Potential effects on population, in terms of impact on public confidence, physical suffering and disruption of daily life, including the loss of essential services (Council Directive 2008/114/Ec on the Indentification and Designation of European Critical Infrastructures and the Assessment of the Need to Improve Their Protection, 2008).

The EU directive quoted in section 2.3 ensures that obligations of owners/operators for what concerns the security of their infrastructure should be made to prevent, or at least limit, the consequences on other nations (Montanari& Querzoni, 2014). The United States of America defines critical infrastructure as physical and cyber systems and assets (whether physical or virtual), that are so vital to the United States that their incapacity or destruction of such systems and assets would have a debilitating impact on security, national economic security, national public health or safety, or any combination of those matters (Department of Homeland Security, 2017b).

Critical infrastructure must be secure and able to withstand and rapidly recover from all hazards. Proactive and coordinated efforts are necessary to strengthen and maintain secure, functioning, and resilient critical infrastructure – including assets, networks, and systems – that are vital to public confidence and the Nation's safety, prosperity, and

well-being. This endeavor is a shared responsibility among Federal, state, local, tribal, and territorial entities, and public and private owners and operators of critical infrastructure (Department of Homeland Security, 2017b).

Cyber threat is one of the major setbacks the Industrial Control Systems has faced in recent years. Oil and Gas installation is categorized as critical infrastructure considering the fact that it fuels the economy of many nations of the world. An attack on an Oil and Gas installation could lead to a catastrophic impact on the environment, lives, investment and world economy. The different forms of cyber-attacks could impact oil and gas production ranging from minor impacts like equipment shutdown, process upset to major catastrophic damages which has the capability to cause environmental impact, damage to equipment and even loss of human lives. Typically, cyber-attack could lead to Denial-of Service (DoS), Distributed Denial-of Service (DDoS), delay in communication among network devices, and even outright loss of control of an industrial network.

Supervisory control and data acquisition (SCADA) systems are key components of the critical infrastructure as they are used to control important sectors such as oil and gas production and transportation pipelines, smart grids, water treatment plants, chemical manufacturing plants, refineries and so on. Malicious intrusion to such systems could lead to catastrophic impact on humans, material and economic damages. Thus, the security of the SCADA system is very important for continuous operations and protection against hostile, cyber-terrorist attacks, also to ensure the integrity of processes and resilient actions (Abou el Kalam, 2021).

Research has shown the lack of preparedness in detecting and securing the Oil and Gas Process Control Network from external threats and attacks. With the advent of inter-communication between Computers, Servers, sensors, transducers, transmitters, controllers, final control elements and other network devices, it has become imperative to ensure all the components of the Process Control Network is protected from intruders or attackers. With the advancement in Industrial Internet of Things (IIOT), early

identification and prevention of attacks which can lead to PCN disaster can be achieved by continuous monitoring using algorithm based smart monitoring systems (Ogu, Achumba, et al., 2022)(Ogu, Chukwudebe, et al., 2022).

The Process Control System of an Oil and Gas installation is robust in nature and integrates the communication from different devices, networks, topologies, protocols and architectures to ensure the safety of life and assets in a highly volatile work environment. It integrates the following: Distributed Control Systems (DCS), Supervisory Control and Data Acquisition Systems (SCADA), Programmable Logic Controllers (PLC), Emergency Shutdown Systems (ESS), Remote Terminal Units (RTU) for Telemetry Systems, Smart Meters, and other Intelligent field Instruments. A typical Process Control System of an Oil and Gas installation may have different architecture which incorporates some or all the systems listed earlier to work as a standalone or an integral component of a multilayered process control network.

The focus of this work is to proffer proactive measures to identify and mitigate against external aggressions and ensure that the integrity, confidentiality, availability and reliability of the individual and collective components of the Process Control Network are fortified against malicious intruder attacks. Attack models will be developed to depict attack scenarios using established methodology, assessed and solutions proffered to boost preparedness for a typical Oil and Gas Process Control Network.

2.1.4 Oil and Gas Facility as a Critical Infrastructure

The oil and gas industry also known as petroleum industry which includes exploration, extraction, transportation refining and marketing of petroleum industry. Petroleum is important to most industries and it is useful in the sustenance of modern civilization and industrialization. Based on the EU Council Directive 2008/114/ec. The European Critical Infrastructure ECI sectors are as listed in the Table 2.1 (Council Directive 2008/114/Ec on the Indentification and Designation of European Critical Infrastructures and the Assessment of the Need to Improve Their Protection, 2008).

Table 2.1 The European Critical Infrastructure (ECI) sectors classification (Council Directive 2008/114/Ec on the Indentification and Designation of European Critical Infrastructures and the Assessment of the Need to Improve Their Protection, 2008)

Sector	**Subsector**	
I Energy	1. Electricity	Infrastructure and facilities for generation and transmission of electricity in respect of supply electricity
	2. Oil	Oil production, refining, treatment, storage and transmission pipelines
	3. Gas	Gas production, refining, treatment, storage and transmission pipelines. LNG terminals
II Transport	4. Road Transport 5. Rail Transport 6. Air Transport 7. Inland Waterways Transport 8. Ocean and short-sea shipping and ports	

As defined by United States Homeland Security, there are 16 critical infrastructure sectors whose assets, systems, and networks, whether physical or virtual, are considered so vital to the United States that their incapacitation or destruction would have a debilitating effect on the security, national economic security, national public health or safety, or any combination thereof (Department of Homeland Security, 2017b). The US Presidential Policy Directive 21 (PPD-21) identifies 16 critical infrastructure sectors as shown in Table 2.2 (The White House - Office of the Press Secretary, 2013).

Table 2.2 The US Critical Infrastructure Sectors (Department of Homeland Security, 2018)

S/No	Sector	Sector-Specific Agency (SSA)
1	Chemical	Department of Homeland Security
2	Commercial Facilities	Department of Homeland Security
3	Communications	Department of Homeland Security
4	Critical Manufacturing	Department of Homeland Security
5	Dams	Department of Homeland Security
6	Defense Industrial Base	Department of Defense
7	Emergency Services	Department of Homeland Security
8	Energy	Department of Energy
9	Financial Services	Department of the Treasury
10	Food and Agriculture	US Department of Agriculture and Department of Health and Human Services
11	Government Facilities	Department of Homeland Security and General Services Administration
12	Healthcare and Public Health	Department of Health and Human Services
13	Information Technology	Department of Homeland Security
14	Nuclear Reactors, Materials and Waste	Department of Homeland Security

15	Transportation Systems	Department of Homeland Security and Department of Transportation
16	Water and Wastewater Systems	Environmental Protection Agency

The US energy infrastructure fuels the economy of the 21st century. Without a stable energy supply, health and welfare are threatened, and the US economy cannot function. The energy infrastructure is divided into three interrelated segments: electricity, oil and natural gas. Presidential Policy Directive 21 identifies the Energy Sector as uniquely critical because it provides an "enabling function" across all critical infrastructure sectors. More than 80 percent of the country's energy infrastructure is owned by the private sector, supplying fuels to the transportation industry, electricity to households and businesses, and other sources of energy that are integral to growth and production across the nation. The reliance of virtually all industries on electric power and fuels means that all sectors have some dependence on the Energy Sector (Department of Homeland Security, 2017a).

The Oil and Gas sector is one of the most important critical infrastructure sectors for economy, housing and transportation. According to market reports, upstream investment reached USD 500B only for 2019, with the global oil demand stabilizing around 1M barrels/day. In Canada, 97% of oil and petroleum products are transported via pipelines. The consumption of natural gas worldwide was recorded to be around 140T cubic feet (Tcf) for 2018 alone, and is projected to increase to 203Tcf by 2040. According to American Petroleum Institute's report of 2019, the US pipeline system (midstream infrastructure) consists of 2.7M miles of pipelines transferring assets between locations. Midstream infrastructure connects to refineries and facilities to distribute oil and gas to the end users (downstream infrastructure) (Stergiopoulos et al., 2020).

2.1.5 Cyber Warfare in the Oil and Gas Industry

The oil and gas industry play a major role in the energy sector of the economy as the operation of most critical assets are dependent on this key industry. Considering the fact that Industrial Control System make use of network devices, data communication protocols and architecture of regular Information Technology (IT) systems, it implies that vulnerabilities are common. Several intentional and unintentional ICS cyber incidents have been recorded in recent times ranging from minor impacts, to significant equipment damage, fatality, process upset. There is a significant difference between the security philosophies of enterprise IT and ICS. The purpose of the enterprise IT security is to protect the servers and the network components from attack while the purpose of ICS security is to protect the safe operation of the critical infrastructure facility regardless of what may befall the rest of the network.

Oil and Gas infrastructures are divided into three broad categories: upstream, midstream and downstream infrastructures. Upstream infrastructures support seismic surveys, exploration activities and drilling operations, while midstream infrastructure is mainly for the transportation of oil and gas products and for providing a link between upstream production and downstream dissemination. The downstream infrastructure focuses on the distributing of assets to consumers, mainly for crude oil and raw/condensed natural gas (Stergiopoulos et al., 2020). As published by Forbes, "We are experiencing a technological revolution, any tech buzzword be it artificial intelligence, big data, IoT with application to many different industries. Automatization and integration expose industries to new vulnerabilities and threats, and the oil and gas industry is no exception" (Forbes, 2017).

With the exploitation of new cost-effective operational concepts, the use of digital technologies and increased dependencies on cyber structures, the oil and gas industry is exposed to new sets of vulnerabilities and threats. A Norwegian company DNV GL writes in an article identifying the biggest cyber security threats to the oil and gas industry. According to the company, cyber-attacks have grown in stature and

sophistication, making them more difficult to detect and defend against, and costing companies increasing sums of money to recover from (Offshore Energy Today, 2015). Like all other sectors of the economy, the Oil and Gas industry has been impacted by the continuous digital growth. Industrial Control Systems (ICS) used to operate in isolation, without bridging over information Technology (IT) infrastructures. Industry 4.0 enabled the integration of multiple industrial technologies in ICT, the engineers can now be able to remotely monitor operations as well as maintain Supervisory Control and Data Acquisition (SCADA) systems in real time. This digital revolution has exposed the once air-gapped Operational Technology (OT) infrastructures to multiple new attack surfaces and attack vectors (Stergiopoulos et al., 2020)(Ahakonye et al., 2023a).

As described by Williams et al., "Our increasing dependence on technology and web-based communication has opened the door for cybersecurity threat, particularly in the oil and gas industry. Petroleum companies face significant threats, such as hydrocarbon installation terrorism, which can cause plant shutdowns resulting from sabotage and interruption of utilities. With the oil and gas sector fueling every aspect of our daily life, the protection of this particular critical infrastructure has never been more crucial. We cannot afford to underestimate the consequences of attacks on the operations and systems that power our lifestyle" (Williams et al., 2015) .

ICS operational technology networks can be penetrated by malicious cyber-attackers. Even though there are firewalls, demilitarized zones, data diodes are security measures to isolate ICS operational networks, it cannot be assumed to stop all penetrations. Even air gapped operational technology networks can be penetrated. Hackers can use infected thumb drives, infected software updates, insider attacks and spear phishing attacks to penetrate heavily isolated and air gapped operational technology networks. The Stuxnet malware is a famous example of a worm which penetrated an air gapped network by exploiting a USB thumb drive autorun vulnerability (Alves et al., 2018).

The consequences of cybersecurity threats to the oil and gas industry as enumerated by Parsons includes but not limited to the following:

a. Plant Sabotage/Shutdown
b. Utilities interruption
c. Production disruption
d. Hydrocarbon Installation terrorism
e. Facility Terrorism
f. Undetected Spills (Williams et al., 2015).

Parsons states that 96% of all security incidents fall into nine basic patterns:

a. Point-of-sale intrusions
b. Crimeware
c. Cyber Espionage
d. Insider misuse
e. Web App attacks
f. Miscellaneous errors
g. Physical theft/loss
h. Payment card skimmers
i. Denial of service (Williams et al., 2015).

As published by Offshore Energy Today, a Norwegian company DNV GL - Oil & Gas says that while the study focused on operations on the Norwegian Continental Shelf, the issues are equally applicable to oil and gas operations anywhere in the world. DNV GL enumerated that the top ten cyber security vulnerabilities include:

a. Lack of cyber security awareness and training among employees
b. Remote work during operations and maintenance
c. Using standard IT product with known vulnerabilities in the production environment
d. A limited cyber security culture among vendors, suppliers and contractors

e. Insufficient separation of data networks
f. The use of mobile devices and storage units including smartphones
g. Data networks between on and offshore facilities
h. Insufficient physical security of data rooms, cabinets and so on
i. Vulnerable software
j. Outdated and ageing control systems in facilities (Offshore Energy Today, 2015).

In 2017, Siemens sponsored Ponemon Institute LLC to conduct independent research titled "The state of cybersecurity in the oil & gas industry: United States" with the purpose of understanding how companies in the oil and gas industry are addressing cybersecurity risks in the Operational Technology (OT) environment. According to the Ponemon report "the deployment of cybersecurity measures in the industry is not keeping pace with the growth of digitization in oil and gas operations. 35% of the respondents to rated their organization's OT cyber readiness as high. With most respondents describing their organization as having low to medium cybersecurity readiness, 68% of respondents say their operations have had at least one security compromise in the past year, resulting in the loss of confidential information or OT disruption (Ponemon Institute, 2017).

According to Forbes, "The possible consequences of cyber-attacks highly depend on the malefactors and their aims. For example, competitors or state-backed hackers are interested in revealing sensitive information. Sabotage, on the other hand, is likely to come from hacktivists (Forbes, 2017).

Kaspersky Industrial Control Systems (ICS) Security Assessment, identified various vulnerabilities leading to obtaining unauthorized access to critical network components which include:

a. Insufficient physical protection of ICS equipment.
b. Vulnerable network architecture, insufficient network protection (including flaws in separation of the ICS network from other networks).

c. Vulnerabilities leading to network traffic interception and redirection (including ones in industrial communication protocols).
d. Vulnerabilities in ICS components, such as SCADA, PLCs, smart meters and so on.
e. Insufficient authentication and authorization in various services.
f. Weak user credentials.
g. Configuration flaws, including excessive user privileges, as well ass non-compliance with security standards and vendors recommendations.
h. Vulnerabilities in communications between the analyzed ICS and other systems.
i. Vulnerabilities caused by errors in applications code (code injections).
j. Vulnerabilities caused by using outdated hardware and software versions without the latest security updates.
k. Information disclosure (Kaspersky Lab, 2019).

2.1.6 Cyber Security Incidents Review

Cybercrimes cost the oil and gas industry, energy and other utility companies an average of $13.2 million each year for lost business opportunities and damaged equipment, higher than in any other industry as documented in Ponemon's survey of 257 businesses. A historical review of cyber-attacks in the oil and gas industry will be useful in analyzing how this threat has evolved over the years (Ponemon Institute, 2017).

December 2002, Venezuela's state oil company became embroiled in a bitter workers industrial action. At the same time of the strike action, there were cases of computer hacking and attacks which caused a significant damage to their operations as most of these operations are controlled centrally by computers. It was alleged that someone, possible an employee who is involved in the general workers strike, remotely accessed a program terminal to erase all PLC programs in port facility. This action which is

termed a sabotage cut Venezuela's national production down to 370,000 barrels per day, compared with 3 million barrels before the industrial action (Polyakov, 2019).

In 2008, malicious hackers interfered with alarms and communications for Baku-Tbilisi-Ceyhan pipeline in Turkey, super-pressurizing crude oil to cause an explosion that resulted in the spilling of more than 30,000 barrels of oil (Williams et al., 2015).

23 October 2009, an explosion happened in Bayamon, Puerto Rico. The fire blazed for three days, forcing residents to flee their homes, investigators termed it a system glitch in the facility's computerized monitoring system. While a storage tank was getting refilled with gasoline from a fuel ship docked along the San Juan harbor. Since the tank's meter malfunctioned, the petrol kept overflowing until it met an ignition source (Polyakov, 2019).

In 2010, STUXNET shocked the industry as the first computer worm to attack SCADA systems. It was used to hijack industrial control systems around the globe, including computers used to manage oil refineries, gas pipelines, and power plants. Although Stuxnet was not designed for Oil and Gas, it seriously affected these companies as well (Williams et al., 2015)(Polyakov, 2019). Stuxnet, a computer worm targeting industrial programmable logic controllers (PLCs) and SCADA systems, was a wake-up call to every industry. Despite the fact that it was not specifically designed to attack the petroleum industry, several oil and gas companies were infected with the virus (Forbes, 2017)(Virvilis & Gritzalis, 2013a).

Stuxnet virus, a malicious and sophisticated computer program that is built to cleave and create independent control remote systems. It is an example of Advanced Persistent Threats (APTs) which does not require its target to connect to the internet, however it can access and establish control over the target using some secondary devices such as thumb drives. In the Iran's nuclear plant attack, Stuxnet hacked windows operating system using four zero-day attacks which could have been spread from USB drives that has shortcut files to start executable codes. It has the ability to reprogram its target once it successfully penetrates and exploits other computers that are connected (peer to peer

or in a private network) to the infected one to harm them. Stuxnet can also update itself using two methods first by learning the peer to peer communication of the violated computer , second method is by trying to connect the violated computer with command and control servers to give them a stealing data and download arbitrary executables (Al-rabiaah, 2018).

Other cybersecurity incidence involving SCADA systems include the following:

a. January 2003, the Accident of "Nuclear power plant" where the SQL slammer worm closed the safety control system in Davis Besse nuclear power plant Ohio USA.
b. The March 2000 "Maroochy Wastewater infrastructure" attack which happened in Queensland Australia resulting in a leak of around 212000 gallons of raw sewage. The communications sent by radio links to wastewater pumping stations were lost, the pumps were not working properly, the alarms put in place to alert the staff of faults were not going off. Initially, the thought was, it is a problem with the new system, but later the engineer monitoring the communication signals discovered that someone was hacking into the system and deliberately causing the problems. The perpetrator of the attack was an insider who spread unauthorized commands of radio to the SCADA system (Alhaidari & Al-Dahasi, 2019) (Ramotsoela et al., 2019).

Ramotsoela et al explained that a disgruntled contractor who was turned down for a position by the municipality, retaliated by launching a devastating attack on the facility using insider knowledge he gained while carrying out system upgrades. The existing preventative measures alone were not adequate to protect the critical infrastructure, so a second layer of security is required (Ramotsoela et al., 2019).

Cybersecurity & Infrastructure Security Agency (CISA) in 2020 issued a Ransomware Impacting Pipeline Operations Alert (AA20-049A) when a ransomware attack affected "the control and communication assets on the Operational Technology (OT) network of a natural gas compression facility. A cyber threat actor used a spear phishing link to

obtain initial access to the organizations information technology (IT) network before pivoting to its OT network. The threat actor then deployed commodity ransomware to encrypt data for impact on both networks. Specific assets experienced a Loss of Availability on the OT network which included Human Machine Interface (HMIs), data historians, and polling servers. The impacted assets were no longer able to read and aggregate real-time operational data reported from the low-level OT devices, resulting in partial Loss of View for human operators. The attack did not impact any programmable logic controllers (PLCs) and at no point did the victim lose control of operations (CISA, 2020).

2.1.7 Advanced Persistent Threats (APTs)

Evolution of technology yielded a highly sophisticated threats called Advanced Persistent Threats which focuses majorly on the critical industrial sector. APTs when they gain unauthorized access to a computer network, have the capability to remain undetected for an extended period and usually do not need the internet for spreading. Stuxnet in 2010 targeted Iran's nuclear program by exploiting four zero-day vulnerabilities in Windows Operating systems.

Recently, a number of different APTs have emerged which increased researcher's attention to analyze them in order to find solutions to protect against existing and future APTs. Al-Rabiaah in his research explained that recent variances of Stuxnet are Duqu, Flame, Shamoon, and Triton (Al-rabiaah, 2018).

Nikos and Dimitris explained that as the complexity and the number of cyber-attacks continuously increase, it is becoming evident that current security mechanisms have limited success in detecting sophisticated threats. Stuxnet, Duqu, Flame and red October have troubled the security community due to their severe complexity and their ability to evade detection – in some cases for several years (Virvilis & Gritzalis, 2013b).

In September 2011, DUQU which is a derivative of Stuxnet was discovered by CrySyS Lab. It is a remote access Trojan (RAT) specialized for cyber espionage that steals data

from computers it infects and targets industrial equipment manufacturers, illegally collecting information about the manufacturers system and other proprietary data. The name Duqu was derived from the fact that its key logger creates temporary files which normally starts with "~DQ…". It has lots of similarities to Stuxnet which includes its modular structure, injection mechanism and a driver that has a fraudulent digital signature on it (Bencsath et al., 2011).

The discoverers of Duqu believed that the Duqu creator is the author of Stuxnet or knew about how Stuxnet was implemented. The target of Duqu was to steal information in order to ease establishing a new harmer attack for different industries. It targeted emails with a Microsoft Word document by exploiting zero-day. Its features includes that it shared some codes with Stuxnet but has no ICS specific attack, it was only designed as a key logger to gather information in preparation for future attack. It used valid digital certificates and deleted itself after 36days. It used communication: HTTP and HTTPS with several proxies on the command and control servers to forward traffics to port 80 and 443 to enable the attackers update Duqu executable files and gather information of the system by Duqu remote server in .jpg for misguiding communications of the network (Al-rabiaah, 2018).

In early 2015, Kaspersky Lab (a security company) detected a cyber intrusion in their Lab and this affected several of their internal systems. Investigation and technical analysis of the attacks showed that the new round of attacks is updated version of the Duqu 2011 which was believed to have been discontinued in 2012 hence the new malware and its associated platform was named DUQU 2.0. Other victims of Duqu 2.0 were found in the Middle East, Asia and Western Countries. In the case of Kaspersky attack, Duqu 2.0 took advantage of zero-day exploit which allowed an unprivileged domain user to elevate credentials to a domain administrator account which grants permission to infect other computers in the domain (Kaspersky Lab, 2015).

CrySyS Lab further investigated the Duqu 2.0 after the attack on Kaspersky in 2015 and they found out that the adversaries behind Duqu malware had come back and active

although they modified their tools to be undetected by old methods, they also strongly reused codes and ideas during their recent attacks. The following were the similarities they found:

a. Similar string decryption routines related to Anti-virus product strings
b. Similar methods, magic numbers, bug and file format related to files encrypted with AES by both threats
c. Same non-standard CBC mode AES encryption used by both threats
d. Extremely similar logging module with exactly the same magic numbers
e. Similar C++ like coding and compiling style (Bencsáth et al., 2015).

In May 2012, Flame a sophisticated attack that targeted cyber espionage of the Middle East was revealed. Flame also known as Flamer, sKyWIper and Skywiper has the capability of attacking computers running Microsoft Windows Operating Systems. It can be spread in two ways through: LAN or USB drives. It has the ability to record screenshots, keyboard activities, network traffics, audio and even skype conversations, it can turn an infected computer into Bluetooth beacons which attempts to download contact information from nearby Bluetooth -enabled devices. It saves the data collected in a local file which it later sends to a command-and-control servers scattered around the world along with locally stored documents. The program then awaits further instructions from these servers. Flame is bigger in size about 20MB and its target is sabotage and collecting data in wide range not to a specific industry. The attackers aim specific files for intelligence purposes such as PDFs, text files and AutoCAD (Al-rabiaah, 2018).

CrySyS Lab comparing sKyWIper or Flame to Duqu or Stuxnet shows that they have many differences and it seems sKyWIper was not made by the same developer team. They also believe that there is a possibility that the attackers hired multiple independent development teams for the same purpose as sKyWIper and Duqu are two different implementations developed for the same requirement specifications. This may be an

approach to increase the robustness of an operation, which can persist even if one of the two (or more) implementations is uncovered (CrySyS Lab, 2012).

In 2012, Shamoon Cyber-attack on Aramco aimed to stop oil and gas production in Saudi Arabia and prevent resource flow to international markets – 30,000 computers were damaged. A malware called Flame, which is spreadable, can record audio, screenshots, and user activities, was used for targeted cyber espionage in the Middle East. An unknown virus infected the computer systems at RasGas Ltd – a major liquefied natural gas exporter in Doha, Qatar (Williams et al., 2015).

Al-rabiaah explains that Shamoon was revealed in August 2012 and was used to attack oil and energy sector in the Middle East especially Saudi Aramco Company where it destroyed over 30,000 workstations. It spread via LAN by copying itself from infected computer to non-infected computers. The main operation of the Shamoon malware targeted Windows NT kernel 32 bit-based versions of Microsoft Windows, wiping, overwriting files and Master Boot Record (MBR) and denying access to the infected computers. Shamoon is less sophisticated than other APTs (Al-rabiaah, 2018).

Triton Malware is the latest and recent APT which came to fore in December 2017. It targeted Industrial Control Systems in Saudi Arabia Petrochemical plant. The malware had the capability to send specific commands to the safety systems controller - Triconex SIS (specific commands example halt). It exploited a vulnerability which is not yet declared in a machines that are running Microsoft Windows operating system (Al-rabiaah, 2018).

On May 7, 2021 the Darkside Ransomware attacked the Colonial Pipeline in the USA. The pipeline which is used in the transportation of gasoline and jet fuel was impacted which affected the computerized equipment managing the pipeline. Darkside actors deployed Darkside Ransomware against the Colonial Pipeline's IT network after accessing the pipeline company's IT network (CISA & FBI, 2021).

2.1.8 Process Control Network Overview

A Process Control Network (PCN) is a network composed of real-time Industrial Control Systems (ICS) which manage, monitor and control industrial infrastructure. PCNs make use of software, hardware, networks and their connectivity for accessing, controlling and transferring data with each other (Nitesh Malviya, 2019).

Typically, as shown in Figure 2.2, a PCN consists of the following: Human Machine Interface (HMI), Programmable Logic Controllers (PLC), Remote Terminal Units (RTU), Master Terminal Units (MTU) or engineering workstations (EWS), network devices like switches, firewalls and routers, process instrumentation which include sensors, transducers, actuators, transmitters. The network components are categorized from Level 0 to Level 3.5 for a PCN.

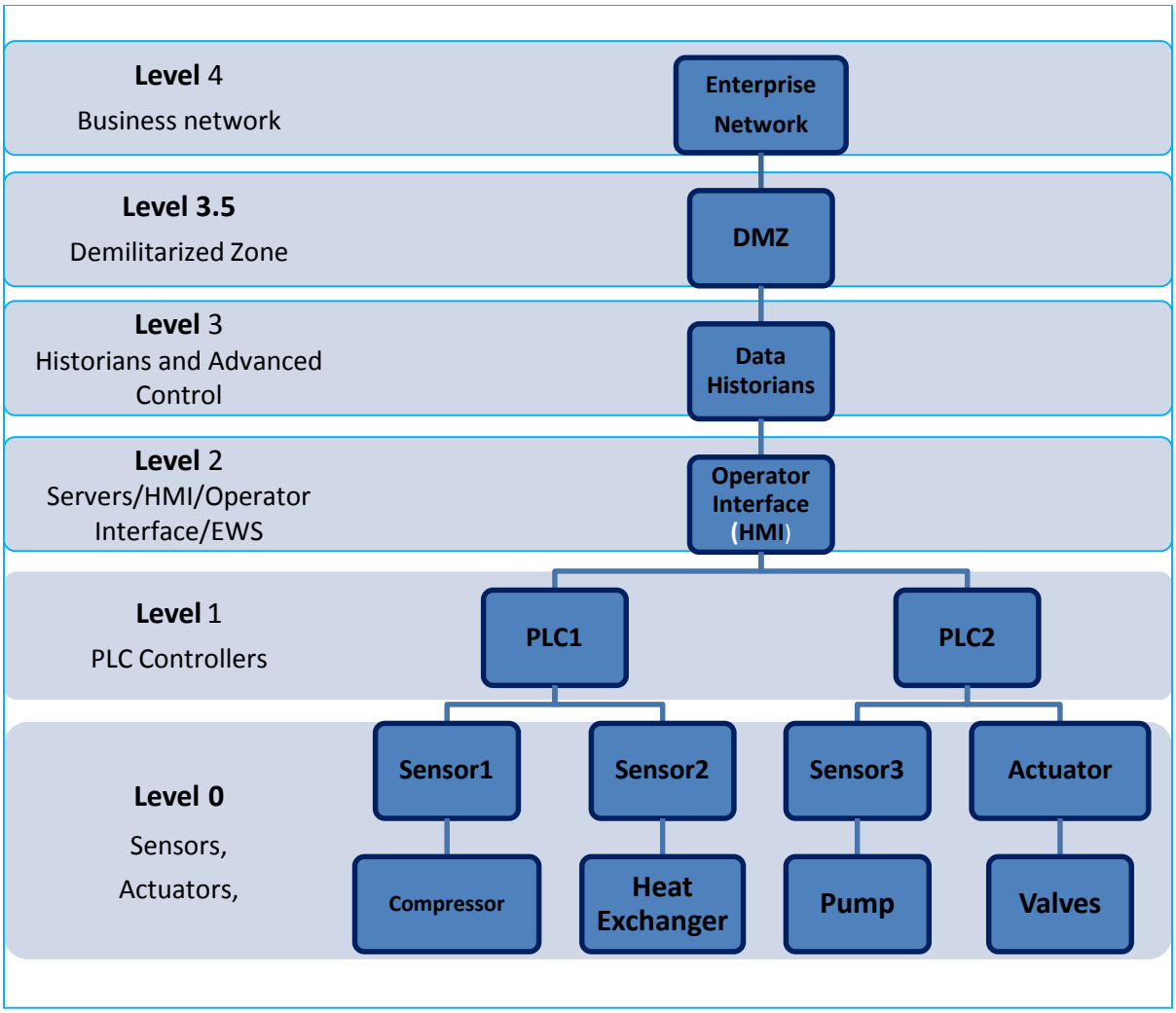


Figure 2.2 Process Control Network Hierarchy

Level 0 – Actual production process equipment, Sensors and actuators for the production process.

Level 1 – Controllers and real time control of the production process

Level 2 – Human Machine Interface (HMI), Supervisory Control, Engineering Work Station (EWS), Servers

Level 3 – Data Historians, Advanced Control

Level 3.5 – Demilitarized Zone (DMZ), Interface between PCN and Business Network

Level 4 – Business or Enterprise Network

Levels 0 to 3.5 are termed the Process Control Network which may consist of different technologies, topologies, protocols and communication mediums. The communication medium for levels 0 to 2 could be standard or proprietary depending on the integration systems and vendor equipment used. Level 3 and 3.5 utilizes standard open systems Ethernet Technology while Level 4 utilizes open systems Ethernet LAN technology.

PCN are generally referred to as Distributed Control Systems (DCS), or Supervisory Control and Data Acquisition (SCADA) systems. Data transfer on PCN can be achieved through different forms of communication medium such as:

i. Ethernet network
ii. Optical Fiber network
iii. Serial Connections
iv. Microwave Radio and Wireless connections
v. Satellite communications

Some of these communication mediums may be standard communication protocols while others may be Vendor Proprietary Networks. Choice of network topology and protocol used may be determined by so many factors like cost of project, choice of technology, infrastructure requirement, communication speed, network reliability, integration requirement, and so on.

2.1.9 Process Control Network of an Oil and Gas Installation

The Process Control Network of an Oil and Gas installation could be designed in different ways which is determined by lots of factors like cost, technology, integration process, environmental conditions, location of facility and so on, a typical example is as shown in Figure 2.3.

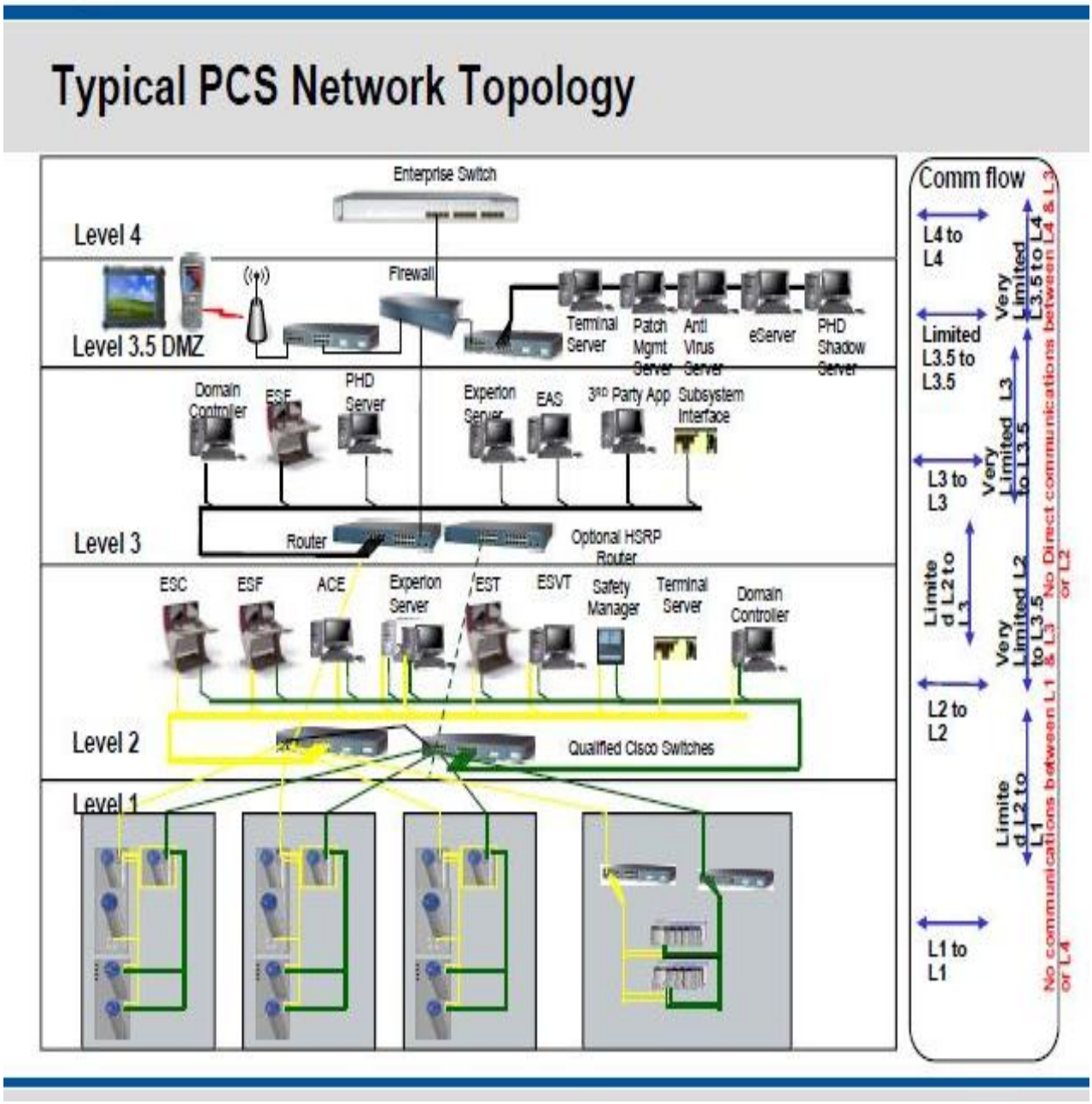


Figure 2.3 Typical Process Control Network Topology (Alston, 2020).

2.1.10 Cyber-Physical Systems

Lanotte, Merro, Muradore and Vigano in their work explained that Cyber-Physical Systems (CPS) are integrations of networking and distributed computing systems with physical processes that monitor and control entities in a physical environment, with

feedback loops where physical processes affect computations and vice versa. In real-time control systems, a hierarchy of sensors, actuators and control processing components are connected to control stations. They may include SCADA, PLC and DCS. Their work focused on integrity and DoS attacks to sensors and actuators of CPS (Lanotte et al., 2017).

Pasqualetti, Dorfler and Bullo in their work proposed a mathematical framework for cyber-physical systems, attacks and monitors. They designed a centralized and distributed attack detection and identification monitor. After a review of the different attack patterns that could happen to a CPS, they inferred that cyber-physical systems are prone to failures and attacks on their physical infrastructure and cyber-attacks on their data management and communication layer (Pasqualetti et al., 2013)(Pasqualetti et al., 2015). In a further work, the attack monitoring methods for cyber-physical systems were classified using model-based and data driven approaches. The data driven approach relies on the data generated in the form of measurements from sensors deployed on the physical system, the model based assumes knowledge of the model of the underlying system (Krishnan & Pasqualetti, 2021).

Model based monitoring methods contends with the difficulty of obtaining reliable model for the underlying system. This in practice is achieved by system identification, for which the available data must be adequately informative, this requirement is difficult to achieve for complex systems. It is not clear if the full system identification is necessary for attack monitoring. All these challenges has increased the popularity of data-driven approaches in monitoring (Krishnan & Pasqualetti, 2021). In recent times, CPS is adapted towards solving complex problems in the society, in the form of coordinated network control distributed systems with active feedback algorithm. Typical examples are smart cities, smart grid, cyber-driven factories and Industry 4.0. Okafor et al. explained in their work that data trustworthiness, reliability and availability is necessary for the actualization of CPS example smart cities with robust system architecture for secured high bandwidth systems and low-latency diffusion (Okafor et al., 2022)

2.1.11 Model of Cyber-Physical Systems Under Attack

Pasqualetti et al., in their work modelled cyber-physical systems under attack as linear time-invariant descriptor systems subject to unknown inputs. This simplified model neglected system nonlinearities and the presence of noise in the dynamics and the measurements. The model has proven useful in studying stability, faults, and attacks. Model of cyber-physical systems under attack is considered using the descriptor system as shown in Equations (2.1) and (2.2) (Pasqualetti et al., 2013)**.**

$$E\dot{x}(t) = Ax(t) + Bu(t) \qquad (2.1)$$

$$y(t) = Cx(t) + Du(t) \qquad (2.2)$$

Where

$x(t) \in \mathbf{R^n}$ *is the state vector*

$u(t) \in \mathbf{R^m}$ *is the input vector*

$y(t) \in \mathbf{R^p}$ *is the output vector*

$E \in \mathbf{R^{nxn}}$ *is the state matrix*

$A \in \mathbf{R^{nxn}}$ *is the state system matrix n* x *n*

$B \in \mathbf{R^{nxm}}$ *is the input matrix n* x *m*

$C \in \mathbf{R^{pxn}}$ *is the output matrix p* x *n*

$D \in \mathbf{R^{pxm}}$ *is the feedforward matrix p* x *m*

The matrix E is singular, while disturbances affecting the plant operation can be shown as the inputs *Bu* and *Du* which are unknown signals. Besides reflecting the genuine failure of system components, these disturbances model the effect of attacks against the cyber-physical system (Pasqualetti et al., 2013).

In their further work, Pasqualetti et al., explained that Deception Attacks refer to the possibility of compromising the integrity of control packets or measurements and are carried out by altering the behavior of sensors and actuators. Denial of Service attacks

compromise the availability of network resources, a typical example is jamming of communication signals. False data injection attacks are output attacks rendering an unstable mode of the system unobservable, and are specific deception attacks in the context of static estimators. Replay Attacks are carried out by hijacking the sensors, recording the readings for a certain time, and repeating such readings while injecting an exogenous signal into the system. Covert Attacks are closed-loop replay attacks, where the output attack is chosen to cancel out the effect on measurements of the state attack. They modelled the cyber physical system under attack as descriptor systems subject to unknown inputs altering the state and the measurements (Pasqualetti et al., 2013)(Pasqualetti et al., 2015).

Combita, Cardenas and Quijano in their work explored how conventional tools from Fault Detection, Isolation and reconfiguration (FDIR) can be used to solve integrity attacks on sensors of legacy Industrial Control Systems. The Fault Detection and Isolation (FDI) methods can be divided into model-based methods and data-based methods. Model-based methods can be quantitative or qualitative while quantitative methods include state estimation for deterministic systems and Kalman filters-based methods for stochastic systems. Their work was focused on the deterministic discrete-time systems using model-based fault detection and isolation methods. They emphasized that model-based methods need accurate model of the plant and the quality of the results is strongly linked with the accuracy of the model (Combita et al., 2017).

In their work they modelled the plant using deterministic discrete-time linear time invariant model to describe the behavior of the plant as:

$$x_{k+1} = Ax_k + Bu_k \qquad (2.3)$$

$$y_k = Cx_k \qquad (2.4)$$

Where

$k \in \mathbf{Z}^+$ *is the discrete time instant*

$x_k \in \mathbf{R^n}$ *is the state vector*

$u_k \in \mathbf{R^m}$ *is the control input vector*

$y_k \in \mathbf{R^p}$ *is the measurement output vector*

$A \in \mathbf{R^{nxn}}$ *is the state system matrix n* x *n*

$B \in \mathbf{R^{nxm}}$ *is the input matrix n* x *m*

$C \in \mathbf{R^{pxn}}$ *is the output matrix p* x *n*

Lanotte et al., in their work explained that the physical plant is supported by a communication network through which the sensor measurements and actuator data are exchanged with controllers and supervisors (example IDS) which are the cyber components also called logics of a CPS. The dynamic behavior of the physical plant of a CPS was represented by means of a discrete-time state-space model as shown in Equations (2.5) and (2.6) (Lanotte et al., 2017).

$$x_{k+1} = Ax_k + Bu_k + w_k \quad (2.5)$$

$$y_k = Cx_k + e_k \quad (2.6)$$

Where

$x_k \in \mathbf{R^n}$ *is the current (physical) state vector*

$u_k \in \mathbf{R^m}$ *is the input (the control actions implemented through actuators)*

$y_k \in \mathbf{R^p}$ *is the output (the measurement from the sensors)*

$w_k \in \mathbf{R^n}$ *is the uncertainty which represents perturbation*

$e_k \in \mathbf{R^p}$ *is the measurement error which represents sensor noise*

$A \in \mathbf{R^{nxn}}$ *is the state system matrix n* x *n*

$B \in \mathbf{R^{nxm}}$ *is the input matrix n* x *m*

$C \in \mathbf{R^{pxn}}$ *is the output matrix p* x *n*

$k \in \mathbf{N}$ *is the discrete time instant*

A, *B* and *C* are matrices modelling the dynamics of the physical system.

x_{k+1} depends on the current state x_k and the corresponding control actions u_k, at the sampling instant $k \in \mathbf{N}$. The state x_k cannot be directly observed: only its measurement y_k can be observed. In their contribution, they used a hybrid process calculus called Calculus of Cyber-Physical systems and Attacks (CCPSA) to specify Cyber Physical Systems and Cyber-physical attacks**.** The cyber-physical systems have two components: a physical component denoting the physical plant (also called environment) of the system, and containing information on state variables, actuators, sensors, evolution law and so on., and a cyber component (Lanotte et al., 2017).

Jithish and Sriram in their modelling of attacks and defenses developed an analytical model for Denial of Service (DoS) and Deception attacks in Network Controlled Systems (NCS). The components of NCS include sensors for data acquisition, controllers and actuators for control and a network for communication. The DoS attacks involve the adversary jamming the communication channels between network components thereby causing the data not to reach the destination or compromising subsystems to prevent data transmission. The deception attacks are characterized by the manipulation of data packets by gaining access to sensor and actuator channels, thereby causing the subsystem to receive false data that is perceived to be true. False data can include tampered measurement data, inaccurate time-stamps, forged sender identities and so on., (Jithish & Sankaran, 2018).

$$\hat{y}[k] = \begin{cases} y[k], k < \varepsilon \\ 0, k \geq \varepsilon \end{cases} \qquad (2.7)$$

$$\hat{y}[k] = \begin{cases} y[k], k < \zeta \\ \eta, k \geq \zeta \end{cases} \qquad (2.8)$$

Equation (2.7) was used to express DoS, where $\hat{y}[k]$represents the sensor output, $y[k]$ corresponds to the legitimate sensor value at a time instant k, ε is a constant that represents the attack time instant.

Equation (2.8) was used to express Deception attack, where $\hat{y}[k]$represents the sensor output, $y[k]$ corresponds to the legitimate sensor value at a time instant k, η is the

manipulated sensor value, ζ is a constant that represents the attack time instant (Jithish & Sankaran, 2018).

The physical plant was represented mathematically in discrete-time state space with the following Equations:

$$x_{k+1} = Ax_k + Bu_k \qquad (2.9)$$

$$y_k = Cx_k \qquad (2.10)$$

Where k = sample time, x_k = state vector, A = System matrix, B = Input matrix, C = output matrix. Equation (2.9) is called the state equation while Equation (2.10) is called the output equation (Jithish & Sankaran, 2018).

2.1.12 Machine Learning in Cyber Attack Detection

Machine learning is a branch of Artificial Intelligence (AI) focused on building applications that learn from data and improve their accuracy over time without being programmed to do so. Algorithms are trained to find patterns and features in massive amounts of data in order to make decisions and predictions based on new data. The better the algorithm, the more accurate the decisions and predictions will become as it processes more data (IBM Cloud Education, 2020a)(El Naqa & Murphy, 2015).

IBM with an age long history in machine learning defined machine learning as a branch of Artificial Intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy (IBM Cloud Education, 2020a).

There are several approaches for categorizing the cyber-attack prediction methods ranging from their use case to mathematical models as shown in figure 2.4.

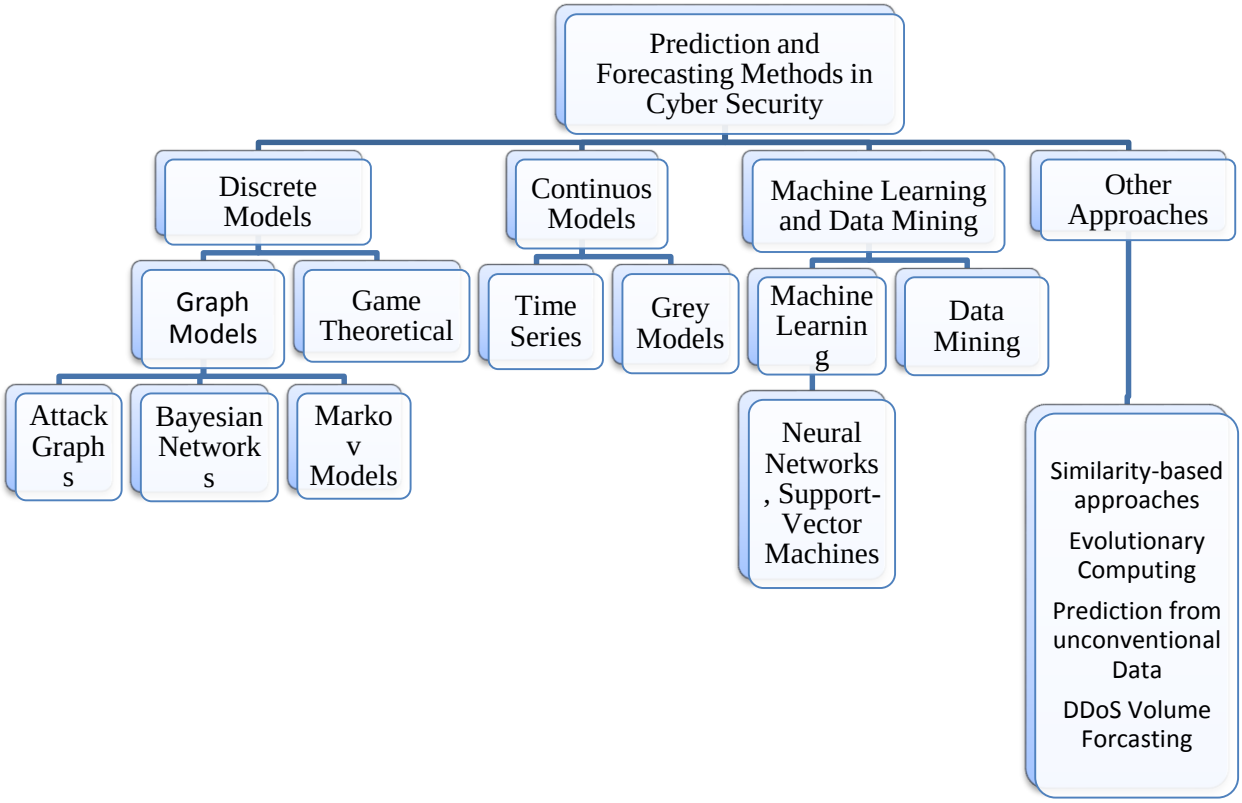


Figure 2.4 Prediction and Forecasting Methods in Cyber Security

The different forms of machine learning algorithms are as shown in Figure 2.5. The applications of machine learning ranges from simple everyday activities like web search, virtual assistance, email spam and malware filtering, image recognition, speech recognition, traffic recognition to more complex scenarios like medical diagnosis, autonomous cars (self-driving cars).

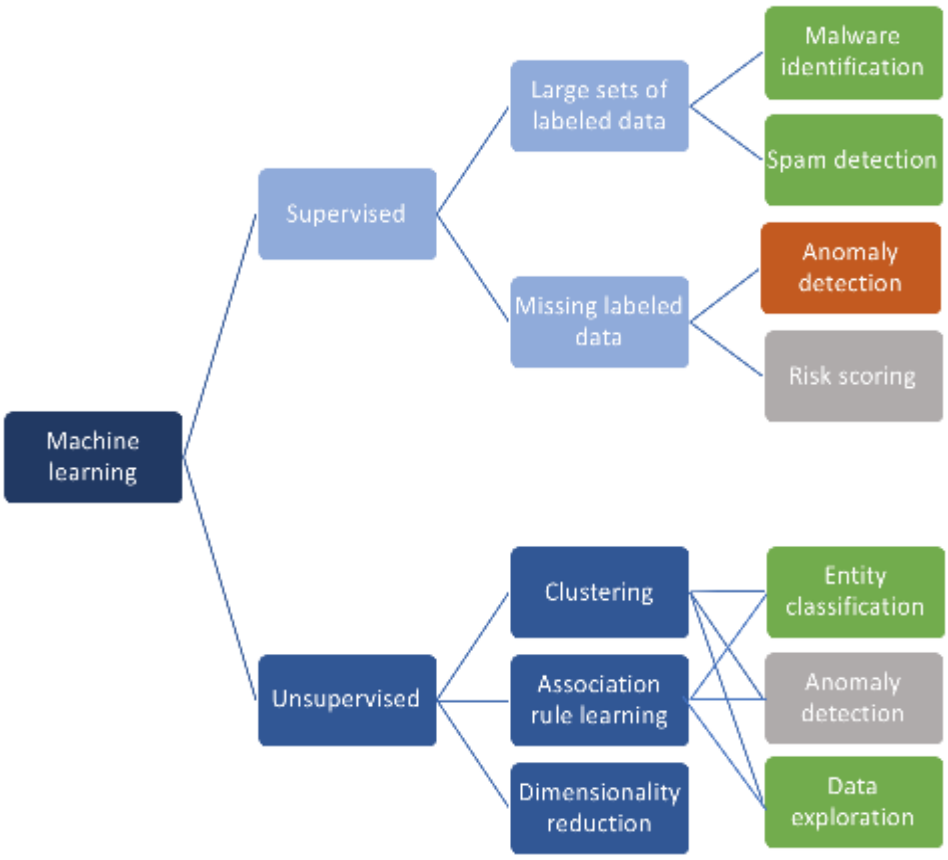


Figure 2.5 Machine Learning Algorithms

Irrespective of the machine learning method used, the working principle remains the same which can be summarized into:

i. Select and prepare the data set for training
ii. Choose an algorithm to train the dataset
iii. Train the algorithm to create a model
iv. Use the model, test, validate and improve the model

2.1.13 Machine Learning Methods

The machine learning methods can be categorized into:

i. Supervised Learning
ii. Unsupervised Learning
iii. Semi-supervised learning
iv. Reinforcement learning.

2.1.13.1 Supervised Machine Learning

This method trains itself on a labeled data set. The data is labeled with information that the machine learning model is being built to determine and that may even be classified in ways the model is supposed to classify the data. This requires less training data than other machine learning methods and makes training easier because, the results of the model can be compared to actual labeled results. Properly labelled data is expensive to prepare, and there is the danger of overfitting, or creating a model so closely tied and biased to the training data that it doesn't handle variations in new data accurately (IBM Cloud Education, 2020a). Supervised learning is defined by its use of labeled datasets to train algorithms that classify data or predict outcomes accurately. As input data is fed into the model, it adjusts its weight through reinforcement learning process, which ensures that the model has been fitted appropriately. The training dataset includes inputs and correct outputs, which allows the model to learn over time (IBM Cloud Education, 2020b).

Wakefield's explanation of supervised learning – the machine is taught by example. The operator provides the machine learning algorithm with a known dataset that includes inputs and outputs, and the algorithm must find a method to determine how to arrive at those inputs and outputs. While the operator knows the correct answers to the problem, the algorithm identifies patterns in data, learns from observations and makes predictions. The algorithm makes predictions and is corrected by the operator – and this

process continues until the algorithm achieves a high level of accuracy/performance. Supervised machine learning addresses classification, regression and forecasting issues (Katrina Wakefield, 2021).

2.1.13.2 Semi-Supervised Learning

This method offers a happy medium between supervised and unsupervised learning. During the training, it uses a smaller labelled data set to guide classification and feature extraction from a larger, unlabeled data set. Semi-supervised learning can solve the problem of having not enough labeled data (or not being able to afford to label enough data) to train a supervised learning algorithm (IBM Cloud Education, 2020a)(El Naqa & Murphy, 2015).

2.1.13.3 Unsupervised Machine Learning

This method trains using unlabeled data sets and uses algorithms to extract meaningful features needed to label, sort and classify the data in real-time, without human intervention. This is less about automating decisions and predictions and more about identifying patterns and relationships in data that humans would miss. An unsupervised learning algorithm can analyze huge volumes of emails and uncover the features and patterns that indicate spam (and keep getting better at flagging spam over time) (IBM Cloud Education, 2020a)(El Naqa & Murphy, 2015). Unsupervised machine learning algorithm discovers hidden patterns or data groupings without the need for human intervention. It provides an exploratory path to view data, allowing businesses to identify patterns in large volumes of data more quickly when compared to manual observation. Unsupervised learning models can comb through large amounts of data and discover a typical data point within a dataset. These anomalies can raise awareness around a faulty equipment, human error or breaches in security (IBM Cloud Education, 2020c)(Ndubuaku et al., 2019).

Unsupervised machine learning can be categorized into:

Clustering – Based on defined criteria, sets of similar data are grouped, segmenting data into several groups and performing analysis on each data set to find the existing patterns.

Dimension reduction – The number of variables are reduced to find the exact information required (Katrina Wakefield, 2021).

Algorithms for Machine Learning There are different types of algorithms with divers functionalities in Machine Learning.

For Labeled Data – Supervised Learning

i. Classification – Categorizes all the variables that form the output.
ii. Regression – The output variables are set as a real number
iii. Decision trees
iv. Instance-based algorithms

For Unlabeled Data – Unsupervised learning

i. Clustering algorithms
ii. Association algorithms
iii. Neural networks

Intrusion Detection – This is an effective method to identify network attacks, and it can give the alarms and defensive measures before or when suffering a great destruction (Shang et al., 2015).

Intrusion detection methods can be divided into two categories:

i. Misuse detection – Also known as knowledge-based detection or signature-based detection. It achieves intrusion detection by matching the known abnormal behaviors. Most enterprise IDSs adopt signature-based detection algorithms which search for predefined patterns (or signatures) in the monitored data in order to detect an ongoing attack which matches one or more signatures. Signature-based approaches usually score high detection capabilities and low false positive rates

against known attacks, but they are not effective when the behavior of attacks get slightly modified, calling for an update of signatures. In particular, they are not meant to detect zero-day attacks, which are novel attacks that cannot be matched to any known signature. When a zero-day attack that exploit newly added or undiscovered system vulnerabilities is identified, its signature needs to be devised and added as a new rule to the IDS (Zoppi, Ceccarelli, Capecchi, et al., 2021). Several IDS solutions exists but they cannot detect these unpattern attacks which may be in the form of DoS, DDoS, MitM or even zero-day attacks (Kulugh et al., 2022).

ii. **Anomaly detection** – Also known as behavior-based detection, identifies the abnormal behaviors through establishing the normal behavior model. Compared with the misuse detection, anomaly detection can detect the intrusion behaviors, which are not known before and it has a lower false negatives, but it has a higher false positives (Shang et al., 2015).

Anomaly Detectors are intended to find unconforming patterns in dataset which differs from the expected behavior of a system (or a network), these unconforming patterns are termed anomalies. It is assumed that ongoing attacks will generate observable anomalies which can be tracked while trending the performance indicators, or system features and it is based on this premise the anomaly-based IDS are built. Unsupervised algorithms are perfect fit for this type of activity as they can adapt themselves to suit the current context of the system and ultimately detect unknown attacks (Zoppi, Ceccarelli, Capecchi, et al., 2021).

2.1.13.4 Types of Anomaly Detection

There are four common classes of Machine Learning applications which include: prediction, anomaly detection, classification and structure discovery. Anomaly Detection may not be used when the training data is labelled or the ratio between the anomaly and the normal is 1:5 or the data is not autocorrelated (Srinath Perera, 2015).

Anomalies or outliers come in three types:

i. **Point Anomalies:** An individual data instance is considered as anomalous with respect to the rest of the data (example purchase with large transaction value).
ii. **Contextual Anomalies:** A data instance is anomalous in a specific context, but not otherwise (Anomaly if occur at a certain time or a certain region example. large spike at the middle of the night)
iii. **Collective Anomalies:** A collection of related data instances is anomalous with respect to the entire dataset, but not individual values. They have two variations:
    a. Events in unexpected order (ordered).
    b. Unexpected value combinations (unordered) (Srinath Perera, 2015).

Ramotsoela et al., in their work categorized intrusion detection systems into signature based intrusion detection which can be termed supervised learning and behavior based intrusion detection which can be termed unsupervised learning (Ramotsoela et al., 2019). Tufan et al. in their work explained that conventional, rule-based intrusion detection systems are usually fast, and can run on low computing resources. So, rule-based IDS is usually appropriate for an environment with many sessions per second in heavy network traffic (Tufan & Tezcan, 2021).

Pu et al in their work classified Intrusion Detection Systems (IDS) as Misuse (also called signature based) detection and anomaly (also called behavior based) detection. Misuse detection approaches are designed to detect attacks by using a database of predefined attack patterns. They are highly effective to detect known attacks and preferred for the low false positive rate, but they are unable to defend the system against unknown attacks, because such attacks do not exist in the predefined pattern list. As network attacks continue to increase in the frequency and diversity, maintaining an updated database is time consuming and unfeasible (Pu et al., 2021)(Teixeira et al., 2018).

The introduction of IDS in PCN could contribute to the reduction in the data processing speed of the network which delays the intercommunication between sensors and

controllers in real-time systems. Typically, antivirus, anti-malware and firewall functionalities can introduce further delay in network service delivery (Ahakonye et al., 2023b)(Nwakanma et al., 2022). Machine Learning approaches are comprehensively used to design Intrusion Detection Systems (IDS) as they have the ability to handle a massive quantity of data, and accordingly, network administrators can consider adequate measures to prevent the same. Feature selection is indispensable as it removes irrelevant/redundant features from the dataset so that the overall competence of the model can be improved (Abrar et al., 2020).

Zoppi et al. explained that differently from signature-based approaches, unsupervised anomaly detection algorithms first model the normal (expected) behavior of a system, then they use this knowledge to find patterns in data that do not conform to the model of such expected behavior: these patterns are called anomalies. Unsupervised algorithms infer patterns from a training set and discover the underlying structure of the data without reference to known outcomes (ie labels are unknown at training time) (Zoppi, Ceccarelli, & Bondavalli, 2021).

2.1.13.5 Reinforcement Learning

This is behavioral machine learning model that is similar to supervised learning, but the algorithm isn't trained using sample data. This model uses trial and error method to learn as it goes. A sequence of successful outcomes will be reinforced to develop the best recommendation or policy for a given problem (IBM Cloud Education, 2020a). Reinforcement learning focuses on regimented learning processes, where a machine learning algorithm is provided with a set of actions, parameters and end values. By defining the rules, the machine learning algorithm then tries to explore different options and possibilities, monitoring and evaluating each result to determine which one is optimal. Reinforcement learning teaches the machine trial and error. It learns from past experiences and begins to adapt its approach in response to the situation to achieve the best possible result (Katrina Wakefield, 2021).

2.1.13.6 Commonly Used Machine Learning Algorithms

There is no single machine learning algorithm that solves all the data problems. Different algorithms work for different situations and may be influenced by factors like, data structure, data quality, data availability, data features, data size, computational power and desired results. The appropriate algorithm must be selected for an identified problem. There exist several algorithms which can be used to solve data problems and they include:

i. **Logistic Regression –** Can be applied to classification problem – Supervised learning
ii. **Linear Regression –** Can be applied to Regression problem – Supervised learning
iii. **Support Vector Machine Algorithm –** Can be applied to Classification problem – Supervised learning
iv. **Decision Tree –** Can be applied to Classification or Regression problem – Supervised Learning
v. **Naïve Bayes –** Can be applied to Classification problems - Supervised learning
vi. **K-Means Clustering Algorithm –** Can be applied to Clustering problems – Unsupervised learning
vii. **Artificial Neural Networks –** Reinforcement Learning
viii. **K-Nearest Neighbor (kNN) –** Supervised Learning
ix. **Random Forest –** Can be applied to Classification or Regression problems – Supervised Learning
x. **Dimensionality Reduction Algorithms –** Reduces data with vast dimension to lesser dimension
xi. **Gradient Boosting Algorithms –** Ensemble of learning algorithms
xii. **Isolation Forest –** This is similar in principle to Random Forest and is built on the basis of decision tree algorithm. It identifies anomalies or outliers.

Elmrabit et al. in their comparison of twelve different machine learning algorithms summarized that there is a wide range of machine learning classification techniques which although they share the same objectives, encompass very different mathematical models, strengths and weaknesses. Each technique attempts to classify data in different circumstances. There are numerous ways to integrate the use of Artificial Intelligence with cyber security. Comparing the performance of various machine learning algorithms will lead to a better understanding and provide an excellent chance of implementing these technologies within industrial systems to reduce the risk of known and unknown attacks (Elmrabit et al., 2020).

Ahakonye et al, in their work on efficient classification of enciphered SCADA emphasized the research challenge and uncertainties establishing the type of machine learning needed for SCADA vulnerability detection. They carried out assessment of discrete machine learning algorithms effective for SCADA network attack detection by applying the most suitable and effective algorithms. The factors they considered in their research include algorithm performance, weight of algorithm which is based on computational complexity and accuracy, model training time and so on. (Ahakonye et al., 2021) (Nwakanma et al., 2022).

2.2 Review of Related Works

Virvilis and Gritzalis in their work did a technical comparison of the malware (focusing on behavior/dynamic analysis of the samples in a controlled environment) and identified the initial common attack patterns, in an effort to identify why current security solutions failed to detect the threats and propose realistic countermeasure for strengthening the defenses against similar threats (Virvilis & Gritzalis, 2013b).

Alhaidari and Al-Dahasi in their work provided a possible security solution via a simulation framework that utilizes the Machine Learning technique to detect Distributed Denial of Service (DDoS) attacks on SCADA systems. They used three Machine Learning algorithms J48, Naïve Bayes and Random Forest to determine attack

patterns. In their conclusion they recommended the use of SCADA system datasets in testing as well as using other models of Machine Learning based on different configuration parameters (Alhaidari & Al-Dahasi, 2019).

Wilson et al in their contributions to smart grid security showed that an unsupervised feature learning for automatic and adaptive attack detection in transmission SCADA systems could be used to improve detection accuracy with reduced reliance to system models and human expertise by using a stacked autoencoder (SAE) based deep learning framework. The outcome of their research captured rich patterns hidden in the data to identify attacks which demonstrates that it has the potential to provide adaptive and automatic threat monitoring in complex smart grid applications. (D. Wilson et al., 2018)

Al-Rabiaah in his work explains that the accelerated evolution of technology created opportunities for attackers who are sophisticated in technology and knowledge to break security which has yielded the Advanced Persistent Threat. He explained that Stuxnet which is a computer virus is an example of APTs and has the ability to destroy industrial systems including SCADA. His work showed that Stuxnet has a number of recent variances in the form of Duqu, Flame, Shamoon and Triton, pointing out the recent trends of information security attacks and their features towards predicting future attacks (Al-rabiaah, 2018).

In their survey of prediction and forecasting methods used in cyber security, Husak et al discussed four main points which includes attack projection, intention recognition, intrusion prediction and network security situation forecasting. Key pertinent questions were raised by their survey which are firstly, what can be predicted in a cyber security domain, second is how usable are the predictions in cyber security and lastly how to evaluate predictions in cyber security as well as what metrics should be used (Husák et al., 2019). Husak et al explained that Intrusion detection is mostly reactive and responds to specific attack patterns or observed anomalies. Taking a proactive approach in which there is a preemptive inference of an upcoming malicious activity which could react to such events before they can cause any harm. Research efforts and progress in

predictions and forecasting in cyber security are not as prominent as attack detection (Husák et al., 2019).

Ramotsoela et al., categorized intrusion detection systems into signature based and behavior-based intrusion detection. Signature based intrusion detection provides a fast way to detect known attacks based on specific patterns that can be attributed to those attacks. Behavior based intrusion detection, also known as anomaly detection, provides greater detection generality in that it can detect previously unknown attacks by analyzing sensed data or traffic patterns to determine which instances deviate from the norm. They noted that a disadvantage of signature-based method is that they are unable to detect unknown attacks. The research compared a number of popular anomaly detection schemes that are based on machine learning (Ramotsoela et al., 2019).

Stergiopoulos et al in their cyber-attacks incident assessment on the Oil and Gas sector explained that numerous publications exist on cyber-physical attacks and defenses, which cover numerous critical infrastructure sectors like the Energy, Health and Telecommunications sector. There has been no systematic approach to catalog, map and classify cybersecurity attacks on the Oil and Gas (O&G) sector. Modern history has already proven that Oil and Gas OT infrastructure is vulnerable against cyberattacks (Stergiopoulos et al., 2020).

Abrar et al in their work observed that lately, various noteworthy methods were proposed to offer a solution to intrusion detection, yet it remains a challenge to build a secure system as the attackers subsequently change their behavior in order to dodge the security mechanism of the system. They used machine learning classifiers as the effectiveness of an Intrusion Detection Systems (IDS) is determined by dimensions of the data, therefore, data was pre-processed to remove unrelated attributes from the dataset (Abrar et al., 2020).

Joloudari et al in their work for the early detection of advanced persistent threat attack using performance analysis of deep learning identified that there have been existing methods for the detection of APT attacks, there is still need for a system that can

perform timely detection when attacks occur in real-time. None of these methods provide a system model capable of detecting a new attack pattern, high generalizability and high flexibility. In their work they used deep learning on the NSL-KDD dataset with result showing that deep learning methods reduces the identified weaknesses in the existing methods in detecting APT attacks by providing high detection accuracy as well as automatic extraction of the main features of the attack(Joloudari et al., 2020).

Al-Abassi et al in their work identified that most intrusion detection systems on ICS are based on traditional IDS, which are majorly designed for IT security and do not consider the imbalance nature of ICS datasets, thereby suffering from low accuracy and high false-positive when they are applied. In their work they used a deep learning model to construct new balanced representations of the unbalanced datasets which are fed into an ensemble deep learning attack detection model specifically designed for an ICS environment (Al-Abassi et al., 2020).

Pu et al in their work explained that as the number and complexity of new attacks continue to rise, hence the necessity for effective and intelligent solutions. Unsupervised machine learning techniques are particularly appealing to intrusion detection systems (IDS) since they can detect known and unknown types of attacks as well as zero-day attacks. In their work they focused on anomaly detection (also known as behavior based) which uses the normal system activity to build normal operation profiles, identifying anomalies as behavior that deviate from the normal ones. The aim of clustering algorithm is separate the given the given unlabeled data into clusters that achieve high inner similarity and outer dissimilarity, without relying on signatures, explicit description of attack classes, or labeled data for training (Pu et al., 2021).

Zoppi et al, in their work presented a quantitative comparison of 17 unsupervised anomaly detection algorithms applied on a multiple attack dataset. They adopted unsupervised anomaly detection algorithms since they are known to be the most suitable way to deal with zero-day attacks (Zoppi, Ceccarelli, Capecchi, et al., 2021). On the selection of the appropriate anomaly detectors, Zoppi et al, emphasized the

assumption that an attack generates observable deviations from the expected behavior and they aim at finding patterns in data that do not conform to the expected behavior of a system. Once an expected behavior is defined, anomaly detectors target deviations from such expectations, protecting against known attacks, zero-day attacks, emerging threats and enhancing existing algorithms. They identified which anomalies are generated when attacks occur and this help to determine which algorithms are more suited to detect specific anomaly classes (Zoppi et al., 2020).

Rosa et al, in their work explained that SCADA security is a large umbrella for different specific problems. Anomalies such as network-based cyber-attacks or physical faults could be further categorized by looking into physical process properties, network SCADA communications or a combination of both. They also explained that it is very unlikely that a single approach of anomaly detection would outperform all the others for all types of anomalies. A typical example is the recent APTs campaigns targeting Industrial Automation and Control Systems (IACS), they typically start by collecting information from the environment over long periods of time undetected (Rosa et al., 2021).

Bierbrauer et al. in their work opined that machine learning techniques for tackling anomaly detection problem in cybersecurity have included semi-supervised methods, deep learning models and graph-based approaches. Two additional unsupervised machine learning methods for anomaly detection are relevant. Isolation Forest exploit a decision tree-like architecture to uncover anomalies, assuming that those anomalies would be most easily split for the rest of the data. In their work they used UNSWB-NB15 data set synthesized and contains 2.5 million observations of labelled packets. They experimented with unsupervised learning Isolation forest and local outlier factor and concluded that both were not able to meet the expectation (Bierbrauer et al., 2021).

Sarker in his work carried out a comprehensive review of several machine learning algorithms that can be applied to enhance the intelligence and the capabilities of an application. He explained the principles of different machine learning techniques and

their applicability in various real-world application domains, such as cybersecurity systems, smart cities, healthcare, e-commerce, agriculture and many more. He also highlighted the challenges and potential research directions based on his study. After a robust review of the different machine learning capabilities, he concluded that, the effectiveness and efficiency of a machine learning-based solution depend on the nature and characteristics of the data, and the performance of the learning algorithm. The application of machine learning is not straightforward, although the current cyberspace enables the production of huge amount of data with very high frequency hence the need to collect useful data for the target machine learning-based applications. Hybrid learning models , example the ensemble of methods which may entail modifying or enhancement of existing techniques or designing new learning methods could yield better results in the future (Sarker, 2021).

Shetty in his vulnerability assessment for cybersecurity using machine learning explained that Dataset is a significant part of machine learning as this provides the data to the algorithm for analyzing and training. The first thing to do before applying any algorithm is to analyze and understand the dataset. He opined that network data cannot be used in raw form rather it has to be pre-processed. He summarized by no single algorithm can be adjudged the best as the algorithms performed better than each other in different aspects of security and incidents, hence the need to tailor the algorithms according to the specific needs (Shetty, 2021).

Zoppi et al. in their review of unsupervised algorithms to detect Zero-Day attacks explained that many Intrusion Detection systems (IDS) show deficiencies in identifying novel zero-day attacks. These attacks exploit new vulnerabilities or known vulnerabilities in novel and different ways and cannot be matched against known signatures. Complexity and dynamicity of systems are rapidly increasing, to the extent that a correct and complete characterization of all possible security threats becomes almost impossible. The growth of hacking activities often exposes systems to brand new attacks and the likelihood of being targeted by zero-day attacks is getting higher

and higher. Regardless of their characteristics, attacks should be timely identified to block an ongoing attack or protect critical assets, unfortunately most IDS cannot effectively deal with zero-day attacks (Zoppi, Ceccarelli, & Bondavalli, 2021).

Ahakonye et al. in their work explained the machine learning model design displaying the phases of the machine learning technique evaluation for SCADA attack classification. In their work, they emphasized the fact that for critical industrial systems, accuracy alone is not the ideal measure to assess the capability of a model; other metrics such as false alarm rate, training time are needed for comparison. Their work was subdivided into 3 main phases: training, testing and model selection phase, as shown in figure 2.6. During the training and testing phase, the various datasets used in the study is split into training and testing sets and imported into the machine learning algorithms after implementing a 5-fold cross validation. The testing set was used to validate the performance of the training set (Ahakonye et al., 2021).

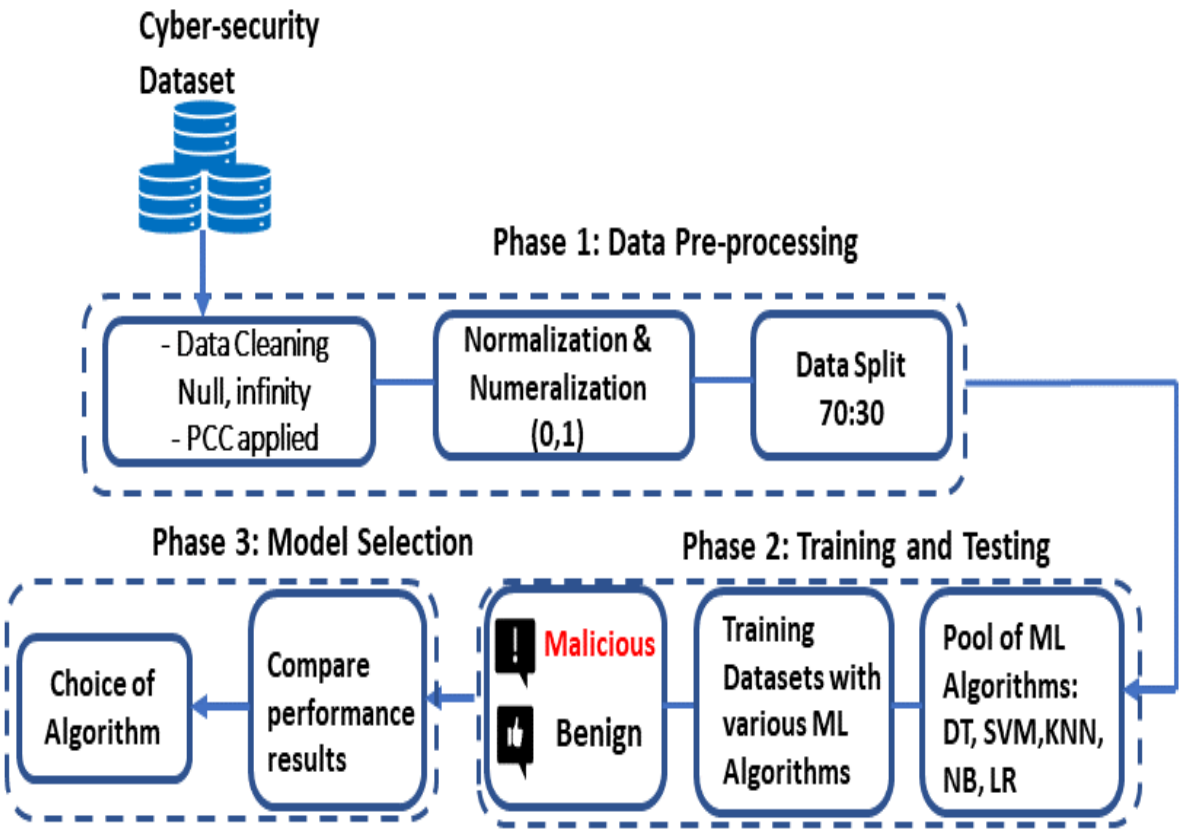


Figure 2.6 Model design displaying the phases of machine learning technique evaluation for SCADA attack classification (Ahakonye et al., 2021).

2.3 Research Gap

This section identifies the limitations of the related works reviewed in section 2.9 which focuses more on the shortcomings this study hopes to address.

Pu et al in their work identified that the number and complexity of new attacks continues to rise, therefore effective and intelligent solutions are necessary. Unsupervised machine learning technique was used in their work for intrusion detection systems since they can detect known and unknown types of attack as well as zero-day attacks. Their work was not able to develop an effective feature selection method from the known public NSL-KDD network attacks datasets used (Pu et al., 2021). The identified gaps include:

i. The known public data NSL-KDD used is not a perfect representation of existing real networks and there is need for data optimization.
ii. Identical and repeated datasets which affect the learning ability of the algorithm and the final output of the algorithm.
iii. Unsupervised Learning anomaly detection approach has the ability to generate lots of false positives which will require human investigation to avoid denial of real network traffics.

Abrar et al in their work on machine learning approach using different algorithms to solve intrusion detection problems, focused on labelling the data as normal or intrusive (supervised learning) and data optimization (Abrar et al., 2020). The following drawbacks were identified:

i. Model may not detect zero-day attacks or emerging threats or unknown attacks.
ii. Limited dataset is publicly accessible which impacted the outcome of the work.
iii. Four feature subsets extracted from the available dataset may not be detailed enough.
iv. The available dataset lacked comprehensiveness and completeness.
v. Inadequate computational power to run complex algorithms

Al-Abassi et al in their work, used a generalized ensemble deep learning-based cyber-attack detection method specifically designed for ICS, constructed new balanced representations from the raw imbalanced dataset which was used in an ensemble deep learning algorithm. (Al-Abassi et al., 2020). The identified gaps include:

i. The need for more real time ICS data samples with characteristic features.
ii. The need to extract more important features from the dataset by identifying the attack types as well as attack location which cannot be derived from the public datasets used.

iii. The need to distinguish between system downtime and actual real time attacks.

Joloudari et al in their work for the early detection of advanced persistent threat attack using performance analysis of deep learning explained that APT attack is one of the most stable and persistent attacks on a system and resides in the system for a long time, even though it is important to detect it early. The artificial intelligence methods was needed for the timely detection of the APT attacks (Joloudari et al., 2020). The identified gaps include:

i. The known public data NSL-KDD used is not a perfect representation of existing real networks and there is need for data optimization.
ii. Could not extract the important features of the dataset as the dataset is not a perfect representation of real network.
iii. The need to subject the same or similar ICS data samples to unsupervised learning methods to simulate a real time network attack detection.

Zoppi et al, in their work observed that unsupervised algorithms such as Isolation Forest, One-Class Support Vector Machines and Self-Organizing Maps show less misclassifications than algorithms belonging to other families. Clustering algorithms which represented a valid alternative due to a good tradeoff between misclassifications and low computational complexity needed for training (Zoppi, Ceccarelli, Capecchi, et al., 2021)(Zoppi et al., 2020). The identified gaps include:

i. Higher number of misclassifications with respect to tricky unknown attacks.
ii. High computational complexity required for some algorithms which may be expensive.
iii. Identification of the different attack detection capabilities of algorithms.

Rosa et al, in their work investigated the possibility of providing a robust and flexible solution to anomaly detection in a complex SCADA system. The framework presented will enable the integration of different techniques and algorithms, rather than being a

replacement for existing domain specific detection tools (Rosa et al., 2021).The identified gaps include:

i. The complexity of managing such an integrated and automated platform with minimal human or operator intervention.
ii. Integration of such a complex system to an existing and operating SCADA installation will require a robust change management process which if not properly managed may lead to a prolonged downtime.
iii. In Oil and Gas production, there is need to prioritize availability of network resources but the complex nature of the design may lead to prolonged troubleshooting when the need arises.
iv. Unavailability of Realtime dataset with all the characteristic features of the different network protocols and possible attack models simulations.

Sarker in his detailed review of different machine learning algorithms concluded that a more in-depth investigation of data collection methods is needed while working on the real-world data. Historical data may contain many ambiguous values, missing values, outliers, and meaningless data. The machine learning algorithm efficiency depends on the data quality, and availability for training and consequently on the resultant model. If the data are bad to learn such as non-representative, poor quality, irrelevant features, or insufficient quantity for training, then the machine learning model may become useless or will produce lower accuracy (Sarker, 2021). The identified gaps include:

i. Data quality issues – to accurately clean and pre-process the diverse data collected from diverse sources is a challenge.
ii. Selecting a proper learning algorithm that is suitable for the target application is a challenge. The outcome of different learning algorithms may vary depending on the data characteristics.
iii. Selecting a wrong learning algorithm would result in producing unexpected outcomes that may lead to loss of effort and impact the accuracy and effectiveness of the model.

Zoppi et al. in their work on zero-day attacks explained that IDS that rely on supervised machine learning algorithms require historical system observations for which a label is known. Unsupervised anomaly detection algorithms do not assume any knowledge on the attacks, they model the expected (normal) behavior of the system and classify any deviation from the normal behavior as anomaly (a suspect activity, possibly an attack). Unsupervised algorithms are meant to synergize with supervised machine learning algorithms but care must be taken as misleading algorithms may let their combination lean towards misclassification. Minimizing the misclassification of unsupervised anomaly detection algorithms is highly desirable and is a key challenge (Zoppi, Ceccarelli, & Bondavalli, 2021).

2.3.1 Summary of Research Gap

This section summarizes the identified gaps from the literature reviewed:

Table 2.3 Summary of identified research gaps

S/No.	Author/Year/Title	Results	Limitations
1.	Pu et al (2021) A Hybrid Unsupervised Clustering-Based Anomaly Detection Method (Pu et al., 2021).	Unsupervised anomaly detection based on clustering technique to minimize false positives.	i. Identical and repeated NSL-KDD datasets which affect the learning ability of the algorithm and the final output. ii. Possibility of generating lots of False Positives which could deny real network traffics
2.	Zoppi et al (2021) Unsupervised Anomaly Detectors to Detect Intrusion in the current threat landscape	Quantitative comparison of 17 Unsupervised anomaly detection algorithms.	i. High rate of misclassification of unknown attacks ii. High computational complexity,

	(Zoppi, Ceccarelli, Capecchi, et al., 2021).		iii. Issues with Different attack detection capabilities
3.	Rosa et al (2021) Intrusion and Anomaly Detection for the next-generation of industrial automation and control systems (Rosa et al., 2021).	Integration of different techniques and algorithms for monitoring of networks.	i. Integration of the complex solution will require prolonged network downtime.
4.	Abrar et al (2020) A Machine Learning Approach for Intrusion Detection System on NSL-KDD Dataset (Abrar et al., 2020).	Machine learning approach using different algorithms to solve intrusion detection problems (supervised learning).	i. Model could not detect zero-day attacks. ii. Not enough details in the public dataset. iii. Inadequate computational power to run complex programs
5.	Joloudari et al (2020) Early detection of the Advanced Persistent Threat attack using performance analysis of deep learning (Joloudari et al., 2020).	Deep Learning and Decision Tree models used to detect APT attacks. Improved detection by improved training and testing.	i. Could not extract important features from the NSL-KDD dataset. Dataset not a perfect representation of real network
6.	Al-Abassi (2020) An ensemble deep learning-based cyber-attack detection in industrial control system (Al-Abassi et al., 2020).	Used a generalized ensemble deep learning-based cyber-attack detection method specifically designed for ICS	i. Could not extract important features from the dataset. ii. Could not distinguish between system downtime and actual real time attacks.
7.	Alhaidari (2019) New Approach to Determine DDoS Attack Patterns on	Improve framework of SCADA system against Distributed Denial of	i. Need for real-time SCADA datasets to be used and other

	SCADA System Using Machine Learning (Alhaidari & Al-Dahasi, 2019).	Service (DDoS) attacks using three Machine Learning Algorithms.	Machine Learning models used in different configuration.
8.	Al-rabiaah (2018) The "Stuxnet" Virus of 2010 as an example of a "APT" and its "Recent" Variances (Al-rabiaah, 2018).	Characteristics, Features and operations of Stuxnet and APTs, their recent trends of attacks and features to predict future attacks.	No known prevention and detection methods for APTs.
9.	Wilson et al (2018) Deep Learning-Aided Cyber-Attack Detection in Power Transmission Systems (D. Wilson et al., 2018).	Unsupervised feature learning framework for automatic and adaptive attack detection in transmission SCADA system.	No consideration for the multiple sources of uncertainty and variation from subsystems like renewable energy, electric vehicles in Smart Grid.
10.	Husak et al (2018) Survey of Attack Projection, Prediction, and Forecasting in Cyber Security (Husák et al., 2019).	Attack projection, Intention recognition, Intrusion prediction and network security situation forecasting.	Predictions and Forecasting in cyber security are not yet as prominent as attack detection.

Based on the extensive and comprehensive Literature Review carried out (Table 2.3), it revealed:

a. Inability of the existing designed systems to show a distinction between system downtime and real network attacks in a process control network gave rise to the need to reduce high False Alarms Rates (FARs) and misclassification of unknown attacks.
b. Previous researchers used simulated datasets and were unable to extract important features of a real-time SCADA network and this gave rise to the need

to use real-time datasets to train, test and validate models to improve accuracy in the detection and monitoring of recent attack types.

c. Researchers had developed models that detected signature-based attacks but were unable to detect un-patterned and unclassified behavior-based attacks, hence the need for models with robust algorithms for real-time detection.
d. Researchers had proffered solutions with complex computations requiring prolonged downtime for system integration and this gave rise to the need for ease of Integration of the designed model to the network for effective attack detection and monitoring, Inter alia.

CHAPTER THREE

MATERIALS AND METHOD

3.1 Materials

The materials and tools used for this research include software and hardware as listed:

i. Real time network data samples of a Process Control Network
ii. Microsoft Visio Application (a software application for drawing)
iii. Machine Learning Algorithms (Supervised and unsupervised machine learning algorithms)
iv. Python 3 Programming Language
v. Emulator Software and Hardware
vi. Computer Hardware
vii. MATLAB
viii. Python Software Library – Matplotlib, Pandas, Keras, PyOD, Numpy, Scikit-learn, Jupyter Notebook, Seaborn
ix. Anaconda Navigator
x. Google Colab

3.1.1 Real-Time Network Data Samples of a Process Control Network

This was sourced from oil and gas facilities. The intent is to extract data samples from Distributed Process Control Networks at various points on the network for different operating parameters of different processes and use same data for the training, validation and simulation of the algorithms.

3.1.2 Microsoft Visio Application (a software application for drawing)

This was used to develop a centralized and distributed Process Control Network Architecture which showed the configured network as-found and the attack

detection/identification monitoring. This Microsoft application will be used to design the network architecture that will show the points of integration of the design systems.

3.1.3 Machine Learning Algorithms

There are different types of machine learning algorithms available. Selection will be based on our application to detect cyber-attacks in the PCN which will be used to build models based on the real time data samples collected for the training and validation of the model in order to make predictions. The models will learn by training with real time data, validation of the training will be done before the predictions. The models when integrated to the Process Control Network will be useful in detecting amorphous cyber-attacks and will have the capability of notifying the network administrator of a malicious attacks for further investigations.

Unsupervised machine learning algorithms which have the ability to identify patterns and behavior will be used in this study to analyze and cluster unlabeled real time datasets of a Process Control Network. These algorithms will discover hidden patterns or data groupings without human intervention. It has the ability to discover similarities and differences in raw unclassified, unlabeled and uncategorized data samples (Ambonati, 2017).

The following machine learning algorithms were used:

Isolation Forest – This is similar in principle to Random Forest and is built on the basis of decision tree algorithm. It identifies anomalies or outliers. It isolates observations by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature.

One Class Support Vector Machine – An unsupervised algorithm that learns a decision function for novelty detection. It has the ability to classify new data as similar or different to the training set.

K-means Clustering – This partitions the dataset into k predefined distinct non-overlapping subgroups (clusters) where each data point belongs to only one group. A cluster is a collection of data points aggregated because of certain similarities.

Long Short-Term Memory Models – This model mimics the operation of the neurons in the human brain.

K-Nearest Neighbors – Has the capability to detect anomalies or outliers.

Deep Learning algorithms – this is a group of machine learning methods which work with the principle of Artificial Neural Networks and could be used for supervised, semi-supervised or unsupervised learning applications.

3.1.4 Python Programming Language

This is a high-level general-purpose programming language and makes it possible to build machine learning model prototypes as well as allow the function testing of the models. The choice of Python 3 for the machine learning modelling is due to the fact that it is flexible, has tools, gives access to libraries and frameworks for Artificial Intelligence and Machine learning. The implementation of machine learning algorithms requires a lot of time and function testing. To reduce development time, there is need to leverage on the python 3 frameworks and libraries. Software libraries as found in python 3 are pre-written codes that developers use to solve common programming tasks, to reach some functionality or perform different actions. Python libraries provide base level items so developers don't have to code them from the beginning each time. With python, there will not be need for compilation of source codes, changes can be quickly made and the results instantly seen without the need for recompilation of the codes, thereby saving hardware resources. There is provision for the integration of Python with other programming languages to achieve desired goals (Kuhlman, 2013)(Python Software Documentation, 2023a)(Python Software Documentation, 2023b).

Python's Libraries - Machine Learning requires continuous data processing, and handling, Python's libraries provides that functionality.

Some of the Python's libraries to be leveraged on this study include:

i. **Pandas** – This is useful for general purpose data analysis, high-level data structures and analysis. It aids the merging, filtering, collation of data and import from external sources like MS Excel.

ii. **Matplotlib** – This is useful for the creation of 2D plots, histograms, charts and other forms of graphical visualization.

iii. **Numpy** – This is useful for high performance scientific computing and data analysis.

iv. **Scikit-learn** – This is useful in handling basic machine learning algorithms like clustering, linear and logistic regressions, regression, classification, neural networks and is designed to work with Python numerical and scientific libraries like Numpy and Scipy.

v. **Keras** – This is for deep learning. This helps in fast calculations and prototyping, as it uses the GPU in addition to the CPU of the computer.

vi. **TensorFlow** – This is for working with deep learning by setting up, training and utilizing artificial neural networks with massive datasets

vii. **Scipy** – This is useful for advanced computing.

viii. **PyOD** – Python Outlier Detector Library

3.1.5 Allen Bradley RSLogix 5000 PLC Emulator Software

The operation of the 3-phase separator control system can be simulated using an Allen Bradley RSLogix 5000 PLC Emulator software which eliminates the need for the actual PLC hardware and its IO modules. There are several ControlLogix processors and decision for a particular choice of processor is dependent on several factors which may include application and cost (Babcock, 2009). However, all the processors use RSLogix 5000 which makes it possible for an application written for one processor to be adapted

and used for another 1756 processor. For the purpose of this research and simulation of the 3-phase separator, an RSLogix 5000 Emulator Software was used which is capable of simulating 1756 processor, Digital Inputs Module, Analog Input Module, Digital Output Module and so on (Rockwell, 2022)(Rockwell, 2011)(Babcock, 2009).

3.1.6 Computer Hardware Requirement

Training and simulation of machine learning algorithms requires huge hardware resources for smooth computation of large amount of data and running of complex algorithms that require powerful computation hardware. The following hardware characteristics are considered while selecting a hardware for Artificial Intelligence technologies: processing speed, graphics processing capability, storage space (HDD or SSD), RAM. The Graphics Processing Unit (GPUs) are considered faster than Central Processing Units (CPUs) with emphasis on speed and high throughput. There are minimum hardware requirements for the effective running of machine learning algorithms. A minimum of 8GB RAM can be used but 16GB RAM and above is recommended for most deep learning tasks. A minimum of 7^{th} generation (Intel Core i7 processor) is recommended. For the storage, SSD is recommended over HDD and the size will be relative to the size of data handled. It is important to note that the larger the data size the higher the computational requirement.

Table 3.1 Design Materials Specification

S/No	Device Name/Tag	Device Type	Process Application	Manufacturer	Quantity Used	System Application
1	Sensor A PT1	Differential Pressure, 4-20mA, HART	3-phase Separator vessel Internal pressure	Yokogawa EJX110A	2	Input Process Control

2	Sensor B PT2	Differential Pressure, 4-20mA, HART	3-phase Separator vessel Internal pressure	Yokogawa EJX110A	2	Input Emergency Shutdown
3	Sensor C	Differential Pressure 4-20mA, HART	Pipeline pressure	Yokogawa EJX110A	4	Input Crude Supply and discharge pipelines
4	Sensor D H1	Level Transmitter 4-20mA, HART	3-phase Separator vessel Water Level	Emerson Rosemount	1	Input Process Control Water Level
5	Sensor E H2	Level Transmitter 4-20mA, HART	3-phase Separator vessel Crude Oil Level	Emerson Rosemount	1	Input Process Control Crude Oil Level
6	Sensor F	Multivariable Flow transmitter 4-20mA, HART	Pipeline Flow	Yokogawa EJX110A	1	Input Process Control Gas discharge pipeline
7	Software Input/Output Module	1789 I/O simulator module	PLC Automation	Allen Bradley RSLogix 5000 Emulator	4	Programmable Logic Controller
8	Hardware Software	Software Emulator Controller	RSLogix 5000	Allen Bradley RSLogix 5000 Emulator	1	Programmable Logic Controller
9	Software	Microsoft Visio 2013	Design Software	Microsoft	1	Software Design Application
10	Software	Python-3 Programming Language Version 3.8.5	Programming Language for Algorithm Application	Python Software foundation	1	Programming Software

11	Software	Pandas Version 1.5.2	Python Software Library	Developer - Wes Mckinney	1	Data manipulation and Analysis
12	Software	Matplotlib Version 3.3.2	Python Software Library	Developer: Michael Droettboom, et al	1	Python Plotting Library
12	Software	Numpy Version 1.20.3	Python Software Library	Developers: Travis Oliphant	1	Open-Source Python Library
13	Software	Keras Version 2.8.0	Python Software Library	Developers: Francois Chollet	1	Open-Source Python Library
14	Software	PyOD Version 1.0.6	Python Software Library	Developers: Yue Zhao, Zain Nasrullah, and Zheng Li.	1	Open-Source Python Toolbox
15	Software	Jupyter Notebook Version 6.1.4	Project Jupyter Web-based interactive	Developers: Fernando Perez et al.	1	Open-Source Python
16	Software	Anaconda Navigator 2016 Version 2.3.2	Data Science Package, Machine Learning Applications	Developers: Anaconda, Inc	1	Machine Learning Applications
17	Software	Scikit-Learn Version 0.23.2	Python Software Library	Developer: David Cournapeau	1	Free Software Machine Learning Library
18	Software	Seaborn Version 0.11.0	Python Software Library	Developer: Michael Waskom	1	Data Visualization Library
19	Software	MATLAB 2021b Engineering Software	Engineering Application	Developer: Mathworks	1	Engineering Application
20	Software	Google Colab	Jupter Notebook on the cloud	Developer: Google research	1	Cloud based Application

21	Hardware	Computer CPU or GPU	Minimum Recommended Specification: 16GB RAM, 7th generation (Intel Core i7 processor Quad Core, 3GHz), SSD, 64bit Windows 11

3.2 Methods

The methodology generally applied in this research is, "Top-Down System of Engineering design (where the entire system's structure is considered first, before the units modular designs)". At the hardware sections of the design, block diagrams, schematic (circuit) diagrams, equivalent circuits and flow diagrams are the adopted techniques employed for the description, modelling and analysis of the circuits, respectively. At the Software development section of the system, the structural engineering tools and techniques used include; Machine learning and Deep learning toolkits, more especially unsupervised machine learning, structured programming languages and packages, such as, Python, MATLAB, PLC emulator software, Flowcharts and Algorithmic representations of normal plant operation as well as the plant under attack. While, statistical principles, graphical representations and standard control systems equations are adopted in results presentations, discussion and validation, respectively.

3.2.1 The Block Diagram of System Design

To ensure a comprehensive approach geared towards the design of a robust system and to grant insight into the sub-system composition, the Top-Down System of Engineering Design was applied as shown in the block diagram of Figure 3.1.

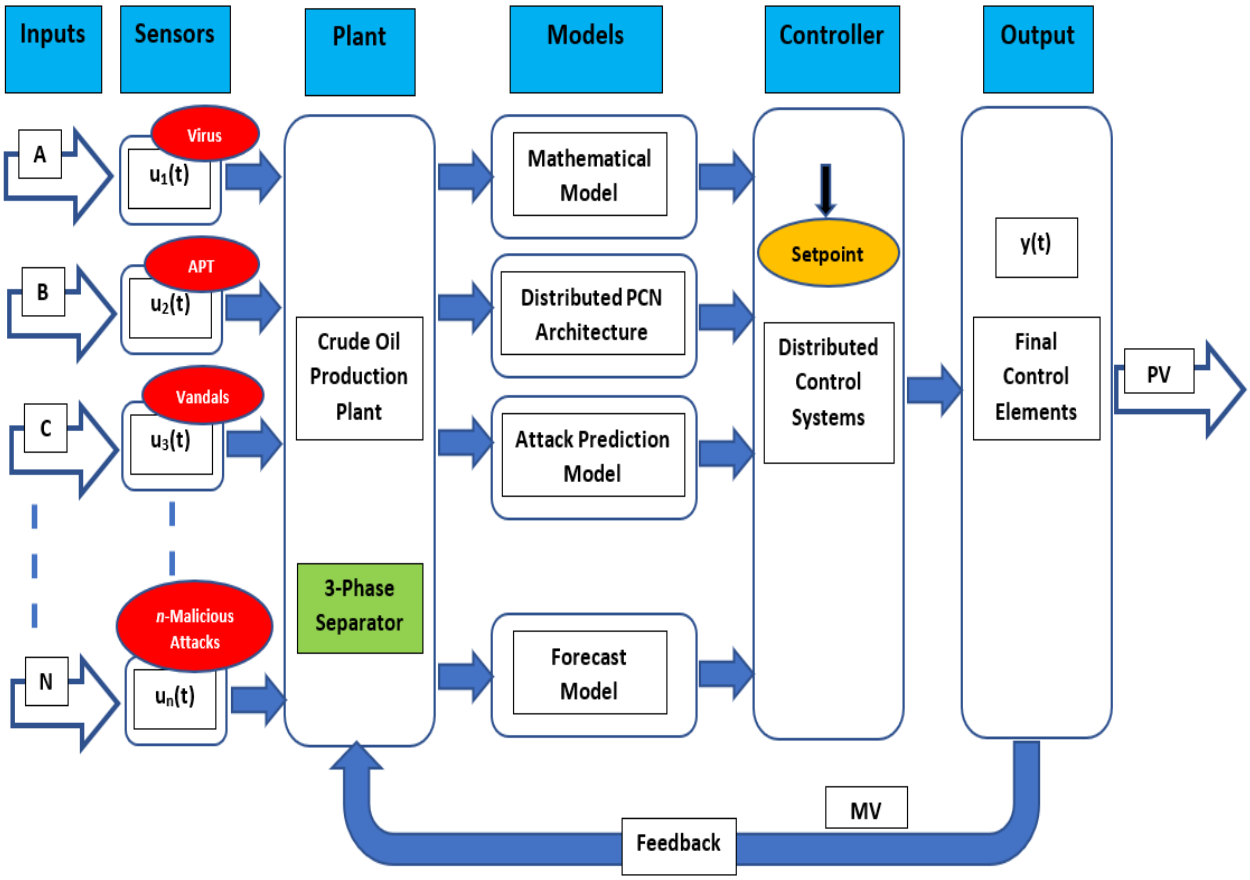


Figure 3.1 Block diagram of System Design

3.2.1.1 System Description

The block diagram in Figure 3.1 shows the entire system design. The various inputs which are measured by the sensors as *u(t)*, when attacked by *n*-malicious attackers which may be in the form of virus attacks, APT attacks, vandalism or theft. The compromised inputs if not detected will be fed into the controller which produces output

signals based on the configured logic and predefined setpoints. The system is designed in such a way that there is a communication between the input sensors, the production plant, the controller and the final control elements.

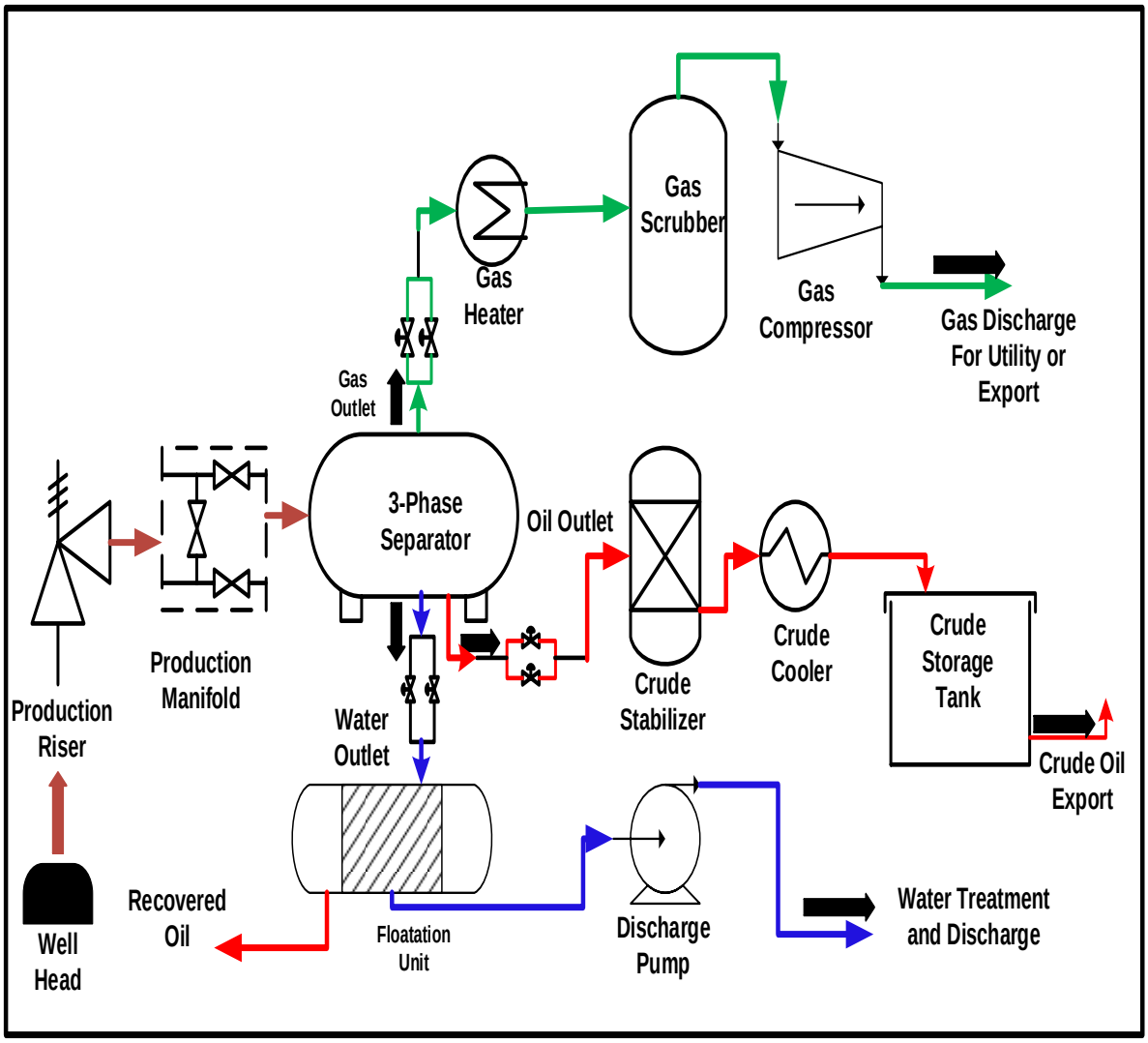


Figure 3.2 Process Flow diagram of Crude Oil Production System

The details of how the entire crude oil production process is tied together is shown in Figure 3.2 while the equivalent block diagram is shown in Figure 3.3. The 3-phase separator is not an isolated unit, rather it is connected to other units which make up the crude oil production system. An attack on any of the instrumentation will have a cascading effect on the entire process which may lead to shut down and possible

escalation if there are associated spillage of the process fluid. The crude from the 3-phase separator vessel flows to the crude stabilizer vessel. Any process upset from the 3-phase separator will have a resultant cascading effect on the crude stabilizer vessel and this will affect the entire process control and may have a devastating effect.

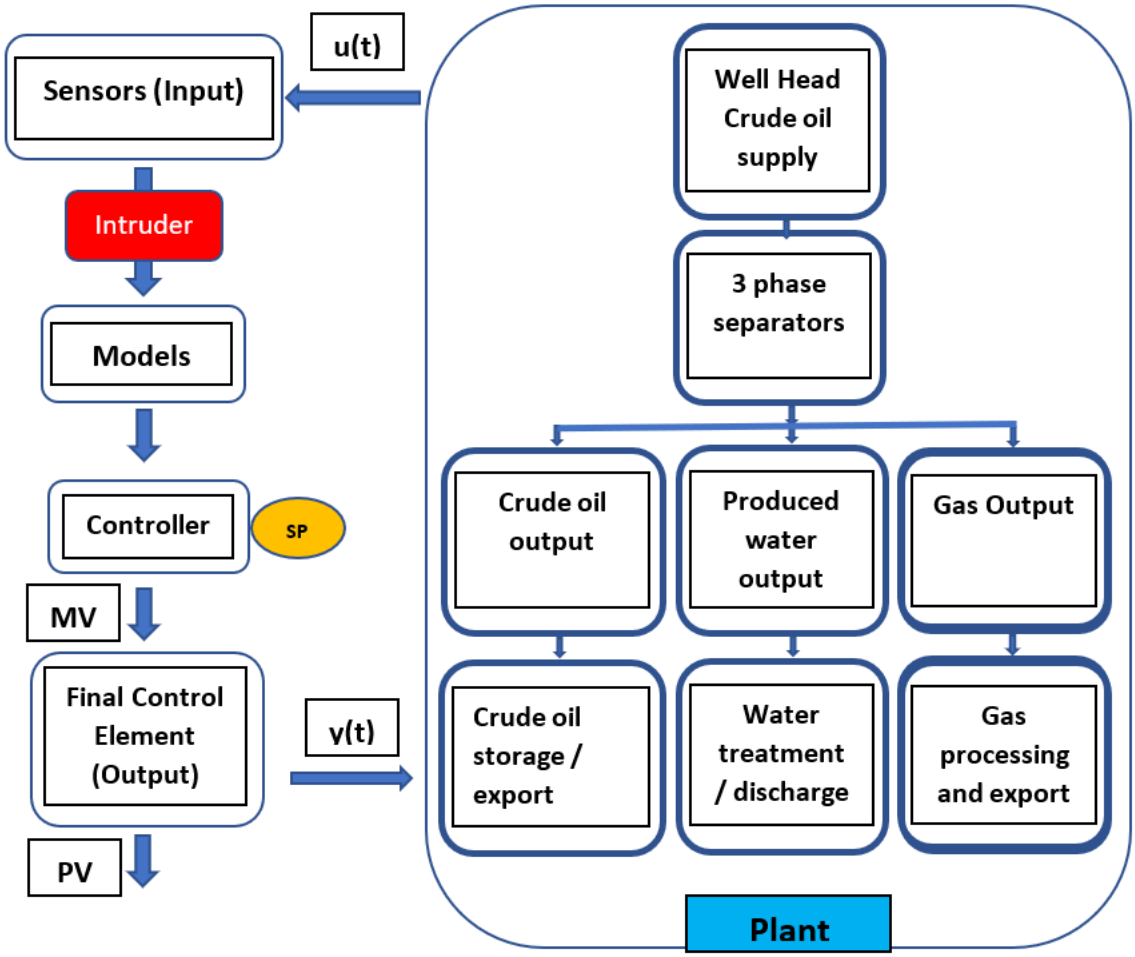


Figure 3.3 Equivalent block diagram of the crude oil production system process control network.

3.2.1.2 Overall System Diagram and the Descriptions

A. Input – The inputs to the system as shown on Figure 3.1 as inputs A, B, C, …., N, which identifies the different forms of input signals to the system which may be in the form of process variables such as pressure, level, flow, temperature, vibration, speed

and so on which may be coming from the interactions with other multivariable systems for effective operations of the system.

B. Sensors - This comprises of all the sensing elements and transducers connected to the fixed/rotating equipment and their interconnections used in measuring the process variables. The process variables measured with these sensors include: pressure, flow, temperature, level and other equipment conditions like vibration, velocity, acceleration, displacement and so on. The signals from these sensing elements serve as inputs to the controllers and are sent through the communication links. The sensors are located on the Layer 0 of the process control network. Real-time data of a live plant was collated and used for the implementation of this research which simulates oil and gas control system. The RSLogix 5000 simulator chassis as shown in Figure 3.4 was used in this simulation. As shown on Figure 3.4, the input/output devices were configured on the slot 4 and slot 5 of the Allen Bradley RSLogix 5000 emulator as 1789 I/O simulator module. The pressure values and level values coming from these respective sensors are continuous in nature, hence are configured on the Analogue input module while the valve statuses, feedbacks and ON/OFF states from switches are discrete in nature and configured on the digital input modules. Most sensors communicate with the controller using 4-20mA electronic signals, there may be need for the conversion of the measured signals from its original physical quantity to electrical signals example pressure to current converter (P/I).

C. The Plant - This comprises of all the fixed/rotating equipment and their interconnections which are necessary for the production of crude oil and they include vessels, pipelines, pump, compressors, heat exchangers and so on. The source of the crude oil is from the well head. Through the manifold, the crude flows to the 3-phase separator, where the process fluid is separated into three components oil, gas and water. Each of the separated component flows through their respective pipelines to their final destination after treatment which maybe for use, discharge or export. In this research

real-time SCADA data was used for the simulation of the process plant operation and these serve as input signals to the controller. Figure 3.2 shows the equivalent block diagram of the crude oil production system/a process control network while Figure 3.3 shows the Process Flow diagram of Crude Oil Production System.

D. Controllers - This comprises of all the controllers/logic solvers which receives the input signals from the sensors/transducers, interprets the signals and makes logical decisions based on the setpoints and configured logic for the automatic control of processes. The controller has the logic for the different control loops with proportional, Integral and Derivative (PID) control algorithms which are varied to get optimal response of the control loop. In this research, different controllers were considered for the different control loops which include pressure controllers, level controllers, flow controllers, programmable logic controllers and so on. The concept used in this research is the distributed control system whereby, the different control loops are tied to the central control system where the logic configuration resides. As seen in Figure 3.4, the Allen Bradley RSLogix 5000 PLC Emulator Controller configured on the slot 3 of the emulator chassis software was used to simulate the process control of the plant. The controller which housed the control logics was responsible for sending out the output signals to the final control elements.

E. The Final Control Element - This comprises of all the output devices which receives the output signals *y(t)* from the controller for the automatic operation of the process equipment. Output devices used in this research include control valves, shutdown valves, blow down valves and their respective actuators. Other output devices include electric motors, pump and compressor drivers. The final control element implements the manipulated value sent from the controller for the effective operation of the process control. The communication between the controller and the final control elements is dependent on the nature of the device used. In this research, the output

command sent from the controller to the valves was 24Volts DC. The valve outputs are configured on the slot 5 of the Allen Bradley RSLogix 5000 emulator as 1789 I/O simulator module.

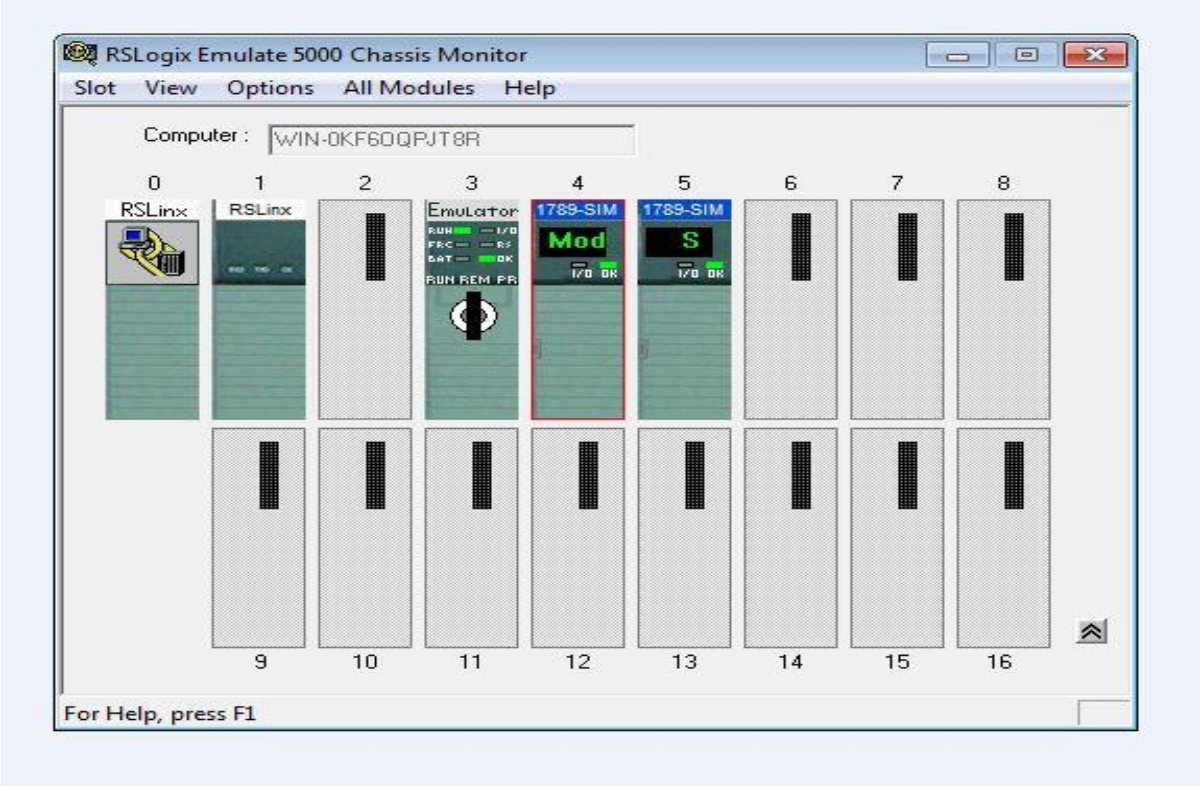


Figure 3.4 Allen Bradley RSLogix 5000 PLC Emulator

The Allen Bradley RSLogix 5000 PLC Emulator as shown in Figure 3.4 is a 16-slot chassis which the software emulator application is used to simulate the operation of the PLC hardware without connecting to any physical hardware. The Slot 0 of the emulator chassis is used for the RSLinx communication driver software which provides the communication to devices using other protocols like ethernet/IP, Devicenet, ControlNet and so on. The emulator processor is configured on the slot 3 of the emulator chassis while the emulator 1789 Simulator I/O modules are configured on the slot 4 and 5 of the chassis.

3.2.1.3 Mathematical Model and Analysis of Process Control Networks for Attacks and Monitoring

From Figure 3.2, the entire process flow diagram of an oil and gas producing plant was detailed, starting from the well head to the final point of export or utilization of the produced oil, gas or water. For the ease of computation, simulation and modelling, more focus will be on the 3-phase separator as shown in Figure 3.5.

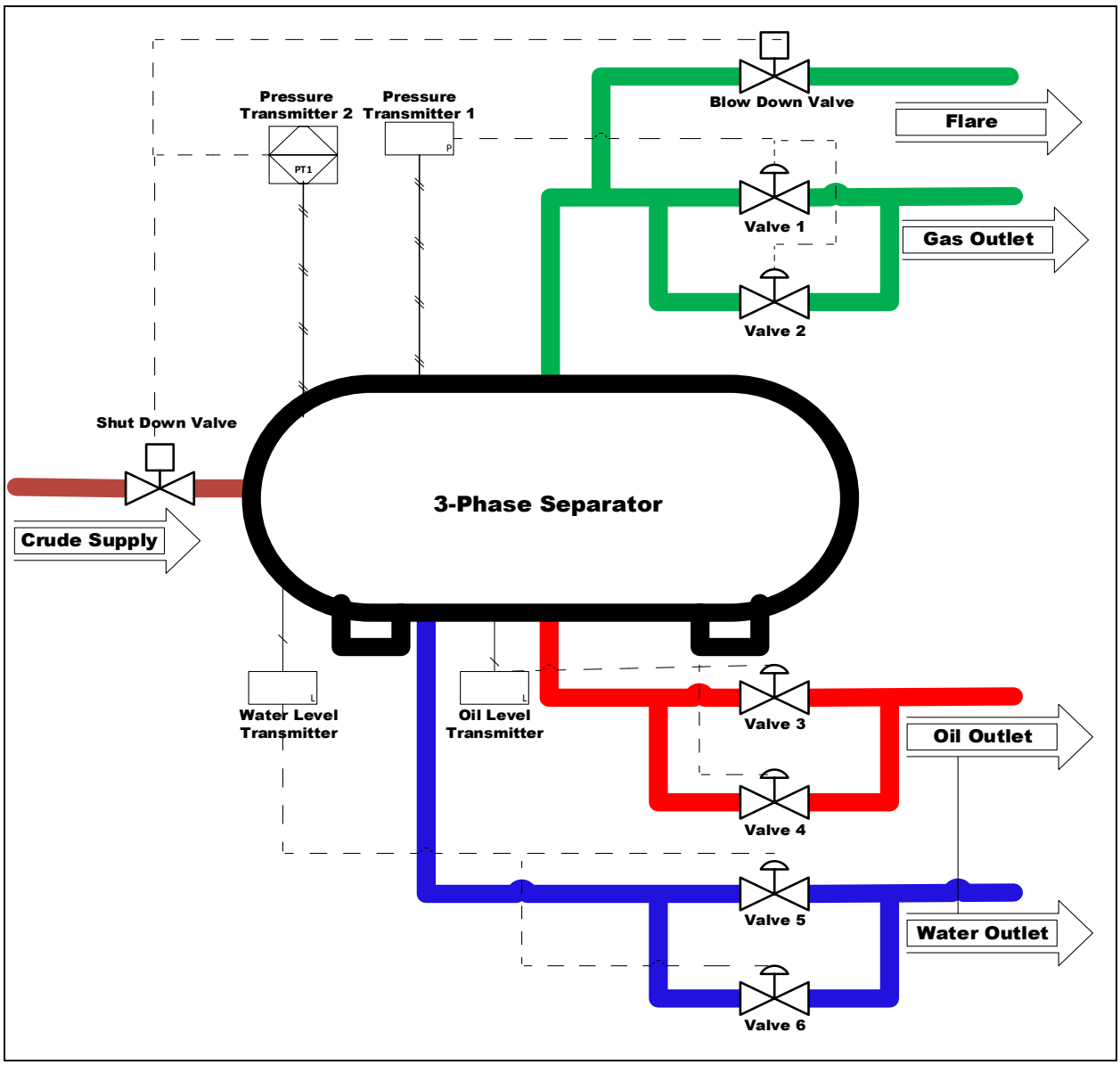


Figure 3.5 Process Flow Diagram of a 3-Phase Separator

The 3-Phase Separator is a gravity vessel that has the capability of separating the crude from the well bore into three phases namely: gas, oil and water (Song et al., 2023)(Abdu Sabir et al., 2022)(Wu et al., 2022)(Jonach et al., 2022). In this case the separator is a horizontal fixed structure and the process variables monitored include pressure, level of water, level of crude, temperature and so on. It is an instrumented vessel which requires continuous monitoring to avoid process upset and over pressurization. The crude is supplied from the oil production manifold into the vessel and the internal configuration of the vessel as well as residence time helps in the separation process. From Figure 3.5, the Pressure Transmitter 2 is connected to the Emergency Shutdown System which has High-High (HH) and Low-Low (LL) setpoints, which if triggered will close the Shutdown Valve and Open the Blowdown valve. The Shutdown valve is a fail-close valve while the Blowdown valve is a fail-open valve. The Blowdown valve when opened depressurizes the vessel by allowing the gas to flow to the flare.

The 3-phase separator has three outlets namely:

a. Gas outlet,
b. Crude outlet and
c. Water outlet respectively.

3.2.1.4 The Gas Outlet Control Loop

The flow of gas on the gas outlet pipe is controlled by valve 1 and valve 2. The input to the control loop is pressure transmitter 1. The gas from the 3-phase separator is further processed to achieve desired results depending on the system design, process parameters and rated pressure. The continuous monitoring of the process variable as well as the gas condition determines what further processing is required. The gas produced from the 3-phase separator could be used for other purposes which may include dehydration and used as fuel gas for power generation, liquefied and prepared

for export or further compressed to boost its pressure and be used for re-injection to the reservoir for the boosting of production. The continuous monitoring of the process parameters is necessary to keep the control loop in check. An attack on this control loop could lead to an unexpected behavior of the outlet valves which can lead to process upset and if not controlled could trigger further system upset, the consequences could be over pressurization which may lead to explosion, loss of containment, oil spillage and eventually fire which may have devastating impact on the people, environment and assets.

3.2.1.5 The Oil Outlet Control Loop

The flow of oil on the oil outlet pipe is controlled by the two outlet valves: valve 3 and valve 4 as shown in Figure. 3.5. The control loop uses the oil level transmitter as the input signal with which to control the two valves. The interface level measurement is used in this case to differentiate the oil level from the produced water level. The produced oil from the 3-phase separator is further processed and this may vary due to the process parameters as well as condition of the oil produced. In this case, the crude oil then flows to the stabilizer column, this is necessary to achieve the Reid Vapor Pressure (RVP) before going to the crude cooler and finally to the crude oil tank. RVP is a required characteristics necessary to regulate the volatility of the crude oil in preparation for export. The continuous monitoring of the process parameters is necessary to keep the control loop in check. An attack on this control loop could lead to an unexpected behavior of the outlet valves which can lead to process upset and if not controlled could trigger further system upset, the consequences could be over pressurization which may lead to explosion, loss of containment, oil spillage and eventually fire which may have devastating impact on the people, environment and assets.

3.2.1.6 The Water Outlet Control Loop

The flow of water on the water outlet pipe is controlled by the two outlet valves: valve 5 and valve 6 as shown in Figure 3.5. The control loop uses the water level transmitter as the input signal with which to control the two outlet valves. The 3-phase separator separates the produced crude from well bore into gas and liquid. The liquid is further separated into oil and gas as a result of the fluid density. Gravitationally the water usually settles at the bottom of the vessel thereby making it possible to drain out when required. The Basic Sediment and Water (BS&W) property of the produced crude determines the number of suspended solids and produced water from the reservoir. The produced water from the 3-phase separator is further processed and this may vary depending on the process parameters as well as the condition of the produced water. In this case the produced water flows to the floatation unit for further separation of oil from water and treatment. The produced water if it meets the specification of the regulatory agency after treatment may be discharged to the effluent waters or may also be reinjected to the well bore to boost the productivity of the reservoir. The continuous monitoring of the process parameters on the water outlet line is necessary to keep the control loop in check. An attack on this control loop could lead to an unexpected behavior of the outlet valves which can lead to process upset and if not controlled could trigger further system upset, the consequences could be over filling of the vessel which may lead to loss of containment, and spillage which may have devastating impact on the people, environment and assets.

Consider the flow of crude oil into the 3-phase separator as $q_{in}(t)$, the level of the crude as h and the flow of crude oil out of the 3-phase separator as $q_o(t)$. The flow out of the 3-phase separator $q_o(t)$ is proportional to the level of crude h in the 3-phase separator as long as the flow is laminar.

$$q_o(t) = kh(t) \qquad (3.1)$$

Where k is a constant.

The rate of change of the crude volume in the 3-phase separator is assuming negligible produced water content:

$$\frac{dV(t)}{dt} = q_{in}(t) - q_o(t) \quad (3.2)$$

This equation can be changed to:

$$\frac{dA_T h(t)}{dt} = q_{in}(t) - q_o(t) \quad (3.3)$$

Where the constant A_T is the cross-sectional area of the vessel

$$\frac{A_T dh(t)}{dt} = q_{in}(t) - q_o(t) \quad (3.4)$$

$$\frac{A_T dh(t)}{dt} = q_{in}(t) - kh(t) \quad (3.5)$$

$$\frac{dh(t)}{dt} = \frac{1}{A_T} q_{in}(t) - \frac{k}{A_T} h(t) \quad (3.6)$$

$$\frac{dh(t)}{dt} + \frac{k}{A_T} h(t) = \frac{1}{A_T} q_{in}(t) \quad (3.7)$$

Transforming Equation (3.7)

$$sH(s) + \frac{k}{A_T} H(s) = \frac{1}{A_T} Q_{in}(s) \quad (3.8)$$

Where $H(s)$ and $Q_{in}(s)$ are the Laplace transforms of *h(t)* and $q_{in}(t)$

Transfer function,

$$\frac{H(s)}{Q_{in}(s)} = \frac{1}{A_T s + k} = \frac{{}^1/_k}{\left({}^{A_T}/_k\right)s + 1} \quad (3.9)$$

The standard transfer function is:

$$\frac{K}{S\tau + 1} \quad (3.10)$$

Where K is the system Gain and τ is the time constant.

From Equation (3.9) ${}^1/_k$ is the System Gain while $\left({}^{A_T}/_k\right)$ is the time constant.

Figure 3.6 shows the possible attack methods to any of the control loops under consideration which include deception attacks, denial-of-service (DoS) attacks and forced data injection attacks and each of them could lead to loss of Confidentiality, Integrity and Availability (CIA) of the process control network. In this scenario, the deception attacks maybe any or combination of the following: incorrect measurement from the sensor or the input devices, incorrect time stamp for measurements, improper identification of the communication node address. The most common attack is the physical attack on the infrastructure or field devices. This form of attack will not be considered in this work as it is expected that the field devices are shielded from unauthorized personnel and there exist physical restrictions to the locations of the field devices. The focus of this work is more on logical or cyber-attacks.

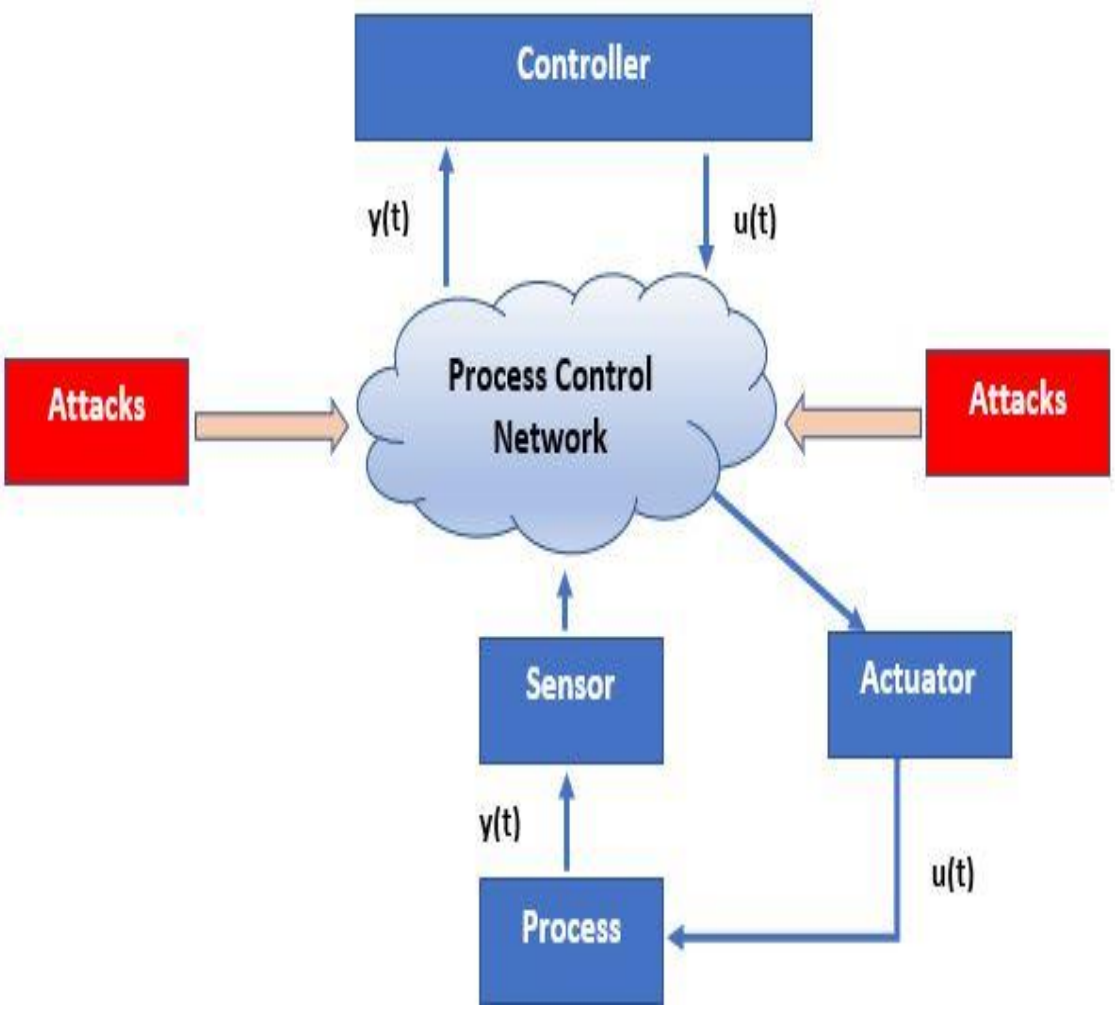


Figure 3.6 Possible attack methods on a Process Control Network

The network in Figure 3.6 shows a Process Control Network which comprises of interconnected components using shared communication resources in a closed loop. The network is necessary for data communication: the sensors are located at the plant for data acquisition and measurement of process state *x(t)*, the measured data is transmitted to the controllers which computes and make logical decisions on the network based on the control input signals received. The control output signal is sent from the controller to the actuators located in the plant which are the final control element or output device, the communication network provides the link between the process plant and the controllers. The sensors also measure and transmit the output feedback *y(t)* back to the controller. The process control network can be rendered

inoperable when an intruder attack any of the mentioned network components by deception, by injecting or manipulating data packets (false-data injection), by blocking data packets from reaching destination denial of service (DoS). Any of these forms of cyber-attacks could lead to process upset which can cause loss of containment or spillages, which may lead to explosion and fire, that can impact people, asset and environment.

The Process Control Systems is made up of the processes, sensors, actuators (final control elements), the process variables and their communication networks. The process variables store certain values of the physical processes, example is level, pressure, temperature in a vessel, flow rates on the pipeline and so on. The state of a process at a given time is determined by the values of its process variables (Ndonda & Sadre, 2022).

3.2.1.7 Model of 3-Phase Separator Under Attack

In this our scenario where there are more variables like oil interface level, water level, ullage, innage, let the oil level be $x_1(t) = h(t)$. The interaction between the process variables on a 3-phase separator under attack can be represented using the state space model as shown in Figure 3.7.

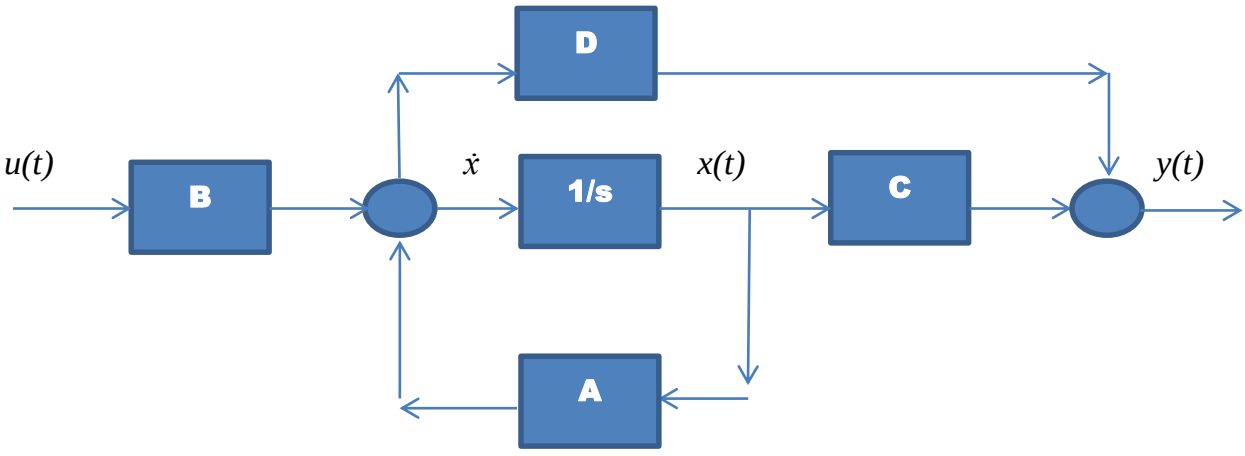


Figure 3.7 State Space Representation of a Process Control Network

Equation (3.7) becomes:

$$\frac{dx_1(t)}{dt} + \frac{k}{A_T}x_1(t) = \frac{1}{A_T}q_{in}(t) \qquad (3.11)$$

$$\frac{dx_1(t)}{dt} = -\frac{k}{A_T}x_1(t) + \frac{1}{A_T}q_{in}(t) \qquad (3.12a)$$

where

$$x(t) = x_1(t)$$

$$u(t) = q_{in}(t) \qquad (3.12b)$$

$$A = -\frac{k}{A_T}$$

$$B = \frac{1}{A_T}$$

Applying equation (3.12b) in the above equation (3.12a), a more general form of a dynamic state equation can be derived as shown in equation (3.13)

$$\frac{dx}{dt} = \dot{x}(t) = Ax(t) + Bu(t) \qquad (3.13)$$

$$y(t) = Cx(t) + Du(t) \qquad (3.14)$$

Where,

$x(t) \in \mathbf{R^n}$ *is the state vector*

$\dot{x}(t) \in \mathbf{R^n}$ *is the differential state vector*

$u(t) \in \mathbf{R^m}$ *is the input vector*

$y(t) \in \mathbf{R^p}$ *is the output vector*

$E \in \mathbf{R^{nxn}}$ *is the state matrix*

$A \in \mathbf{R^{nxn}}$ *is the state system matrix n* x *n*

$B \in \mathbf{R^{nxm}}$ *is the input matrix n* x *m*

$C \in \mathbf{R^{pxn}}$ *is the output matrix p* x *n*

$D \in \mathbf{R^{pxm}}$ *is the feedforward matrix p* x *m*

Modelling the control loop under attack as linear time-invariant system with unknown inputs while neglecting noise in the measurement and system non-linearities. Equation (3.13) is the State Equation while Equation (3.14) is the output equation. The model as shown in Equations (3.13) and (3.14) represents a genuine system component failure, physical attacks or cyber physical attack caused by malicious intruder.

Considering a complex multivariable system like a 3-phase separator as shown in Figure 3.8, with interactions between different process fluids like the crude oil level and the produced water level, let $h_1$be level of produced water plus crude oil in the 3-phase separator while h_2 is the level of crude oil only inside the 3-phase separator. Assuming both constituents have uniform cross sectional area $A_T = 1$

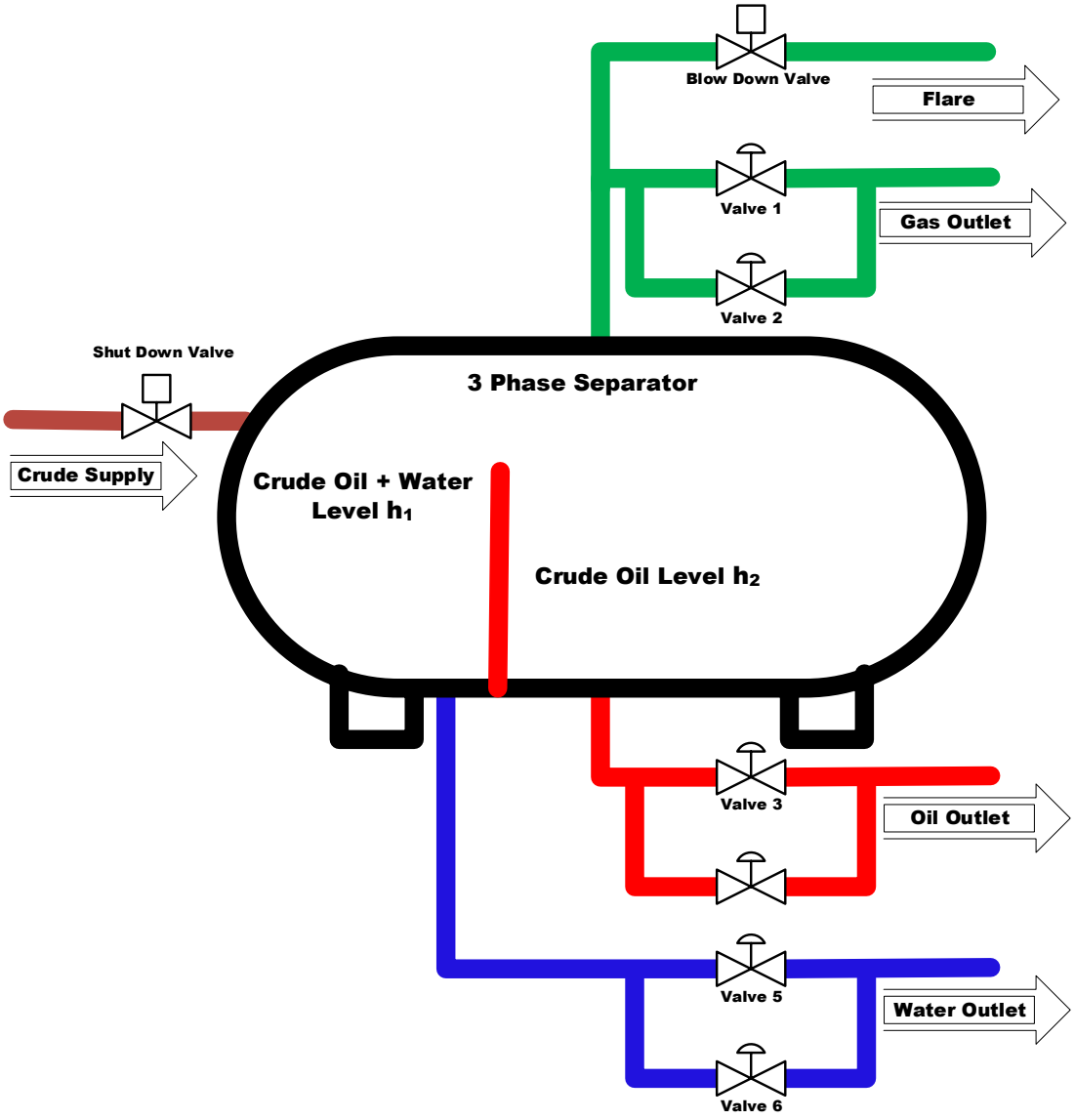


Figure 3.8 Modelling of a 3-Phase Separator

Consider the flow of crude oil into the 3-phase separator as $q_{in}(t)$,

Flow of crude oil across the baffle in the separator is $q_2(t) = k_2(h_1 - h_2)$,

The outlet crude oil flow out of the vessel $q_3 = k_3 h_2$

From Equation (3.7)

$$\frac{dh_1(t)}{dt} = -k_2(h_1 - h_2) + q_{in}(t) \qquad (3.15)$$

$$\frac{dh_2(t)}{dt} = k_2(h_1 - h_2) - k_3 h_2 \qquad (3.16)$$

This can further be rearranged into matrices

$$\begin{pmatrix} \frac{dh_1(t)}{dt} \\ \frac{dh_2(t)}{dt} \end{pmatrix} = \begin{pmatrix} -k_2 & k_2 \\ k_2 & -(k_2 + k_3) \end{pmatrix} \begin{pmatrix} h_1(t) \\ h_2(t) \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} q_{in} \qquad (3.17)$$

This is the same form for Equation (3.13):

Where matrices $A = \begin{pmatrix} -k_2 & k_2 \\ k_2 & -(k_2+k_3) \end{pmatrix}$ and $B = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

The matrix ***A*** determines the dynamics of the system ẋ . The matrix ***B*** is how the system responds to inputs *u(t)*. Where k_1, k_2, *and* k_3 are constants.

With this, the state equation will still be sufficient for the study of the physical properties of the system with more variables. For systems with large variables, the following can be used:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}$$

$$u = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_m \end{bmatrix}$$

$$y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_p \end{bmatrix}$$

$x(t)$ is an *n* vector of states. The input variables $u(t)$ include water level, oil level, pressure, flow and temperature. In this research, the input variable u(t) considered is pressure (Bar).

$u(t)$ is an m vector of controls and can be called the manipulated variables $y(t)$ is an l vector of outputs measured variables.

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix}$$

$$B = \begin{bmatrix} b_{11} & b_{12} & b_{13} & \dots & b_{1m} \\ b_{21} & b_{22} & b_{23} & \dots & b_{2m} \\ b_{31} & b_{32} & b_{33} & \dots & b_{3m} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ b_{n1} & b_{n2} & b_{n3} & \dots & b_{nm} \end{bmatrix}$$

$$C = \begin{bmatrix} c_{11} & c_{12} & c_{13} & \dots & c_{1n} \\ c_{21} & c_{22} & c_{23} & \dots & c_{2n} \\ c_{31} & c_{32} & c_{33} & \dots & c_{3n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ c_{p1} & c_{p2} & c_{p3} & \dots & c_{pn} \end{bmatrix}$$

$$D = \begin{bmatrix} d_{11} & d_{12} & d_{13} & \dots & d_{1m} \\ d_{21} & d_{22} & d_{23} & \dots & d_{2m} \\ d_{31} & d_{32} & d_{33} & \dots & d_{3m} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ d_{p1} & d_{p2} & d_{p3} & \dots & d_{pm} \end{bmatrix}$$

The matrices *A*, *B*, *C* and *D* can be constant or time-varying.

3.2.2 Design of a Centralized and Distributed Process Control Network Architecture for Attack Detection and Identification Monitoring

Typically, a PCN architecture consists of the following: Human Machine Interface (HMI), Programmable Logic Controllers (PLC), Remote Terminal Units (RTU), Master Terminal Units (MTU), engineering workstations (EWS), network connections and network devices like switches, firewalls and routers, process instrumentation which include sensors, transducers, actuators, transmitters, converters and so on.

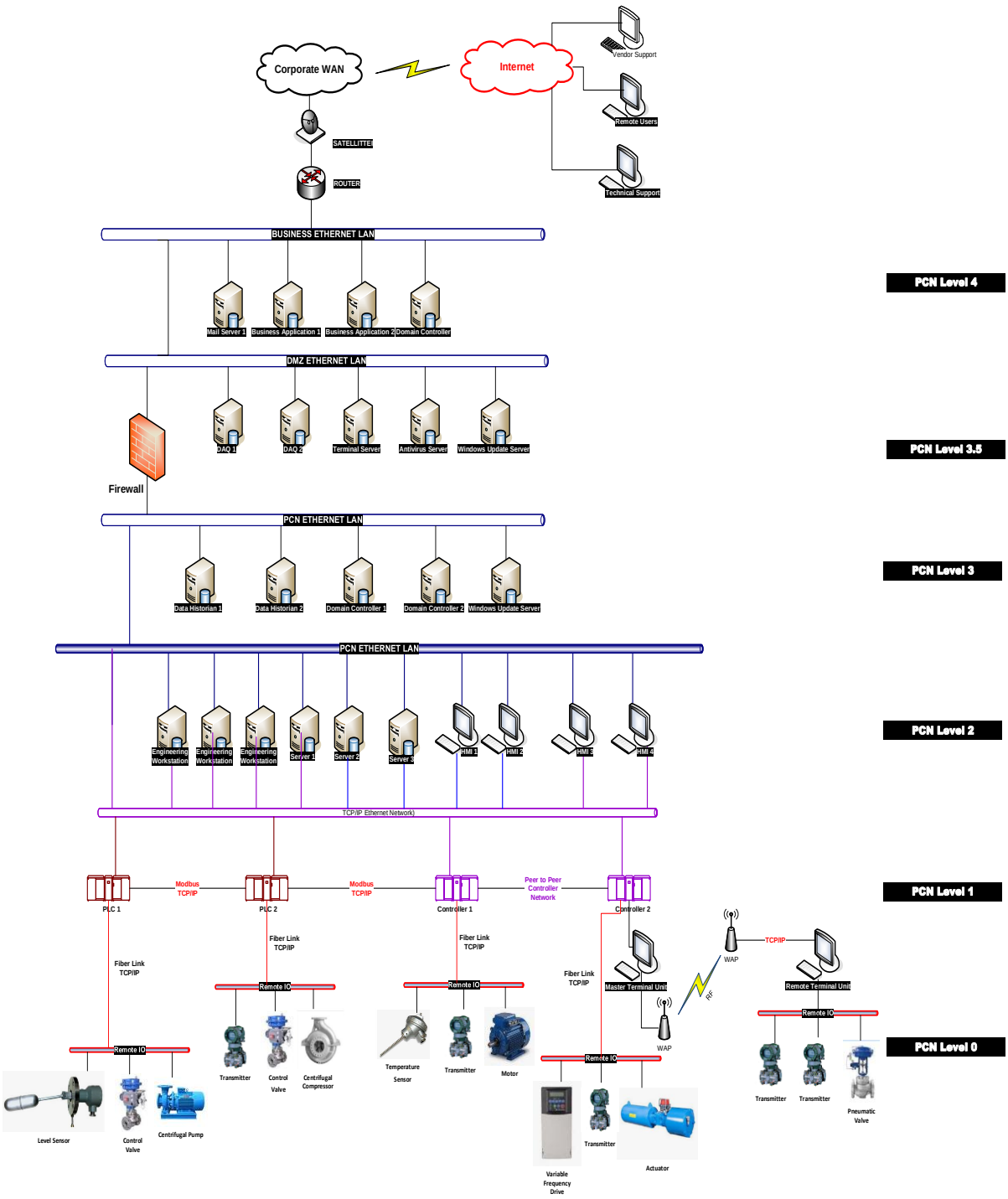


Figure 3.9 A typical PCN Architecture Design

The network components are categorized from Level 0 to Level 3.5 for a PCN as shown in Figure 3.9.

Level 0 – This is the actual production process equipment level – This level houses the input and output devices which include sensors, transmitters, actuators and other final control elements for the production process. The wiring of these instruments is terminated to Remote Input/Output Cabinets through the field Junction Boxes.

Level 1 – At this level, the Logic Solvers, Controllers and other real time control systems of the production process are located. The controllers may be physically located at remote locations to ensure sufficient security and restricted access control. Fiber technology may be used to route the connections from the Remote Input/Output units to the controller where ethernet/IP may not be used due to distance limitations. Where fiber technology is used there will be need for fiber to ethernet converters on the network. There is every need to ensure that the communication between the level 0 devices and level 1 devices are seamless to avoid lags in data communication.

Level 2 – This level houses the Human Machine Interface (HMI) which is usually the field operators' interface, Supervisory Controls, Engineering Work Station (EWS), other critical control Servers. The HMI is the interface through which the plant operator interacts with the plant operations for process interventions. Technical adjustments on the control loop like tunning and programming of control loops are usually done using the Engineering workstations at this level. Supervisory Control is necessary to monitor the functionality and availability of respective components of the system. It is recommended to integrate the security server with the capability of detecting and preventing unauthorized data exchange at this level to be able to monitor the data exchange between the field devices and the controllers in a real time plant.

Level 3 – This level houses Data Historians, Advanced Control Servers, Domain Controllers, Windows Update Servers and other servers which are necessary for the operation of the process control network. The data historians are servers which have the capability of logging all the process data for configured tags of a production plant in real time and storing it for a predefined duration. The stored historical process data are needed for trending, comparisons and analysis of the behavior of processes. Domain

controllers are necessary for security authentication request within the network. Windows update servers are necessary for the deployment of operating system security patches to the machines on the network.

Level 3.5 – This level is called Demilitarized Zone (DMZ), it is the interface between PCN and the Business Network which provides an extra layer of security based on the firewall configured rules. A firewall which serves as a layer of protection separates the PCN from the DMZ. The servers at this level may be connected to other networks including the internet. Terminal servers which is necessary to authenticate remote users who may log on from the business network is housed on the DMZ layer. The Antivirus server necessary for the routine update of the virus definitions and security patch servers are also housed on this level as both may be connected to the internet for real time updates. The firewall at the DMZ level helps to block every other communication traffics from the business network except the traffics that are permitted and the rules are configured on the firewall.

Level 4 – Business Network also known as Corporate Network is the network that organizations use to run their businesses which may include mail servers, domain controllers and business application servers. All the vendors who may be providing support services like troubleshooting and maintenance may connect to the PCN through the internet and the business LAN. Also, some employees who may be at remote locations may connect to shared resources on the PCN from the business network through the terminal server. Organizations technical support team can also view real time data from the process plant through the DMZ. External threats through the internet may exploit the logon details of authorized personnel which include the remote employee or the vendor technical support personnel who may be connecting through the internet to the business network then to the PCN. Adequate controls may have been applied on the firewalls to prevent access of unauthorized person but there is a limitation of exploited account.

3.2.3 Developing a Defense System capable of Detecting False Data Injection Attack, Prediction and Forecasting Model by Algorithm

The applied approach for categorizing the cyber-attack prediction methods in this research based on the defined use case and application for forecasting anomaly detection is as shown in Figure 3.10. Different algorithms were used to simulate the attacks models and it was observed that some algorithms are more accurate in detecting outliers than the others

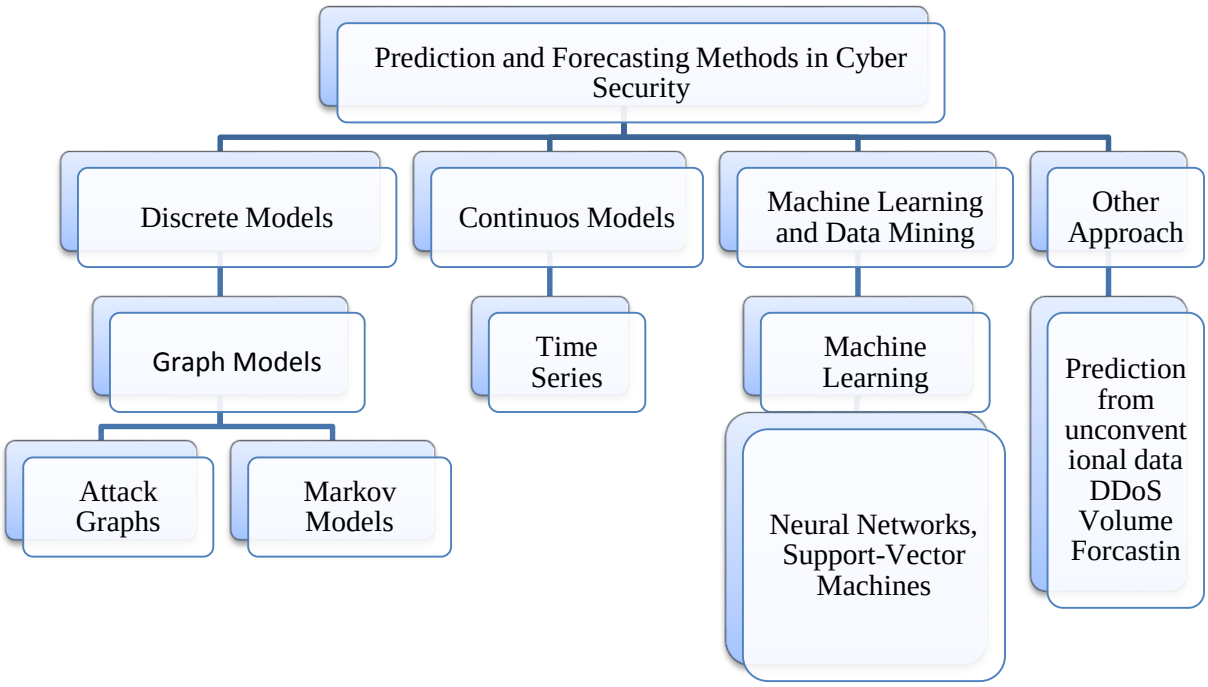


Figure 3.10 Applied Prediction and Forecasting Methods in Cyber Security

3.2.3.1 Unsupervised Machine Learning Algorithms

The study used Unsupervised Machine Learning to study the nature of communication packets from the SCADA system of a Process Control Network in order to be able to

detect anomaly data packets as well as predict Cyber Attacks. The choice of unsupervised machine learning is to be able to find all kinds of unknown patterns in the learning data set since the data structures are of different types and the attacks have been amorphous in nature. The choice of Machine Learning Unsupervised Anomaly Detection Algorithm for the study is to aid identify outliers in a continuous data flow between the field devices and the control systems which is focused developing a proactive method of detecting and preventing cyber-attacks on an Industrial Control System.

i. The attacker's objective to compromise the integrity of the process control data packets thereby changing the response of the final control element.
ii. The process control network is equipped with monitoring devices which is used in controlling the linear time-invariant system.
iii. A compromise of the input data packet influences the resultant output of the system and could be termed anomalies.

Different machine learning algorithms were used while simulating the attack detections and the parameters compared to decide on the best algorithm of choice, they are:

A. Isolation Forest – This is similar in principle to Random Forest and is built on the basis of decision tree algorithm. It identifies anomalies or outliers. It isolates observations by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature.

B. **One Class Support Vector Machine –** An unsupervised algorithm that learns a decision function for novelty detection. It has the ability to classify new data as similar or different to the training set.

C. **K-means Clustering –** This partitions the dataset into k predefined distinct non-overlapping subgroups (clusters) where each data point belongs to only one group. A cluster is a collection of data points aggregated because of certain similarities.

D. **Long Short-Term Memory Models** – This model which mimics the operation of the neurons in the human brain.

E. **K-Nearest Neighbors** – Has the capability to detect anomalies or outliers in a dataset.

F. **Deep Learning algorithms –** group of machine learning algorithms with the principle of Artificial Neural Networks and could be used for supervised, semi-supervised or unsupervised learning applications.

3.2.4 Simulation of the System Models Designed

From Figure 3.5, the 3-phase separator has transmitters sensing input variables such as pressure, level of oil, level of water, oil-water interface level, temperature and so on. The 68,722 real-time dataset was collated from a 3-phase separator over a period of two years between 2018 and 2020. The product of this 3-phase separator is crude oil, gas and water, each of these products are handled separately as outputs from the separator. The Pressure Transmitters PT-1 and PT-2 sensed the process vessel pressure inside the 3-phase separator. While the PT-1 is connected to the process control used to operate the control valve-1 and valve-2 on the gas outlet line., the PT-2 is connected to the emergency shutdown system used to operate the inlet shutdown valve (SDV) and the process Blowdown Valve (BDV) going to the flare. The water level transmitter H-1 is also connected to the process control system and used to operate the valve-5 and valve-6 on the water outlet line, while the oil level transmitter H-2 is connected to the process control system and used to operate the valve-3 and valve-4 on the oil outlet line.

For the period under review, a total of 68,722 data samples of only the pressure variable on the 3-phase separator was used for ease of simulation, modelling, analysis and interpretation. Python 3.0 was used to show a deeper insight to the data collated as shown in Figure 3.11. Imported matplot library into the Python 3 and did a plot of the dataset with 68,722 samples without anomalies as shown in Figure 3.11.

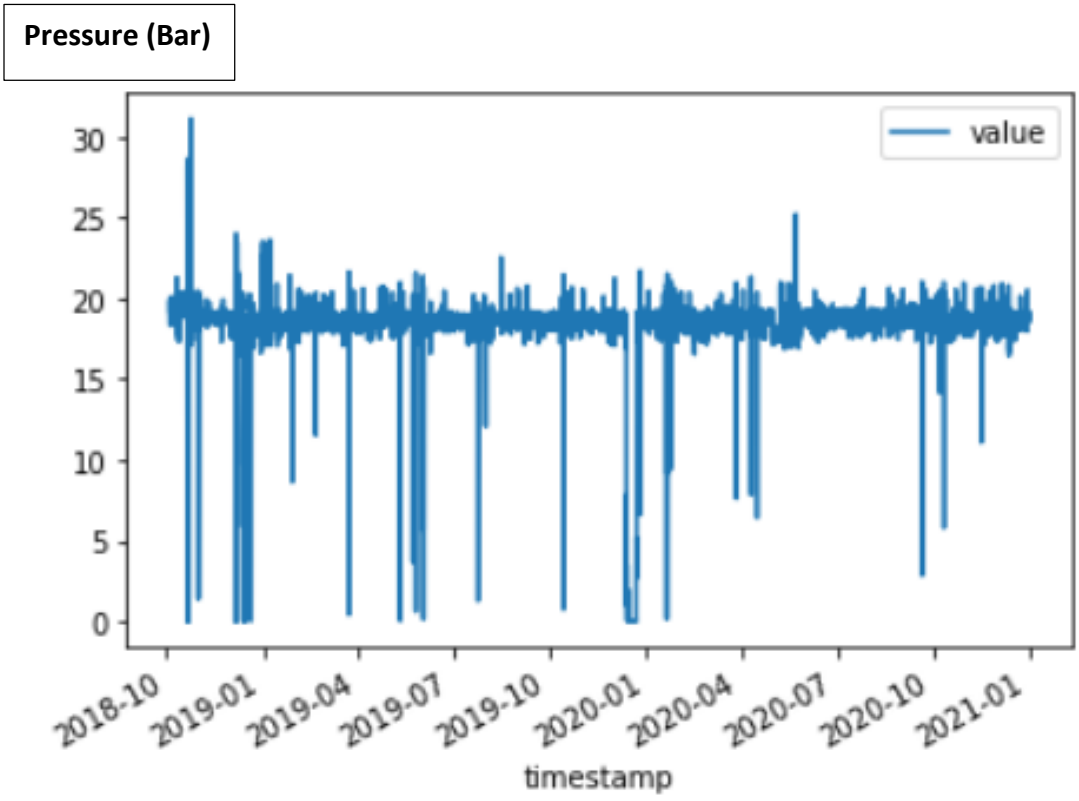


Figure 3.11 Plot of Pressure (value) against Date and Time with 68,722 raw data samples without anomaly

From the plot of Figure 3.11 which was achieved using Matplotlib embedded in python software, it can be deduced that the normal operating pressure values hovered about 18Bar – 20Bar with the maximum value about 30Bar. This implies that for the period of over 2 years between 2018 and 2020, there was no pressure HH trip recorded fom this 3-phase separator. The average pressure was calculated using Python 3 and its value is 18.408501150828183Bar. However, some lower pressure values shown on the plot indicates $p \rightarrow 0$Bar and this can be termed as equipment, process or facility shutdowns. Although, the pressure of the 3-phase separator $p \rightarrow 0$ can be termed an anomaly since it also generates an alarm for the console operator to investigate and resolve.

The design philosophy for the 3-phase separation system is such that all the instruments used in sensing the different process variables will be configured as safety instrumented systems with setpoints to trigger alarm and trip points where alarms may lead to annunciation only but trips will cause shutdowns. The configuration will be

implemented on the controllers through the engineering workstation, it should be audible enough to attract the attention of the plant operator and may be integrated with the plant public address system to ensure all personnel are alerted adequately. The purpose of alarm configuration is to ensure personnel are adequately protected as well the equipment from unsafe process conditions. Immediately the console operator hears the alarm, there is need for the investigation of the alarm and the outcome of such investigation will determine what line of action to take which maybe as trivial as log and monitor alarm or as complex as initiate a controlled plant shutdown. Process alarms which are usually generated where preset setpoint for process variables are exceeded. The preset variable level could be Low, High, Low-Low, High-High as the case may be.

Low (L) Alarm– The Low alarm annunciation will trigger when the value of the measured variable goes below the setpoint for low.

High (H) Alarm – The High alarm annunciation will trigger when the value of the measured variable goes above the setpoint for high.

Low-Low (LL) Trip - The Low-Low trip which includes alarm annunciation and shutdown interlock will be triggered when the value of the measured variable has passed the Low setpoint and reaches a lower preset setpoint which is used as a trip point.

High-High (HH) Trip - The High-High trip which includes alarm annunciation and shutdown interlock will be triggered when the value of the measured variable has passed the High setpoint and reaches a higher preset setpoint which is used as a trip point.

These setpoints are usually configured on the controller. While the Low and the High setpoint can be modified on the HMI by the operator for ease of monitoring and intervention**,** the HH and the LL can only be modified through the EWS as this has higher consequence like shutdowns and cascaded effects.

Considering the redundant pressure transmitter P1 and P2 on the 3-phase separator ranged 0 – 100 Bars, the normal operating range of the transmitters is between 15 Bar and 30Bar. For field values above 15Bar and below 30Bar, there is no alarm and the operation is considered normal.

The setpoint is as follows:

Low (L) Alarm – 15Bar (For values 15Bar and below the controller generates a Low alarm annunciation).

High (H) Alarm – 30Bar (For values 30Bar and above the controller generates a High alarm annunciation.

Low-Low (LL) Trip – 5Bar (At 5Bar the controller generates a Low-Low annunciation alarm as well as a shutdown trip interlock).

High-High (HH) Trip – 50Bar (At 50Bar the controller generates a High-High annunciation alarm as well as a shutdown trip interlock).

Shutdown trip interlock are usually latched which means they will usually remain in the alarm state even if the measured value returns to a safe state until the operator investigates the alarm, restores the signal to a safe state and acknowledges the alarm on the HMI. The alarm condition must be restored to a normal condition before an acknowledgement can be successfully executed.

3.2.4.1 Process Setpoints for Devices on the 3-phase Separator

For a safe operation of the crude oil production plant there is need to set operating boundaries and setpoints, as this aids in the determination of process upsets and normal operating parameters. The Table 3.2 shows the setpoints used in the safe operation of the 3-phase separator. The detection of anomalies and possible prevention of cyber-attacks with its automation will be made possible if there is a clear distinction between the different pressure values and categorization to avoid false alarms and unnecessary system downtime. The experienced console operator must understand the different

categorizations of annunciations like Low Alarm, Medium Alarms, High Alarms to be able to distinguish between equipment downtime, system downtime, facility downtime and a real cyber-attack, and possible escalation procedures. For equipment downtime, system downtime and facility downtime there may be need for further investigation by the console operator and possible resolution. However, any alarm that mimics cyber-attacks will require full escalation and the procedures for investigation and recovery must be followed at all times.

Table 3.2 Process Control Transmitter Setpoints

Description	**Pressure Transmitters PT-1 (Bar)**	**Pressure Transmitters PT-2 (Bar)**	**Water Level Transmitter H-1 (%)**	**Oil Level Transmitter H-2 (%)**
Normal Operation	15 - 30	15 - 30	30 - 60	30 - 60
Low Alarm	< 15	< 15	< 30	< 30
Low-Low Alarm	< 5	< 5	< 20	< 20
High Alarm	> 30	> 30	> 60	> 60
High-High Alarm	> 50	> 50	> 70	> 70

3.2.4.2 Control Narratives for the 3-Phase Separator Instruments

The Table 3.3 shows the cause-and-effect chart for the pressure transmitter PT-1 on the 3-phase separator as configured on the process control system, while Table 3.4 shows the cause-and-effect chart for the pressure transmitter PT-2 on the 3-phase separator as configured on the emergency shutdown system. Table 3.5 shows the cause-and-effect chart for the water level transmitter H1 on the 3-phase separator as configured on the process control system, while Table 3.6 shows the cause-and-effect chart for the oil level transmitter H2 on the 3-phase separator as configured on the process control system.

Table 3.3 Pressure Transmitter PT-1 Cause and Effect Chart for Process Control

Description	**Pressure Value (Bar)**	**Process Operation Valve Position**	**Comment**
Normal Operation	15 – 30	Valve-1 = Open, Valve-2 = Open SDV = Open BDV = Closed	Normal Operation
Pressure Alarm Low (PAL)	< 15	Valve-1 = Closed Valve 2 = Closed SDV = Open BDV = Closed	Console Operation investigates cause of alarm (LAL)
Pressure High Alarm (PAH)	> 30	Valve-1 = Closed Valve 2 = Closed SDV = Open BDV = Closed	Console Operation investigates cause of alarm (HAL)
Pressure Low-Low Alarm (PLL)	< 5	Valve-1 = Closed Valve 2 = Closed SDV = Closed BDV = Open	Trip condition Process Shutdown And Gas Blowdown (LSD)
Pressure High-High Alarm (PHH)	> 50	Valve-1 = Closed Valve 2 = Closed SDV = Closed BDV = Open	Trip condition Process Shutdown And Gas Blowdown (HSD)

Table 3.4 Pressure Transmitter PT-2 Cause and Effect Chart for Emergency Shutdown

Description	**Pressure Value (Bar)**	**Process Operation Valve Position**	**Comment**
Normal Operation	15 – 30	SDV = Open BDV = Closed	Normal Operation
Pressure Alarm Low (PAL)	< 15	SDV = Open BDV = Closed	Low Alarm (LAL)

Pressure High Alarm (PAH)	> 30	SDV = Open BDV = Closed	High Alarm (HAL)
Pressure Low-Low Alarm (PLL)	< 5	SDV = Closed BDV = Open	Trip condition, Process Shutdown and Gas Blowdown (LSD)
Pressure High-High Alarm (PHH)	> 50	SDV = Closed BDV = Open	Trip condition, Process Shutdown and Gas Blowdown (HSD)

Table 3.5 Water Level Transmitter H-1 Cause and Effect Chart for Process Control

Description	**Level Value (%)**	**Process Operation Valve Position**	**Comment**
Normal Operation	30 – 60	Valve-5 = Open, Valve-6 = Open SDV = Open BDV = Closed	Normal Operation
Level Alarm Low (H1_AL)	< 30	Valve-5 = Closed Valve 6 = Closed SDV = Open BDV = Closed	Console Operator investigates and resolve cause of alarm (LAL)
Level High Alarm (H1_AH)	> 60	Valve-5 = Closed Valve 6 = Closed SDV = Open BDV = Closed	Console Operator investigates and resolve cause of alarm (HAL)
Level Low-Low Alarm (H1_LL)	< 20	Valve-5 = Closed Valve 6 = Closed SDV = Closed BDV = Open	Trip condition Process Shutdown And Gas Blowdown (LSD)
Level High-High Alarm (H1_HH)	> 70	Valve-5 = Closed Valve 6 = Closed SDV = Closed BDV = Open	Trip condition Process Shutdown And Gas Blowdown (HSD)

Table 3.6 Oil Level Transmitter H-2 Cause and Effect Chart for Process Control

Description	**Level Value (%)**	**Process Operation Valve Position**	**Comment**
Normal Operation	30 – 60	Valve-3 = Open, Valve-4 = Open SDV = Open BDV = Closed	Normal Operation
Level Alarm Low (H2_AL)	< 30	Valve-3 = Closed Valve 4 = Closed SDV = Open BDV = Closed	Console Operator investigates and resolve cause of alarm (LAL)
Level High Alarm (H2_AH)	> 60	Valve-3 = Closed Valve 4 = Closed SDV = Open BDV = Closed	Console Operator investigates and resolve cause of alarm (HAL)
Level Low-Low Alarm (H2_LL)	< 20	Valve-3 = Closed Valve 4 = Closed SDV = Closed BDV = Open	Trip condition Process Shutdown And Gas Blowdown (LSD)
Level High-High Alarm (H2_HH)	> 70	Valve-3 = Closed Valve 4 = Closed SDV = Closed BDV = Open	Trip condition Process Shutdown And Gas Blowdown (HSD)

3.2.5 Validation of the Developed Models for Attacks Test Cases

To be able to validate the developed models for the different test cases of attacks simulation, the PLC logic was used for the first test case and the different conditions for the input variable pressure PT-1 was explored. This outcome is applicable to all the other input variables as the case may be. The different machine learning algorithm test cases were also simulated using the pressure dataset and other existing datasets.

3.2.5.1 Validation of Developed Models for Attacks Test Cases using PLC Logic

Considering Figure 3.8 showing the interaction between the input variables, process values (PV), manipulated variable (MV), disturbance variable (DV) and the controller setpoint with respect to the expected output (OP). It has been shown that any form of attack on any aspect of the control system for this multivariable system will impact the output of the system which will in turn lead to an impact to people, asset and environment with a huge cost implication. To test the achieved results and verify the impact of disturbance in the control system of the 3-phase separator, a PLC logic was developed using Allen Bradley RSLogix 5000 PLC Emulator which showed the impact of the changes in the inputs and the resultant effect on the output. With the PLC emulator system as setup in Figure 3.4, the inputs from the different transmitters serves as the analog inputs as shown in logic rungs of Figure 3.12, while Figures 3.13 and 3.14 showed the output conditions (valve positions) in response to the analog inputs.

Case 1 - Normal Operation: In this case, pressure in the vessel as shown on the transmitters PT-1 and PT-2 was 20Bar. Also, the value of the water level transmitter is 49% while that of the oil level transmitter is 31%. These values are within the normal operating range and all the outlet valves for gas outlet line, oil outlet line and water outlet line are in open positions respectively. From Figure 3.13, during normal operations all outlet valves are in open position. The Shutdown valve is Open during normal operation while the Blowdown valve is closed, as shown in Figure 3.14. This test case results are applicable to the inputs from the water lines and the oil lines respectively. Figures 3.12, 3.13, 3.14 shows the representation of the different process conditions (Normal Operation) as shown in Table 3.3.

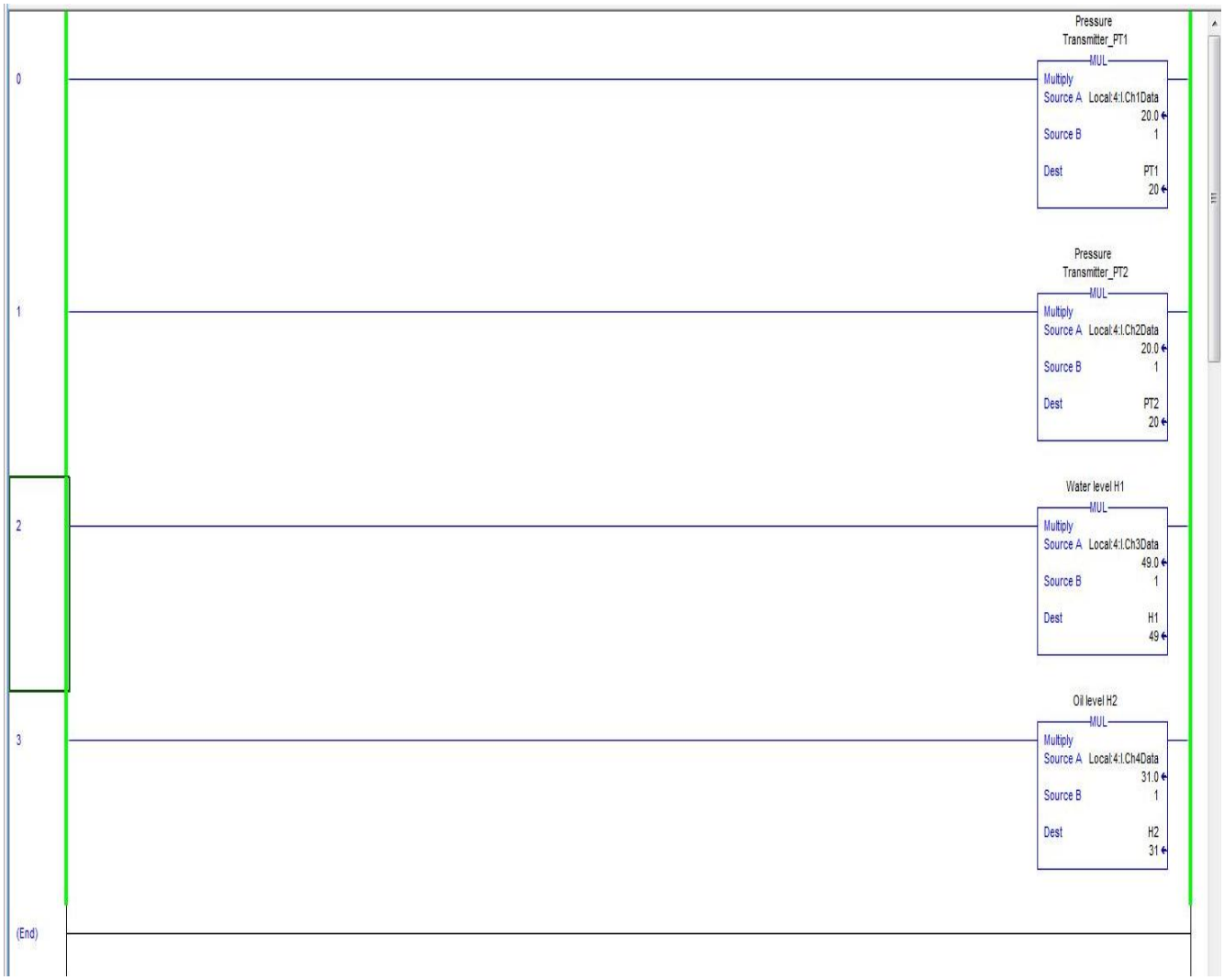


Figure 3.12 PLC Logic of Analog Inputs during Normal Operation PT1 = 20Bar

Figure 3.13 PLC Logic showing the outlet valve sequence during Normal Operation

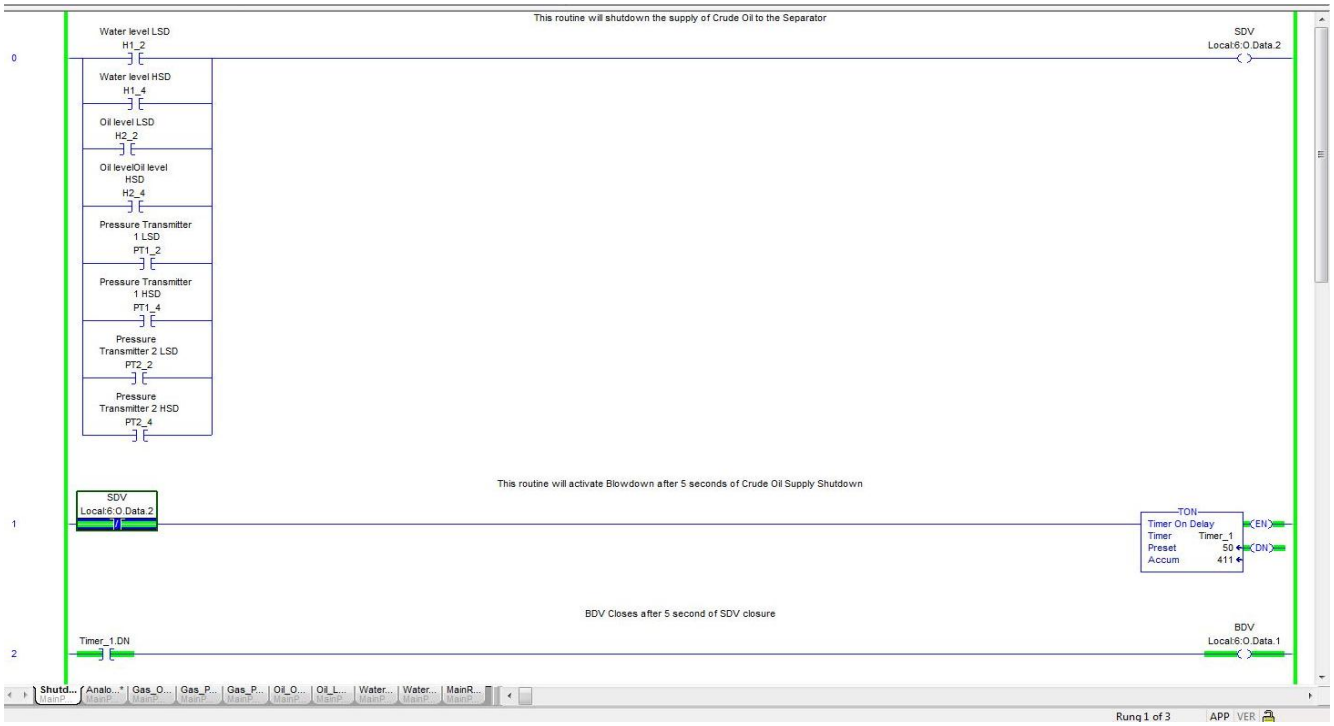


Figure 3.14 PLC Logic for Shutdown/Blowdown valve sequence during Normal Operation

Case 2 – Low Alarm Active Condition: When any of the input variables drops in value to the point of Low Alarm. In this case, the pressure value in the vessel drops to 14Bar which is the < 15 Bar setpoint while the water level remains 49% and oil level remains 31%. This condition generates a low alarm active (LAL) as shown in Figure 3.15 and Figure 3.16, which triggers a console annunciation for Pressure PT-1 Low Alarm for the console operator to investigate and resolve. Meanwhile, based on the cause-and-effect chart of Table 3.3, this low alarm active condition causes the controller to send a close command to the respective gas product outlet valve, as shown in Figure 3.17. However, the Shutdown valve still remains Open, while the Blowdown valve remains closed, as shown in Figure 3.14. This test case results are applicable to the inputs from the water lines and the oil lines respectively. Figures 3.15, 3.16, 3.17 shows the representation of the different process conditions (Low Alarm Conditions) as shown in Table 3.3.

Figure 3.15 PLC Logic for Analog Pressure Low Alarm for PT1=14Bar

Figure 3.16 Logic for Pressure Low Alarm active (LAL Condition) for PT1

Figure 3.17 Logic configuration for Gas Outlet Line Valve closed – LAL condition

Case 3 – High Alarm Active Condition: When any of the input variables increases in value to the point of High Alarm. In this case, the pressure value in the vessel increases to 34Bar which is the > 30Bar setpoint for high alarm (HAL) while the water level remains 49% and oil level remains 31%. This condition generates a high alarm active as shown in Figure 3.17, which triggers a console annunciation for Pressure PT-1 High Alarm for the console operator to investigate and resolve. Meanwhile, based on the cause-and-effect chart of Table 3.3, this high alarm active condition causes the controller to send a close command to the respective gas product outlet valve, as shown in Figure 3.20. However, the Shutdown valve still remains Open, while the Blowdown valve remains closed, as shown in Figure 3.14. This test case results are applicable to the inputs from the water lines and the oil lines respectively. Figures 3.18, 3.19, 3.20 shows the representation of the different process conditions (High Alarm Active Conditions) as shown in Table 3.3.

Figure 3.18 PLC Logic for Analog Pressure High Alarm for PT1=34Bar

Figure 3.19 Logic for Pressure High Alarm active (HAL Condition) for PT1

Figure 3.20 Logic configuration for Gas Outlet Line Valve closed – HAL condition

Case 4 – Low-Low Shutdown Active Condition: When any of the input variables which includes pressure of gas PT-1 or PT-2 or both, level of water H-1 or level of oil H-2 or both drops in value to the point of shutdown trip point (LSD) as shown on the cause-and-effect chart, the shutdown valve will trip closed while the blowdown valve will trip open. In this case, the pressure in the vessel drops to 3bar which is < 5 Bar, based on the cause-and-effect chart of Table 3.3, the actuator of the shutdown valve receives a valve close command while the actuator of the blowdown valve receives an open command, this is as shown in Figure 3.23. However, the Logic configuration for Gas Outlet Line Valve remains closed as shown in the LAL condition of Figure 3.17. The LL condition triggers a critical alarm on the console as the production process will be shutdown and full gas blowdown, the BDV opens 5 seconds after the of SDV has closed. This test case results are applicable to the inputs from the water lines and the oil lines respectively. Figures 3.21, 3.22, 3.23 shows the representation of the different process conditions (Low-Low Shutdown Conditions) as shown in Table 3.3.

Figure 3.21 PLC Logic for Analog Pressure Low-Low Trip for PT1=3Bar

Figure 3.22 Logic for Pressure Low-Low Alarm active (LSD Condition) for PT1

Figure 3.23 Logic for Shutdown/Blowdown valves during Low-Low active Condition

Case 5 – High-High Shutdown Active Condition: When any of the input variables which includes pressure of gas PT-1 or PT-2 or both, level of water H-1 or level of oil H-2 or both increases in value to the point of shutdown trip point (HSD) as shown on the cause-and-effect chart, the shutdown valve will trip closed while the blowdown valve will trip open. In this case, the pressure in the vessel has increased to 52bar which is > 50 Bar, based on the cause-and-effect chart of Table 3.3, the actuator of the shutdown valve receives a valve close command while the actuator of the blowdown valve receives an open command, this is as shown in Figure 3.26. However, the Logic configuration for Gas Outlet Line Valve remains closed as shown in the HAL condition of Figure 3.20. The HH condition triggers a critical alarm on the console as the production process will be shutdown and full gas blowdown, the BDV opens 5 seconds after the of SDV has closed. This test case results are applicable to the inputs from the water lines and the oil lines respectively. Figures 3.24, 3.25, 3.26 shows the representation of the (Sutdown/Blowdown Conditions) as shown in Table 3.4.

Figure 3.24 PLC Logic for Analog Pressure High-High Trip for PT1=52Bar

Figure 3.25 Logic for Pressure High-High Alarm active (HSD Condition) for PT1

Figure 3.26 Logic for Shutdown/Blowdown valves during High-High active Condition

3.2.5.2 Validation of Developed Models using Machine Learning Test Cases

Based on the collated dataset as plotted in Figure 3.11, it can be deduced that no extreme high pressure values from the dataset was identified, it will be necessary to have a robust system which is capable of detecting all forms of anomalies which include very high and very low anomalies. To be able to model and simulate anomalies of very high values using the collated data stream from the 3-phase separator system, values were injected at specific dates to show extremely high pressure values in the 3-phase separator and this is typically a case of Man-in-the-Middle Attack (MitM).

The details of the injected anomalies:

1. From 23:06 hours December 25, 2018 to 04:23 hours December 26, 2018 an extreme high pressure value 98.99999Bar was injected to the dataset.
2. From 23:03 hours December 31, 2019 to 03:29 hours January 1, 2020 an extreme high pressure value 97.88888Bar was injected to the dataset.
3. The 3-phase separator pressure transmitter failed on the September 9, 2019 at 12:14 hours to an extreme high pressure valuue of 99.99999Bar. The transmitter was later fixed and returned back to service at 14:40 hours, same day.
4. There was a HH process upset whiich caused a trip of the production plant at 10:14 hours on August 31, 2020 and this jerked up the pressure value in thhe 3-phase separator to 51.15665Bar. The process upset was resolved and the faciility returned back to normal operation at 11:35 hours same day.

After the injection of the anomalies into the dataset with 68,722 data samples, Imported matplot library into the Python 3 and did a plot of the dataset with anomalies as shown in Figure 3.27. Critical analysis of the SCADA Pressure data with 68,722 data samples showed anomalies which were identified. These anomalies were categorized to be cyber attacks, process upsets and production downtimes. These extreme HH condition was also introduced to confirm the ability of the models to thouroughly identify cyber attack scenarios. Consider the plot of the raw dataset of 68,722 collated from the real-time

SCADA plant as shown in figure 3.11. With the attack from malicious intruders as injected and shown in Figure 3.27, it was observed that the maximum range shifted from about 30Bar to about 100Bar which has crossed the shutdown value and will cause a closure of the SDV and an opening of BDV as shown in the cause-and-effect chart of Table 3.3.

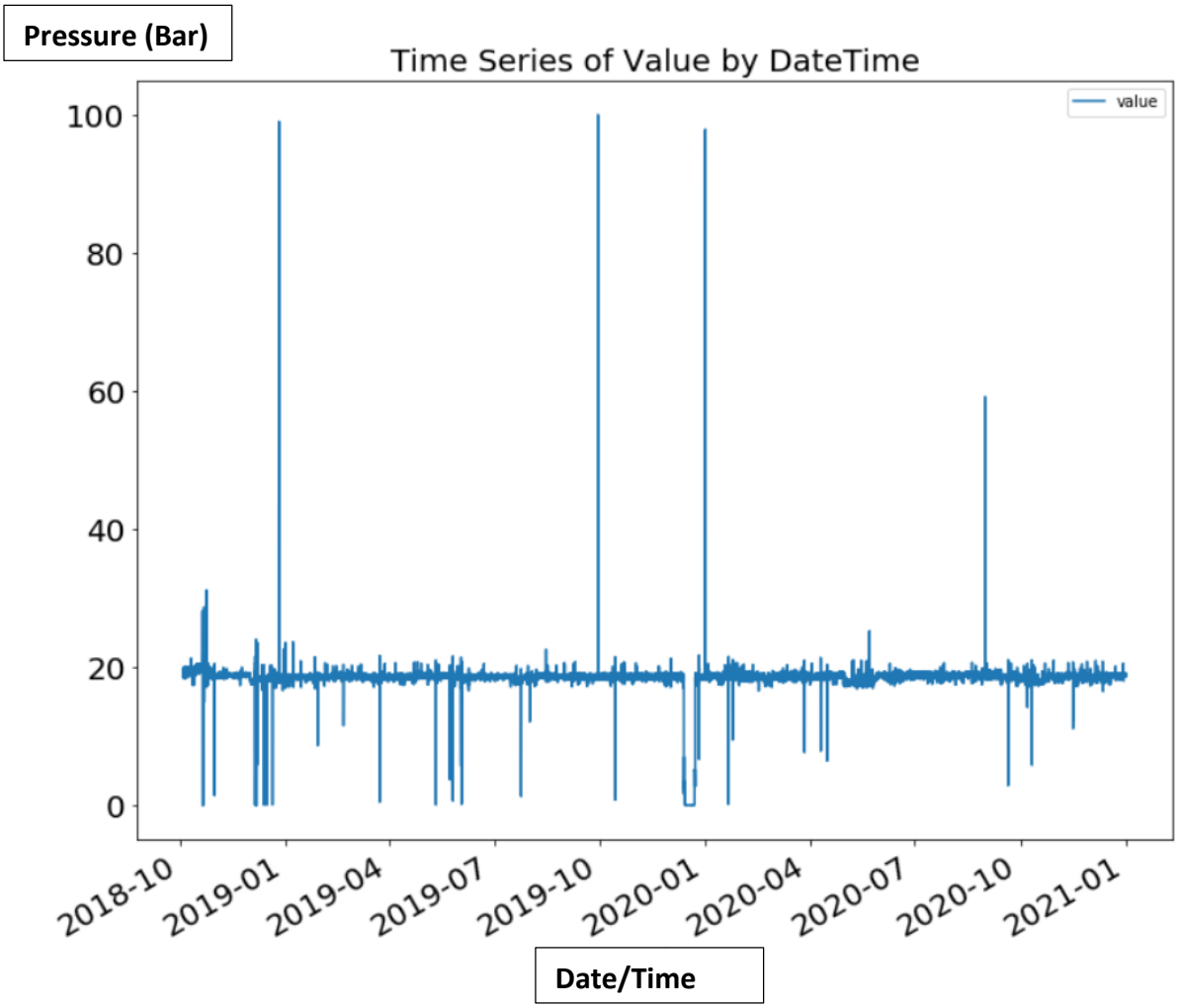


Figure 3.27 Plot of Pressure (value) against Date and Time, 68,722 data samples with anomalies injected.

Data Description

The Figure 3.27 shows the plot of 68,722 data samples with injected extreme values. After thhe injection of the anomalies into the dataset with 68,722 data samples, Imported matplot library into the Python 3 and did a plot of the dataset with anomalies as shown in Figure 3.27. Critical analysis of the SCADA Pressure data with 68,722 data

samples showed anomalies which were identified. These anomalies were categorized to be cyber attacks, process upsets and production downtimes. The extreme HH condition introduced helped verify the ability of the models to thouroughly identify cyber attack scenarios. From the plot of Figure 3.27, it can be deduced that the normal pressure values hovered about 18Bar – 20Bar with the maximum value about 99Bar. The average pressure was calculated using Python 3 and its value is 18.46955747413078Bar. With the injection of the anomalies as shown Figure 3.27, the dataset lies between -0.018000Bar minimum value and 99.99999Bar maximum value with a total of 68,722 values collected over a period from midnight of October 4, 2018 till midnight of December 31, 2020. Other variables like height of water h_1, height of oil h_2 and Temperature was also collated but for the ease of simulation, data presentation and computation, a choice was made to use only the pressure value for simulation.

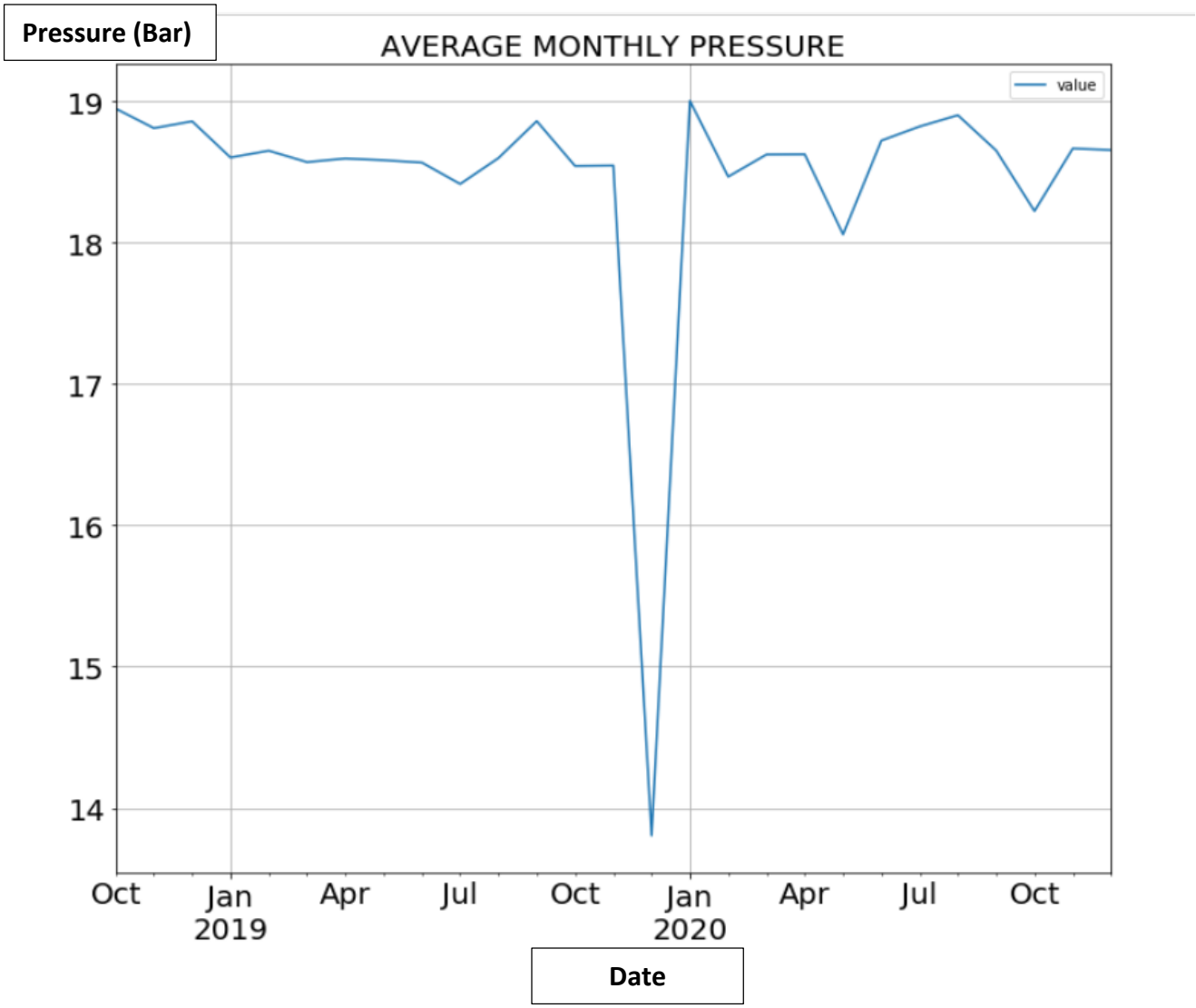


Figure 3.28 Plot of Average Monthly Pressure (value) against Date and Time

As can be seen in Figure 3.28, the average monthly pressure for the period under review lies between 18.0Bar and 18.8Bar with the exception of Dec 2019 when there was a deep in the trend which could be as a result of possible shutdown. The minimum pressure values at the end of each month shows that the year 2019 recorded the most minimum pressure values as captured by the plot of Figure 3.28, the months of March, May, June, July, October and December till January 2020. December 2019 had values close to 0Bar.

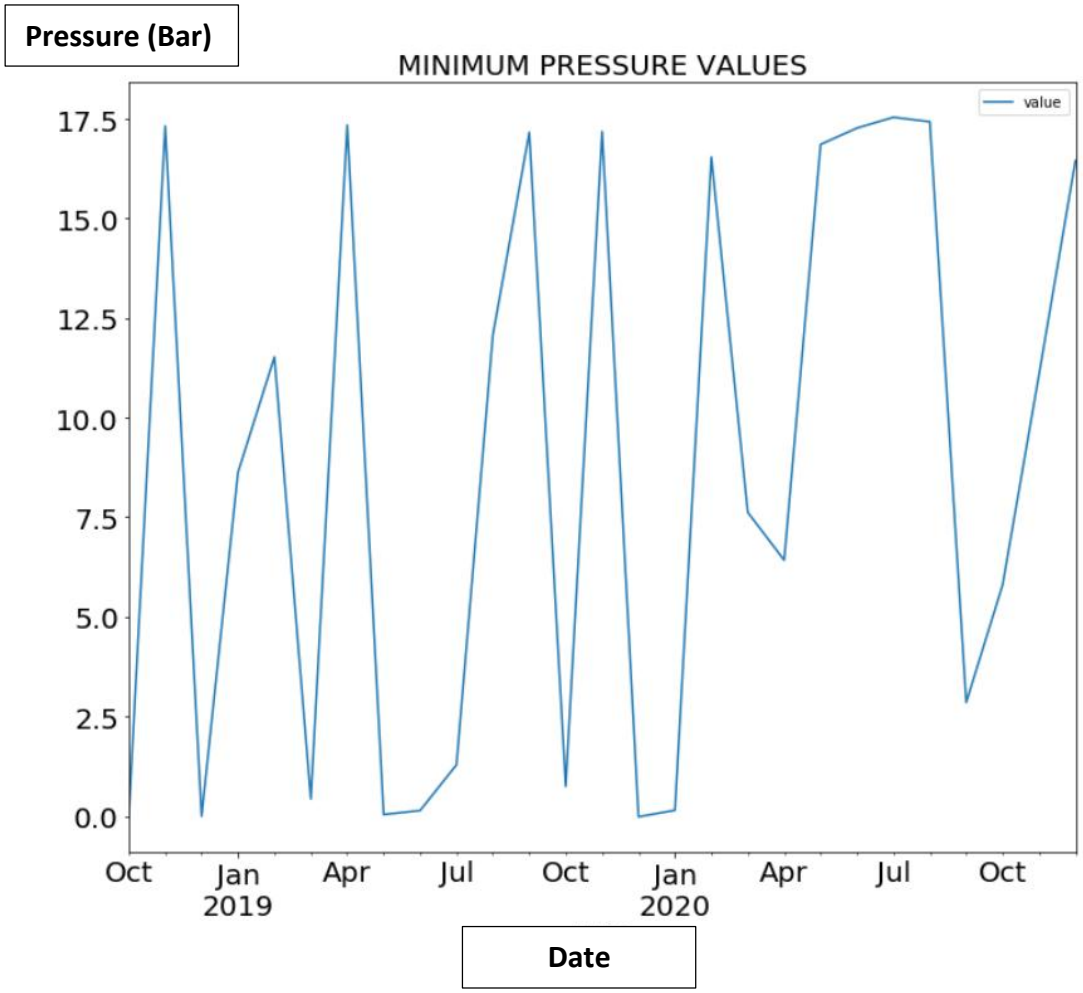


Figure 3.29 Plot of Minimum Pressure (value) against Date and Time

From Figure 3.29, Plotting the maximum monthly pressure values showed that there are few times we had the values rise very high even beyond the HH setpoint which is 50Bar. A plot of the date and Time against the pressure is shown in Figure 3.29 and this clearly shows the high anomalies which will need the investigation of the console operator to determine the possible causes even to escalation points.

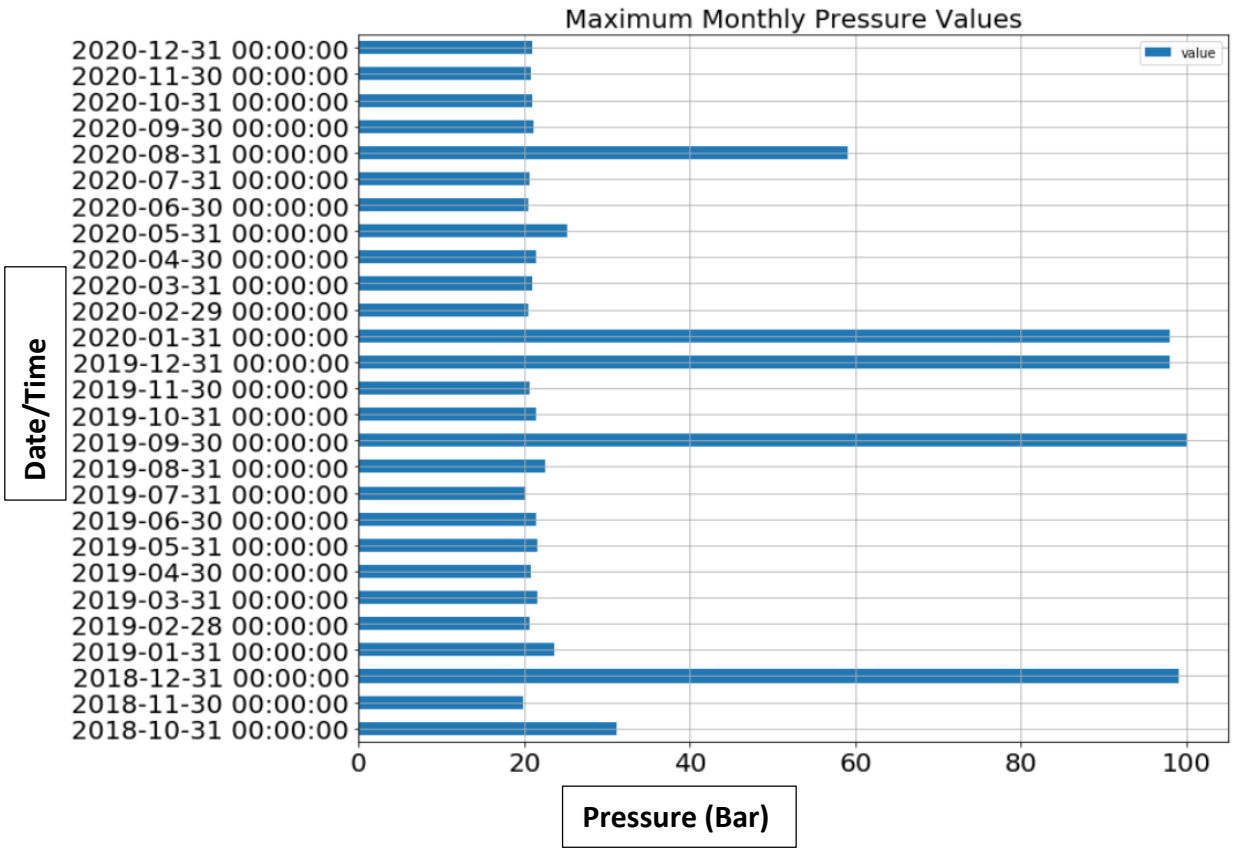


Figure 3.30 Plot of Date and Time against Maximum Monthly Pressure (value)

From Figure 3.30, it can be seen that abnormal maximum pressure values were detected in August 2020 with a high value of 58.0Bar, also in December 2019 and January 2020 with values a s high as 98.0Bar. December 2018 and September 2019 detected the highest values about 99.0Bar. The mentioned dates are considered outliers which are anomalies when compared to the average pressure readings.

To validate the outcome of this research, a thorough comparison was made between the results achieved and that of the other researchers who used WUSTL and ORNL datasets (*Tommy Morris - Industrial Control System (ICS) Cyber Attack Datasets*, n.d.)(Teixeira et al., 2018) in the training of their models. It is important to state that the false alarm rate recorded with the SCADA was zero as compared to other datasets used by other researchers, this is as shown in table 3.7.

Table 3.7 Comparative Analysis of Top Performing Machine Learning Classifiers with the SCADA dataset

COMPARATIVE ANALYSIS OF TOP PERFORMING MACHINE LEARNING CLASSIFIERS			
Performance Metrics	**SCADA PRESSURE DATASET**	**WUSTL-SCADA-2018 DATASET**	**ORNL POWER GRID DATASET**
Accuracy (%)	100	100	95.1
False Alarm Rate (FAR) (#)	0	412	241
Prediction Speed (obs/sec)	1000000	4100000	2500
Computation Time (s)	0.45488	5.6605	4.8021
Model	Coarse Tree	Medium Tree	Bagged Tree

The three main Datasets used in this research are:

1. WUSTL-IIoT-2018
2. ICS-SCADA ORNL
3. Real-time SCADA dataset

CHAPTER FOUR

RESULTS AND DISCUSSION

4.1 Results

The results obtained from this research are presented as follows:

4.1.1 Results from the Design of a Secured Process Control Network Architecture

The Figure 4.1 shows the designed Distributed Process Control Network under data-injection external attack. The newly introduced security server which houses the models for cyber-attack detection and with the capability of monitoring the real time data exchange between the input/output devices and the controllers is integrated at the Layer 2 of the PCN alongside the Engineering Workstations with direct access to the data exchange between the controllers and the field devices. The security server will be able to detect and prevent data injection and annunciate at the HMI for the console operator and the network administrators for detailed investigation and resolute actions to be taken.

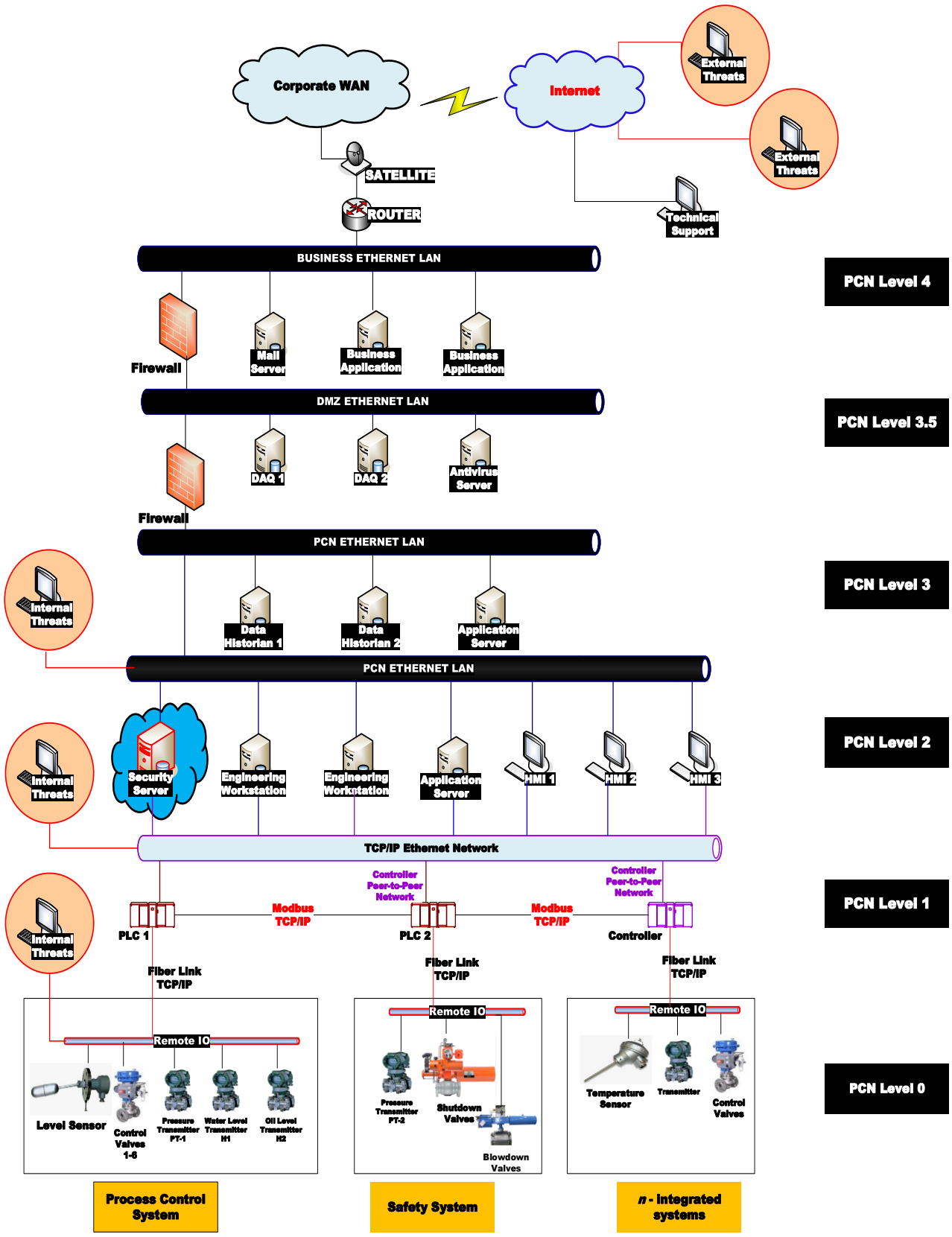

Corporate WAN
Internet
External Threats
External Threats
SATELLITE
ROUTER
Technical Support
BUSINESS ETHERNET LAN
PCN Level 4
Firewall
Mail Server
Business Application
Business Application
DMZ ETHERNET LAN
PCN Level 3.5
DAQ 1
DAQ 2
Antivirus Server
Firewall
PCN ETHERNET LAN
PCN Level 3
Internal Threats
Data Historian 1
Data Historian 2
Application Server
PCN ETHERNET LAN
PCN Level 2
Internal Threats
Security Server
Engineering Workstation
Engineering Workstation
Application Server
HMI 1
HMI 2
HMI 3
TCP/IP Ethernet Network
Internal Threats
Controller Peer-to-Peer Network
Controller Peer-to-Peer Network
PCN Level 1
PLC 1
Modbus TCP/IP
PLC 2
Modbus TCP/IP
Controller
Fiber Link TCP/IP
Fiber Link TCP/IP
Fiber Link TCP/IP
Remote IO
Remote IO
Remote IO
Level Sensor
Control Valves 1-6
Pressure Transmitter PT-1
Water Level Transmitter H1
Oil Level Transmitter H2
Pressure Transmitter PT-2
Shutdown Valves
Blowdown Valves
Temperature Sensor
Transmitter
Control Valves
PCN Level 0
Process Control System
Safety System
n - Integrated systems

Figure 4.1 Distributed Process Control Network with Real Time Security

The integration of the security server will aid in the detection of insider threats which may result to anomalous data exchange between the sensors, transmitters, controllers and the final control elements in the plant operation. All other external threats such as internet threats which may come as a result of interface between the PCN and the Business network can also be detected and prevented by this system. The 3-phase separator with pressure transmitters HH setpoints set to 50 Bar, there is a risk of catastrophe if the system fails to detect early of extreme pressure values which can either cause a collapse or explosion of the vessels. For extreme high values, the pressure safety valve on the vessel which is set at 78 Bar is the next line of protection. The pressure safety valve (PSV) is designed to prevent excessive pressure buildup in the vessel, it reliefs the excess pressure by opening the valve discharge and closes back as soon as normal condition is restored. The PSV is a mechanical device which is known as the last line of defense of pressurized equipment which implies that if all the automated systems fail, it will help to rescue the pressure buildup and its failure to act could lead to catastrophic failure which may include explosion due to over pressurization.

4.1.2 Results from Intrusion Detection System in PCN using Machine Learning

The model implementation process is categorized into training, testing and validation. The 68,722 real-time data used in this work is sub divided into 60% for training of the models, 25% for validation of the models and 15% for testing phase of the models. The various datasets were imported into the machine learning algorithms and a fivefold cross validation was implemented. The performance of the models was verified using testing set.

The Performance Metrics (PM) used in evaluating the different models which serves as a guide in the selection of the best model include:

(i) Accuracy of the model,

(ii) Receiver Operating Characteristics (ROC),

(iii) Confusion Matrix (CM),

(iv) Training time,

(v) The model's Mis-Classification Error (MCE),

(vi) Prediction Speed

The goal of this research is to detect anomalies; hence all outliers are expected to be detected by the designed system as well as annunciate to attract the attention of the console operator, who will use his knowledge of the system to define if there is need to carry out further investigation or to escalate. From section 3.2.4.1, it was stated that LL=5Bar, L=15Bar, H=30Bar, HH=50Bar. This implies that the normal operating pressure range for this 3-phase separator is 15 – 30Bar. Pressure values outside the operating range is considered an outlier and needs the console operator's intervention to address. Using the collated 68,722 dataset, different algorithms were experimented with to be able to identify best choice of algorithm.

4.1.2.1 Isolation Forest Algorithm Model for Anomaly Detection

This is an unsupervised machine learning algorithm which has the capability of detecting outliers from a dataset. It is an algorithm for anomaly detection which detects anomalies by modeling isolated data points or outliers rather than modeling the normal points. The outliers are isolated by randomly selecting a feature from the given set of features and then randomly selecting a split value between the maximum and minimum values of that feature. Applying the Isolation Forest Algorithm to our Dataset, with the contamination parameter set to the value of 0.1, the algorithm was able to detect all extremely high and low values as shown in Figure 4.2. The contamination parameter refers to the expected outliers in the dataset which is used when fitting to define the threshold on the dataset.

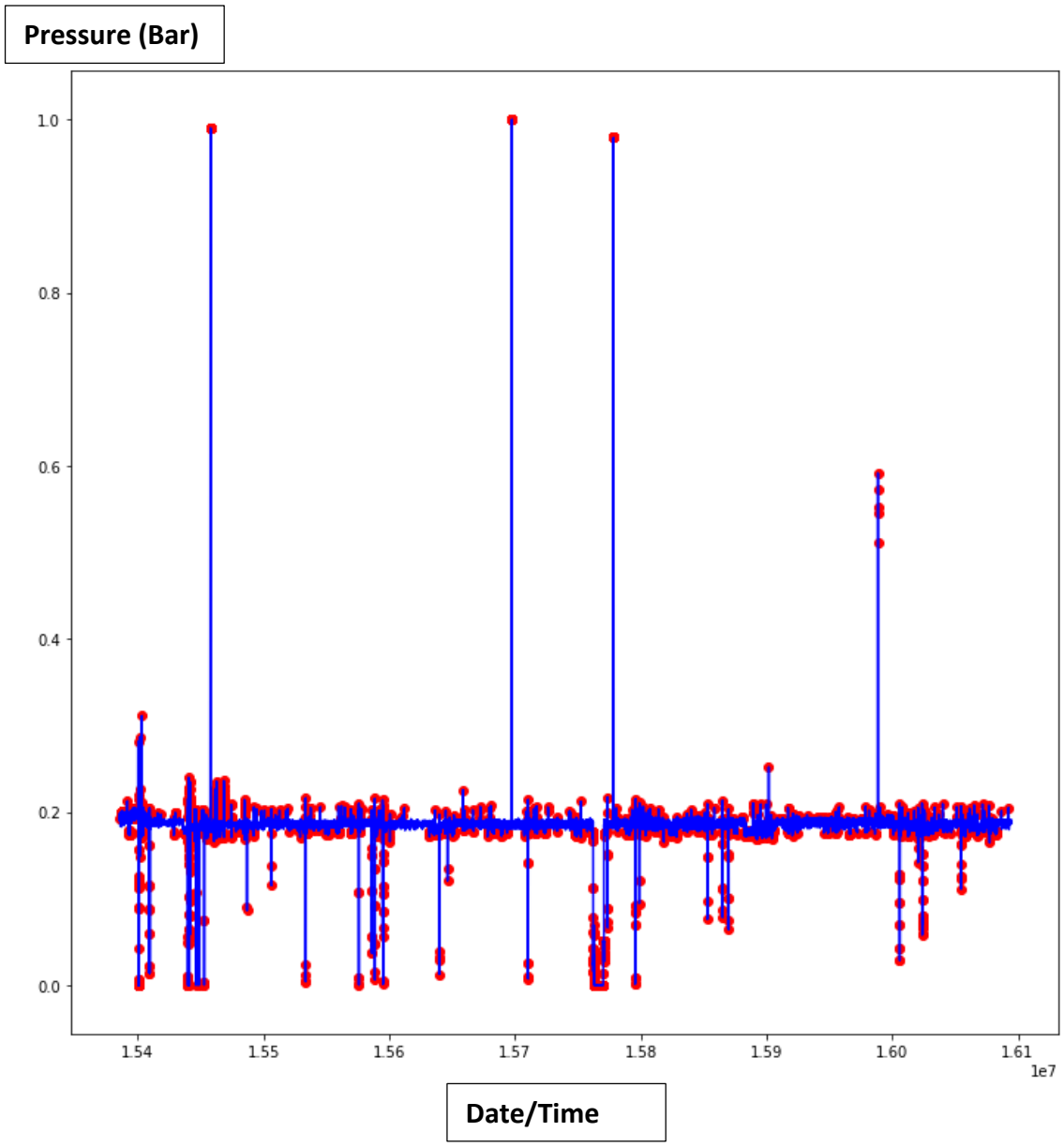


Figure 4.2 Plot of Isolation Forest Algorithm with contamination parameter set to 0.1 (y-axis is the pressure while the x-axis is the Date and Time)

The Figure 4.2 shows the plot of pressure against Date/Time using Isolation Forest Algorithm as detailed in Appendix D. The setting of the contamination parameter to 0.1 implies that 10 percent of the data was set as outlier and this resulted to 100 percent detection of all anomalies with 6837 anomalies detected out of 68,722 values in the

dataset. This setting could be termed too sensitive, as it detected changes in the pressure values.

With a change on the outlier value to 0.01 which is 1 percent of the entire dataset, the algorithm was able to generate outliers at the extreme low-pressure values. The plot is as shown in Figure 4.3.

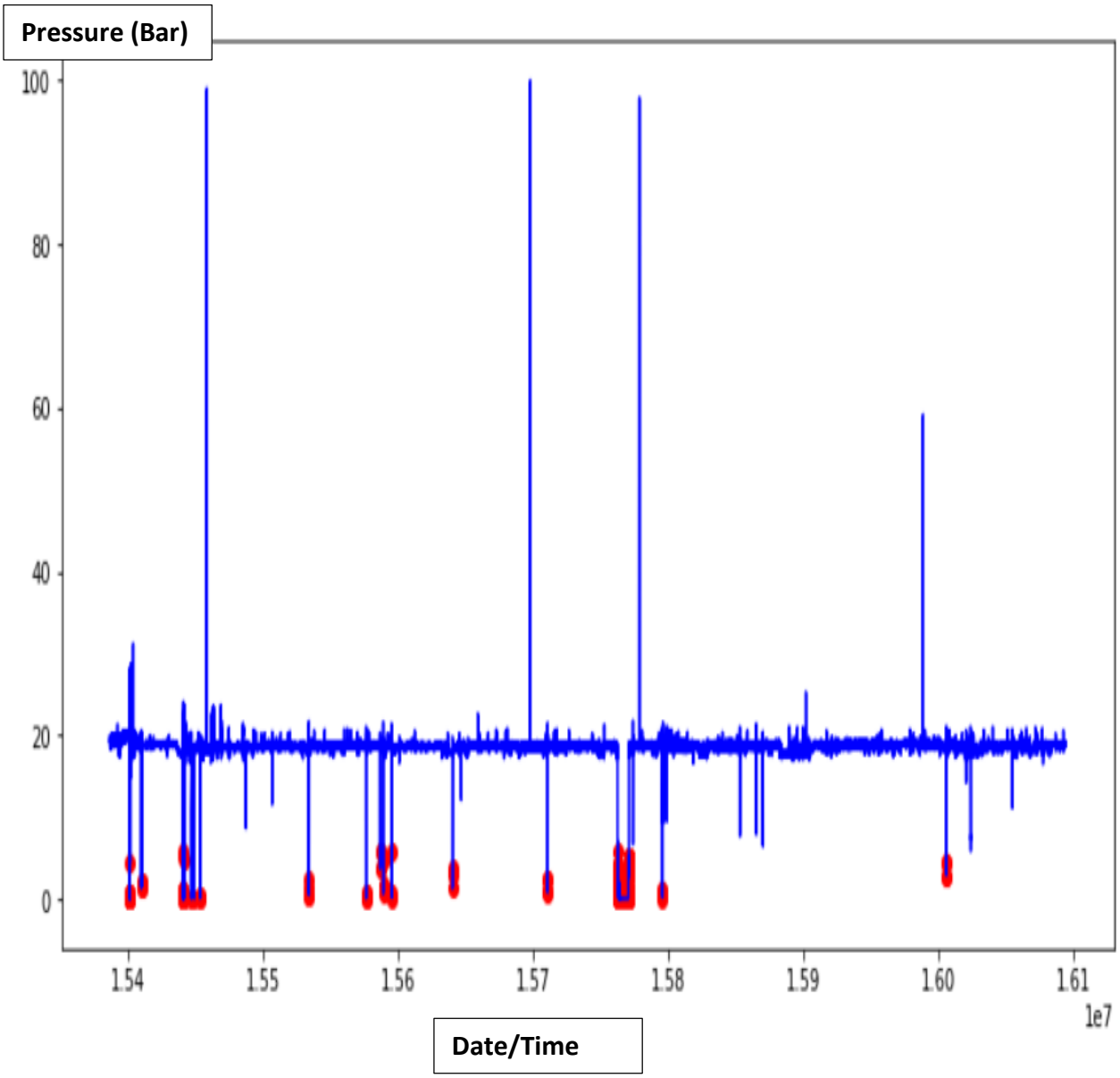


Figure 4.3 Plot of Isolation Forest Algorithm anomaly detection with 68,722 dataset, contamination parameter set to 0.01 (y-axis is the pressure while the x-axis is the Date and Time)

The Figure 4.3 shows the plot of pressure against Date/Time using Isolation Forest Algorithm as detailed in Appendix D.

4.1.2.2 Long Short-Term Memory Models for Outlier Detection

Long Short-Term Memory (LSTM) Models is based on Artificial Neural Network (ANN) principles with application in Artificial Intelligence and Deep Learning. LSTM has feedback as compared to other standard Artificial NN. This includes Recurrent Neural Network (RNN) which has the ability to process single datapoints like images as well as sequential data like video or speech. RNN algorithm works based on the principle of the human brain and has the ability to uncover the underlying relationships in sequence and can memorize long-term dependencies. LSTM is composed of a cell, an input gate, an output gate and a forget gate. It has the ability to determine how long to hold an old information, when to forget and remember and the connections between new input and old memory.

Implementation of LSTM using our current dataset of 68,722 data sample

Epoch: This indicates the number of passes of the entire training dataset the machine learning algorithm has completed.

Batch Size: This is the number of samples processed before the model is updated. The size of a batch must be more than or equal to one and less than or equal to the number of samples in the training dataset.

Time Steps: These are intervals of data samples and is a definition of how long in time is the data sample.

Working with a dataset of 68,722, a time step of 34361, a batch size of 32 and 20 epochs. We had a RMSE of 0.192. A threshold of 0.42225963644459513 and anomalies were detected as shown in the plot of Figure 4.4.

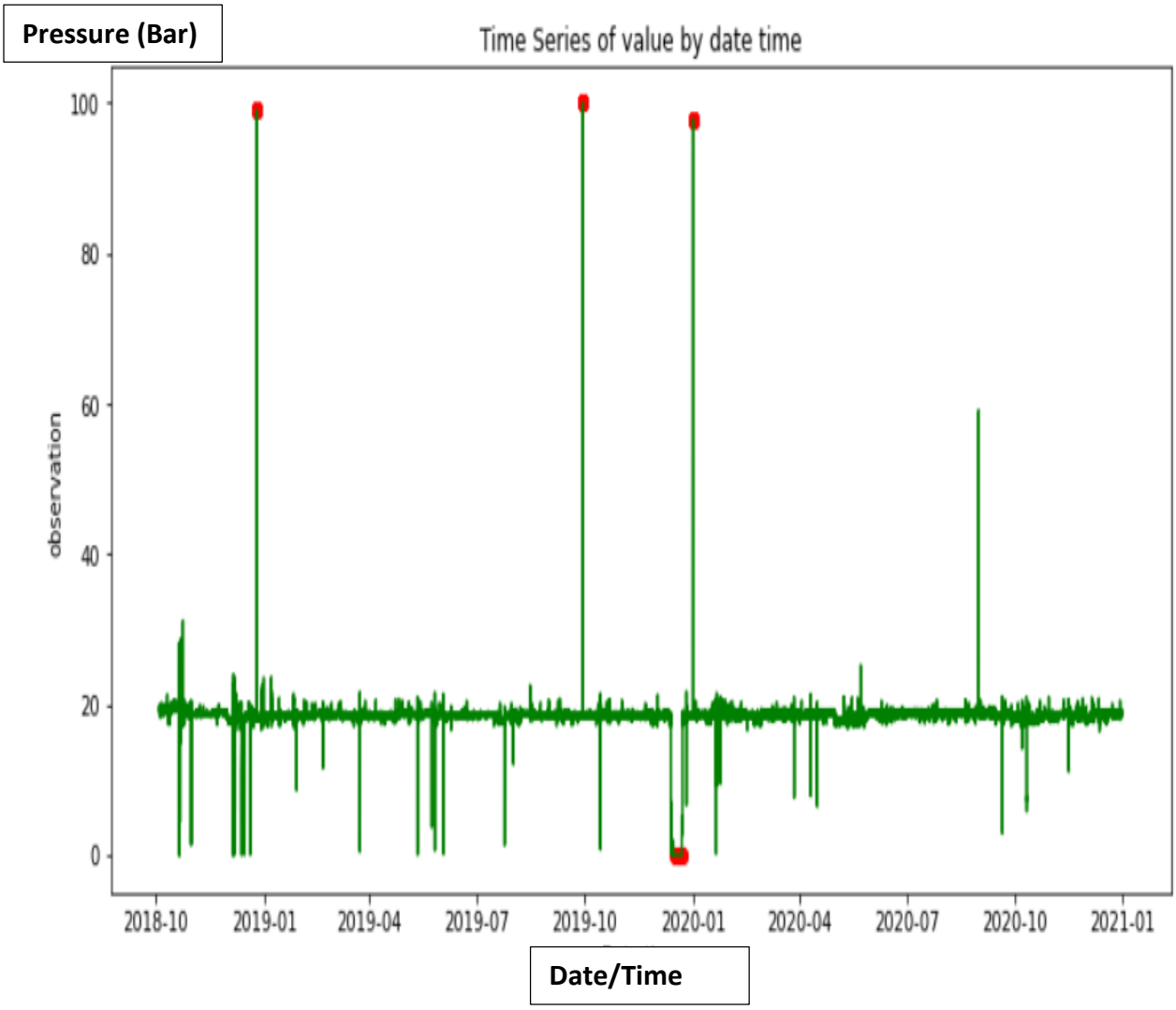


Figure 4.4 Plot of LSTM Algorithm anomaly detection with a dataset of 68,722, time step of 34361, batch size of 32 and 20 epochs (y-axis is the pressure while the x-axis is the Date and Time)

The Figure 4.4 shows the plot of pressure against Date/Time using LSTM algorithm as detailed in Appendix E. Working with a dataset of 68,722, a time step of 34361, a batch

size of 128 and 20 epochs. We had a RMSE of 0.090. A threshold of 0.21178787271031924 and anomalies were detected as shown in the plot of Figure 4.4.

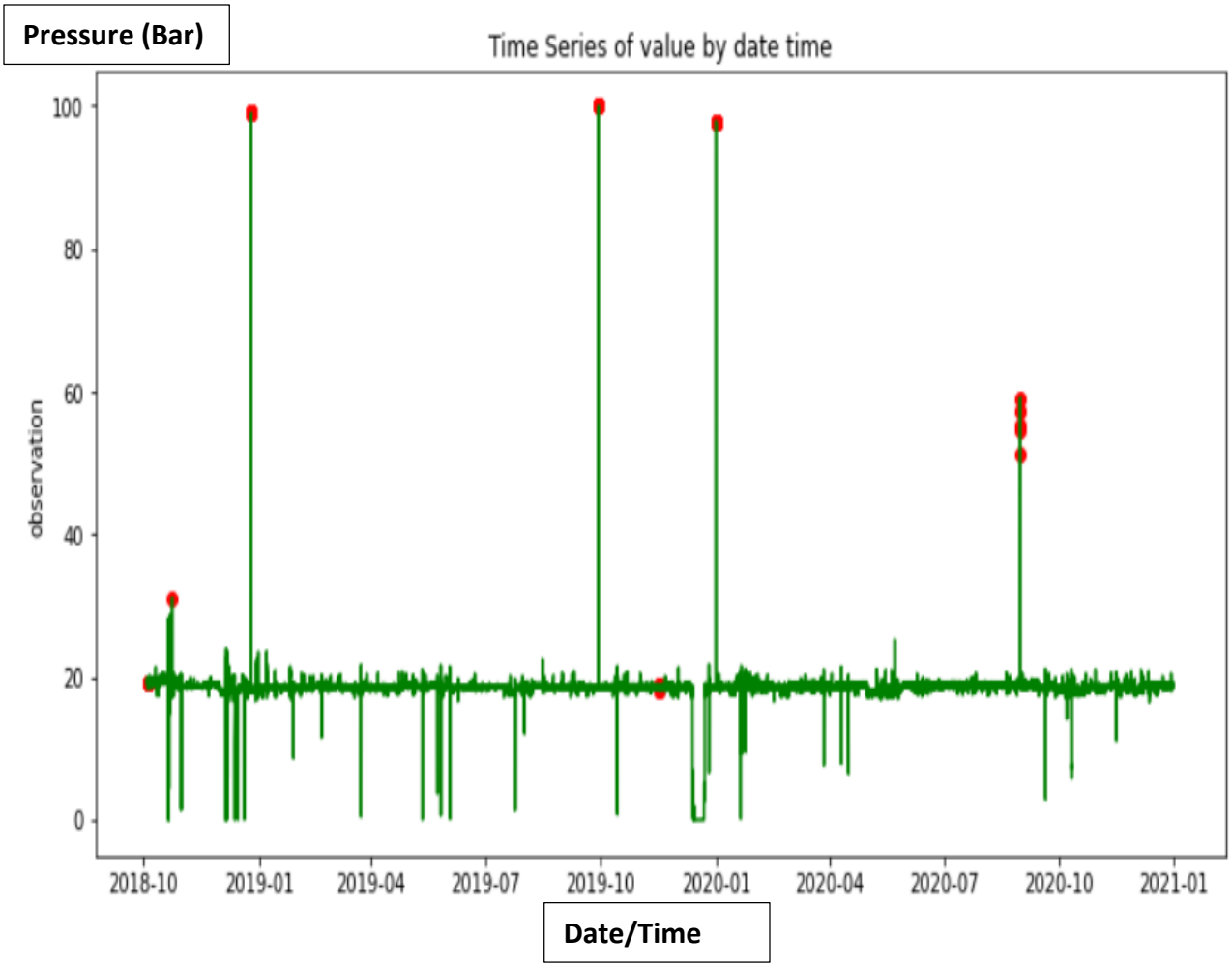


Figure 4.5 Plot of LSTM Algorithm anomaly detection with a dataset of 68,722, time step of 34361, batch size of 128 and 20 epochs (y-axis is the pressure while the x-axis is the Date and Time)

The Figure 4.5 shows the plot of pressure against Date/Time using LSTM algorithm as detailed in Appendix E. Working with a dataset of 68,722, a time step of 2, a batch size of 32 and 20 epochs. We had a RMSE of 0.006. A threshold of 0.06170250556823232 and anomalies were detected as shown in the plot of Figure 4.5.

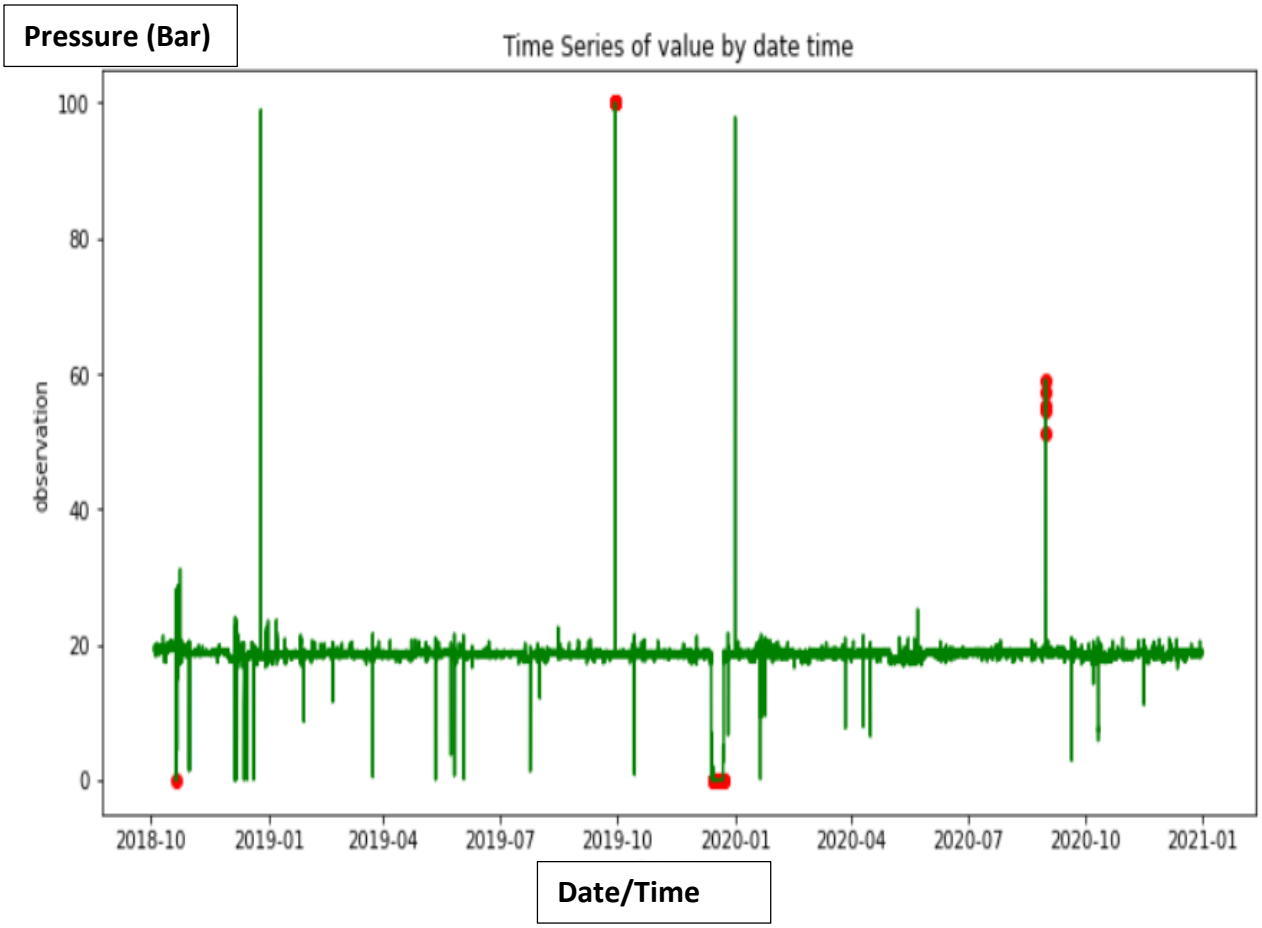


Figure 4.6 Plot of LSTM Algorithm anomaly detection with a dataset of 68,722, time step of 2, batch size of 32 and 20 epochs (y-axis is the pressure while the x-axis is the Date and Time)

The Figure 4.6 shows the plot of pressure against Date/Time using LSTM algorithm as detailed in Appendix E. Working with a dataset of 68,722, a time step of 2, a batch size of 128 and 20 epochs. We had a RMSE of 0.006. A threshold of 0.05809943435319962 and anomalies were detected as shown in the plot of Figure 4.6.

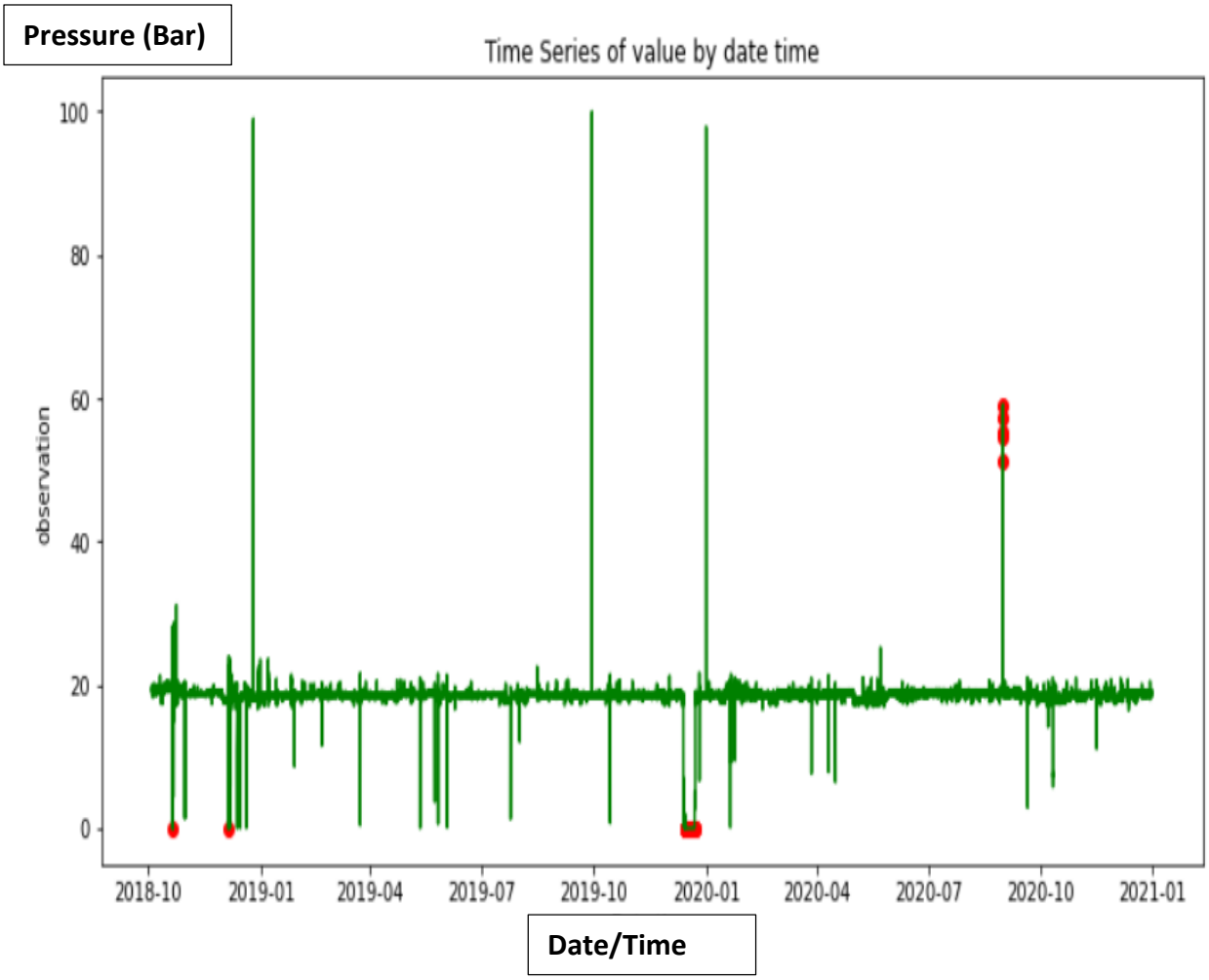


Figure 4.7 Plot of LSTM Algorithm anomaly detection with a dataset of 68,722, time step of 2, batch size of 128 and 20 epochs (y-axis is the pressure while the x-axis is the Date and Time)

The Figure 4.7 shows the plot of pressure against Date/Time using LSTM algorithm as detailed in Appendix E. Comparing Figures 4.4, 4.5, 4.6 and 4.7 it will be observed that, working with a dataset of 68,722, a time step of 34361, a batch size of 128 and 20 epochs detected the extreme high anomalies best with a RMSE of 0.090. A threshold of 0.21178787271031924 and anomalies were detected as shown in the plot of Figure 4.7.

4.1.2.3 Python Outlier Detection (PyOD) Algorithm

PyOD is an open-source Python toolbox for performing scalable outlier detection on multivariable data. It provides access to a wide range of outlier detection algorithms, including established outlier ensembles (Zhao et al., 2019)(Zhao, 2022b)(Han et al., 2022). Applied the 68,722 SCADA pressure dataset from the 3-phase separator to four different PyOD algorithms and the results are as follows:

4.1.2.3.1 Inter Quartile Range Algorithm (IQR)

This is used to measure variability of the dataset by dividing a dataset into quartiles. The sorted data is split into 4 equal parts Q_1, Q_2 and Q_3 referred to as 1st, 2nd and 3rd quartiles with Q_1 representing 25%, Q_2 – 50% and Q_3 – 75% of the data.

If a dataset has $\frac{2n}{2n+1}$ data points

Q_1 = median of the dataset

Q_2 = median of n smallest data points

Q_3 = median of n highest data points

IQR is the range between the 1st and the 3rd quartiles namely Q_1 and Q_3.

$$IQR = Q3 - Q1 \quad (4.1)$$

Datapoints below Q_1 – 1.5 IQR or above Q_3 + 1.5 IQR are termed anomalies.

Applying this rule to the 68,722 SCADA pressure dataset using Python returned the following:

The lower threshold detected is 17.701954000000004Bar

The upper threshold detected is 19.5607055999999996Bar

The detected anomalies are as plotted in Figure 4.8.

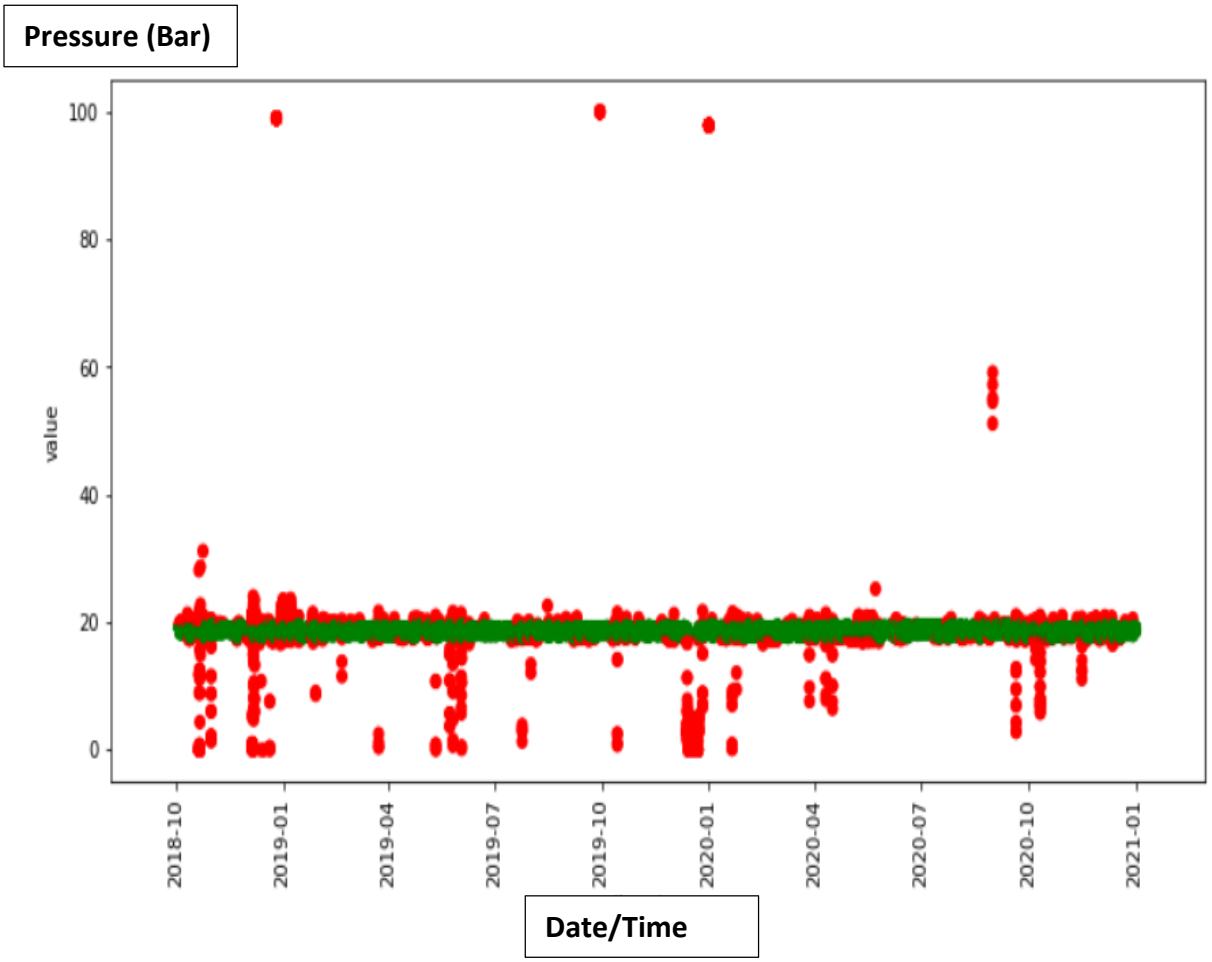


Figure 4.8 Plot of Inter Quartile Range (IQR) Algorithm anomaly detection with a dataset of 68,722, (y-axis is the pressure while the x-axis is the Date and Time). As detailed in Appendix F.

4.1.2.3.2 k-Nearest Neighbors (KNN) Outlier Detection Algorithm

KNN detects anomalies using the distance to the k^{th}Nearest neighbors as the outlier score (Zhao, 2022b). This is based on the fact that distant and standalone neighbors are considered anomalies.

Applying the 68,722 pressure data samples using the kNN algorithm generated a threshold of 0.0002099999999991553. Plotting the result of the algorithm is as shown in Figure 4.9.

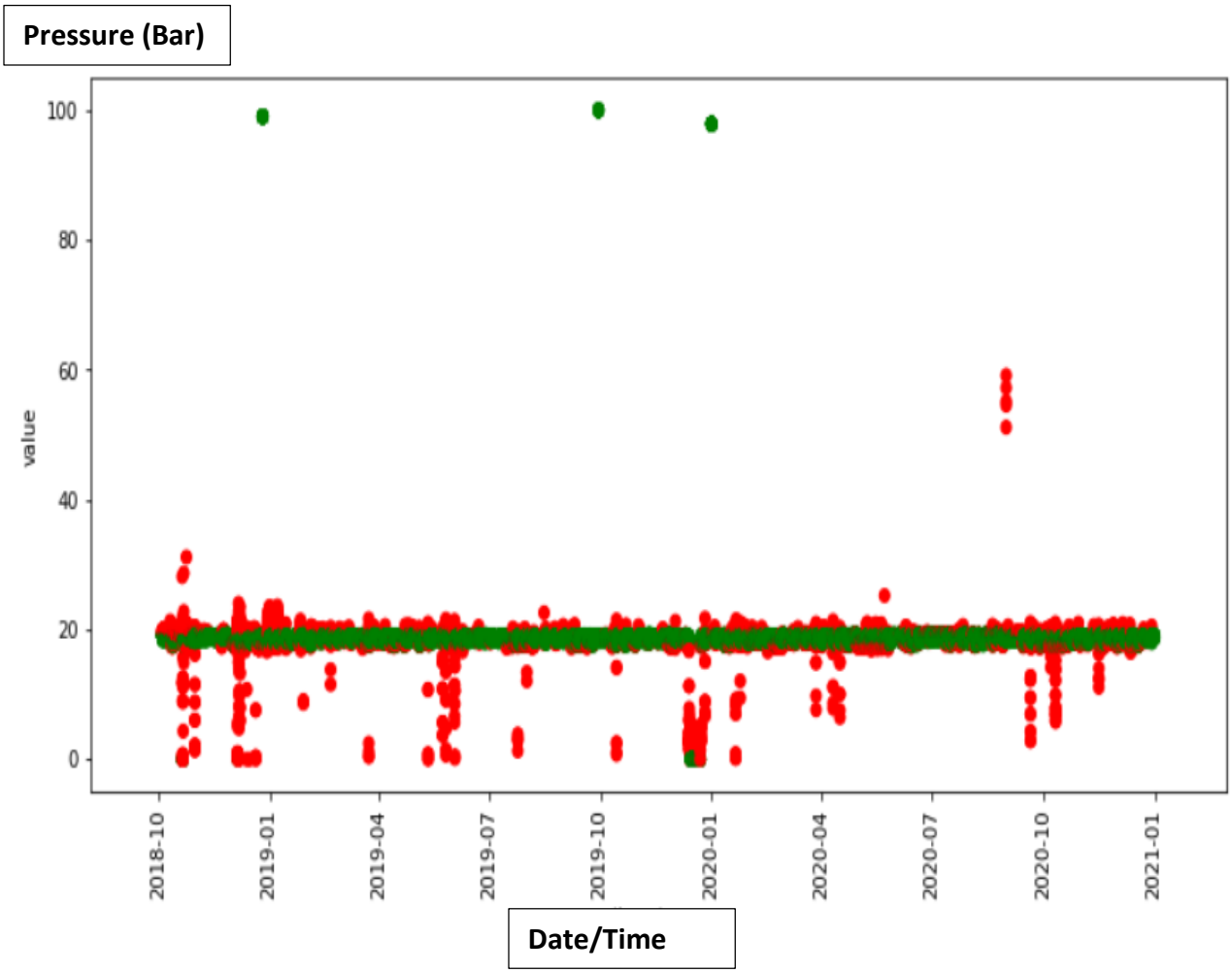


Figure 4.9 Plot of k Nearest Neighbor (kNN) Algorithm anomaly detection with a dataset of 68,722, (y-axis is the pressure while the x-axis is the Date and Time). As detailed in Appendix F.

4.1.2.3.3 Local Outlier Factor (LOF) Algorithm

This unsupervised anomaly detection model computes the local density deviation of a given data point concerning its neighbors. It considers as outliers the samples that have a substantially lower density than their neighbors. It is local in that the anomaly score depends on how isolated the object is with respect to the surrounding neighborhood (Zhao, 2022a).

Applying the 68,722 pressure data samples using the LOF algorithm generated a threshold of 1.0797704251094205. Plotting the result of the algorithm is as shown in Figure 4.10.

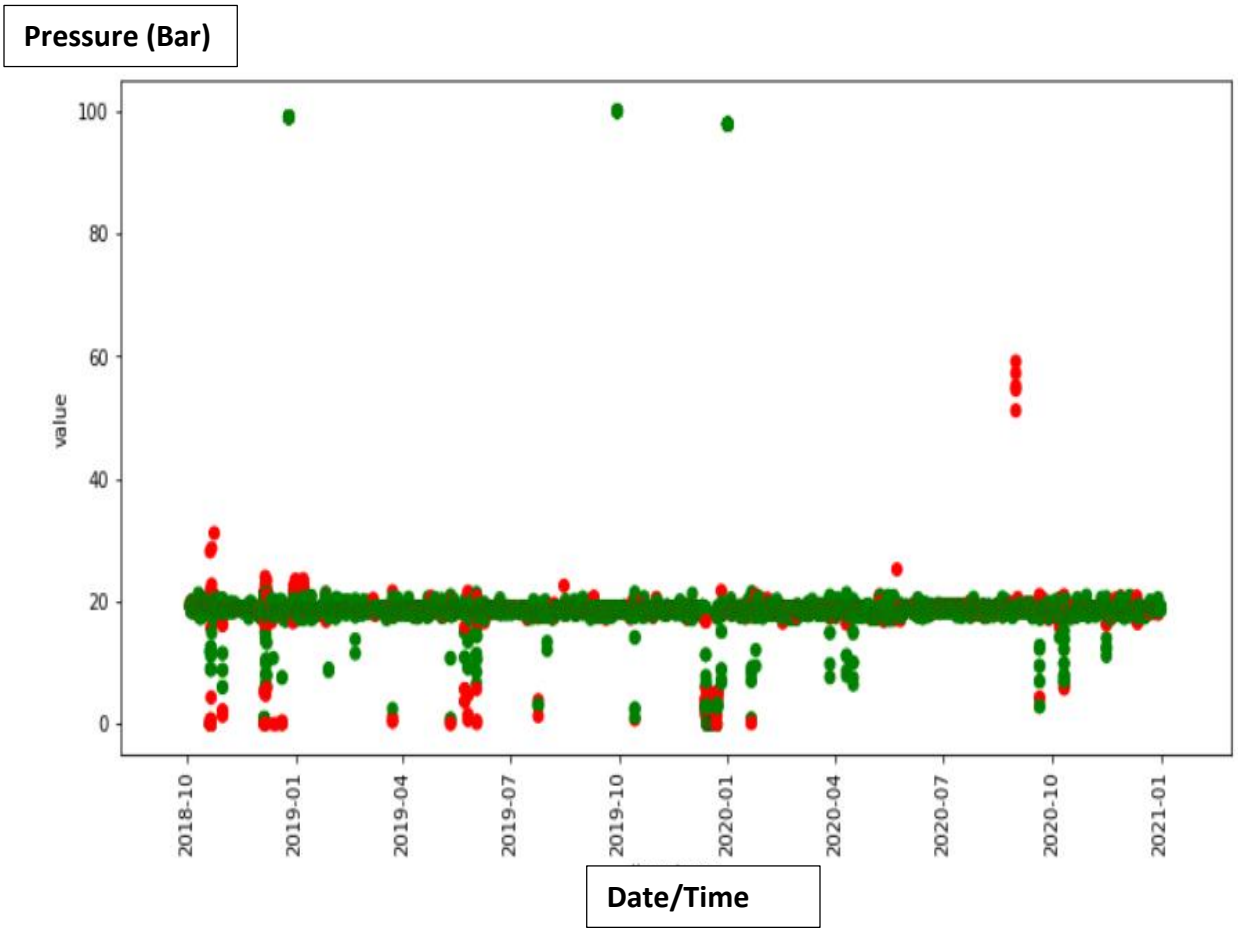


Figure 4.10 Plot of Local Outlier Factor (LOF) Algorithm anomaly detection with a dataset of 68,722, (y-axis is the pressure while the x-axis is the Date and Time). As detailed in Appendix F.

4.1.2.4 Decision Tree Algorithms

This has wide range of applications and can be easily combined with one or more algorithms to develop more complex models capable of solving complex problems. Further simulations were run using different forms of Decision Trees Algorithms and the results are as summarized in Table 4.1.

4.2 Discussion of Results

4.2.1 Designed PCN Architecture Results

The integration of the newly developed security server which houses the models is capable of detecting malicious behavious in the data signal exchange between the sensors, the transmitters, the controllers and the final control element in the PCN. This layer 2 device helps to mitigate possible armophous cyber-attacks which may have catastrophic effects including process safety incidents. The point of inegration is robust in nature and can capture both insider and outsider threats with minimal computational requirements. On sensing an anomaly, the inegrated systems sends annunciation to the HMI to alert the console operator who has the responsibility to investigate the alarm, abort or further escalate the investigation. This developed system which is based on the Open System Interconnection (OSI) is not particular to any vendor and has the ability to communicate with systems that were designed with OSI standards.

4.2.2 Machine learning Algorithms Applied to PCN Results

A. The Isolation Forest Algorithm Model for Anomaly Detection – This used less memory and computing resources, it generated results very fast but not very good for the detection of high-pressure value anomalies as can be seen in Figure 4.3. The goal of this research is not to detect all changes in the pressure value as outliers but to detect the pressure values that are extremely low or extremely high or both. It is important to note that extreme low-pressure values could mean system shutdown or equipment shutdown while extreme high-pressure values will likely be a malicious attack.

B. The LSTM Algorithm Model for Anomaly Detection - Based on Figures 4.5, 4.6, 4.7, 4.8 it can be deduced that the anomalies detected by the model shown on Figure 4.6 with a time step of 34361, a batch size of 128 and 20 epochs, RMSE of 0.090 an a threshold of 0.21178787271031924 is more reliable because it was able to detect all the extremely high values perfectly and can be used in real-time detection of extremely

high values, as it learns what is normal from the dataset, defines a threshold, and detects anomalies with less manipulation.

C. The interquartile range algorithm - This identified upper and lower thresholds as detected by the algorithm helped to detect the outliers as shown in the plot of Figure 4.9. Like the Isolation Forest, the interquartile range algorithm was able to detect and tag pressure changes anomalous. The lower threshold detected is 17.701954000000004Bar. The upper threshold detected is 19.5607055999999996Bar. This algorithm has high sensitivity which will likely generate false alarms.

D. *k*-Nearest Neighbor Algorithm - This algorithm was able to detect some anomalies but could not detect the extreme high-pressure values accurately as shown in Figure 4.10. The accuracy of this algorithm with this dataset is poor.

E. Local Outlier Factor - This algorithm was able to detect some anomalies but performed poorly in the accurate detection of extreme high-pressure values as shown in Figure 4.11. The accuracy of the algorithm with this dataset is less than that of kNN.

Several other algorithms/algorithmic adaptations were used to simulate the 68,722 SCADA pressure datasets collated from the 3-phase separator for a period of more than 2 years. The following comparisons were made with results from two other existing datasets used by other researchers, which are WUSTL Dataset with 1,048,575 data samples and ORNL Power Grid Dataset with 4912 data samples. Comparison of the behavior of the realt-time WUSTL datasets using Minimum Classifcation Error (MCE), prediction speed, accuracy and training time for different algorithms is as shown in Table 4.1.

Table 4.1 Behaviour of the existing WUSTL dataset using different algorithms

Algorithm	Accuracy (%)	Training Time (ms)	MCE	Prediction Speed (obs/sec)
Decision Trees (DT)				
Fine Tree (FT)	98.1	4.5778	5096	1100000
Medium Tree (MT)	97	3.8937	7974	1100000

Coarse Tree (CT)	95	3.5817	13357	1200000
Optimizable Tree	99.1	127.26	2345	1100000
Discriminant Analysis				
Linear Discriminant (LDR)	92.8	4.5025	19350	1000000
Quadratic Discriminant (QDR)	92.9	3.8003	19225	990000
Optimizable Discriminant	92.9	284.61	19216	100000
Logistic Regression (LR)	93.6	9.4867	0	1100000
Naïve Bayes				
Gaussian Naive Bayes (GNB)	92.8	8.3104	19349	110000
Kernel Naive Bayes (KNB)	89.7	3379.1	27849	340
Optimizable NB	96	3039.9	4992	3100
Support vector Machines (SVMs)				
Linear SVM	36	7252.9	172670	3000
Quadratic SVM	10.6	5742.1	241081	330000
Cubic SVM	44.6	10046	149331	250000
Fine Gaussian SVM	95.4	809.21	12292	12000
Medium Gaussian SVM	94.5	836.05	14835	9800
Coarse Gaussian SVM	92.9	937.93	19192	9300
Nearest Neighbors				
Fine KNN	99.2	5.5776	2163	540000
Medium KNN	99	8.9119	2578	260000
Coarse KNN	98.4	16.92	4394	91000
Cosine KNN	98.8	204.06	3339	5900
Cubic KNN	99	12.953	2613	130000
Weighted KNN	99.2	8.8344	2134	250000
Ensemble Learning (EL)				
Boosted Tree (BST)	97.5	38.673	6812	2200000
Bagged Tree (BT)	99.4	85.488	1512	1600000
RSUBoosted Tree (RUBT)	95.9	28.591	10936	2100000
Subspace KNN (SKNN)	99.2	79.149	2083	560000
Subspace Discriminant (SD)	92.8	29.92	19348	130000

Comparison of the behavior of the realt-time SCADA Pressure datasets using Minimum Classifcation Error (MCE), prediction speed, accuracy and training time for different algorithms is as shown in Table 4.2.

Table 4.2 Behaviour of the SCADA Pressure Training dataset with Anomaly using different algorithms

Algorithm	Accuracy (%)	Training Time (mS)	MCE	Prediction Speed (obs/sec)
Decision Trees				
Fine Tree (FT)	100	1.1708	0	1200000
Medium Tree (MT)	100	1.0781	0	1300000
Coarse Tree (CT)	100	0.45488	0	1000000
Optimizable Tree	100	21.323	0	1300000
Discriminant Analysis				
Linear Discriminant (LDR)	100	1.843	24	1100000
Quadratic Discriminant (QDR)	99.2	1.1597	518	1600000
Optimizable Discriminant	100	25.029	24	1600000
Logistic Regression (LR)	100	3.205	NA	1100000
Naïve Bayes				
Gaussian Naive Bayes (GNB)	99.2	1.4947	518	1400000
Kernel Naive Bayes (KNB)	100	65.633	8	4500
Optimizable NB	100	918.96	8	3800
Support vector Machines (SVMs)				
Linear SVM	100	7.3065	25	780000
Quadratic SVM	100	383.79	17	1500000
Cubic SVM	80.2	1657.3	13588	930000
Fine Gaussian SVM	100	7.433	5	610000
Medium Gaussian SVM	100	5.3155	1	760000
Coarse Gaussian SVM	100	5.1452	20	1100000
Optimized SVM	100	7490.9	25	1100000
Nearest Neighbors				

Fine KNN	100	3.6447	0	820000
Medium KNN	100	2.0989	5	460000
Coarse KNN	99.9	3.5228	35	130000
Cosine KNN	99.9	17.422	35	17000
Cubic KNN	100	2.3157	5	380000
Weighted KNN	100	2.1524	0	450000
Ensemble Learning (EL)				
Boosted Tree (BST)	99.9	5.0025	35	1200000
Bagged Tree	100	8.5874	0	320000
Subspace Discriminant	100	4.5421	24	260000
Subspace KNN	100	12.777	0	93000
RUSBoosted Trees	100	2.4396	20	960000
Optimized Ensemble	100	232.87	0	530000

Table 4.3 Behaviour of the SCADA Pressure Validation dataset with Anomaly using different algorithms

Algorithm	**Accuracy (%)**	**Training Time (mS)**	**MCE**	**Prediction Speed (obs/sec)**
Decision Trees				
Fine Tree (FT)	100	1.3865	0	1300000
Medium Tree (MT)	100	1.2055	0	1100000
Coarse Tree (CT)	100	0.53595	0	1300000
Optimizable Tree	100	21.323	0	1300000

Comparing the accuracy of algorithms used in the SCADA Pressure data with 68,722 data samples analysis and that used by other data sets the following was observed:

Table 4.4 Showing Features of Model Parameters used

FEATURES OF MODEL PARAMETERS	
Models	**Parameters**
Observations	SCADA Pressure Dataset = 68,722 Samples WUSTL Dataset =1048575 Samples ORNL Power Grid Dataset = 4912 Samples

Predictors	SCADA Pressure Dataset = 1 WUSTL Dataset = 6 ORNL Power Grid Dataset = 76
Class Responses	SCADA Pressure Dataset = 2 WUSTL Dataset = 2 ORNL Power Grid Dataset = 2
Result Presentation	Bar Chart, Tables, Confusion Matrix, Receiver Operator Characteristics
Decision Tress	Maximum Splits = 100, Split Criterion = Gini's Diversity Index Preset = Fine Tree, Medium Tree, Coarse Tree, Optimizable Tree, Maximum number of splits: 901, Split criterion: Maximum deviance reduction, Optimizer: Bayesian optimization, No of Iterations: 30
Extreme Learning Machine (ELM)	Maximum Splits = 20, Method = AdaBoost, Learning Rate = 0.1, Preset = Boosted, Bagged, RSUBoosted, Subspace Discriminant, Learning Type = Decision Tree, Discriminant, Subspace dimension: 3, Number of learners: 30
Support Vector Machine (SVM)	Kernel Function = Linear, Gaussian, Kernel Scale = 0.00101952, 1, Automatic Box Constraint Level = 1.75813, 1, 26, Multi class Method = One Vs One, One Vs All, Preset = CGSVM, MGSVM, FGSVM
Logistic Regression	Preset: Logistic Regression
k-Nearest Neighbor (KNN)	Neighbors = 1 & 10, Distance Weight = Squared inverse, Equal Distance Metric = Euclidean

	Preset = Weighted, Fine, Cosine
Discriminant Analysis	Linear discriminant, quadratic discriminant

Table 4.5 Models and their accuracy used for the WUSTL-SCADA-2018 Dataset

WUSTL-SCADA-2018 DATASET	
Machine Learning Classifiers	**Accuracy (%)**
Decision Trees (DT)	100
Logistics Regression	99.6
Ensemble Subspace Discriminant	93.1
K-Nearest Neighbors (KNN)	68

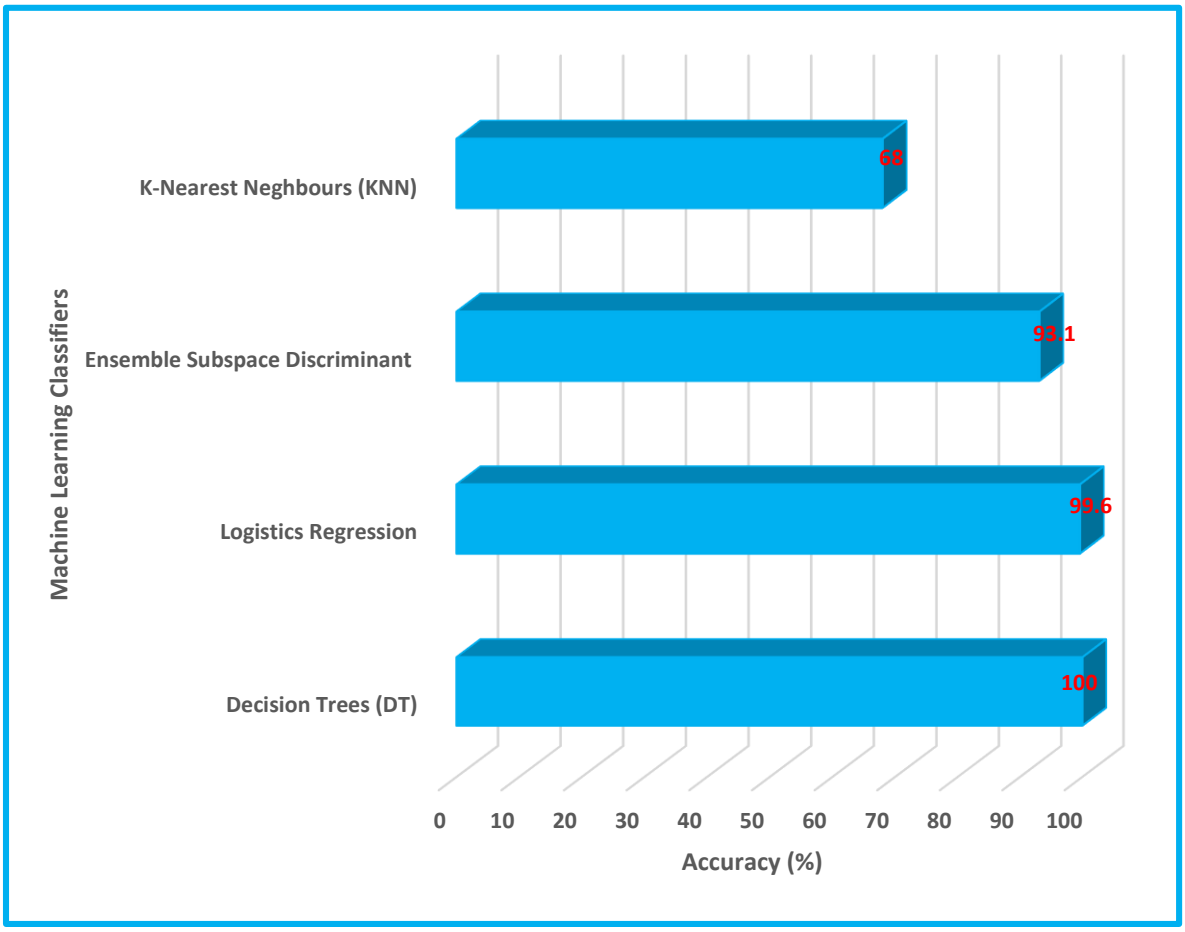


Figure 4.11 Plot of Models results and their accuracy used for the WUSTL-SCADA-2018 Dataset

Table 4.6 Models and their accuracy used for the ORNL Power Grid Dataset

ORNL POWER GRID DATASET	
Machine Learning Classifiers	**Accuracy (%)**
Ensemble Learning (BAGGED)	95.1
K-Nearest Neighbors (KNN)	94.4
Decision Trees (DT)	84.2
Discriminant Analysis (QDR)	52.4

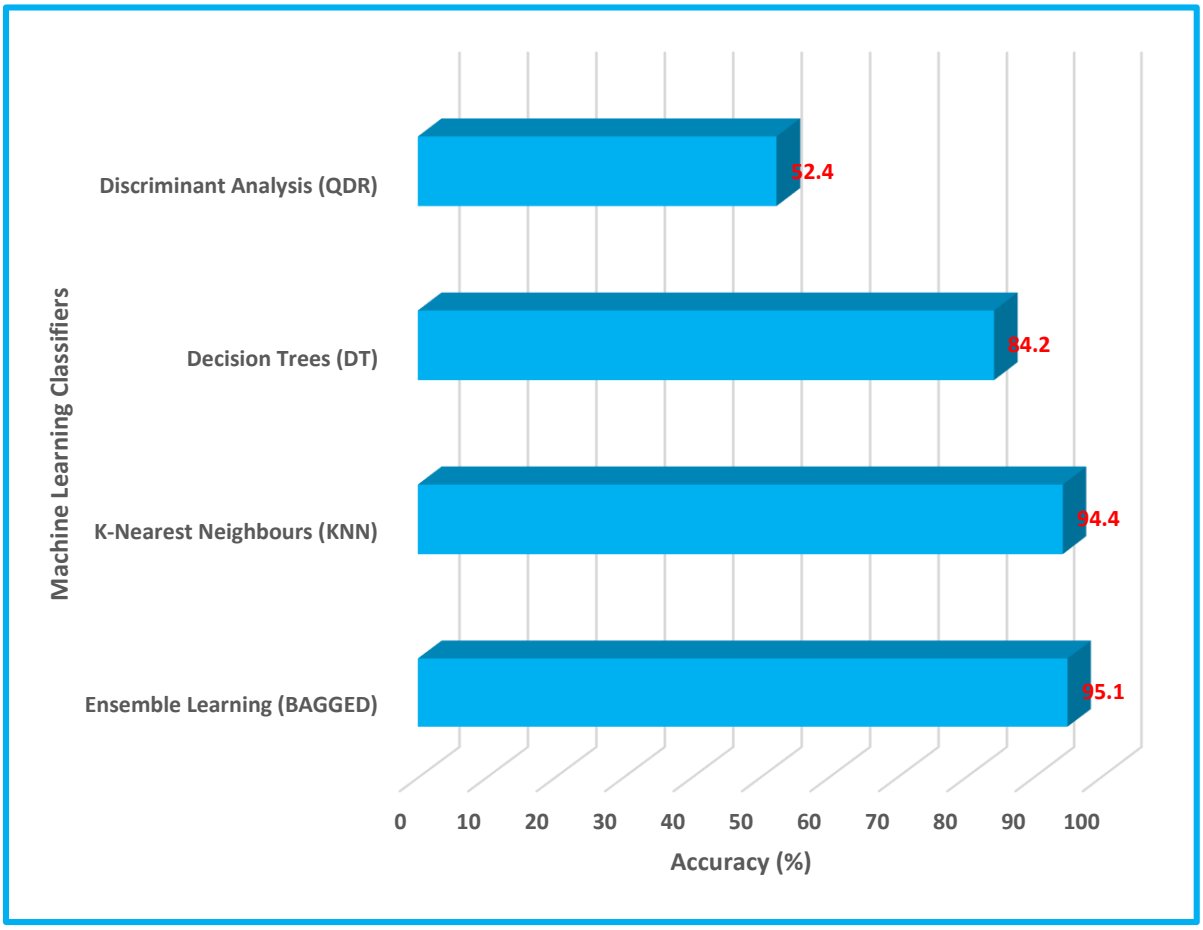


Figure 4.12 Plot of Models results and their accuracy used for the ORNL Power Grid Dataset

Table 4.7 Models and their accuracy used for the SCADA Pressure Dataset

SCADA PRESSURE DATASET	
Machine Learning Classifiers	**Accuracy (%)**
Decision Trees (DT)	100
Ensemble Learning (BOOSTED)	99.9
Discriminant Analysis (QDR)	99.2
Support Vector Machine (CSVM)	80.2

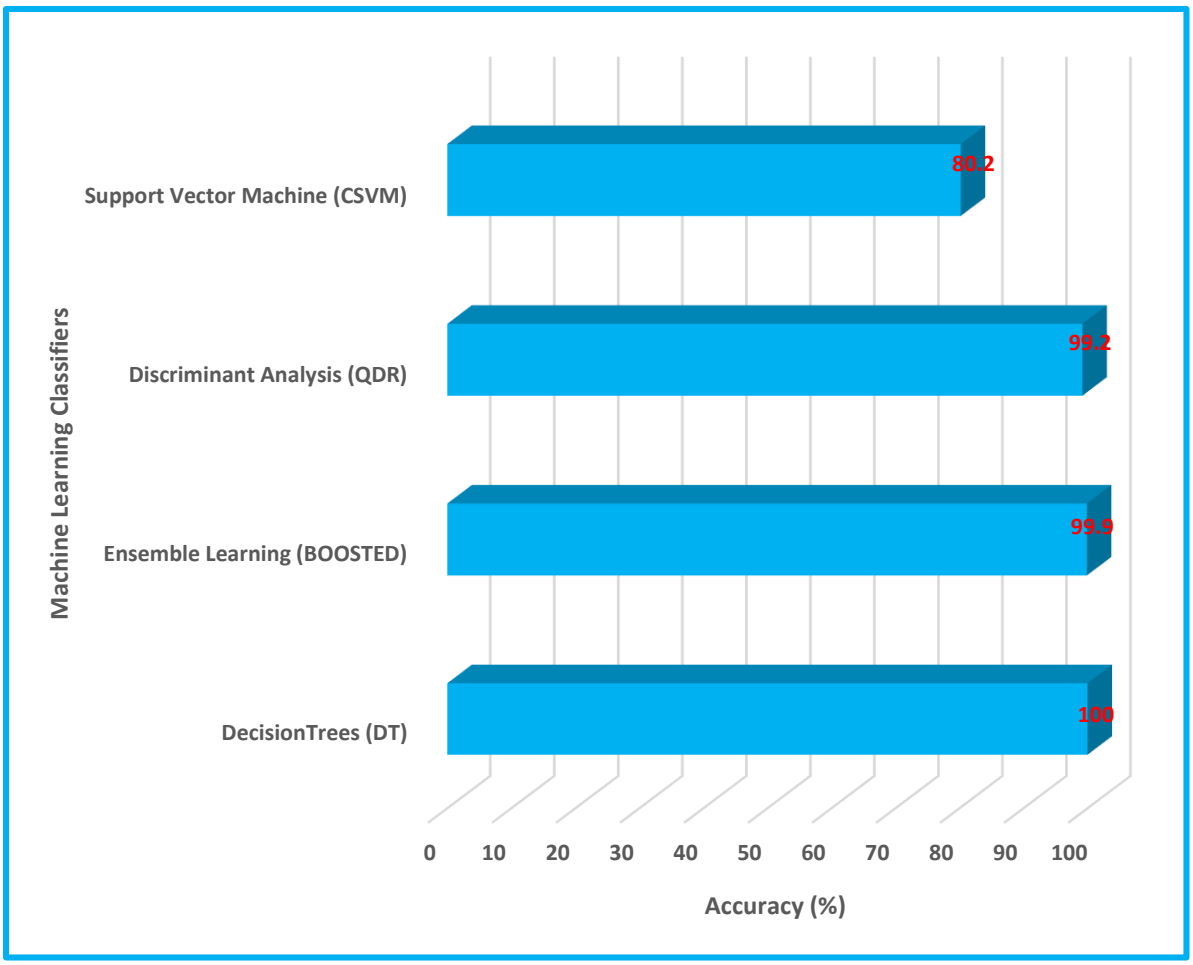


Figure 4.13 Plot of Models results and their accuracy used for the SCADA Pressure Dataset

Table 4.8 Comparative Analysis of Top Performing Machine Learning Classifiers

COMPARATIVE ANALYSIS OF TOP PERFORMING MACHINE LEARNING CLASSIFIERS			
Performance Metrics	**SCADA PRESSURE DATASET**	**WUSTL-SCADA-2018 DATASET**	**ORNL POWER GRID DATASET**
Accuracy (%)	100	100	95.1
False Alarm Rate (FAR) (#)	0	412	241
Prediction Speed (obs/sec)	1000000	4100000	2500
Computation Time (s)	0.45488	5.6605	4.8021

Model	Coarse Tree	Medium Tree	Bagged Tree

Table 4.9 Comparative Analysis of Least Performing Machine Learning Classifiers

COMPARATIVE ANALYSIS OF LEAST PERFORMING MACHINE LEARNING CLASSIFIERS			
Performance Metrics	SCADA PRESSURE DATASET	WUSTL-SCADA-2018 DATASET	ORNL POWER GRID DATASET
Accuracy (%)	80.2	93.1	52.4
False Alarm Rate (FAR) (#)	13588	72009	2339
Prediction Speed (obs/sec)	930000	110000	120000
Computation Time (s)	1657.3	101.64	1.6364
Models	Cubic SVM	Subspace Discriminant	Quadratic Discriminant

Confusion Matrix also known as error Matrix is used to visualize the performance of the algorithms. This tabular layout categorizes the performance of the algorithm into actual class and predicted class which makes it easy to confirm if there exists a confusion between the actual and predicted classes.

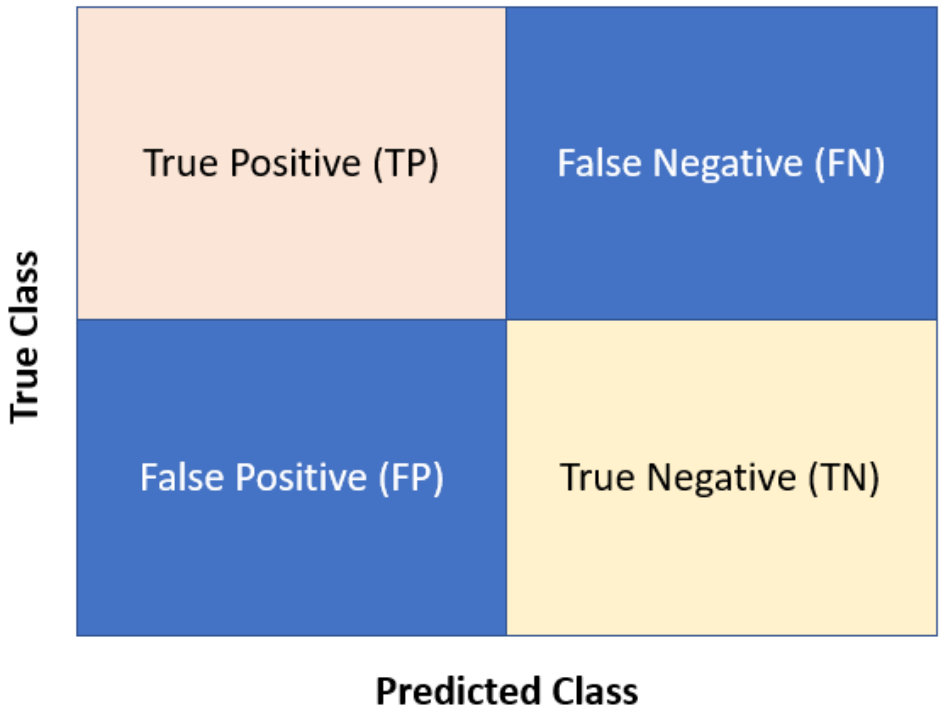


Figure 4.14 Plot of Confusion Matrix

The confusion matrix outcomes as shown in Figure 4.14 can be used to depict the performance of different machine learning algorithms

True Positive (TP) – Model correctly predicts positive outcome

True Negative (TN) – Model correctly predicts negative outcome

False Positive (FP) – Model wrongly predicts positive outcome

False Negative (FN) – Model wrongly predicts negative outcome

Plotting the Confusion Matrix of the Algorithm with Best Performance using the different Datasets is as follows:

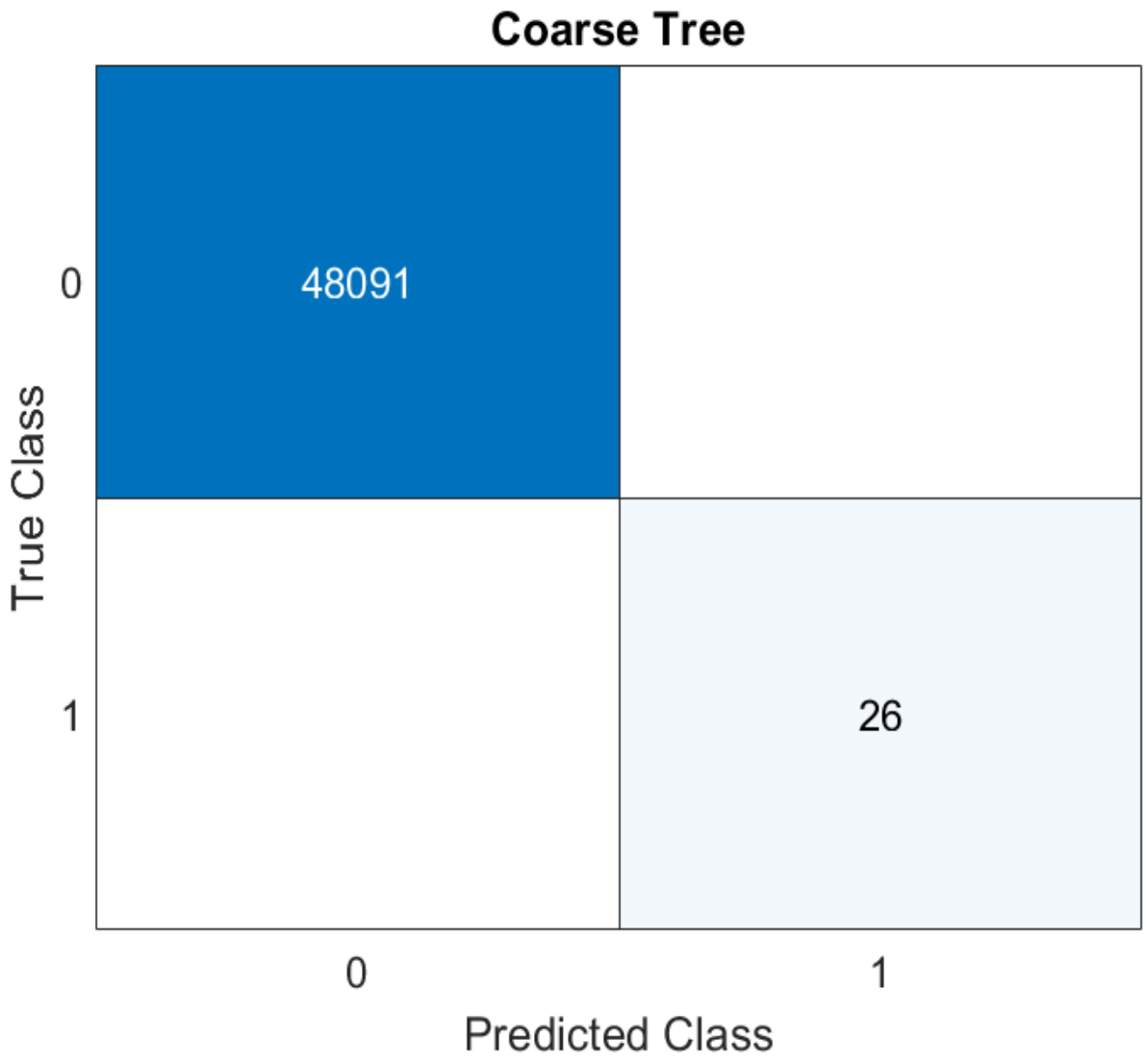


Figure 4.15 Plot of Confusion Matrix of the Tree Algorithm with Best Performance using SCADA Pressure Dataset

The confusion matrix outcomes as shown in Figure 4.15 shows:

True Positive (TP) – 48091

True Negative (TN) – 26

False Positive (FP) – 0

False Negative (FN) – 0

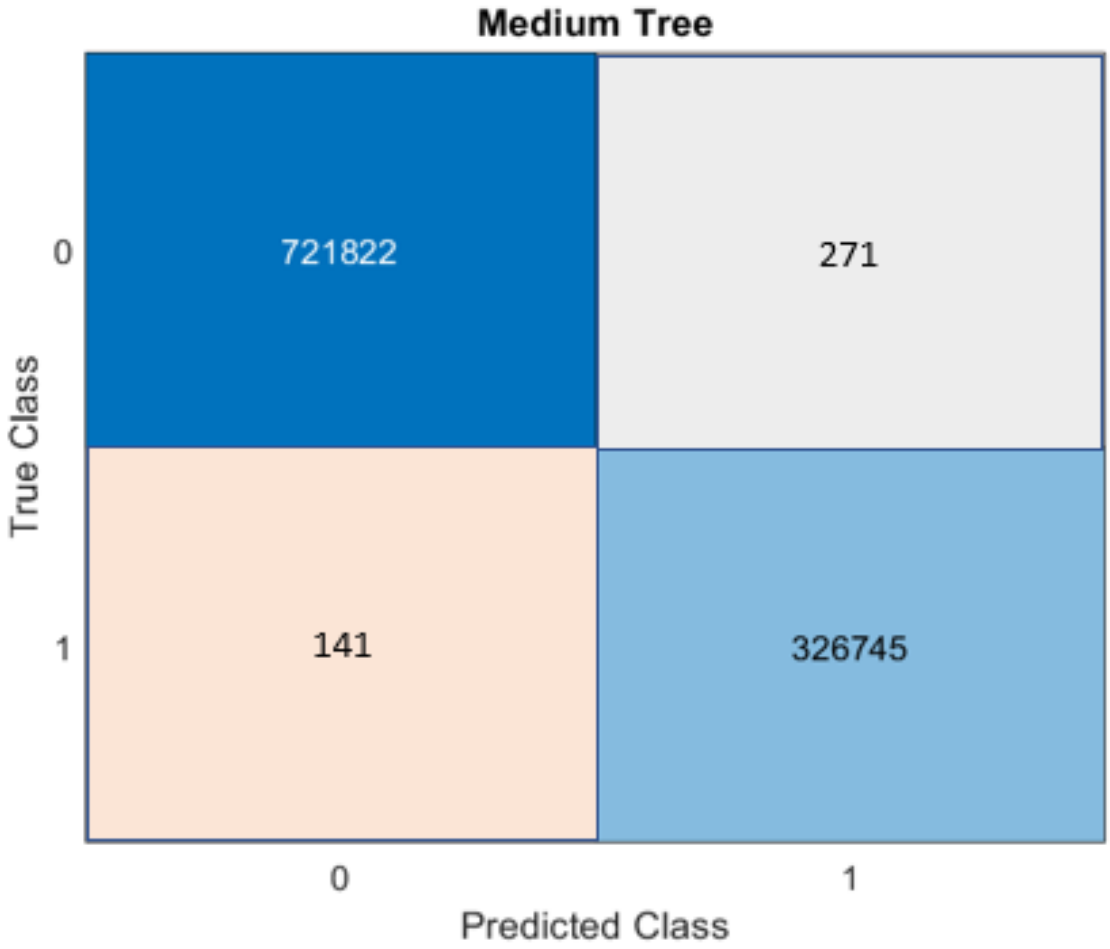


Figure 4.16 Plot of Confusion Matrix of the Tree Algorithm with Best Performance in WUSTL-SCADA-2018 Dataset

The confusion matrix outcomes as shown in Figure 4.16 shows:

True Positive (TP) – 721822

True Negative (TN) – 326745

False Positive (FP) – 141

False Negative (FN) – 271

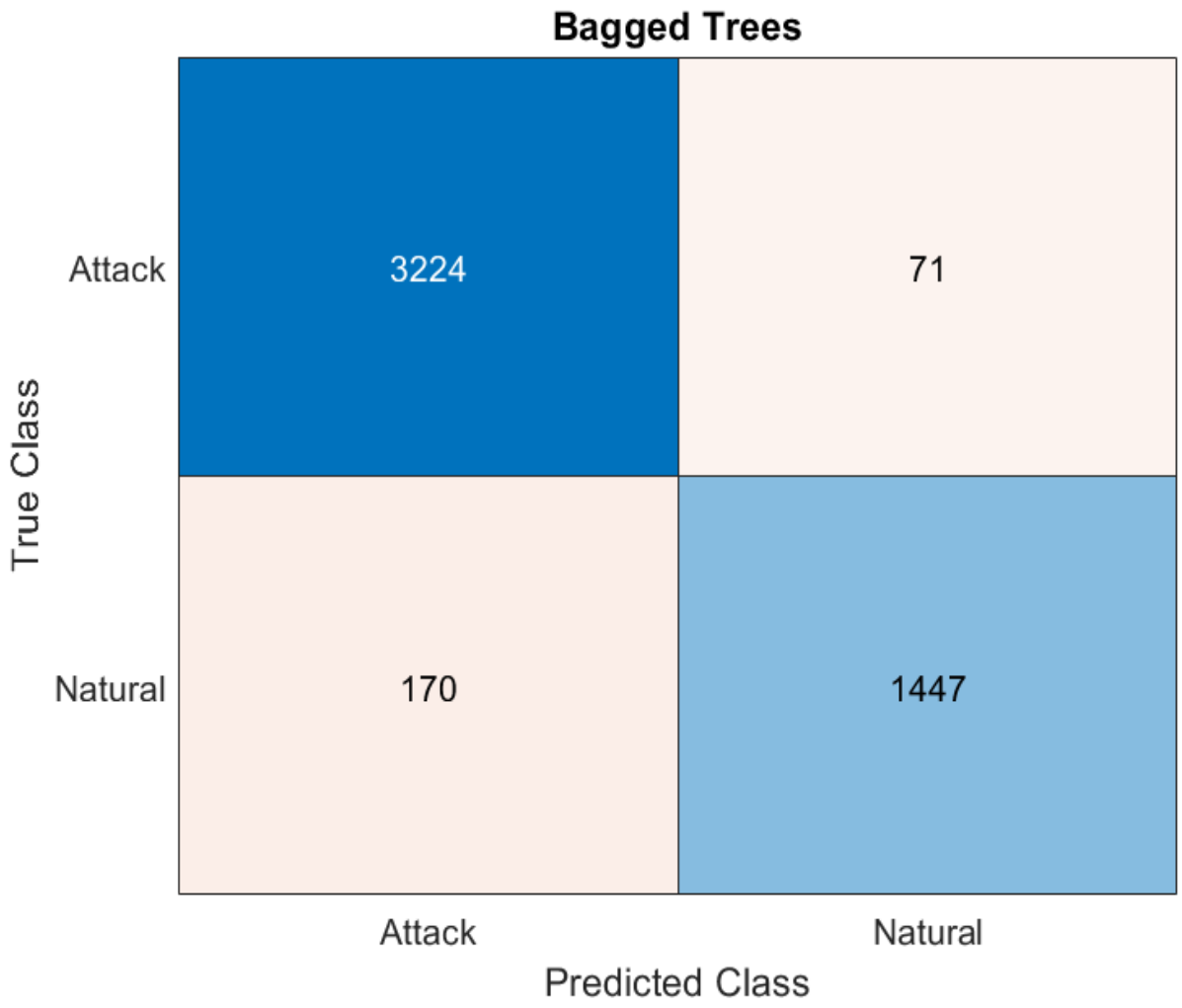


Figure 4.17 Plot of Confusion Matrix of the Tree Algorithm with Best Performance using ORNL (Power Grid) SCADA Dataset

The confusion matrix outcomes as shown in Figure 4.17 shows:

True Positive (TP) – 3224

True Negative (TN) – 1447

False Positive (FP) – 170

False Negative (FN) – 71

Plotting the Confusion Matrix of the Algorithm with Worst Performance using the different Datasets is as follows:

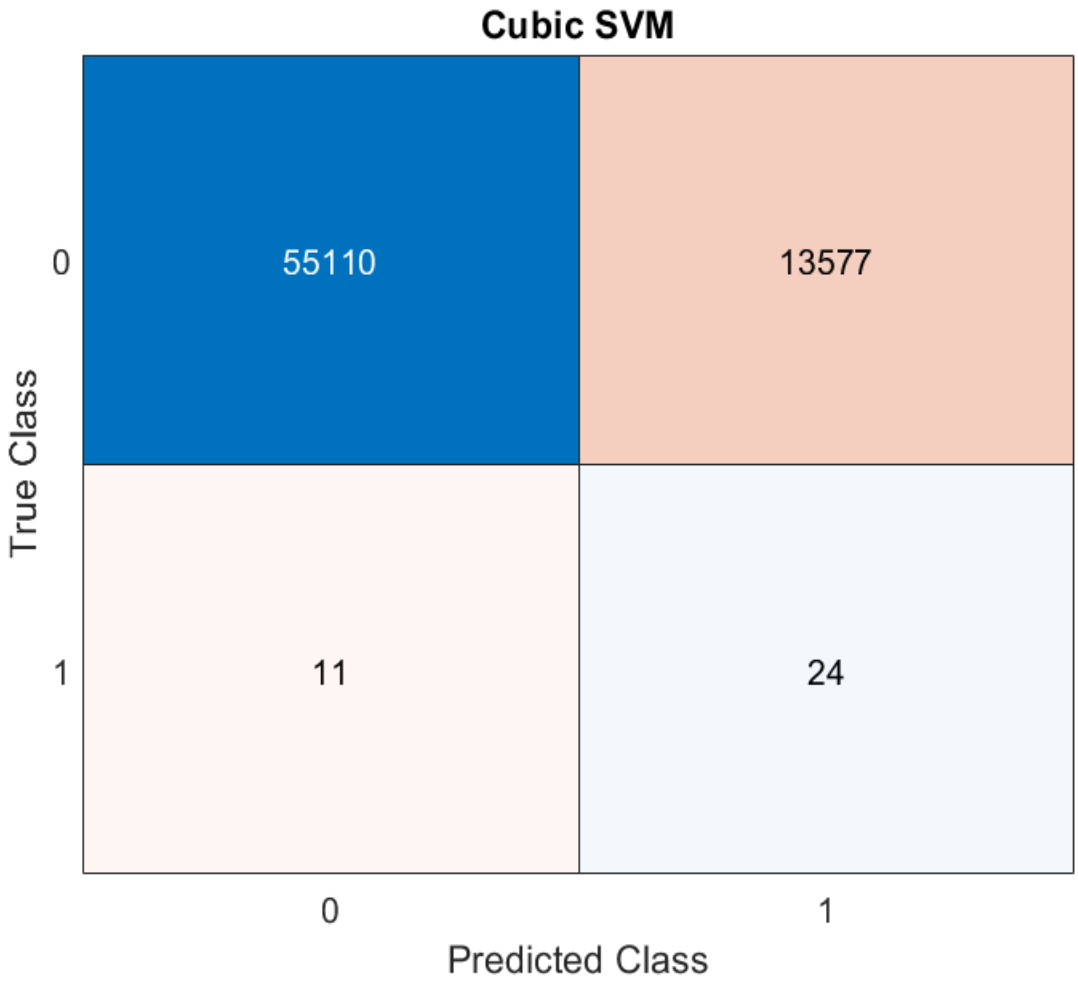


Figure 4.18 Plot of Confusion Matrix of the Algorithm with Worst Performance in SCADA Pressure Dataset

The confusion matrix outcomes as shown in Figure 4.18 shows:

True Positive (TP) – 55110

True Negative (TN) – 24

False Positive (FP) – 11

False Negative (FN) – 13577

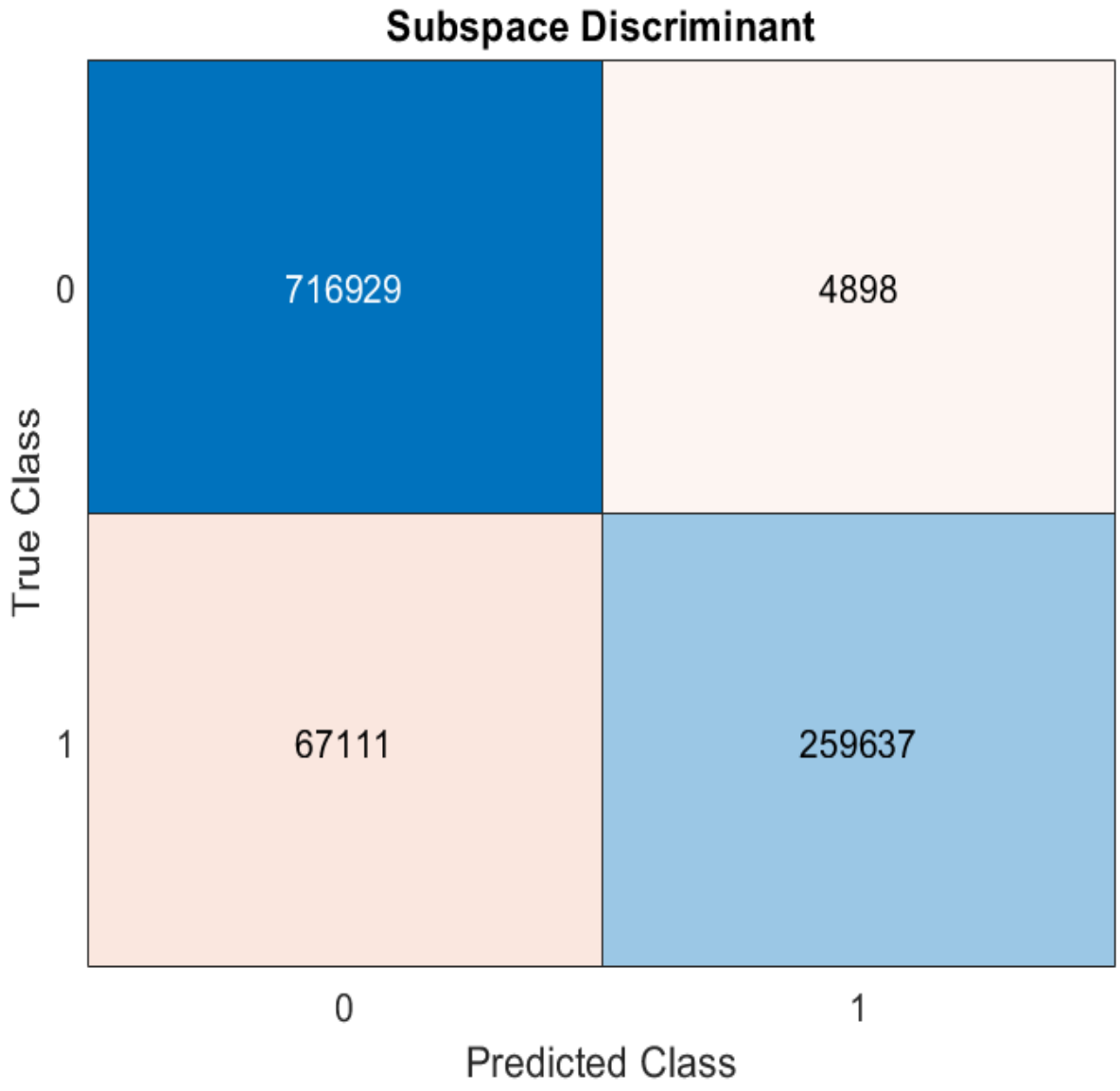


Figure 4.19 Plot of Confusion Matrix of the Algorithm with Worst Performance in WUSTL-SCADA-2018 Dataset

The confusion matrix outcomes as shown in Figure 4.19 shows:

True Positive (TP) – 716929

True Negative (TN) – 259637

False Positive (FP) – 67111

False Negative (FN) – 4898

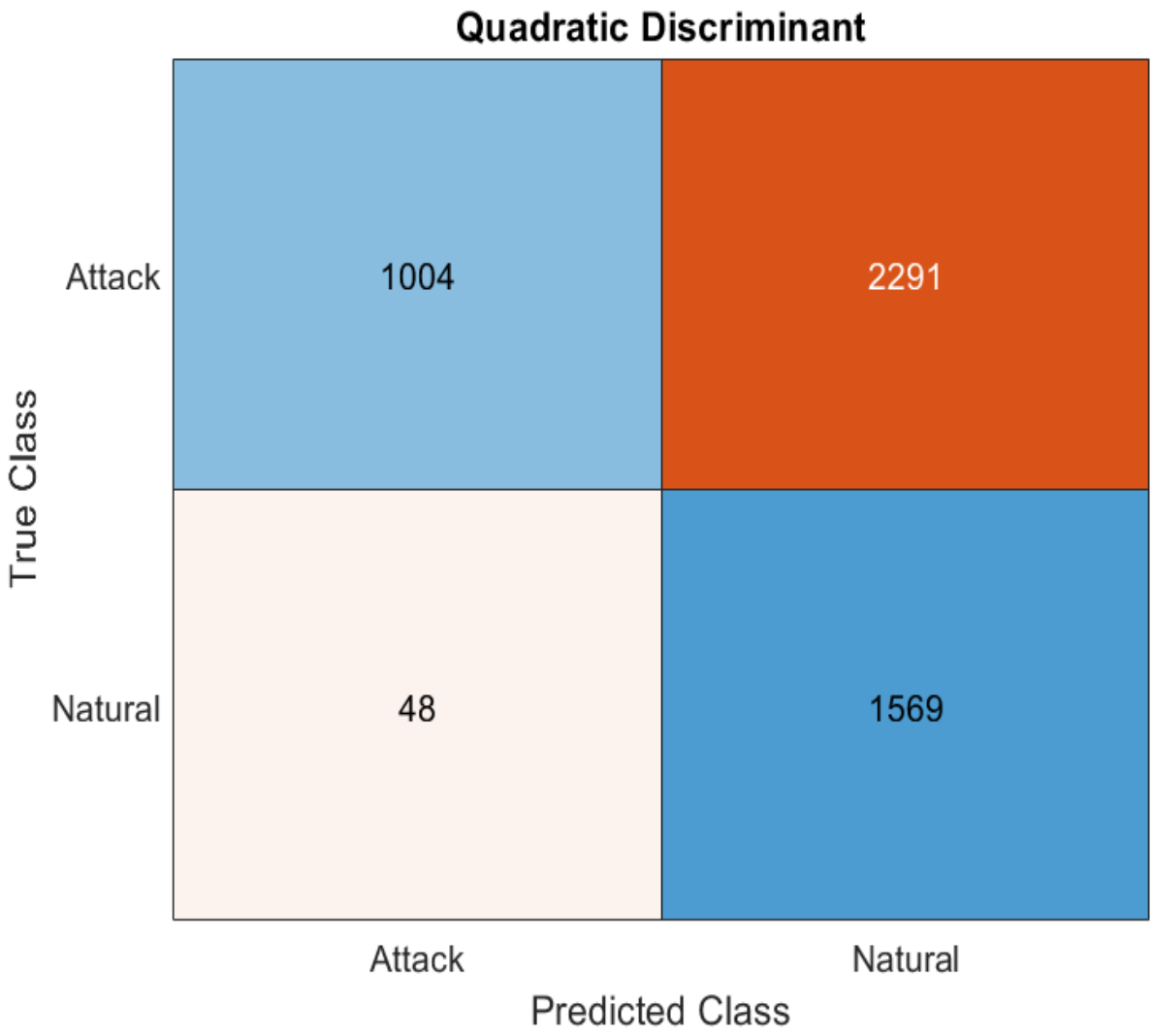


Figure 4.20 Plot of Confusion Matrix of the Algorithm with Worst Performance in ORNL (Power Grid) SCADA Dataset

The confusion matrix outcomes as shown in Figure 4.20 shows:

True Positive (TP) – 1004

True Negative (TN) – 1569

False Positive (FP) – 48

False Negative (FN) – 2291

The Receiver Operator Characteristics (ROC) is another graphical way of evaluating the performance of a machine learning algorithm.

Receiver Operator Characteristics (ROC) graph was used to provide a simple summary of the information from the Confusion Matrix.

The **True Positive Rate** is shown on the y-axis and this is the **sensitivity** of the model

$$\text{True Positive Rate} = \text{Sensitivity} = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

The **False Positive Rate** is shown on the x-axis and this is same as **1 - specificity** of the model

$$\text{False Positive Rate} = 1 - \text{Specificity} = \frac{False\ Positives}{False\ Positives + True\ Negatives}$$

$$\text{False Negative Rate} = \frac{False\ Negatives}{False\ Negatives + True\ Positives}$$

FNR is when the model fails to detect an anomalyy and classifies it as normal.
The Area Under the Curve AUC, which makes it easy to compare from one ROC to another was also shown in the plots.

Accuracy or Classification Rate (CR) is the measure of the accuracy of the IDS in detecting anomaly in traffic transmission which is a ratio of all the True instances as against all the other instances

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

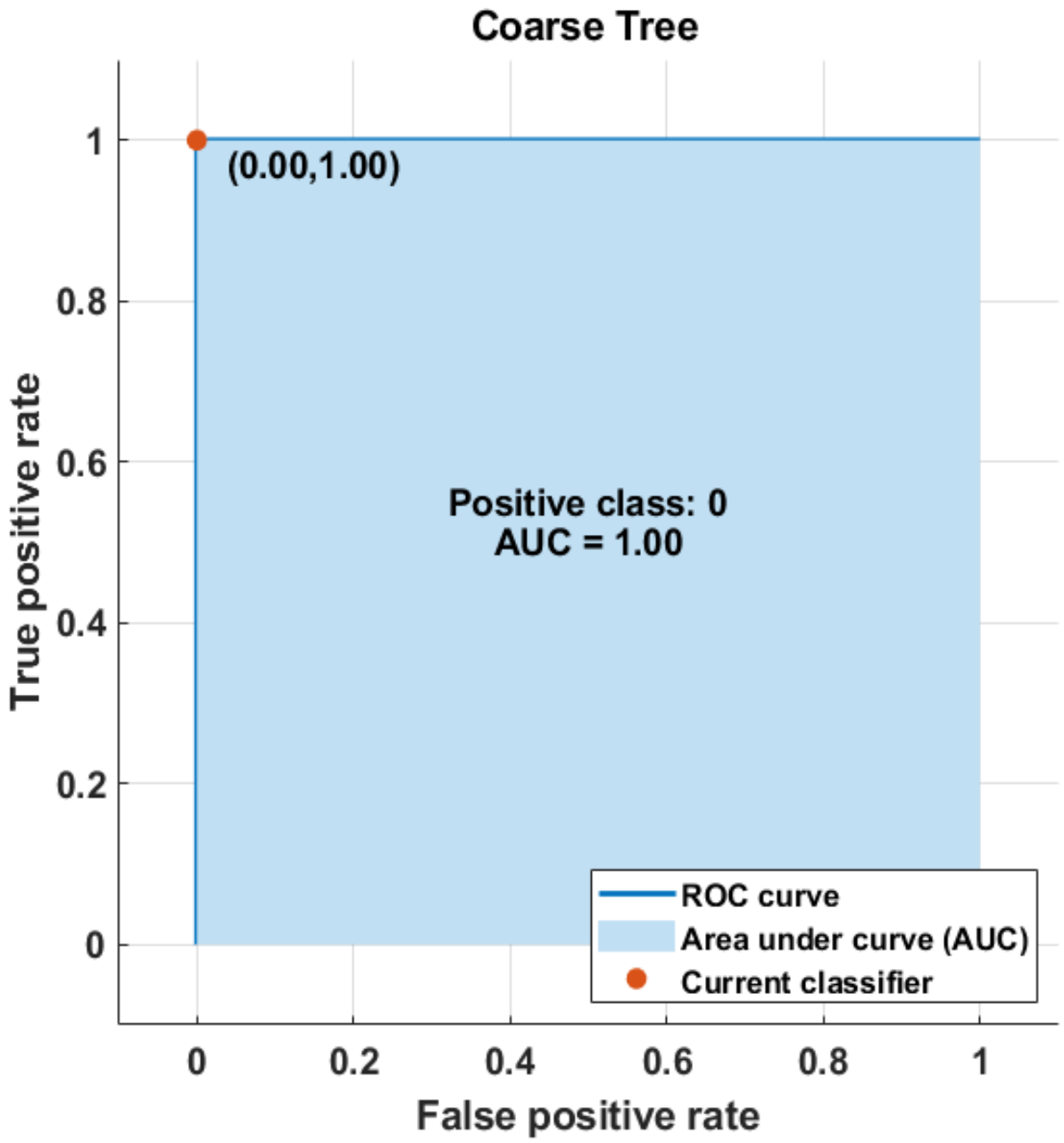


Figure 4.21 Plot of ROC of the Tree Algorithm with Best Performance in SCADA Pressure Dataset.

The Plot of ROC of Figure 4.21 shows AUC of the Coarse Tree Algorithm which presented the Best Performance using the SCADA Pressure Dataset.

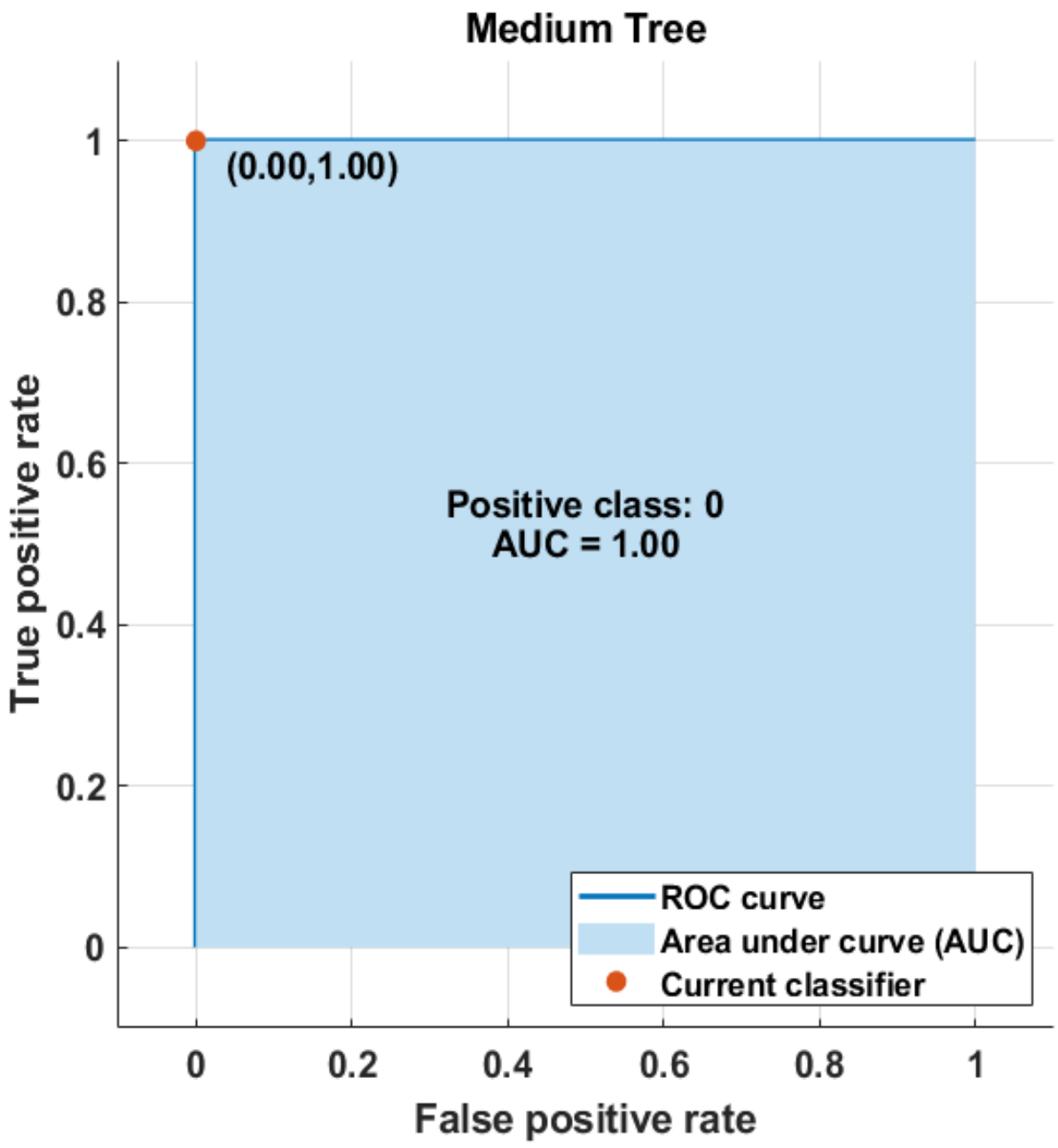


Figure 4.22 Plot of ROC of the Tree Algorithm with Best Performance in WUSTL-SCADA-2018 Dataset.

The Plot of ROC of Figure 4.22 shows AUC of the Medium Tree Algorithm which presented the Best Performance using the WUSTL-SCADA-2018 Dataset.

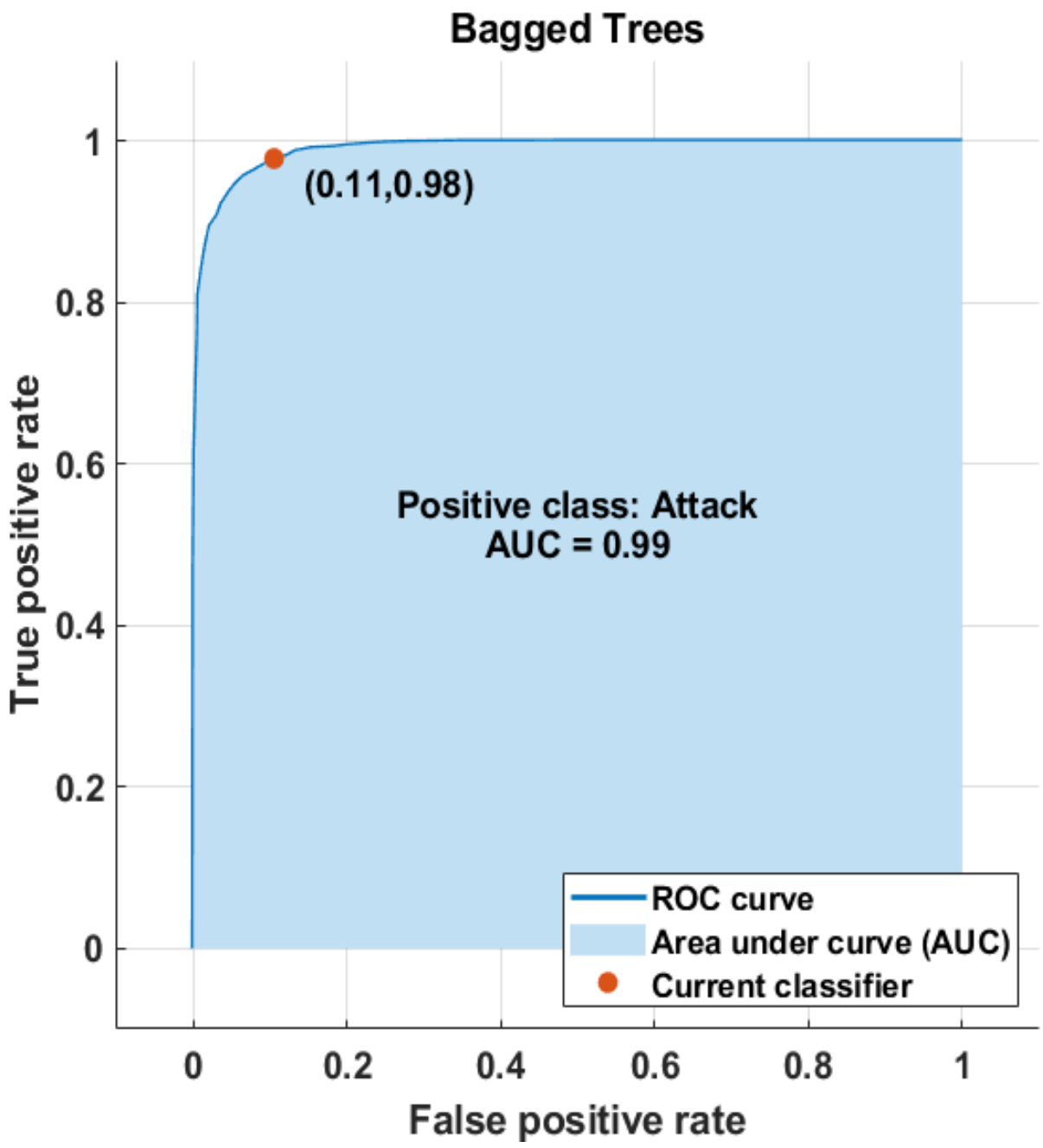


Figure 4.23 Plot of ROC of the Tree Algorithm with Best Performance in ORNL (Power Grid) SCADA Dataset

The Plot of ROC of Figure 4.23 shows AUC of the Bagged Tree Algorithm which presented the Best Performance using the ORNL (Power Grid) SCADA Dataset.

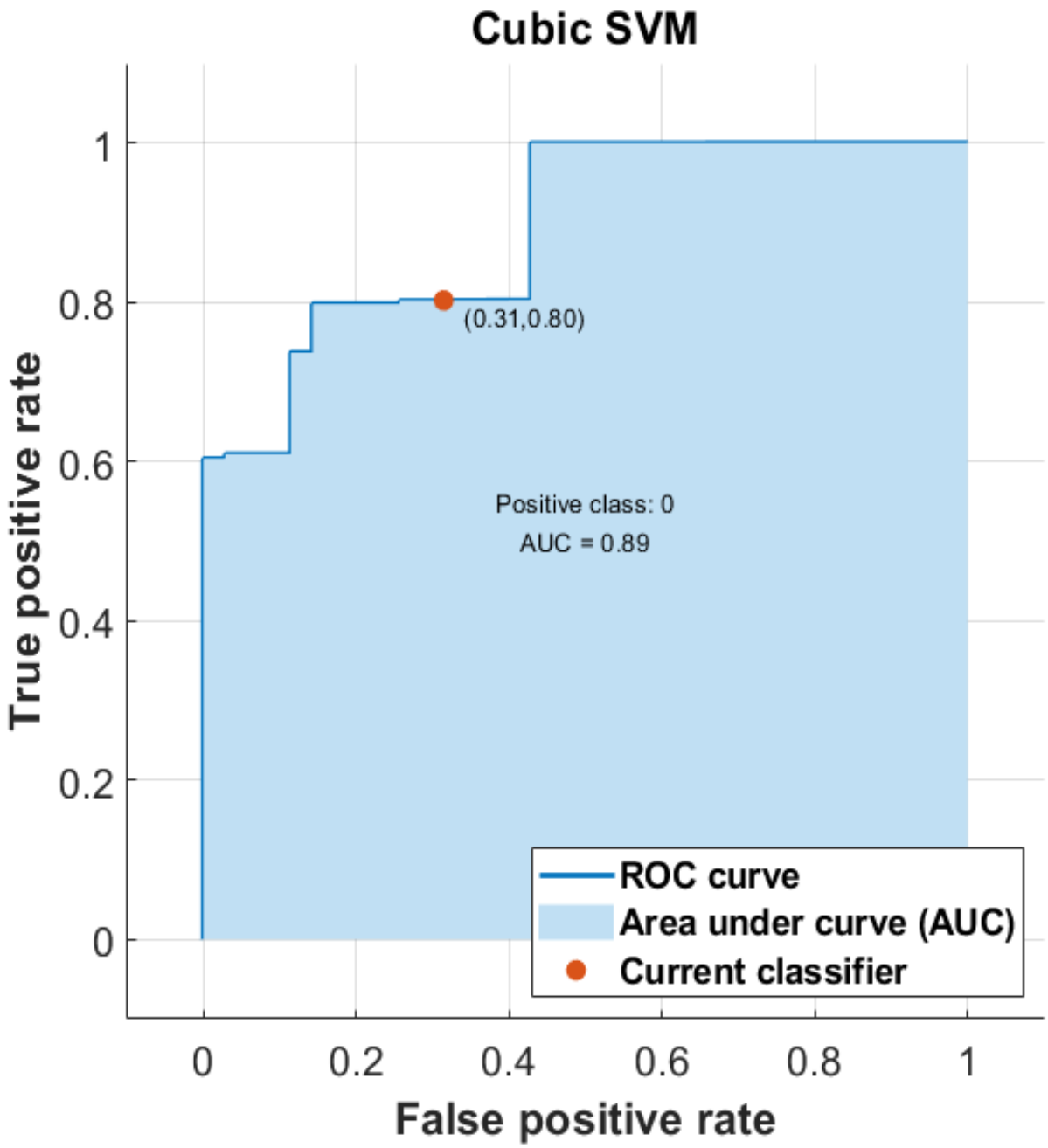


Figure 4.24 Plot of ROC of the Algorithm with Worst Performance in SCADA Pressure Dataset.

The Plot of ROC of Figure 4.24 shows AUC of the Cubic SVM Algorithm which presented the Worst Performance using the SCADA Pressure Dataset.

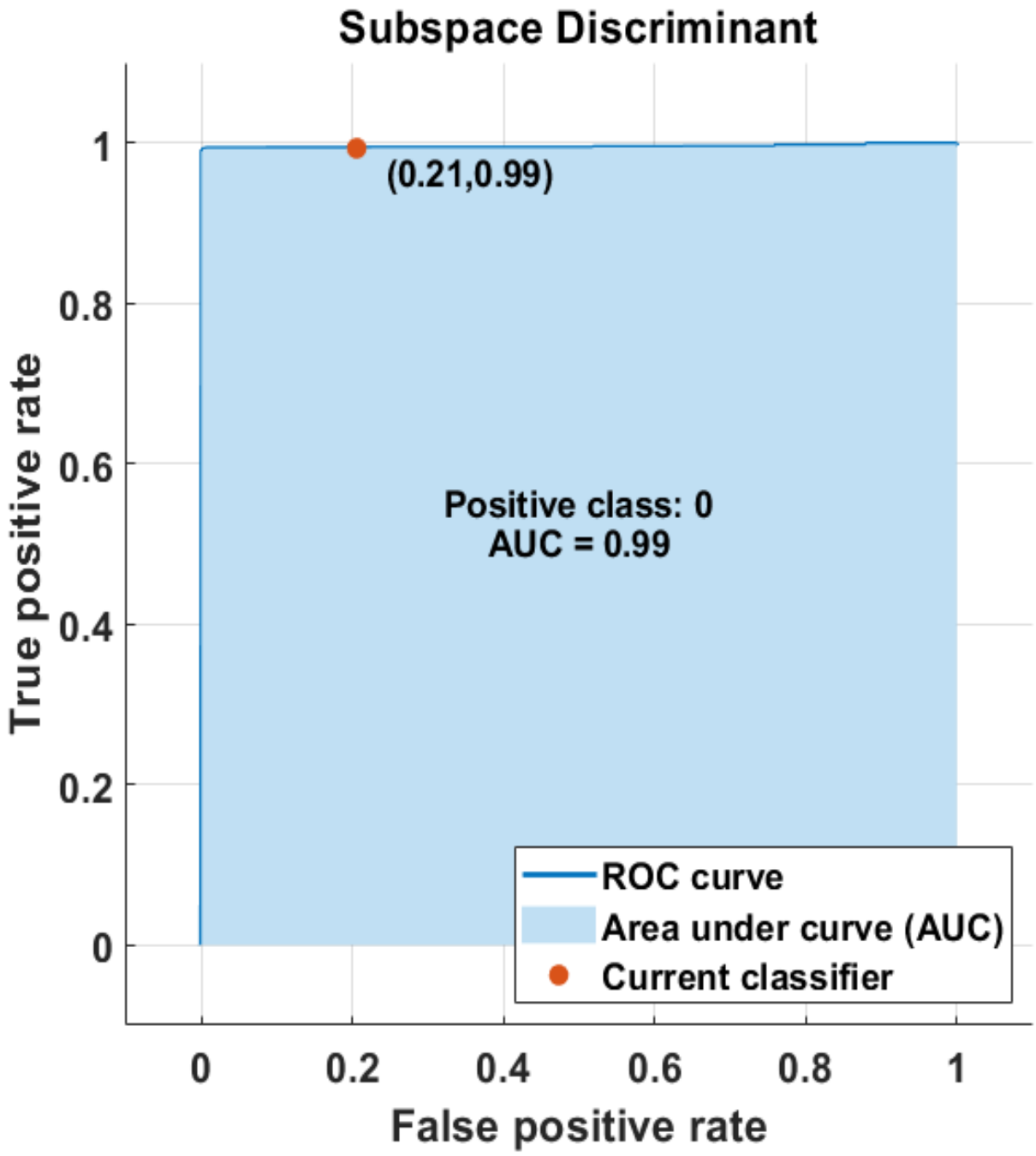


Figure 4.25 Plot of ROC of the Algorithm with Worst Performance in WUSTL-SCADA-2018 Dataset.

The Plot of ROC of Figure 4.25 shows AUC of the Subspace Discriminant Algorithm which presented the Worst Performance using the WUSTL-SCADA-2018 Dataset.

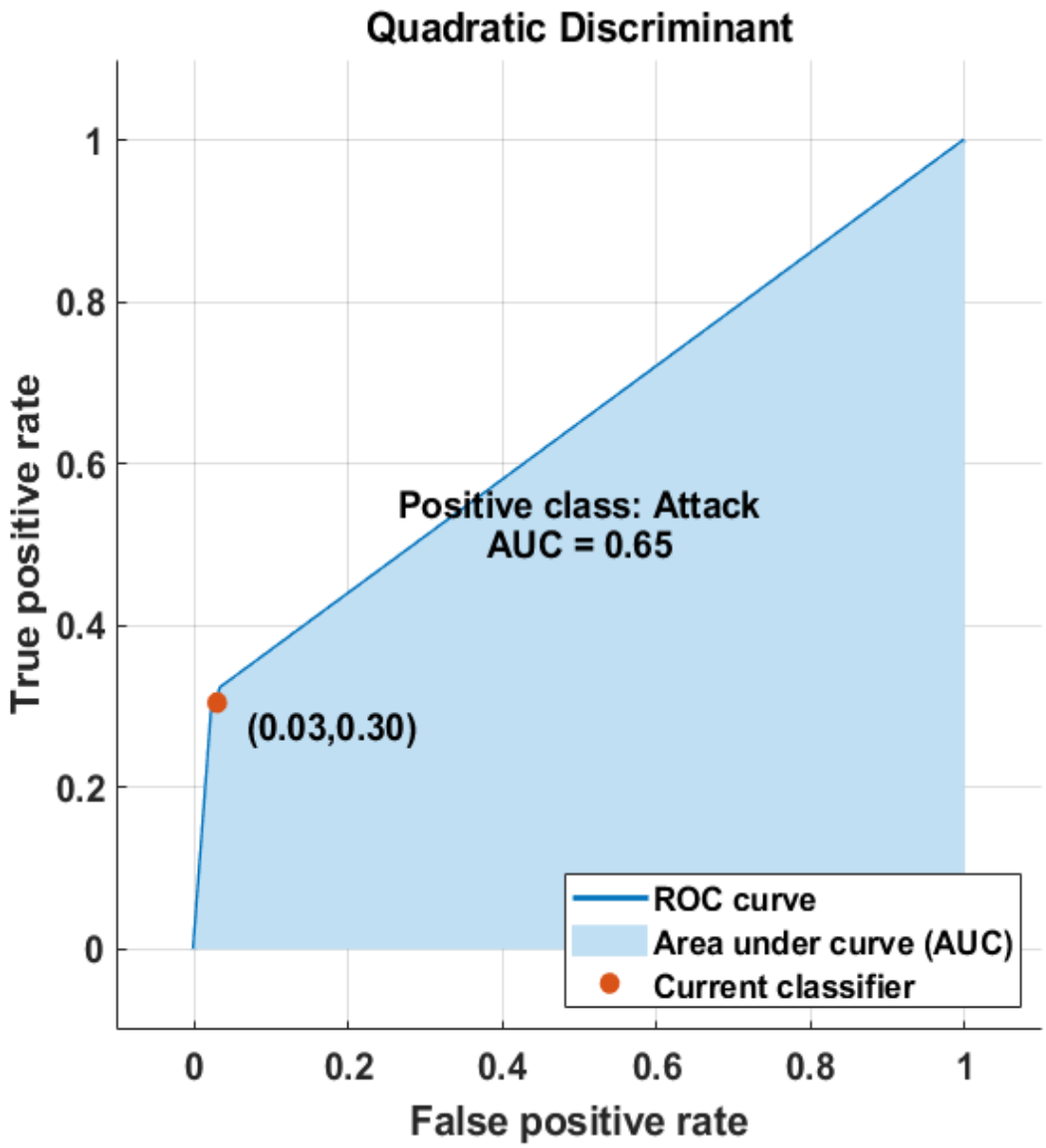


Figure 4.26 Plot of ROC of the Algorithm with Worst Performance in ORNL (Power Grid) SCADA Dataset.

The Plot of ROC of Figure 4.26 shows AUC of the Quaratic Discriminant Algorithm which presented the Worst Performance using the WUSTL-SCADA-2018 Dataset. From the numerous simulations ran, it was carefully observed that best algorithm to detect anomalies based on the available dataset with data size of 68,722 is the Coarse Tree algorithm. The accuracy of the model is 100% with 0 False Alarm Rate and prediction speed of 1000000 observations/second. The computation time is 0.45488.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

At the successful completion of the research, the following conclusion were drawn:

a. The detection and prevention of amorphous cyber-attacks in process control networks of oil and gas installation was achieved using the outlined methods as planned.
b. During the course of this research, different methods were used to achieve the similar result which include: development of mathematical models to simulate attack models, design a centralized and distributed PCN architecture and to design a centralized and distributed PCN with attack detection, identification and monitoring using machine learning.
c. There is a possibility of process safety incidence which could be in the form of loss of containment, fire and explosion if cyber-attacks are not detected early enough. Typical example is a situation where there is inability to detect extreme low values of pressure in the 3-phase separator, this could lead to collapse of the vessel thereby exposing the highly volatile process fluid. On the other hand, the over-pressurization of the process vessel with an assumption that the PSV fails to activate, may lead to the explosion of the vessel with the release of the pressurized volatile process fluid and this will likely lead to fire and further escalation.
d. The selection of the appropriate machine learning algorithm for the model development required the comparison of the different Performance Metrics used in evaluating the different models which served as a guide in the selection of the best model. The performance metrics include: Accuracy of the model, Receiver Operating Characteristics (ROC), Confusion Matrix (CM), Training time, The model's Mis-Classification Error (MCE), Prediction Speed, False Alarm Rate, Computation Time and so on.

e. It was observed that Isolation Forest algorithm with contamination parameter of 0.01, used less memory and computing resources as well as generated results very fast but detected more of extremely low-pressure values but failed to detect the extreme high values. While the Long Short-Term Memory Models when implemented with same 68,722 dataset with a step of 34361, batch size of 128 and 20 epochs generated a RMSE of 0.090, a threshold of 0.211787877271031924 and was able to detect extremely high-pressure anomalies but failed to detect the extreme low values. Adjusting any or all of the epoch, batch size, time step failed to improve the model. The *k*-Nearest Neighbor Algorithm - was able to detect some anomalies but could not detect the extreme high-pressure values accurately as shown in Figure 4.9. The accuracy of the algorithm with this dataset is about 76%.
f. A Plot of Confusion Matrix of the Tree Algorithm with Best Performance using the 68,722 real-time SCADA Pressure Dataset shows zero false positives as compared to other WUSTL and ORNL datasets used by other researchers which produced 141 and 170 false positives respectively.
g. A plot of Receiver Operator Characteristics (ROC) of the best performed Tree Algorithm using the 68,722 real-time SCADA Pressure Dataset shows coarse tree produced best result with zero false positives and better Area Under Curve (AUC) while WUSTL and ORNL showed in medium tree and bagged tree respectively with lesser AUC.
h. Machine learning Anomaly detection algorithms, proved very effective in identifying, detection and prevention of amorphous intrusions into the PCN of an Oil and Gas infrastructure.
i. From the numerous simulations ran, it was carefully observed that best algorithm to detect anomalies based on the available dataset with data size of 68,722 is the Coarse Tree algorithm. The accuracy of the model is 100% with 0 False Alarm Rate and prediction speed of 1000000 observations/second. The computation time is 0.45488.
j. Comparative analysis of top performing machine learning classifiers using the 68,722 SCADA dataset, WUSTL-SCADA-2018 dataset and ORNL power grid

dataset as shown in Table 4.8 indicated that Coarse Tree Algorithm produced significant results when combined parameters are considered. The extensive simulations produced results with high precision, accuracy and reliability with reduced false alarm rates. Hence, the preference for Coarse Tree algorithm in the detection of cyber-attacks in PCN.

k. Comparative analysis of least performing machine learning classifiers as shown in Table 4.9 using the 68,722 SCADA dataset, WUSTL-SCADA-2018 dataset and ORNL power grid dataset as shown in Table 4.9 indicated Cubic Support Vector Machine Algorithm had the least result using the 68,722 SCADA dataset with low accuracy, high false alarm rate, extended prediction speed and prolonged computation time.

l. With these developed models, the successful mitigation of cyber-attacks on Oil and Gas infrastructure will go a long way protecting the PCN and preventing economic sabotage and this has economic benefits world economy.

5.2 Recommendations

Based on the findings and conclusions from this research the following recommendations are made:

a. More data samples: In order to develop very accurate and reliable anomaly detection models, the algorithms must be trained, validated and tested using sufficient data samples. The data used for the development of the models contained 68,722 data sets from a real-time PCN. Potent data sources are recommended for a more reliable result. At any point in time a real time PCN should have potent data source of not less than 68,722 to achieve accurate and reliable results.

b. Hardware and CPU Limitations: Adequate care was taken in choosing the algorithms used to simulate cyber-attack models, as some algorithms seem to use less memory space and

less computation resources. The ability to develop and simulate accurate models depends on the computational resources of the hardware used. Within the course of the research some algorithms could not run effectively on the local computer due to limited CPU and RAM resources (Recommended CPU Speed of 3GHz, RAM 16GB),

c. Model Integrity and trust**:** Implementation of machine learning using different algorithms poses some critical challenges which may be associated with the integrity of the models. It is important to use algorithms from trusted and reliable sources as attackers can modify algorithms with malicious intent and thereby expose the details of the application for more stealthy and devastating attacks.

d. online resources (Google Colab and Kaggle) were used to be able to access GPUs with larger computation resources. Graphics Processing Units (GPUs) are quite expensive to purchase but online resources have made it available at no cost. While using the online resources, there is need to ensure data privacy and security of the original work to avoid pilfering of the model details. Where possible, it will be good to invest in the purchase of high-end computational hardware.

e. There are other challenges with ready-made algorithms which include limited possibility of model tuning and manipulations by end users, model transparency and associated trust issues.

f. Innovation in the design of PCN Architecture: Integration of a dedicated system which has the capability to detect and prevent cyber-attacks through the identification of anomalies in data communication will go a long way in shielding critical infrastructures from malicious cyber-attacks. This will increase the robustness and efficiency of the system,

reduce downtimes as a result of cyber-attacks and save the possible huge cost implication of cyber-attacks.

g. Alarm categorization and console operator training: As shown in the course of the work, the system developed has the capability of identifying anomalies based on the thresholds set. It was also shown that the system can capture extremely low values as anomalies which is likely the situation during process upsets and facility shutdown. The alarm system can be configured to annunciate for the different scenarios with different tones say High Alarm, Low Alarm, Medium Alarm, as this goes a long way to help the console operator know how and when to react based on preset conditions. Extreme high values should trigger thorough investigation as it could be cyber-attacks, as the process lines and vessels are not meant to handle such values without explosion. Adequate training is to be provided to the console operator to be able to understand the criticality of these alarms and a standard procedure on what to do based on the operating philosophy.

5.3 Contributions to Knowledge

a. This real-time modelled system was able to vividly distinguish between normal shutdown, process upsets and cyber-attacks induced shutdowns. This distinction is important for Oil and Gas facilities with 99% runtime, as its negligence can lead to process safety incidence and may further escalate culminating to fatality and economic loss to the host nation with long recovery time.
b. The research concluded, was able to establish various modes of cyber-attacks identifying points of vulnerabilities, their safety mitigative measures and obvious consequences associated with their occurrences in Oil and Gas industries. It further proffered solutions to avert their heinous hazardous effects on people, asset and the environment respectively.

c. The result of this research, was able to unravel and/or determine the prevalent points of vulnerabilities where data could be tampered with in Oil and Gas industry flow lines.
d. Machine learning technique was deployed for the detection and prevention of cyber-attacks on facilities in Oil and Gas industry.

References

Abdu Sabir, B. M., Elamin, I. H., & Sadiq, H. R. (2022). Dynamic Modelling and Simulation of A Three-Phase Gravity Separator. *Journal of Karary University for Engineering and Science*, *11*(1), 1–19. https://doi.org/10.54388/jkues.v2i2.190

Abou el Kalam, A. (2021). Securing SCADA and critical industrial systems: From needs to security mechanisms. *International Journal of Critical Infrastructure Protection*, *32*, 100394. https://doi.org/10.1016/j.ijcip.2020.100394

Abrar, I., Ayub, Z., Masoodi, F., & Bamhdi, A. M. (2020). A Machine Learning Approach for Intrusion Detection System on NSL-KDD Dataset. *Proceedings - International Conference on Smart Electronics and Communication, ICOSEC 2020*, *Icosec*, 919–924. https://doi.org/10.1109/ICOSEC49089.2020.9215232

Ahakonye, L. A. C., Nwakanma, C. I., Lee, J. M., & Kim, D.-S. (2023a). Agnostic CH-DT Technique for SCADA Network High-Dimensional Data-Aware Intrusion Detection System. *IEEE Internet of Things Journal*, *January*, 1–1. https://doi.org/10.1109/jiot.2023.3237797

Ahakonye, L. A. C., Nwakanma, C. I., Lee, J. M., & Kim, D. S. (2021). Efficient Classification of Enciphered SCADA Network Traffic in Smart Factory Using Decision Tree Algorithm. In *IEEE Access* (Vol. 9, pp. 154892–154901). https://doi.org/10.1109/ACCESS.2021.3127560

Ahakonye, L. A. C., Nwakanma, C. I., Lee, J. M., & Kim, D. S. (2023b). SCADA intrusion detection scheme exploiting the fusion of modified decision tree and Chi-square feature selection. *Internet of Things (Netherlands)*, *21*(December 2022), 100676. https://doi.org/10.1016/j.iot.2022.100676

Al-Abassi, A., Karimipour, H., Dehghantanha, A., & Parizi, R. M. (2020). An ensemble deep learning-based cyber-attack detection in industrial control system. *IEEE Access*, *8*, 83965–83973. https://doi.org/10.1109/ACCESS.2020.2992249

Al-rabiaah, S. (2018). The Stuxnet Virus of 2010 As an Example of A APT and Its

Recent Variances. *2018 21st Saudi Computer Society National Computer Conference (NCC)*, 1–5. https://doi.org/10.1109/NCG.2018.8593143

Al-Rabiaah, S. (2018). The “Stuxnet” Virus of 2010 As an Example of A “APT” and Its “Recent” Variances. *21st Saudi Computer Society National Computer Conference, NCC 2018*, 1–5. https://doi.org/10.1109/NCG.2018.8593143

Alanazi, M., Mahmood, A., & Chowdhury, M. J. M. (2023). SCADA vulnerabilities and attacks: A review of the state- of- the- art and open issues. *Computers and Security*, *125*. https://doi.org/10.1016/j.cose.2022.103028

Alhaidari, F. A., & Al-Dahasi, E. M. (2019). New approach to determine DDoS attack patterns on SCADA system using machine learning. *2019 International Conference on Computer and Information Sciences, ICCIS 2019*, 1–6. https://doi.org/10.1109/ICCISci.2019.8716432

Alston, R. (2020). Process Control Networks Secure Architecture Design. In *Honeywell Process Automation*. https://www.honeywellprocess.com/library/marketing/notes/Secure Network Architecture.pdf

Alves, T., Das, R., & Morris, T. (2018). Embedding Encryption and Machine Learning Intrusion Prevention Systems on Programmable Logic Controllers. *IEEE Embedded Systems Letters*, *10*(3), 99–102. https://doi.org/10.1109/LES.2018.2823906

Ambonati, V. (2017). *Unsupervised Anomaly Detection | Kaggle*. https://www.kaggle.com/code/victorambonati/unsupervised-anomaly-detection/notebook

American Petroleum Institute. (2005). Security Guideline for the Petroleum Industry. *American Petroleum Institute*, *3rd Editio*(April 2005), Pages: 1, 31. http://www.nj.gov/dep/rpp/brp/security/downloads/API Security Guidance 3rd Edition.pdf

Amin, S., Litrico, X., Sastry, S., & Bayen, A. (2013a). Cyber security of water SCADA systems-part I: Analysis and experimentation of stealthy deception attacks. *IEEE Transactions on Control Systems Technology*, *21*(5), 1963–1970.

Amin, S., Litrico, X., Sastry, S., & Bayen, A. (2013b). Cyber security of water SCADA systems-part II: Attack detection using enhanced hydrodynamic models. *IEEE Transactions on Control Systems Technology*, *21*(5), 1679–1693.

Babcock, B. N. (2009). P L C P r o g r a m m i n g w i t h R S L o g i x 5 0 0 0. *Modern Media*.

Beaver, J. M., Borges-Hink, R. C., & Buckner, M. A. (2013). An evaluation of machine learning methods to detect malicious SCADA communications. *Proceedings - 2013 12th International Conference on Machine Learning and Applications, ICMLA 2013*, *2*, 54–59. https://doi.org/10.1109/ICMLA.2013.105

Bencsáth, B., Ács, G., Molnár, G., Vaspöri, G., & Buttyán, L. (2015). *Duqu 2.0: A comparison to Duqu* (Vol. 0). http://www.crysys.hu/duqu2/duqu2.pdf

Bencsath, B., Pek, G., Buttyan, L., & Felegyhazi, M. (2011). Duqu: A Stuxnet-like malware found in the wild. In *CrySyS Lab Technical Report* (Vol. 14). https://www.crysys.hu/publications/files/bencsathPBF11duqu.pdf

Bierbrauer, D. A., Chang, A., Kritzer, W., & Bastian, N. D. (2021). Anomaly Detection in Cybersecurity: Unsupervised, Graph-Based and Supervised Learning Methods in Adversarial Environments. *Cryptography and Security (Cs.CR); Artificial Intelligence (Cs.AI); Machine Learning (Stat.ML)*.

CISA. (2020). *Ransomware Impacting Pipeline Operations (Alert AA20-049A)| CISA*. https://us-cert.cisa.gov/ncas/alerts/aa20-049a

CISA, & FBI. (2021). Darkside Ransomware : Best Practices for Preventing Business Disruption from Ransomware Attacks. In *Product ID: AA21-131A*. https://us-cert.cisa.gov/ncas/alerts/aa21-131a

Cisco.com worldwide. (2019a). *Cyber Attack - What Are Common Cyberthreats? - Cisco*. Cisco.Com. https://www.cisco.com/c/en/us/products/security/common-cyberattacks.html

Cisco.com worldwide. (2019b). *What Is a DDoS Attack? Distributed Denial of Service - Cisco*. Cisco.Com. https://www.cisco.com/c/en/us/products/security/what-is-a-ddos-attack.html

Cisco.com Worldwide. (2019). *What Is Phishing? - Types of Phishing Attacks - Cisco*. Cisco.Com. https://www.cisco.com/c/en/us/products/security/email-security/what-is-phishing.html

Combita, L. F., Cardenas, A. A., & Quijano, N. (2017). Mitigation of sensor attacks on legacy industrial control systems. *2017 IEEE 3rd Colombian Conference on Automatic Control, CCAC 2017 - Conference Proceedings*, *2018-Janua*, 1–6. https://doi.org/10.1109/CCAC.2017.8276404

CrySyS Lab. (2012). sKyWIper (a.k.a. Flame a.k.a. Flamer): A complex malware for targeted attacks. In *Laboratory of Cryptography and System Security (CrySyS Lab): Vol. Version 1*. www.crysys.hu

Department of Homeland Security. (2017a). *Energy Sector | Homeland Security*. Department of Homeland Security. https://www.dhs.gov/energy-sector

Department of Homeland Security. (2017b). *What Is Critical Infrastructure? | Homeland Security*. Department of Homeland Security. https://www.dhs.gov/what-critical-infrastructure

Department of Homeland Security. (2018). *Critical Infrastructure Sectors | Homeland Security*. Department of Homeland Security. https://www.dhs.gov/critical-infrastructure-sectors

El Naqa, I., & Murphy, M. J. (2015). What Is Machine Learning? In *Machine Learning in Radiation Oncology* (pp. 3–11). https://doi.org/10.1007/978-3-319-18305-3_1

Elgin M. Brunner, & Suter, M. (2008). International CIIP Handbook 2008/2009. *Center for Security Studies, ETH Zurich, 2008*. http://e-collection.library.ethz.ch/view/eth:31095

Elmrabit, N., Zhou, F., Li, F., & Zhou, H. (2020). Evaluation of machine learning algorithms for Anomaly Detection. *International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*, 65. https://doi.org/10.1109/CyberSecurity49315.2020.9138871

Forbes. (2017). *Cyber Security Risks To Be Aware Of In The Oil And Gas Industries*. Forbes Media LLC. https://www.forbes.com/

Gönen, S., Sayan, H. H., Yılmaz, E. N., Üstünsoy, F., & Karacayılmaz, G. (2020). False data injection attacks and the insider threat in smart systems. *Computers and Security*, *97*, 101955. https://doi.org/10.1016/j.cose.2020.101955

Han, S., Hu, X., Huang, H., Jiang, M., & Zhao, Y. (2022). ADBench: Anomaly Detection Benchmark. *NeurIPS 2022*, *NeurIPS*. http://arxiv.org/abs/2206.09426

Hill, R. (2015). Dealing with cyber security threats: International cooperation, ITU, and WCIT. *2015 7th International Conference on Cyber Conflict: Architectures in Cyberspace*, *2015-Janua*(2015 © NATO CCD COE Publications, Tallinn), Pages: 119-134. https://doi.org/10.1109/CYCON.2015.7158473

Hunga, M. O., & Adishi, E. (2017). Oil Theft, Illegal Bunkering and Pipeline Vandalism: It's Impact on Nigeria Economy, 2015 -2016. *IIARD International Journal of Economics and Business Management ISSN*, *3*(2), 2489–65. www.iiardpub.org

Husák, M., Komárková, J., Bou-Harb, E., & Čeleda, P. (2019). Survey of attack projection, prediction, and forecasting in cyber security. *IEEE Communications Surveys and Tutorials*, *21*(1), 640–660. https://doi.org/10.1109/COMST.2018.2871866

IBM Cloud Education. (2020a, July 15). *What is Machine Learning? | IBM*. IBM

Cloud Education. https://www.ibm.com/cloud/learn/machine-learning

IBM Cloud Education. (2020b, August 19). *Supervised Learning*. IBM Cloud Education. https://www.ibm.com/cloud/learn/supervised-learning

IBM Cloud Education. (2020c, September 21). *Unsupervised Learning*. IBM Cloud Education. https://www.ibm.com/cloud/learn/unsupervised-learning

International Standard Organization. (2018). ISO/IEC 27000:2018(en), Information technology — Security techniques — Information security management systems — Overview and vocabulary. *ISO/IEC 27000:2018 (E), 5th Editio*, Pages: 1-5. https://www.iso.org/obp/ui/#iso:std:iso-iec:27000:ed-5:v1:en

International Telecommunication Union. (2008). Series X: Data Networks, Open System Communications and Security: Telecommunication security - Overview of cybersecurity. *ITU-T X.1205 Recommendation*, *1205*(Rec. ITU-T X.1205 (04/2008)), Pages: 2-3. https://www.itu.int/rec/T-REC-X.1205-200804-I

Irmak, E., & Erkek, I. (2018). An overview of cyber-attack vectors on SCADA systems. *6th International Symposium on Digital Forensic and Security, ISDFS 2018 - Proceeding*, *2018-Janua*, 1–5. https://doi.org/10.1109/ISDFS.2018.8355379

Jang-jaccard, J., & Nepal, S. (2014). Journal of Computer and System Sciences - A survey of emerging threats in cybersecurity. *Journal of Computer and System Sciences*, *80*(5), Pages: 973-993. https://doi.org/10.1016/j.jcss.2014.02.005

Jeff Melnick. (2018). *Top 10 Most Common Types of Cyber Attacks*. Https://Blog.Netwrix.Com/2018/05/15/Top-10-Most-Common-Types-of-Cyber-Attacks/. https://blog.netwrix.com/2018/05/15/top-10-most-common-types-of-cyber-attacks/

Jithish, J., & Sankaran, S. (2018). Securing networked control systems: Modeling attacks and defenses. *2017 IEEE International Conference on Consumer Electronics-Asia, ICCE-Asia 2017*, *2018-Janua*, 7–11.

https://doi.org/10.1109/ICCE-ASIA.2017.8309317

Johnson, D. (1999). *Open Systems in Process Control Are They the Answer? | Control Engineering*. Control Engineering. https://www.controleng.com/articles/open-systems-in-process-control-are-they-the-answer/

Joloudari, J. H., Haderbadi, M., Mashmool, A., Ghasemigol, M., Band, S. S., & Mosavi, A. (2020). Early detection of the advanced persistent threat attack using performance analysis of deep learning. *IEEE Access*, *8*, 186125–186137. https://doi.org/10.1109/ACCESS.2020.3029202

Jonach, T., Jordan, C., Haddadi, B., & Harasek, M. (2022). Modelling and Simulation of 3-Phase Separators in the Oil and Gas Industry with Emphasis on Water Quality. *Chemical Engineering Transactions*, *94*(May), 1009–1014. https://doi.org/10.3303/CET2294168

Kaspersky. (2019). *APT trends report Q2 2019*. https://securelist.com/apt-trends-report-q2-2019/91897/

Kaspersky. (2022). *APT trends report Q2 2022*. https://securelist.com/apt-trends-report-q2-2022/106995/

Kaspersky Lab. (2015). *The Duqu 2.0 - Technical Details (V2.1)* (Vol. 1, Issue June). https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/07205202/The_Mystery_of_Duqu_2_0_a_sophisticated_cyberespionage_actor_returns.pdf

Kaspersky Lab. (2019). *Kaspersky ICS Security Assessment*. Kaspersky Lab 2019. https://media.kaspersky.com/en/business-security/enterprise/ics-security-assessment-datasheet.pdf

Katrina Wakefield. (2021). *A guide to the types of machine learning algorithms | SAS UK*. SAS Institute UK. https://www.sas.com/en_gb/insights/articles/analytics/machine-learning-algorithms.html

Khraisat, A., & Alazab, A. (2021). A critical review of intrusion detection systems in the internet of things: techniques, deployment strategy, validation strategy, attacks, public datasets and challenges. *Cybersecurity*, *4*(1). https://doi.org/10.1186/s42400-021-00077-7

Krishnan, V., & Pasqualetti, F. (2021). Data-Driven Attack Detection for Linear Systems. *IEEE Control Systems Letters*, *5*(2), 671–676. https://doi.org/10.1109/LCSYS.2020.3005102

Kuhlman, D. (2013). A Python Book. *A Python Book*, 1–227.

Kulugh, V. E., Mbanaso, U. M., & Chukwudebe, G. (2022). Cybersecurity Resilience Maturity Assessment Model for Critical National Information Infrastructure. *SN Computer Science*, *3*(3). https://doi.org/10.1007/s42979-022-01108-x

Lanotte, R., Merro, M., Muradore, R., & Vigano, L. (2017). A Formal Approach to Cyber-Physical Attacks. *Proceedings - IEEE Computer Security Foundations Symposium*, 436–450. https://doi.org/10.1109/CSF.2017.12

Malviya, N. (2019). *Process control network (PCN) evolution | Infosec Resources*. Infosec Institute, Inc. https://resources.infosecinstitute.com/topic/process-control-network-pcn-evolution/

Marchetti, M., Guido, A., Pierazzi, F., & Colajanni, M. (2016). Countering Advanced Persistent Threats through security intelligence and big data analytics. *International Conference on Cyber Conflict, CYCON*, *2016-Augus*, 243–261. https://doi.org/10.1109/CYCON.2016.7529438

Merriam-Webster Dictionary. (2018). *Cyberattack | Definition of Cyberattack by Merriam-Webster*. 2018 Merriam-Webster, Incorporated. https://www.merriam-webster.com/dictionary/cyberattack

Merriam-Webster Dictionary. (2019). *Amorphous | Definition of Amorphous by Merriam-Webster Dictionary*. https://www.merriam-webster.com/dictionary/amorphous#synonyms

Mohammed, Abubakar Sadiq and Saxena, Neetesh and Rana, O. (2022). Wheels on the Modbus - Attacking ModbusTCP Communications. *Proceedings of the 15th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 288–289. https://doi.org/https://doi.org/10.1145/3507657.3529654

Montanari, L., & Querzoni, L. (2014). Critical Infrastructure Protection : Threats , Attacks and Countermeasures. *TENACE*.

Nasr, P. M., & Varjani, A. Y. (2014a). Alarm based anomaly detection of insider attacks in SCADA system. *Smart Grid Conference 2014, SGC 2014*. https://doi.org/10.1109/SGC.2014.7090881

Nasr, P. M., & Varjani, A. Y. (2014b). Petri net model of insider attacks in SCADA system. *2014 11th International ISC Conference on Information Security and Cryptology, ISCISC 2014*, 55–60. https://doi.org/10.1109/ISCISC.2014.6994022

Ndonda, G. K., & Sadre, R. (2022). Exploiting the Temporal Behavior of State Transitions for Intrusion Detection in ICS/SCADA. *IEEE Access*, *10*(September), 111171–111187. https://doi.org/10.1109/ACCESS.2022.3213080

Ndubuaku, M. U., Anjum, A., & Liotta, A. (2019). Unsupervised anomaly thresholding from reconstruction errors. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11874 LNCS*, 123–129. https://doi.org/10.1007/978-3-030-34914-1_12

Nisioti, A., Mylonas, A., Yoo, P. D., & Katos, V. (2018). From intrusion detection to attacker attribution: A comprehensive survey of unsupervised methods. *IEEE Communications Surveys and Tutorials*, *20*(4), 3369–3388. https://doi.org/10.1109/COMST.2018.2854724

Nitesh Malviya. (2019, August 27). *Process Control Network (PCN) Evolution*. Infosec Resources. https://resources.infosecinstitute.com/topic/process-control-network-pcn-evolution/

Nwakanma, C. I., Ahakonye, L. A. C., Njoku, J. N., Eze, J., & Kim, D. S. (2022). Effective Industrial Internet of Things Vulnerability Detection Using Machine Learning. *Proceedings of the 5th International Conference on Information Technology for Education and Development: Changing the Narratives Through Building a Secure Society with Disruptive Technologies, ITED 2022, November*, 0–8. https://doi.org/10.1109/ITED56637.2022.10051622

Offshore Energy Today. (2015). *Top 10 cyber security threats for oil and gas industry | Offshore Energy Today*. DNV GL. https://www.offshoreenergytoday.com/top-10-cyber-security-threats-for-oil-and-gas-industry/

Ogu, R. E., Achumba, I. E., Okoronkwo, C. D., Chukwudebe, G. A., & Chukwuchekwa, N. (2022). An IoT Solution for Air Quality Monitoring and Hazard Identification for Smart City Development. *Proceedings of the 2022 IEEE Nigeria 4th International Conference on Disruptive Technologies for Sustainable Development, NIGERCON 2022*. https://doi.org/10.1109/NIGERCON54645.2022.9803129

Ogu, R. E., Chukwudebe, G. A., Achumba, I., Chukwuchekwa, N., & Ezenugu, I. A. (2022). A Robust IoT-based Air Quality Monitoring Node for Multi-Location Deployment. *International Journal of Engineering Research and Technology*, *11*(03), 146–151.

Okada, K. (2021, March). Protecting Industrial Infrastructure from Cyber-attacks Toshiba's " Security Operation Center Services for Industrial Control Systems" | DiGiTAL T-SOUL | TOSHIBA DIGITAL SOLUTIONS CORPORATION. *Toshiba Digital Solutions Corporation Volume 36*. https://www.global.toshiba/ww/company/digitalsolution/articles/tsoul/36/003.html

Okafor, K. C., Ndinechi, M. C., & Misra, S. (2022). Cyber-physical network architecture for data stream provisioning in complex ecosystems. *Transactions*

on Emerging Telecommunications Technologies, *33*(4), 1–31. https://doi.org/10.1002/ett.4407

Oxford Dictionary. (2018). *cyberattack | Definition of cyberattack in English by Oxford Dictionaries*. 2018 Oxford University Press. https://en.oxforddictionaries.com/definition/cyberattack

Parian, C., Guldimann, T., & Bhatia, S. (2020). Fooling the Master: Exploiting Weaknesses in the Modbus Protocol. *Procedia Computer Science*, *171*(2019), 2453–2458. https://doi.org/10.1016/j.procs.2020.04.265

Pasqualetti, F., Dorfler, F., & Bullo, F. (2013). Attack detection and identification in cyber-physical systems. *IEEE Transactions on Automatic Control*, *58*(11), 2715–2729. https://doi.org/10.1109/TAC.2013.2266831

Pasqualetti, F., Dörfler, F., & Bullo, F. (2015). Control-theoretic methods for cyberphysical security: Geometric principles for optimal cross-layer resilient control systems. *IEEE Control Systems*, *35*(1), 110–127. https://doi.org/10.1109/MCS.2014.2364725

Peerlkamp, S., & Nieuwenhuis, M. (2010). Process Control Network Security. *Security*, *1689428*, 1–64. http://www.jbisa.nl/

Polyakov, A. (2019). *Oil and Gas Cyber Security 101*. Infosec Institute. https://erpscan.com/press-center/blog/oil-and-gas-cyber-security-questions-and-answers/

Ponemon Institute. (2017). *The State of Cybersecurity in the Oil & Gas Industry: United States* (Issue February). https://news.usa.siemens.biz/sites/siemensusa.newshq.businesswire.com/files/press_release/additional/Cyber_readiness_in_Oil__Gas_Final_4.pdf

Pu, G., Wang, L., Shen, J., & Dong, F. (2021). A hybrid unsupervised clustering-based anomaly detection method. *Tsinghua Science and Technology*, *26*(2), 146–153. https://doi.org/10.26599/TST.2019.9010051

Python Software Documentation. (2023a). *General Python FAQ — Python 3.11.1 documentation*. Python Software Foundation. https://docs.python.org/3/faq/general.html#what-is-python

Python Software Documentation. (2023b). *The Python Standard Library — Python 3.11.1 documentation*. Python Software Foundation. https://docs.python.org/3/library/

Ramotsoela, D. T., Hancke, G. P., & Abu-Mahfouz, A. M. (2019). Attack detection in water distribution systems using machine learning. *Human-Centric Computing and Information Sciences*, *9*(1). https://doi.org/10.1186/s13673-019-0175-8

Rockwell, A. (2011). RS Logix Emulate5000. *Rockwell, Automation, November*.

Rockwell, A. (2022). Studio 5000 Logix Emulate Getting Results Guide. *Rockwell, Automation, November*.

Rosa, L., Cruz, T., Freitas, M. B. de, Quitério, P., Henriques, J., Caldeira, F., Monteiro, E., & Simões, P. (2021). Intrusion and anomaly detection for the next-generation of industrial automation and control systems. *Future Generation Computer Systems*, *119*, 50–67. https://doi.org/10.1016/j.future.2021.01.033

Sarker, I. H. (2021). Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Computer Science*, *2*(3), 1–21. https://doi.org/10.1007/s42979-021-00592-x

Shang, W., Zeng, P., Wan, M., Li, L., & An, P. (2015). SECURITY AND COMMUNICATION NETWORKS 2016. *Intrusion Detection Algorithm Based on OCSVM in Industrial Control System*, *9:1040–104*(December). https://doi.org/10.1002/sec.1398

Shetty, N. (2021). Vulnerability Assessment for Cybersecurity using Machine learning. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.3884044

Singh, V. K., Ebrahem, H., & Govindarasu, M. (2019). Security Evaluation of Two

Intrusion Detection Systems in Smart Grid SCADA Environment. *2018 North American Power Symposium, NAPS 2018*, 1–6. https://doi.org/10.1109/NAPS.2018.8600548

Smurthwaite, M., & Bhattacharya, M. (2020). Convergence of IT and SCADA: Associated security threats and vulnerabilities. *IOP Conference Series: Materials Science and Engineering*, *790*(1). https://doi.org/10.1088/1757-899X/790/1/012041

Song, S., Liu, X., Li, C., Li, Z., Zhang, S., Wu, W., Shi, B., Kang, Q., Wu, H., & Gong, J. (2023). Dynamic Simulator for Three-Phase Gravity Separators in Oil Production Facilities. *ACS Omega*, *8*(6), 6078–6089. https://doi.org/10.1021/acsomega.2c08267

Srinath Perera. (2015, November 7). *Introduction to Anomaly Detection: Concepts and Techniques | My views of the World and Systems*. Https://Iwringer.Wordpress.Com/2015/11/17/Anomaly-Detection-Concepts-and-Techniques/. https://iwringer.wordpress.com/2015/11/17/anomaly-detection-concepts-and-techniques/

Stergiopoulos, G., Gritzalis, D. A., & Limnaios, E. (2020). Cyber-Attacks on the Oil Gas Sector: A Survey on Incident Assessment and Attack Patterns. *IEEE Access*, *8*, 128440–128475. https://doi.org/10.1109/ACCESS.2020.3007960

Tang, S., Liu, Z., & Wang, L. (2020). Power System Reliability Analysis Considering External and Insider Attacks on the SCADA System. *Proceedings of the IEEE Power Engineering Society Transmission and Distribution Conference*, *2020-Octob*, 0–4. https://doi.org/10.1109/TD39804.2020.9299922

Teixeira, M. A., Salman, T., Zolanvari, M., Jain, R., Meskin, N., & Samaka, M. (2018). SCADA system testbed for cybersecurity research using machine learning approach. *Future Internet*, *10*(8). https://doi.org/10.3390/fi10080076

Council Directive 2008/114/ec on the indentification and designation of European

critical infrastructures and the assessment of the need to improve their protection, Official Journal of the European Union 75 (2008). https://doi.org/EUR 23665 EN ISSN 1018-5593

The White House - Office of the Press Secretary. (2013). *Presidential Policy Directive/PPD-21 -- Critical Infrastructure Security and Resilience | whitehouse.gov*. Presidential Policy Directive/PPD-21. https://obamawhitehouse.archives.gov/the-press-office/2013/02/12/presidential-policy-directive-critical-infrastructure-security-and-resil

Tommy Morris - Industrial Control System (ICS) Cyber Attack Datasets. (n.d.). Retrieved May 29, 2023, from https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets

Tufan, E., & Tezcan, Ci. (2021). Anomaly-Based Intrusion Detection by Machine Learning : A Case Study on Probing Attacks to an Institutional Network. *IEEE Access*, *9*. https://doi.org/10.1109/ACCESS.2021.3068961

Virvilis, N., & Gritzalis, D. (2013a). The big four - What we did wrong in advanced persistent threat detection? *Proceedings - 2013 International Conference on Availability, Reliability and Security, ARES 2013*, 248–254. https://doi.org/10.1109/ARES.2013.32

Virvilis, N., & Gritzalis, D. (2013b). The big four - What we did wrong in advanced persistent threat detection? *Proceedings - 2013 International Conference on Availability, Reliability and Security, ARES 2013*, 248–254. https://doi.org/10.1109/ARES.2013.32

Williams, J., Talbot, B., & Zaidi, S. (2015). CYBERSECURITY THREATS TO THE OIL & GAS INDUSTRY Are You at Risk ? In *www.parsons.com/cyber*. https://www.parsons.com/wp-content/uploads/2017/08/Cybersecurity-Oil-Gas.pdf

Wilson, D., Tang, Y., Yan, J., & Lu, Z. (2018). Deep Learning-Aided Cyber-Attack

Detection in Power Transmission Systems. *IEEE Power and Energy Society General Meeting*, *2018-Augus*, 1–5. https://doi.org/10.1109/PESGM.2018.8586334

Wilson, G. (2014). The Nigerian State and Oil Theft in the Niger Delta Region of Nigeria. *Journal of Sustainable Development in Africa*, *16*(1), 69–81.

Wu, F., Huang, K., Li, H., & Huang, C. (2022). Analysis and Research on the Automatic Control Systems of Oil–Water Baffles in Horizontal Three-Phase Separators. *Processes*, *10*(6). https://doi.org/10.3390/pr10061102

Zhao, Y. (2022a). *All Models - pyod 1.0.7 documentation*. https://pyod.readthedocs.io/en/latest/pyod.models.html#pyod.models.lof.LOF

Zhao, Y. (2022b). *pyod Documentation Release 1.0.6*. https://pyod.readthedocs.io/en/latest/

Zhao, Y., Nasrullah, Z., & Li, Z. (2019). PyOD: A python toolbox for scalable outlier detection. *Journal of Machine Learning Research*, *20*, 1–7.

Zoppi, T., Ceccarelli, A., & Bondavalli, A. (2021). Unsupervised Algorithms to Detect Zero-Day Attacks: Strategy and Application. *IEEE Access*, *9*, 90603–90615. https://doi.org/10.1109/ACCESS.2021.3090957

Zoppi, T., Ceccarelli, A., Capecchi, T., & Bondavalli, A. (2021). Unsupervised Anomaly Detectors to Detect Intrusions in the Current Threat Landscape. *ACM/IMS Transactions on Data Science*, *2*(2), 1–26. https://doi.org/10.1145/3441140

Zoppi, T., Ceccarelli, A., Salani, L., & Bondavalli, A. (2020). On the educated selection of unsupervised algorithms via attacks and anomaly classes. In *Journal of Information Security and Applications* (Vol. 52). https://doi.org/10.1016/j.jisa.2020.102474

APPENDIX A: Data Exploration

Importing Libraries

In [1]:
```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec
%matplotlib inline


import seaborn as sns
```

```
C:\Users\Mnesoma\Anaconda3\lib\site-packages\pandas\compat\_optional.py:138: UserWarnin
g: Pandas requires version '2.7.0' or newer of 'numexpr' (version '2.6.9' currently inst
alled).
  warnings.warn(msg, UserWarning)
```

In [2]:
```
Data = pd.read_csv('Pressure_with_Anomaly_2.csv')
```

Converting Timestamp to Datetime

In [3]:
```
Data['datetime'] = pd.to_datetime(Data["timestamp"])
```

In [4]:
```
Data.head()
```

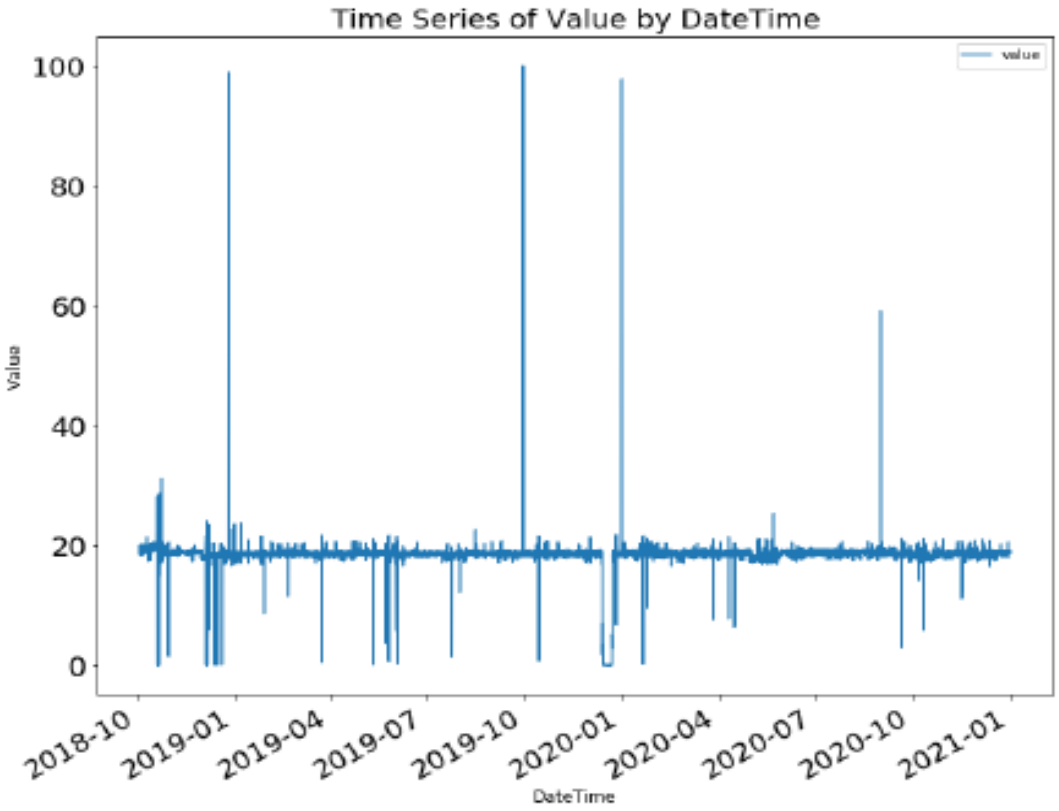
Out[4]:

	timestamp	value	datetime
0	10/4/2018 0:00	19.19495	2018-10-04 00:00:00
1	10/4/2018 0:27	19.04453	2018-10-04 00:27:00
2	10/4/2018 0:28	19.26260	2018-10-04 00:28:00
3	10/4/2018 0:37	19.05637	2018-10-04 00:37:00
4	10/4/2018 0:46	19.25317	2018-10-04 00:46:00

Time Series of Value by DateTime

In [5]:
```
Data.plot(x= 'datetime', y= 'value', figsize = (12,10), fontsize = 20)
plt.xlabel('DateTime', fontsize = 12)
plt.ylabel('Value', fontsize = 12)
plt.title('Time Series of Value by DateTime', fontsize = 20)
```

Out[5]:
```
Text(0.5, 1.0, 'Time Series of Value by DateTime')
```



Setting Datetime as Index

In [6]:
```
Data = Data.set_index('datetime')
```

In [7]:
```
Data.drop('timestamp', axis = 1, inplace = True)
```

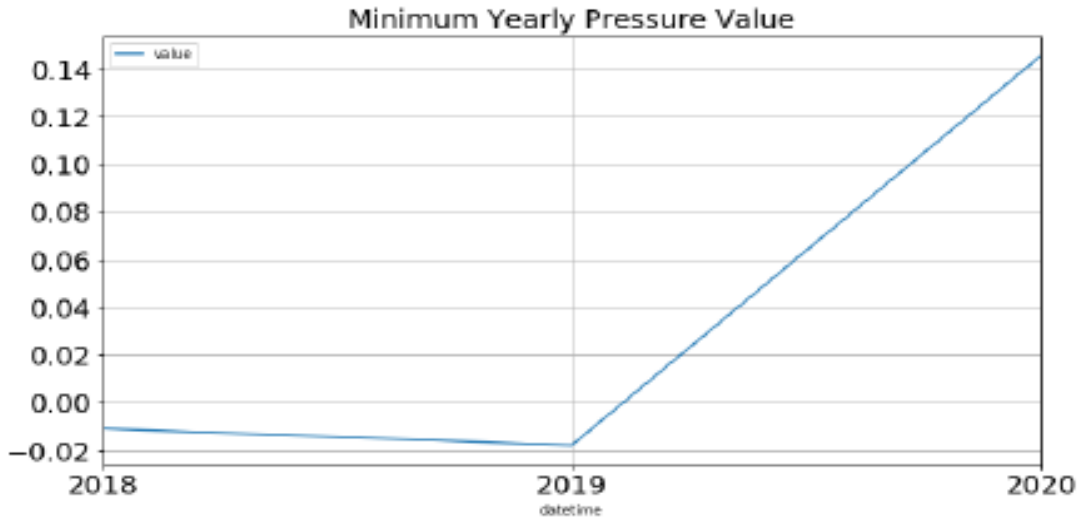
In [8]:
```
Data.head()
```

Out[8]:

datetime	value
2018-10-04 00:00:00	19.19495
2018-10-04 00:27:00	19.04453
2018-10-04 00:28:00	19.26260
2018-10-04 00:37:00	19.05637
2018-10-04 00:46:00	19.25317

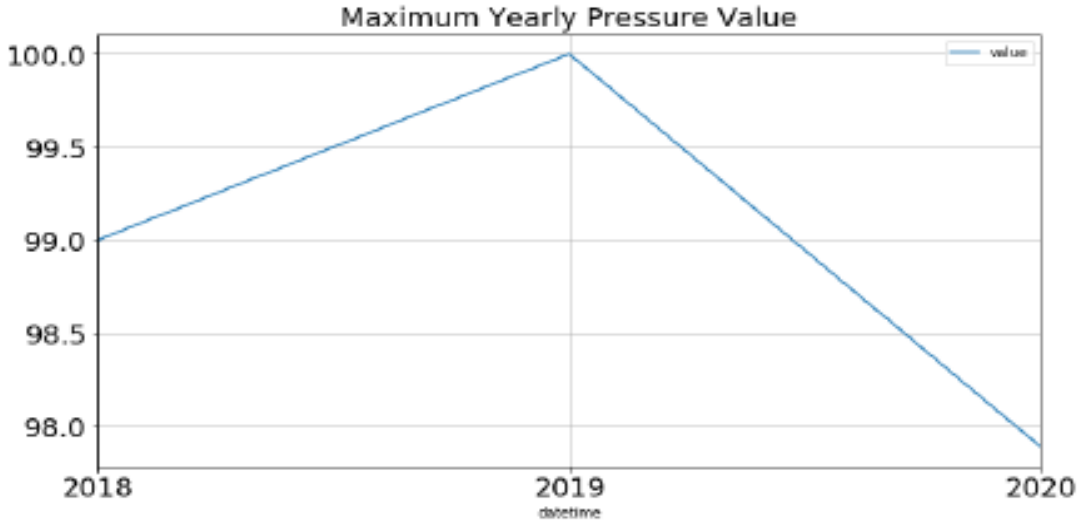
Minimum Yearly Pressure Value

In [9]:
```
Data.resample(rule='A').min().plot(kind='line', figsize=( 12 , 6 ), fontsize =20)
plt.title('Minimum Yearly Pressure Value', fontsize = 20)
plt.grid()
```



Maximum Yearly Pressure Values

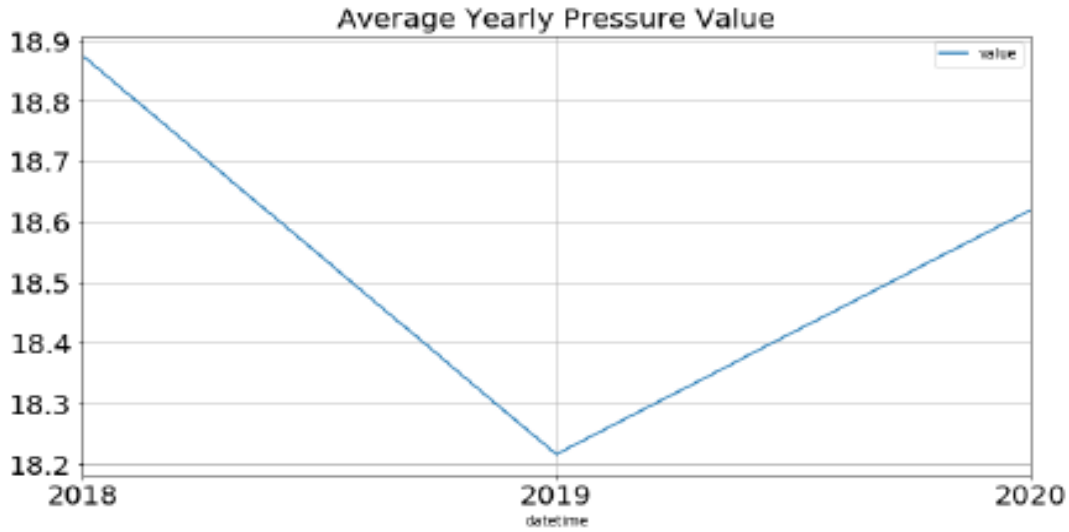
In [11]:
```
Data.resample(rule='A').max().plot(kind = 'line', figsize = (12,6), fontsize = 20)
plt.title('Maximum Yearly Pressure Value', fontsize = 20)
plt.grid()
```



Average Yearly Pressure Value

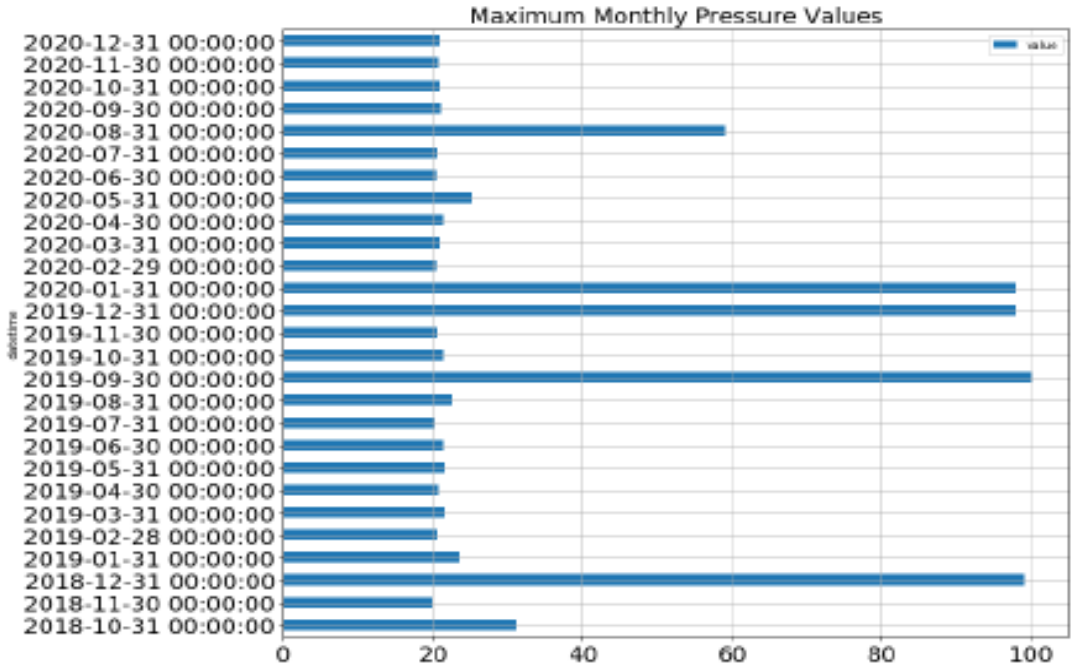
In [13]:
```
Data.resample(rule='A').mean().plot(kind = 'line', figsize = (12,6), fontsize = 20)
plt.title('Average Yearly Pressure Value', fontsize = 20)
```

```
plt.grid()
```

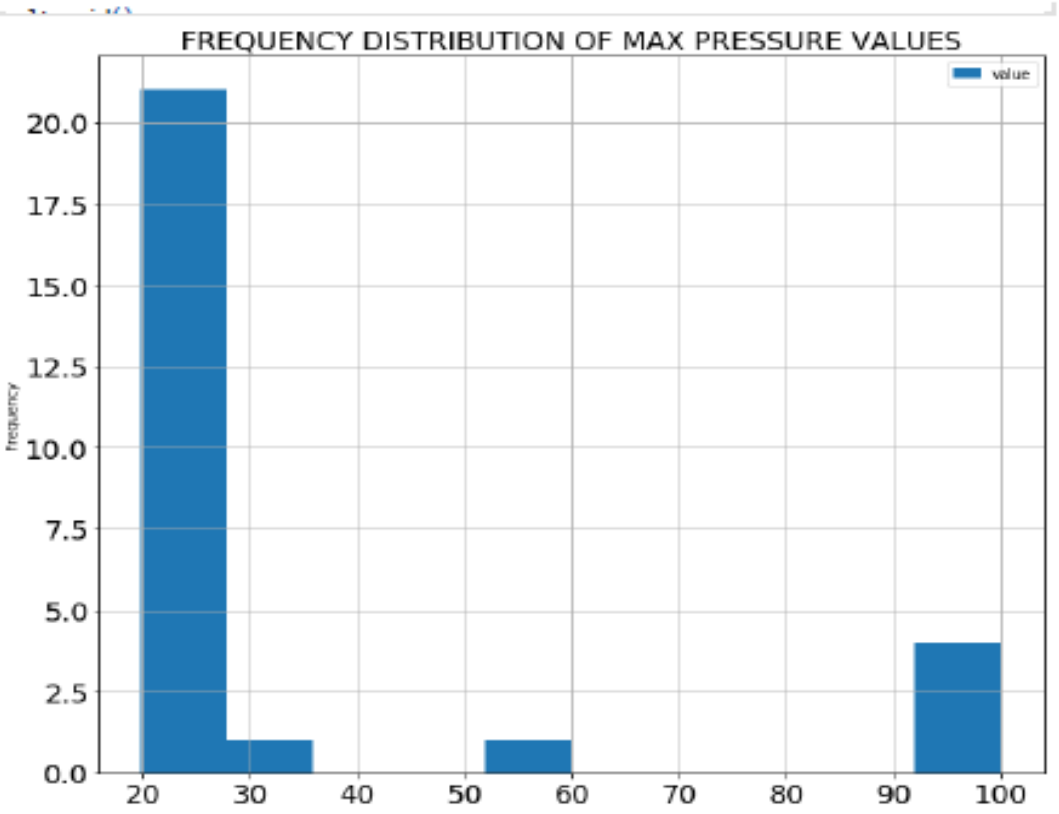


Maximum Monthly Pressure Values

In [14]:
```
Data.resample(rule='M').max().plot(kind = 'barh', figsize = (12,10), fontsize = 20)
plt.title('Maximum Monthly Pressure Values', fontsize = 20)
plt.grid()
```



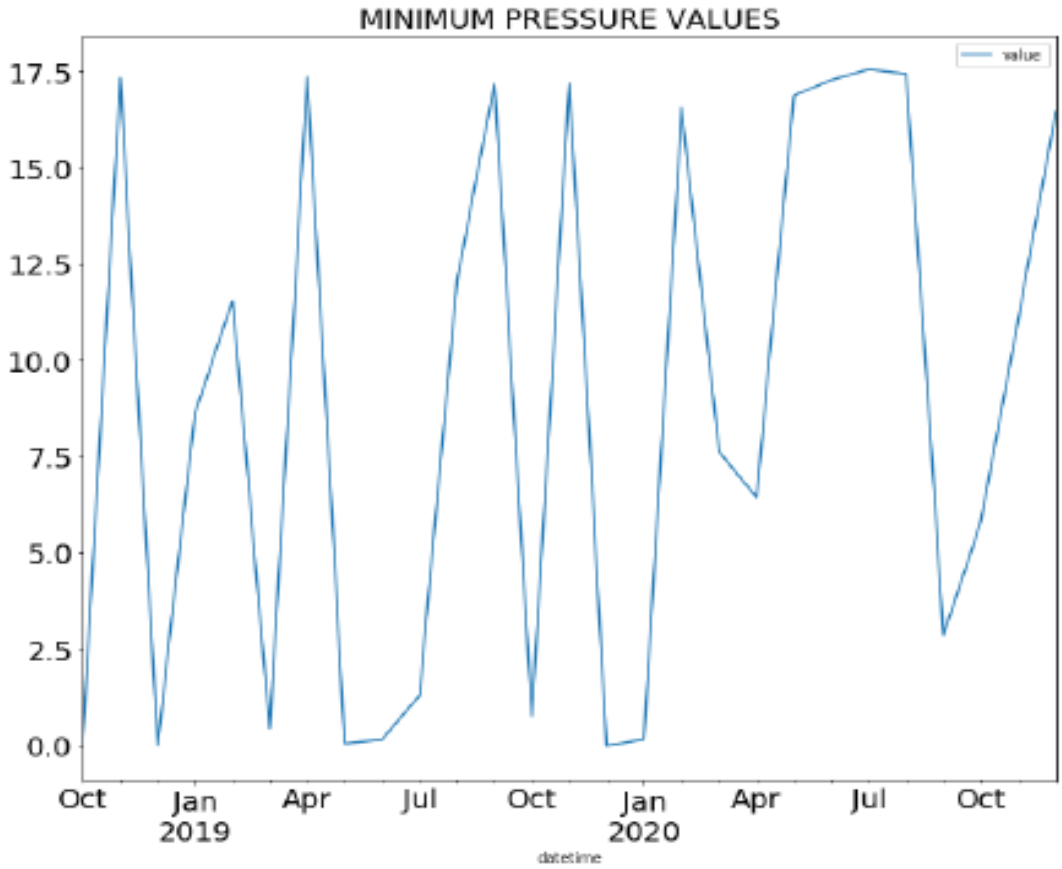
In [15]:
```
Data.resample(rule='M').max().plot(kind = 'hist', figsize = (12,10), fontsize = 20)
plt.title('FREQUENCY DISTRIBUTION OF MAX PRESSURE VALUES', fontsize = 20)
```



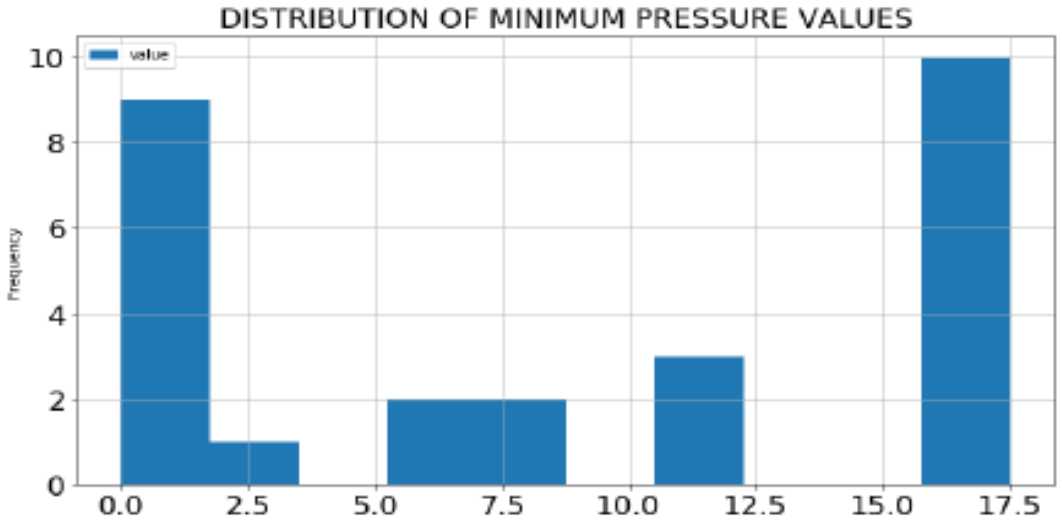
Minimum Monthly Pressure Values

In [16]:
```
Data.resample(rule='M').min().plot(kind = 'line', figsize = (12,10), fontsize = 20)
plt.title('MINIMUM PRESSURE VALUES', fontsize = 20)
plt.grid
```

Out[16]:
```
<function matplotlib.pyplot.grid(b=None, which='major', axis='both', **kwargs)>
```

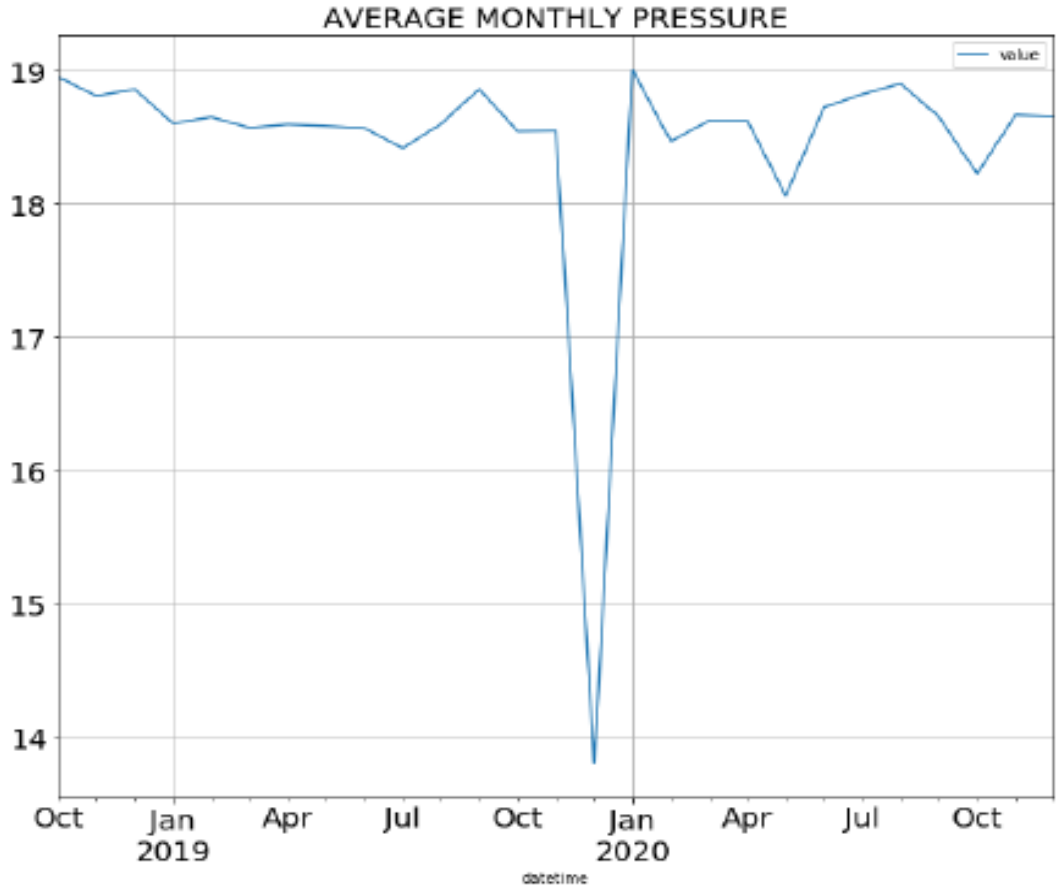


In [18]:
```
Data.resample(rule='M').min().plot(kind = 'hist', figsize = (12,6), fontsize = 20)
plt.title('DISTRIBUTION OF MINIMUM PRESSURE VALUES', fontsize = 20)
plt.grid()
```



Average Monthly Pressure

In [20]:
```
Data.resample(rule='M').mean().plot(kind = 'line', figsize = (12,10), fontsize = 20)
plt.title('AVERAGE MONTHLY PRESSURE', fontsize = 20)
plt.grid()
```



In [ ]:

APPENDIX B: Data Handling Pressure Values without Anomalies

In [1]:
```
import pandas as pd
import numpy as np
import seaborn
import matplotlib.dates as md
from matplotlib import pyplot as plt
from mpl_toolkits.axes_grid1 import host_subplot
import mpl_toolkits.axisartist as AA
from sklearn import preprocessing
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn.covariance import EllipticEnvelope
from pyemma import msm
from sklearn.ensemble import IsolationForest
from sklearn.svm import OneClassSVM
```

In [2]:
```
df = pd.read_csv("C:/Users/RexEng1/Documents/Test/Pressure_without_Anomaly_2_used.csv")
```

In [3]:
```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 68722 entries, 0 to 68721
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   timestamp  68722 non-null  object 
 1   value      68722 non-null  float64
dtypes: float64(1), object(1)
memory usage: 1.0+ MB
None
```

In [4]:
```
# check the timestamp format and frequence
print(df['timestamp'].head(10))
```

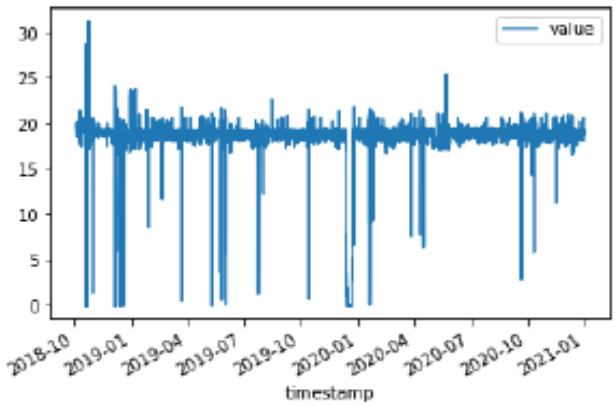
```
0    10/4/2018 0:00
1    10/4/2018 0:27
2    10/4/2018 0:28
3    10/4/2018 0:37
4    10/4/2018 0:46
5    10/4/2018 1:04
6    10/4/2018 1:16
7    10/4/2018 1:35
8    10/4/2018 1:48
9    10/4/2018 2:18
Name: timestamp, dtype: object
```

In [5]:
```
# check the temperature mean
print(df['value'].mean())
```

```
18.408501150828183
```

In [6]:
```
# change the type of timestamp column for plotting
df['timestamp'] = pd.to_datetime(df['timestamp'])
# plot the data
df.plot(x='timestamp', y='value')
```

Out[6]:
```
<AxesSubplot:xlabel='timestamp'>
```

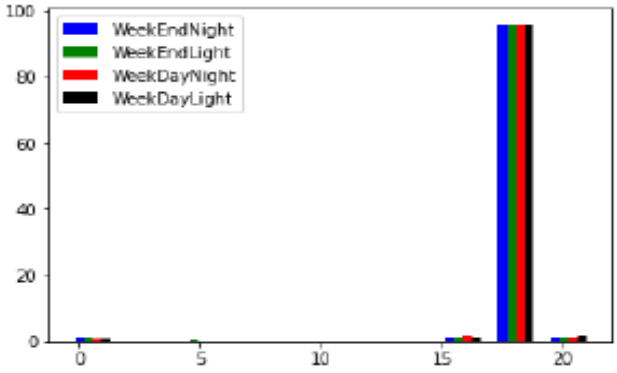


In [7]:
```
# the hours and if it's night or day (7:00-22:00)
df['hours'] = df['timestamp'].dt.hour
df['daylight'] = ((df['hours'] >= 7) & (df['hours'] <= 22)).astype(int)
```

In [8]:
```
# the day of the week (Monday=0, Sunday=6) and if it's a week end day or week day.
df['DayOfTheWeek'] = df['timestamp'].dt.dayofweek
df['WeekDay'] = (df['DayOfTheWeek'] < 5).astype(int)
# An estimation of anomly population of the dataset (necessary for several algorithm)
outliers_fraction = 0.01
```

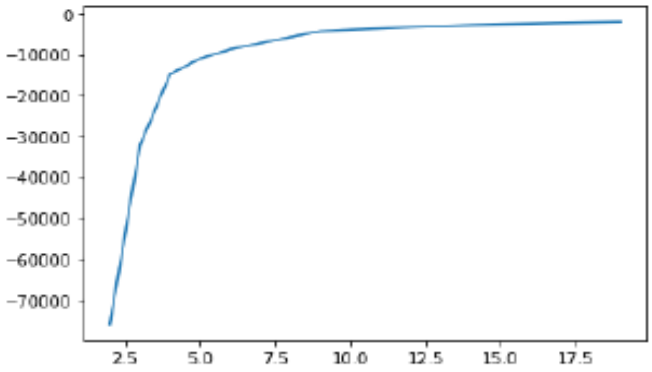
In [9]:
```
# time with int to plot easily
df['time_epoch'] = (df['timestamp'].astype(np.int64)/100000000000).astype(np.int64)
```

In [10]:
```
# creation of 4 distinct categories that seem useful (week end/day week & night/day)
df['categories'] = df['WeekDay']*2 + df['daylight']

a = df.loc[df['categories'] == 0, 'value']
b = df.loc[df['categories'] == 1, 'value']
c = df.loc[df['categories'] == 2, 'value']
d = df.loc[df['categories'] == 3, 'value']

fig, ax = plt.subplots()
a_heights, a_bins = np.histogram(a)
b_heights, b_bins = np.histogram(b, bins=a_bins)
c_heights, c_bins = np.histogram(c, bins=a_bins)
d_heights, d_bins = np.histogram(d, bins=a_bins)

width = (a_bins[1] - a_bins[0])/6

ax.bar(a_bins[:-1], a_heights*100/a.count(), width=width, facecolor='blue', label='Week
ax.bar(b_bins[:-1]+width, (b_heights*100/b.count()), width=width, facecolor='green', la
ax.bar(c_bins[:-1]+width*2, (c_heights*100/c.count()), width=width, facecolor='red', la
ax.bar(d_bins[:-1]+width*3, (d_heights*100/d.count()), width=width, facecolor='black',

plt.legend()
plt.show()
```



In [11]:
```
# Take useful feature and standardize them
data = df[['value', 'hours', 'daylight', 'DayOfTheWeek', 'WeekDay']]
min_max_scaler = preprocessing.StandardScaler()
np_scaled = min_max_scaler.fit_transform(data)
data = pd.DataFrame(np_scaled)
# reduce to 2 importants features
pca = PCA(n_components=2)
data = pca.fit_transform(data)
# standardize these 2 new features
min_max_scaler = preprocessing.StandardScaler()
np_scaled = min_max_scaler.fit_transform(data)
data = pd.DataFrame(np_scaled)
```

In [13]:
```
# calculate with different number of centroids to see the loss plot (elbow method)
n_cluster = range(2, 20)
kmeans = [KMeans(n_clusters=i).fit(data) for i in n_cluster]
scores = [kmeans[i].score(data) for i in range(len(kmeans))]
fig, ax = plt.subplots()
ax.plot(n_cluster, scores)
plt.show()
```



In [14]:
```
# Not clear for me, I choose 15 centroids arbitrarily and add these data to the central
df['cluster'] = kmeans[14].predict(data)
df['principal_feature1'] = data[0]
df['principal_feature2'] = data[1]
df['cluster'].value_counts()
```

APPENDIX C: Data Handling Pressure Values with Anomalies

In [24]:
```
import pandas as pd
import numpy as np
import seaborn
import matplotlib.dates as md
from matplotlib import pyplot as plt
from mpl_toolkits.axes_grid1 import host_subplot
import mpl_toolkits.axisartist as AA
from sklearn import preprocessing
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn.covariance import EllipticEnvelope
from pyemma import msm
from sklearn.ensemble import IsolationForest
from sklearn.svm import OneClassSVM
```

In [25]:
```
df = pd.read_csv("C:/Users/RexEng1/Documents/Test/Pressure_with_Anomaly_2.csv")
```

In [26]:
```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 68722 entries, 0 to 68721
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   timestamp  68722 non-null  object 
 1   value      68722 non-null  float64
dtypes: float64(1), object(1)
memory usage: 1.0+ MB
None
```

In [27]:
```
# check the timestamp format and frequence
print(df['timestamp'].head(10))
```

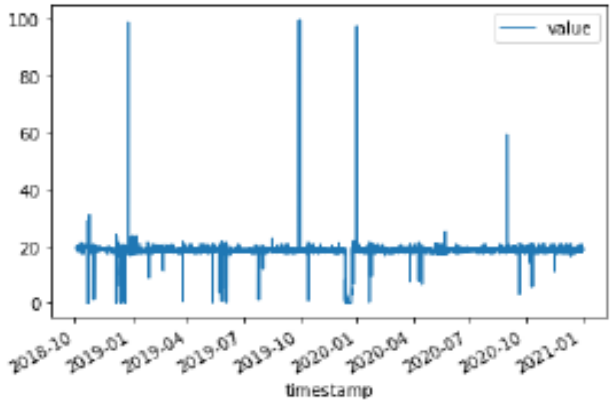
```
0    10/4/2018 0:00
1    10/4/2018 0:27
2    10/4/2018 0:28
3    10/4/2018 0:37
4    10/4/2018 0:46
5    10/4/2018 1:04
6    10/4/2018 1:16
7    10/4/2018 1:35
8    10/4/2018 1:48
9    10/4/2018 2:18
Name: timestamp, dtype: object
```

In [28]:
```
# check the temperature mean
print(df['value'].mean())
```

```
18.46955747413078
```

In [29]:
```
# change the type of timestamp column for plotting
df['timestamp'] = pd.to_datetime(df['timestamp'])
# plot the data
df.plot(x='timestamp', y='value')
```

Out[29]:
```
<AxesSubplot:xlabel='timestamp'>
```

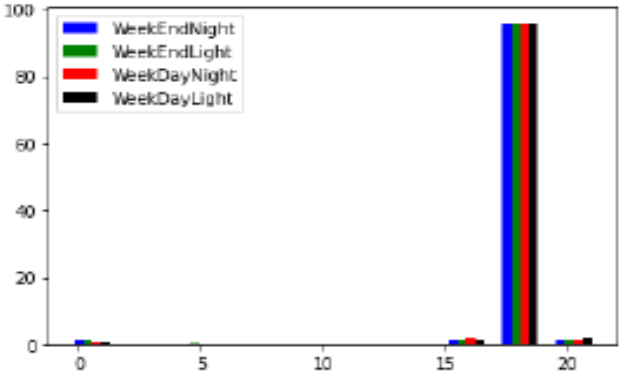


In [30]:
```
# the hours and if it's night or day (7:00-22:00)
df['hours'] = df['timestamp'].dt.hour
df['daylight'] = ((df['hours'] >= 7) & (df['hours'] <= 22)).astype(int)
```

In [31]:
```
# the day of the week (Monday=0, Sunday=6) and if it's a week end day or week day.
df['DayOfTheWeek'] = df['timestamp'].dt.dayofweek
df['WeekDay'] = (df['DayOfTheWeek'] < 5).astype(int)
# An estimation of anomly population of the dataset (necessary for several algorithm)
outliers_fraction = 0.01
```

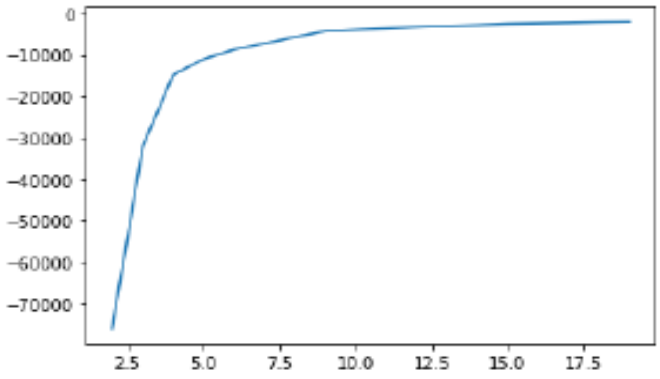
In [32]:
```
# time with int to plot easily
df['time_epoch'] = (df['timestamp'].astype(np.int64)/100000000000).astype(np.int64)
```

In [33]:
```
# creation of 4 distinct categories that seem useful (week end/day week & night/day)
df['categories'] = df['WeekDay']*2 + df['daylight']

a = df.loc[df['categories'] == 0, 'value']
b = df.loc[df['categories'] == 1, 'value']
c = df.loc[df['categories'] == 2, 'value']
d = df.loc[df['categories'] == 3, 'value']

fig, ax = plt.subplots()
a_heights, a_bins = np.histogram(a)
b_heights, b_bins = np.histogram(b, bins=a_bins)
c_heights, c_bins = np.histogram(c, bins=a_bins)
d_heights, d_bins = np.histogram(d, bins=a_bins)

width = (a_bins[1] - a_bins[0])/6

ax.bar(a_bins[:-1], a_heights*100/a.count(), width=width, facecolor='blue', label='Week
ax.bar(b_bins[:-1]+width, (b_heights*100/b.count()), width=width, facecolor='green', la
ax.bar(c_bins[:-1]+width*2, (c_heights*100/c.count()), width=width, facecolor='red', la
ax.bar(d_bins[:-1]+width*3, (d_heights*100/d.count()), width=width, facecolor='black',

plt.legend()
plt.show()
```



In [34]:
```
# Take useful feature and standardize them
data = df[['value', 'hours', 'daylight', 'DayOfTheWeek', 'WeekDay']]
min_max_scaler = preprocessing.StandardScaler()
np_scaled = min_max_scaler.fit_transform(data)
data = pd.DataFrame(np_scaled)
# reduce to 2 importants features
pca = PCA(n_components=2)
data = pca.fit_transform(data)
# standardize these 2 new features
min_max_scaler = preprocessing.StandardScaler()
np_scaled = min_max_scaler.fit_transform(data)
data = pd.DataFrame(np_scaled)
```

In [35]:
```
# calculate with different number of centroids to see the loss plot (elbow method)
n_cluster = range(2, 20)
kmeans = [KMeans(n_clusters=i).fit(data) for i in n_cluster]
scores = [kmeans[i].score(data) for i in range(len(kmeans))]
fig, ax = plt.subplots()
ax.plot(n_cluster, scores)
plt.show()
```



In [36]:
```
# Not clear for me, I choose 15 centroids arbitrarily and add these data to the central
df['cluster'] = kmeans[14].predict(data)
df['principal_feature1'] = data[0]
df['principal_feature2'] = data[1]
df['cluster'].value_counts()
```

APPENDIX D: Isolation Forest Algorithm Implementation

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib
import seaborn
import matplotlib.dates as md
from matplotlib import pyplot as plt
from sklearn import preprocessing
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn.covariance import EllipticEnvelope
from sklearn.ensemble import IsolationForest
from sklearn.svm import OneClassSVM
```

```
C:\Users\Mnesoma\Anaconda3\lib\site-packages\pandas\compat\_optional.py:138: UserWarnin
g: Pandas requires version '2.7.0' or newer of 'numexpr' (version '2.6.9' currently inst
alled).
  warnings.warn(msg, UserWarning)
```

In [5]:
```
Data = pd.read_csv('Pressure_with_Anomaly_2.csv')
```

In [6]:
```
# preprocess 'time' to timedate objects
Data['timestamp'] = pd.to_datetime(Data['timestamp'])
```

In [1]:
```

```

```
C:\Users\Mnesoma\Anaconda3\lib\site-packages\pandas\compat\_optional.py:138: UserWarnin
g: Pandas requires version '2.7.0' or newer of 'numexpr' (version '2.6.9' currently inst
alled).
  warnings.warn(msg, UserWarning)
```

In [7]:
```
Data['hours'] = Data['timestamp'].dt.hour
Data['mins'] = Data['timestamp'].dt.minute
Data['year'] = Data['timestamp'].dt.year
Data['month'] = Data['timestamp'].dt.month
Data['day'] = Data['timestamp'].dt.day
```

In [8]:
```
# An estimation of anomly population of the dataset (necessary for several algorithm)
outliers_fraction = 0.01
```

In [9]:
```
# extracting time as an integer
Data['time_epoch'] = (Data['timestamp'].astype(np.int64)/100000000000).astype(np.int64)
```

```
C:\Users\Mnesoma\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: FutureWarning: cas
ting datetime64[ns] values to int64 with .astype(...) is deprecated and will raise in a
future version. Use .view(...) instead.
```

In [17]:
```
data = Data[['value', 'hours', 'mins', 'year', 'month', 'day']]
min_max_scaler = preprocessing.StandardScaler()
np_scaled = min_max_scaler.fit_transform(data['value'])


data = pd.DataFrame(np_scaled)
```

In [18]:
```
model = IsolationForest(contamination = outliers_fraction)
model.fit(data)
```

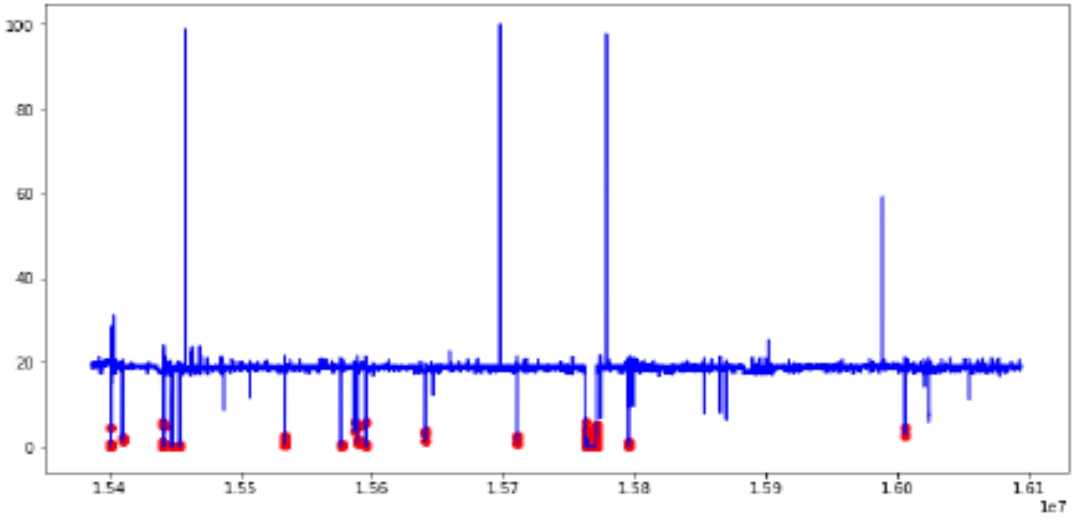
Out[18]:
```
IsolationForest(contamination=0.01)
```

In [19]:
```
Data['anomaly25'] = pd.Series(model.predict(data))
Data['anomaly25'] = Data['anomaly25'].map( {1: 0, -1: 1} )
print(Data['anomaly25'].value_counts())
```

```
0    68046
1      676
Name: anomaly25, dtype: int64
```

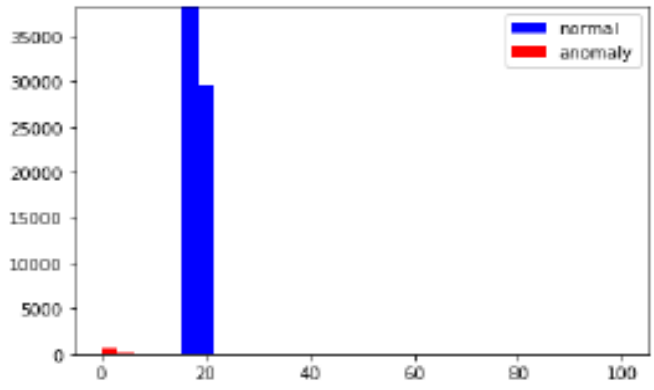
In [20]:
```
# visualisation of anomaly throughout time (viz 1)
fig, ax = plt.subplots(figsize = (12,6))
a = Data.loc[Data['anomaly25'] == 1, ['time_epoch', 'value']] #anomaly
ax.plot(Data['time_epoch'], Data['value'], color='blue')
ax.scatter(a['time_epoch'],a['value'], color='red')
plt.show()
```

```
C:\Users\Mnesoma\Anaconda3\lib\site-packages\matplotlib\cbook\__init__.py:1402: FutureWa
rning: Support for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and wi
ll be removed in a future version.  Convert to a numpy array before indexing instead.
  x[:, None]
C:\Users\Mnesoma\Anaconda3\lib\site-packages\matplotlib\axes\_base.py:276: FutureWarnin
g: Support for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will b
e removed in a future version.  Convert to a numpy array before indexing instead.
  x = x[:, np.newaxis]
C:\Users\Mnesoma\Anaconda3\lib\site-packages\matplotlib\axes\_base.py:278: FutureWarnin
g: Support for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will b
e removed in a future version.  Convert to a numpy array before indexing instead.
  y = y[:, np.newaxis]
```



In [16]:
```
# visualisation of anomaly with temperature repartition (viz 2)
a = Data.loc[Data['anomaly25'] == 0, 'value']
b = Data.loc[Data['anomaly25'] == 1, 'value']
fig, axs = plt.subplots()
axs.hist([a,b], bins=32, stacked=True, color=['blue', 'red'], label = ['normal',
'anomaly'])
plt.legend()
plt.show()
```

```
C:\Users\Mnesoma\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:3208: VisibleDepr
ecationWarning: Creating an ndarray from ragged nested sequences (which is a list-or-tup
le of lists-or-tuples-or ndarrays with different lengths or shapes) is deprecated. If yo
u meant to do this, you must specify 'dtype=object' when creating the ndarray.
  return asarray(a).size
C:\Users\Mnesoma\Anaconda3\lib\site-packages\matplotlib\cbook\__init__.py:1420: VisibleD
eprecationWarning: Creating an ndarray from ragged nested sequences (which is a list-or-
tuple of lists-or-tuples-or ndarrays with different lengths or shapes) is deprecated. If
you meant to do this, you must specify 'dtype=object' when creating the ndarray.
  X = np.atleast_1d(X.T if isinstance(X, np.ndarray) else np.asarray(X))
```



In [ ]:

APPENDIX E: Long Short-Term Memory Algorithm Implementation

Importing Libraries for Data Manipulation and Visualization

In [ ]:
```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec
%matplotlib inline

import seaborn as sns
```

In [ ]:
```
Data = pd.read_csv('Pressure_with_Anomaly_2.csv')
```

In [ ]:
```
Data.head()
```

Out[ ]:

	timestamp	value
0	10/4/2018 0:00	19.19495
1	10/4/2018 0:27	19.04453
2	10/4/2018 0:28	19.26260
3	10/4/2018 0:37	19.05637
4	10/4/2018 0:46	19.25317

In [ ]:
```
Data['datetime'] = pd.to_datetime(Data["timestamp"])
```

In [ ]:
```
Data.head()
```

Out[ ]:

	timestamp	value	datetime
0	10/4/2018 0:00	19.19495	2018-10-04 00:00:00
1	10/4/2018 0:27	19.04453	2018-10-04 00:27:00
2	10/4/2018 0:28	19.26260	2018-10-04 00:28:00
3	10/4/2018 0:37	19.05637	2018-10-04 00:37:00
4	10/4/2018 0:46	19.25317	2018-10-04 00:46:00

In [ ]:
```
Data.shape
```

Out[ ]:
```
(68722, 3)
```

In [ ]:
```
Data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 68722 entries, 0 to 68721
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   timestamp  68722 non-null  object
```

```
 1   value      68722 non-null  float64
dtypes: float64(1), object(1)
memory usage: 1.0+ MB
```

In [ ]:
```
Data.isnull().any()
```

Out[ ]:
```
timestamp    False
value        False
dtype: bool
```

Data.describe(include = 'datetime')

In [ ]:
```
Data.duplicated().sum()
```

Out[ ]:
```
46
```

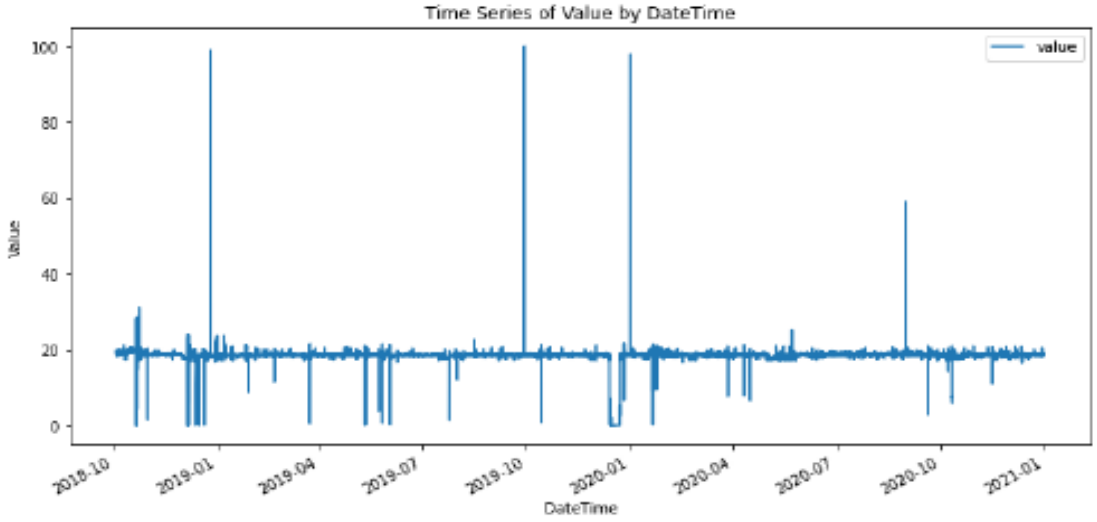
In [ ]:
```
Data.describe()
```

Out[ ]:

	value
count	68722.000000
mean	18.469557
std	2.947190
min	-0.018000
25%	18.340900
50%	18.635130
75%	18.921760
max	99.999990

In [ ]:
```
Data.plot(x= 'datetime', y= 'value', figsize = (12,6))
plt.xlabel('DateTime')
plt.ylabel('Value')
plt.title('Time Series of Value by DateTime')
```

Out[ ]:
```
Text(0.5, 1.0, 'Time Series of Value by DateTime')
```



In [ ]:
```
Data.drop(['timestamp'], axis = 1, inplace = True)
```

In [ ]:
```
Data.head()
```

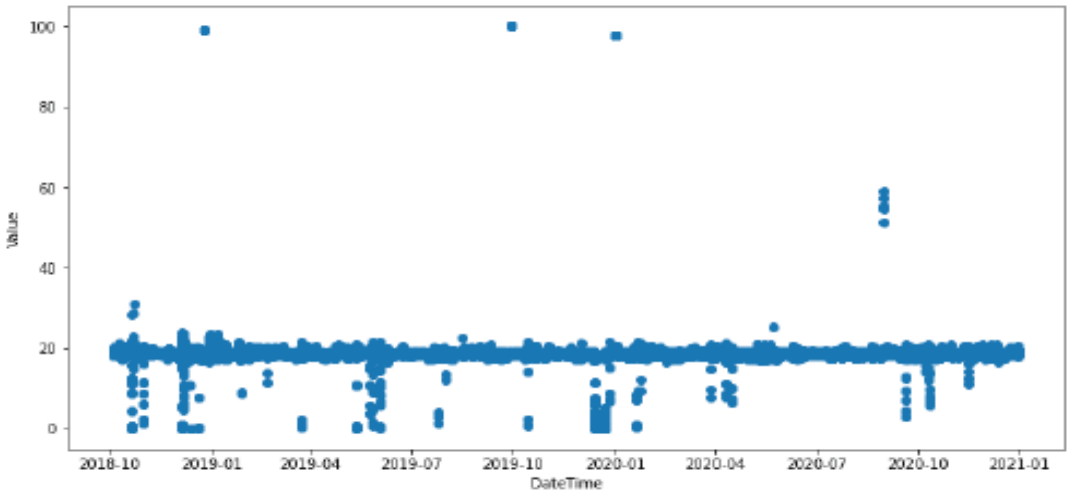
Out[ ]:

	value	datetime
68721	18.79747	2020-12-31 23:59:00
68719	18.67054	2020-12-31 23:48:00
68718	18.99389	2020-12-31 23:41:00
68717	19.03570	2020-12-31 23:02:00
68716	18.73181	2020-12-31 22:53:00

In [ ]:
```
Data.shape
```

Out[ ]:
```
(68676, 2)
```

In [ ]:
```
plt.figure(figsize = (12,6))
x = Data['datetime']
y = Data['value']
plt.scatter(x,y)
plt.xlabel('DateTime')
plt.ylabel('Value')
plt.show()
```



Importing Modules for Model Building

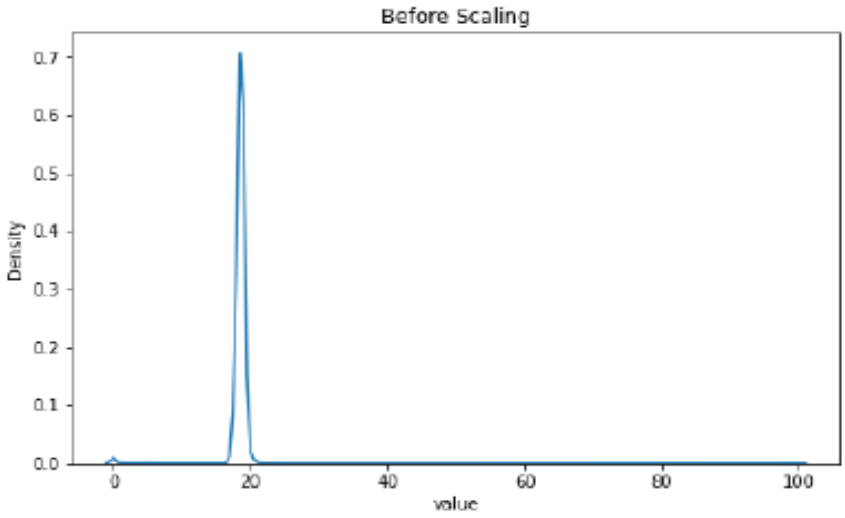
In [17]:
```
import keras
from keras import optimizers
from keras import losses
from keras.models import Sequential, Model
from keras.layers import Dense, Input, Dropout, Embedding, LSTM
from keras.optimizers import RMSprop, Adam, Nadam
from keras.preprocessing import sequence
from keras.callbacks import TensorBoard
```

In [ ]:
```
import sklearn
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, roc_auc_score
from sklearn.preprocessing import MinMaxScaler
```

In [ ]:
```
import tensorflow
import sys
```

In [ ]:
```
fig, (ax1) = plt.subplots(ncols = 1, figsize=(8,5))
ax1.set_title('Before Scaling')
sns.kdeplot(Data['value'], ax = ax1)
```

Out[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f562035a710>
```



Scaling Data

In [ ]:
```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler(feature_range = (0, 1))
Data['scaled value'] = pd.DataFrame(scaler.fit_transform(pd.DataFrame(Data['value'])),
```

In [ ]:
```
Data.shape
```

Out[ ]:
```
(68722, 3)
```

In [ ]:
```
Data.head()
```

Out[ ]:

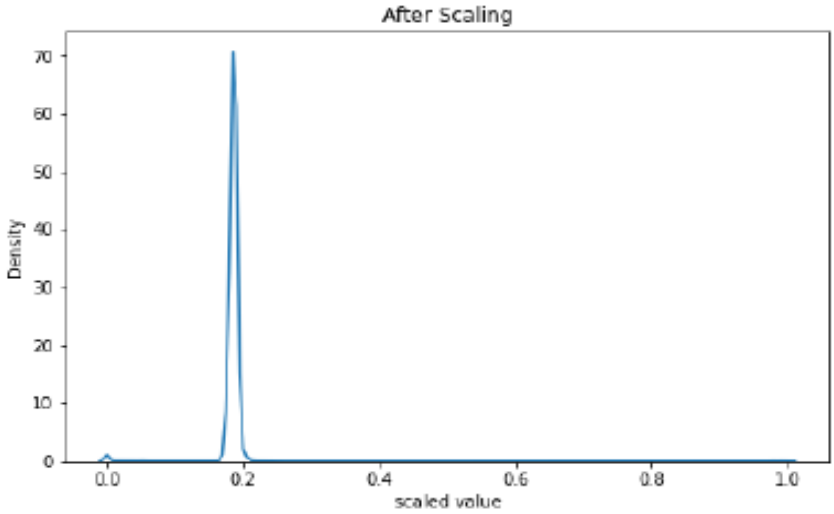
	value	datetime	scaled value
0	19.19495	2018-10-04 00:00:00	0.192095
1	19.04453	2018-10-04 00:27:00	0.190591
2	19.26260	2018-10-04 00:28:00	0.192771
3	19.05637	2018-10-04 00:37:00	0.190709
4	19.25317	2018-10-04 00:46:00	0.192677

In [ ]:
```
Data['scaled value'].count()
```

Out[ ]:
```
68722
```

In [ ]:
```
fig, (ax1) = plt.subplots(ncols = 1, figsize=(8,5))
ax1.set_title('After Scaling')
sns.kdeplot(Data['scaled value'], ax = ax1)
```

Out[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0bd370f690>
```



Building The Long Short Term Memory Model

In [61]:
```
time_steps = 34361
metric = 'mean_absolute_error'
```

In [89]:
```
model = Sequential()
model.add(LSTM(units = 128, activation = 'tanh', input_shape = (time_steps, 1), return_

model.add(Dense(1, activation = 'sigmoid'))

model.compile(optimizer = 'adam',loss = 'mean_absolute_error', metrics = [metric])
model.summary()
```

```
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_6 (LSTM)               (None, 34361, 128)        66560

 dense_6 (Dense)             (None, 34361, 1)          129

=================================================================
Total params: 66,689
Trainable params: 66,689
Non-trainable params: 0
_________________________________________________________________
```

In [90]:
```
sequence = np.array(Data['scaled value'])
print(sequence)
time_steps = 34361
samples = len(sequence)
trim = samples % time_steps
subsequences = int(samples/time_steps)
sequence_trimmed = sequence[:samples - trim]

print(samples, subsequences)
```

```
[0.19209494 0.19059101 0.19277132 ... 0.18685179 0.18812086 0.18812086]
68722 2
```

Out[90]: (2, 34361, 1)

Training Algorithm

In [91]:
```
training_dataset = sequence_trimmed

batch_size = 128
epochs = 20

model.fit(x = training_dataset, y = training_dataset, epochs = epochs, verbose = 1,
          validation_data = (training_dataset, training_dataset),
         callbacks =[TensorBoard(log_dir = f'../logs/{0}' '{tensorlog}')])
```

```
Epoch 1/20
1/1 [==============================] - 26s 26s/step - loss: 0.3123 - mean_absolute_erro
r: 0.3123 - val_loss: 0.3029 - val_mean_absolute_error: 0.3029
Epoch 2/20
1/1 [==============================] - 24s 24s/step - loss: 0.3029 - mean_absolute_erro
r: 0.3029 - val_loss: 0.2926 - val_mean_absolute_error: 0.2926
Epoch 3/20
1/1 [==============================] - 22s 22s/step - loss: 0.2926 - mean_absolute_erro
r: 0.2926 - val_loss: 0.2806 - val_mean_absolute_error: 0.2806
Epoch 4/20
1/1 [==============================] - 22s 22s/step - loss: 0.2806 - mean_absolute_erro
r: 0.2806 - val_loss: 0.2657 - val_mean_absolute_error: 0.2657
Epoch 5/20
1/1 [==============================] - 21s 21s/step - loss: 0.2657 - mean_absolute_erro
r: 0.2657 - val_loss: 0.2456 - val_mean_absolute_error: 0.2456
Epoch 6/20
1/1 [==============================] - 21s 21s/step - loss: 0.2456 - mean_absolute_erro
r: 0.2456 - val_loss: 0.2151 - val_mean_absolute_error: 0.2151
Epoch 7/20
1/1 [==============================] - 21s 21s/step - loss: 0.2151 - mean_absolute_erro
r: 0.2151 - val_loss: 0.1604 - val_mean_absolute_error: 0.1604
Epoch 8/20
1/1 [==============================] - 21s 21s/step - loss: 0.1604 - mean_absolute_erro
r: 0.1604 - val_loss: 0.0336 - val_mean_absolute_error: 0.0336
Epoch 9/20
1/1 [==============================] - 22s 22s/step - loss: 0.0336 - mean_absolute_erro
r: 0.0336 - val_loss: 0.1295 - val_mean_absolute_error: 0.1295
Epoch 10/20
1/1 [==============================] - 21s 21s/step - loss: 0.1295 - mean_absolute_erro
r: 0.1295 - val_loss: 0.1524 - val_mean_absolute_error: 0.1524
Epoch 11/20
1/1 [==============================] - 21s 21s/step - loss: 0.1524 - mean_absolute_erro
r: 0.1524 - val_loss: 0.1581 - val_mean_absolute_error: 0.1581
Epoch 12/20
1/1 [==============================] - 22s 22s/step - loss: 0.1581 - mean_absolute_erro
r: 0.1581 - val_loss: 0.1590 - val_mean_absolute_error: 0.1590
Epoch 13/20
1/1 [==============================] - 22s 22s/step - loss: 0.1590 - mean_absolute_erro
r: 0.1590 - val_loss: 0.1575 - val_mean_absolute_error: 0.1575
Epoch 14/20
1/1 [==============================] - 21s 21s/step - loss: 0.1575 - mean_absolute_erro
r: 0.1575 - val_loss: 0.1542 - val_mean_absolute_error: 0.1542
Epoch 15/20
1/1 [==============================] - 21s 21s/step - loss: 0.1542 - mean_absolute_erro
r: 0.1542 - val_loss: 0.1492 - val_mean_absolute_error: 0.1492
Epoch 16/20
```

```
1/1 [==============================] - 21s 21s/step - loss: 0.1492 - mean_absolute_erro
r: 0.1492 - val_loss: 0.1422 - val_mean_absolute_error: 0.1422
Epoch 17/20
1/1 [==============================] - 21s 21s/step - loss: 0.1422 - mean_absolute_erro
r: 0.1422 - val_loss: 0.1330 - val_mean_absolute_error: 0.1330
Epoch 18/20
1/1 [==============================] - 20s 20s/step - loss: 0.1330 - mean_absolute_erro
r: 0.1330 - val_loss: 0.1211 - val_mean_absolute_error: 0.1211
Epoch 19/20
1/1 [==============================] - 21s 21s/step - loss: 0.1211 - mean_absolute_erro
r: 0.1211 - val_loss: 0.1058 - val_mean_absolute_error: 0.1058
Epoch 20/20
1/1 [==============================] - 21s 21s/step - loss: 0.1058 - mean_absolute_erro
r: 0.1058 - val_loss: 0.0866 - val_mean_absolute_error: 0.0866
```

Out[91]: <keras.callbacks.History at 0x7f55b90fa090>

In [92]:

```
import math
from sklearn.metrics import mean_squared_error

sequence = np.array(Data['scaled value'])
print(sequence)
time_steps = 34361
samples = len(sequence)
trim = samples % time_steps
subsequences = int(samples/time_steps)
sequence_trimmed = sequence[:samples - trim]

print(samples, subsequences)
sequence_trimmed.shape = (subsequences, time_steps, 1)
print(sequence_trimmed.shape)

testing_dataset = sequence_trimmed
print("testing_dataset: ", testing_dataset.shape)

testing_pred = model.predict(x=testing_dataset)
print("testing_pred: ", testing_pred.shape)

testing_dataset = testing_dataset.reshape((testing_dataset.shape[0]*testing_dataset.sha
print("testing_dataset: ", testing_dataset.shape)

testing_pred = testing_pred.reshape((testing_pred.shape[0]*testing_pred.shape[1]), test
print("testing_pred: ", testing_pred.shape)
errorsDF = testing_dataset - testing_pred
print(errorsDF.shape)
rmse = math.sqrt(mean_squared_error(testing_dataset, testing_pred))
print('Test RMSE: %.3f' % rmse)
```

```
[0.19209494 0.19059101 0.19277132 ... 0.18685179 0.18812086 0.18812086]
68722 2
(2, 34361, 1)
testing_dataset:  (2, 34361, 1)
WARNING:tensorflow:6 out of the last 2154 calls to <function Model.make_predict_functio
n.<locals>.predict_function at 0x7f55b90e4c20> triggered tf.function retracing. Tracing
is expensive and the excessive number of tracings could be due to (1) creating @tf.funct
ion repeatedly in a loop, (2) passing tensors with different shapes, (3) passing Python
objects instead of tensors. For (1), please define your @tf.function outside of the loo
p. For (2), @tf.function has reduce_retracing=True option that can avoid unnecessary ret
racing. For (3), please refer to https://www.tensorflow.org/guide/function#controlling_r
etracing and https://www.tensorflow.org/api_docs/python/tf/function for  more details.
1/1 [==============================] - 3s 3s/step
testing_pred:  (2, 34361, 1)
```

```
testing_dataset:  (68722, 1)
testing_pred:  (68722, 1)
(68722, 1)
Test RMSE: 0.090
```

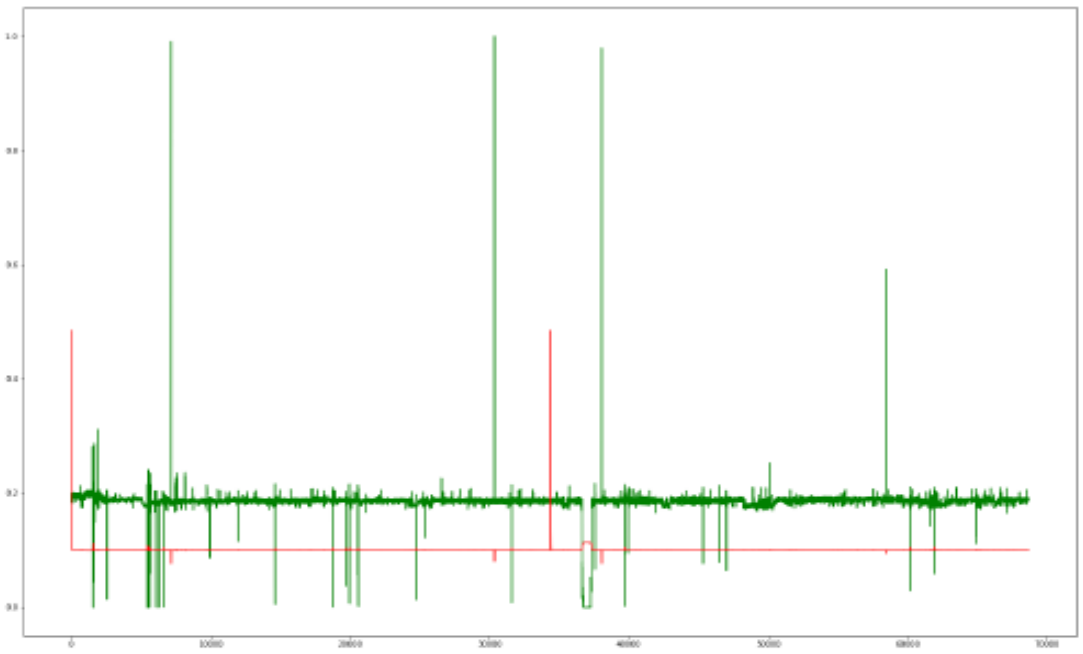
Getting Threshold Value

In [93]:
```
#based on cutoff after sorting errors
dist = np.linalg.norm(testing_dataset - testing_pred, axis=-1)

scores =dist.copy()
print(scores.shape)
scores.sort()
cutoff = int(0.999 * len(scores))
print(cutoff)
#print(scores[cutoff:])
threshold= scores[cutoff]
print(threshold)
```

```
(68722,)
68653
0.21178787271031924
```

In [94]:
```
plt.figure(figsize=(24,16))
plt.plot(testing_dataset, color='green')
plt.plot(testing_pred, color='red')
```

Out[94]:
```
[<matplotlib.lines.Line2D at 0x7f55b9a070d0>]
```

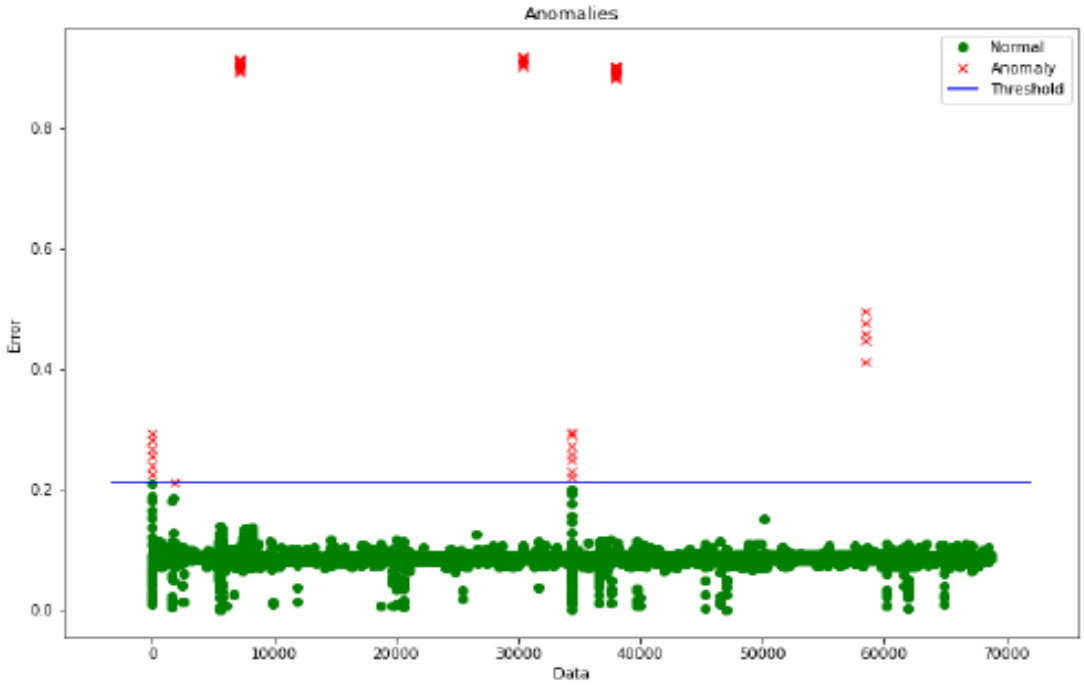


In [1]:
```
#Label the records anomalies or not based on threshold
z = zip(dist >= threshold, dist)

y_label=[]
```

```
error = []
for idx, (is_anomaly, dist) in enumerate(z):
    if is_anomaly:
        y_label.append(1)
    else:
        y_label.append(0)
    error.append(dist)
```

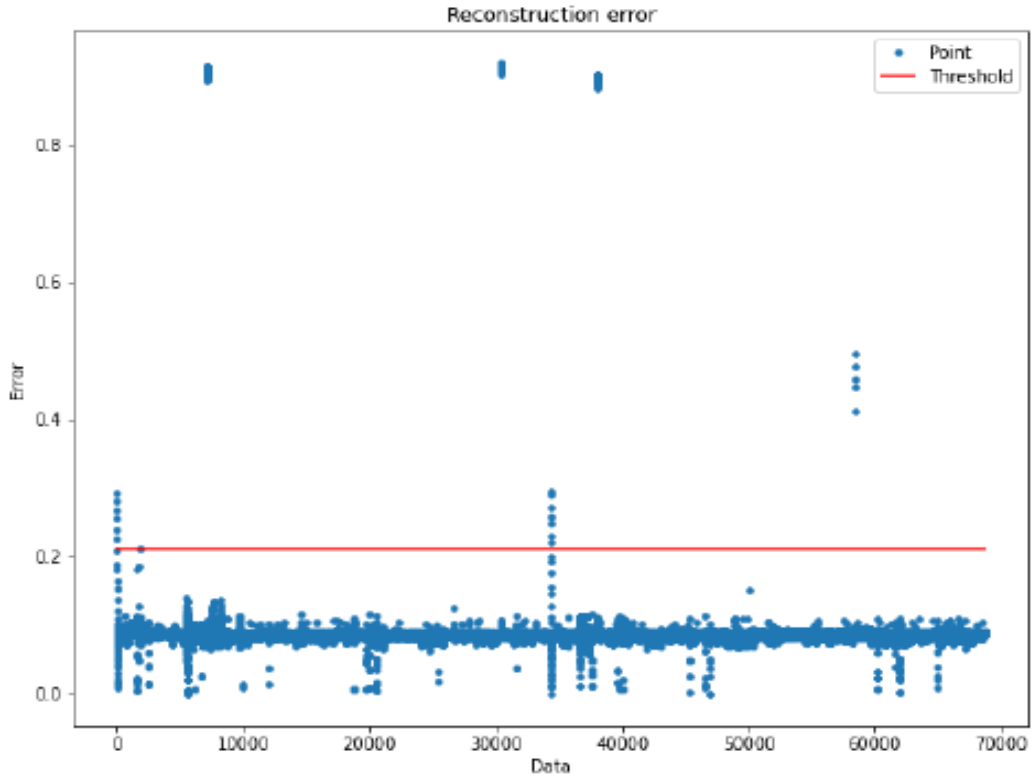
In [98]:
```
class Visualization:
    labels = ["Normal", "Anomaly"]

    def draw_anomaly(self, y, error, threshold):
        groupsDF = pd.DataFrame({'error': error,
                                 'true': y}).groupby('true')

        figure, axes = plt.subplots(figsize=(12, 8))

        for name, group in groupsDF:
            axes.plot(group.index, group.error, marker='x' if name == 1 else 'o', lines
                    color='r' if name == 1 else 'g', label="Anomaly" if name == 1 else

        axes.hlines(threshold, axes.get_xlim()[0], axes.get_xlim()[1], colors="b", zord
        axes.legend()

        plt.title("Anomalies")
        plt.ylabel("Error")
        plt.xlabel("Data")
        plt.show()

    def draw_error(self, error, threshold):
        plt.figure(figsize=(10, 8))
        plt.plot(error, marker='o', ms=3.5, linestyle='',
                 label='Point')

        plt.hlines(threshold, xmin=0, xmax=len(error)-1, colors="r", zorder=100, label=
        plt.legend()
        plt.title("Reconstruction error")
        plt.ylabel("Error")
        plt.xlabel("Data")
        plt.show()
```

In [99]:
```
viz = Visualization()
viz.draw_anomaly(y_label, error, threshold)
```



In [100…

```
viz.draw_error(error, threshold)
```



In [101…
```
Anomaly_Data = pd.DataFrame({'Datetime': Data['datetime'], 'observation': Data['value']
                             'error': error, 'anomaly': y_label})
Anomaly_Data.head(5)
```

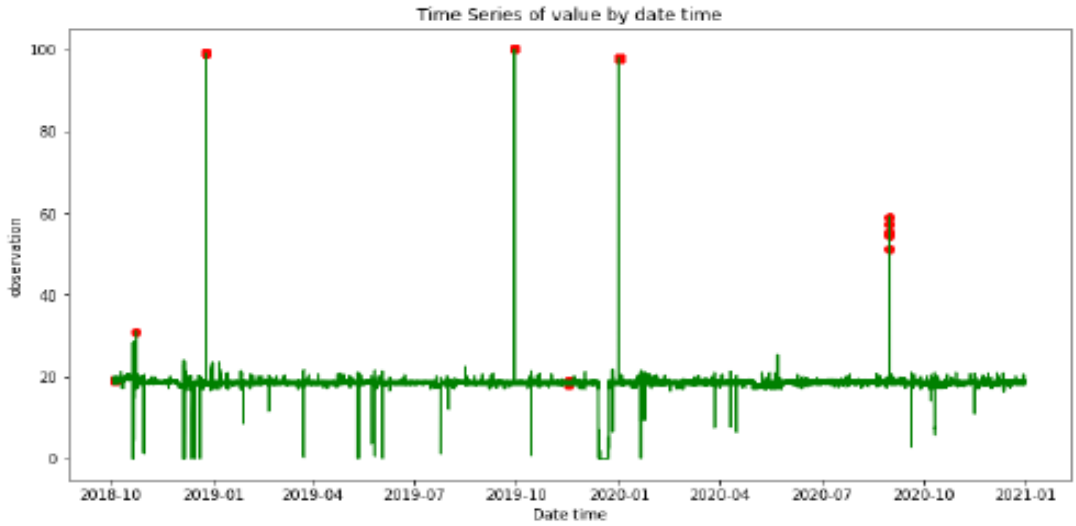
Out[101…

	Datetime	observation	error	anomaly
0	2018-10-04 00:00:00	19.19495	0.293084	1
1	2018-10-04 00:27:00	19.04453	0.282277	1
2	2018-10-04 00:28:00	19.26260	0.267057	1
3	2018-10-04 00:37:00	19.05637	0.255544	1
4	2018-10-04 00:46:00	19.25317	0.239548	1

Visualizing Anomalies Detected

In [102…
```
figure, axes = plt.subplots(figsize=(12, 6))
axes.plot(Anomaly_Data['Datetime'], Anomaly_Data['observation'], color='g')
anomaliesDF = Anomaly_Data.query('anomaly == 1')
axes.scatter(anomaliesDF['Datetime'].values, anomaliesDF['observation'], color='r')
plt.xlabel('Date time')
plt.ylabel('observation')
plt.title('Time Series of value by date time')
```

Out[102… `Text(0.5, 1.0, 'Time Series of value by date time')`



APPENDIX F: Python Outlier Detection Algorithm Implementation

In [1]:
```
import warnings
import numpy as np
import pandas as pd
from pyod.models.mad import MAD
from pyod.models.knn import KNN
from pyod.models.lof import LOF
import matplotlib.pyplot as plt
from sklearn.ensemble import IsolationForest

pd.set_option('max.rows', 250)
warnings.filterwarnings('ignore')
```

```
C:\Users\Mnesoma\Anaconda3\lib\site-packages\pandas\compat\_optional.py:138: UserWarnin
g: Pandas requires version '2.7.0' or newer of 'numexpr' (version '2.6.9' currently inst
alled).
  warnings.warn(msg, UserWarning)
```

In [2]:
```
Data = pd.read_csv('Pressure_with_Anomaly_2.csv')
```

In [3]:
```
Data['timestamp'] = pd.to_datetime(Data["timestamp"])
```

In [48]:
```
def fit_model(model, Data, column='value'):

    df = Data.copy()
    data_to_predict = Data[column].to_numpy().reshape(-1, 1)
    predictions = model.fit_predict(data_to_predict)
    df['Predictions'] = predictions

    return df

def plot_anomalies(df, x='timestamp', y='value'):

    # categories will be having values from 0 to n
    # for each values in 0 to n it is mapped in colormap
    categories = df['Predictions'].to_numpy()
    colormap = np.array(['g', 'r'])

    f = plt.figure(figsize=(12, 6))
    f = plt.scatter(df[x], df[y], c=colormap[categories])
    f = plt.xlabel(x)
    f = plt.ylabel(y)
    f = plt.xticks(rotation=90)
    plt.show()
```

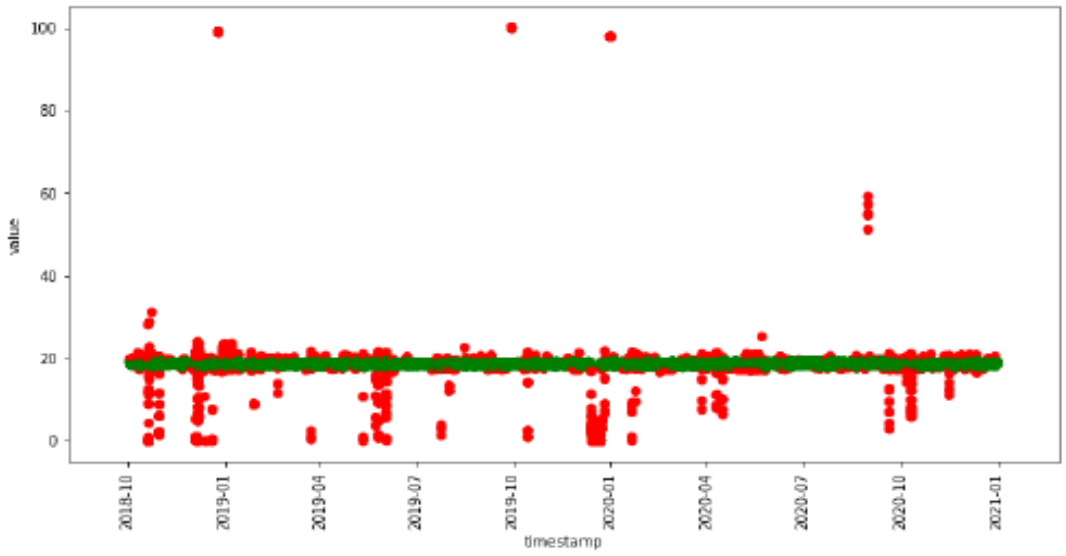
Inter Quartile Range

In [7]:
```
def find_anomalies(value, lower_threshold, upper_threshold):

    if value < lower_threshold or value > upper_threshold:
        return 1
    else: return 0

def iqr_anomaly_detector(Data, column='value', threshold=1.1):

    df = Data.copy()
    quartiles = dict(Data[column].quantile([.25, .50, .75]))
```

```
    quartile_3, quartile_1 = quartiles[0.75], quartiles[0.25]
    iqr = quartile_3 - quartile_1

    lower_threshold = quartile_1 - (threshold * iqr)
    upper_threshold = quartile_3 + (threshold * iqr)

    print(f"Lower threshold: {lower_threshold}, \nUpper threshold: {upper_threshold}\n"

    df['Predictions'] = Data[column].apply(find_anomalies, args=(lower_threshold, upper
    return df
```

In [49]:
```
iqr_df = iqr_anomaly_detector(Data)
plot_anomalies(iqr_df)
```

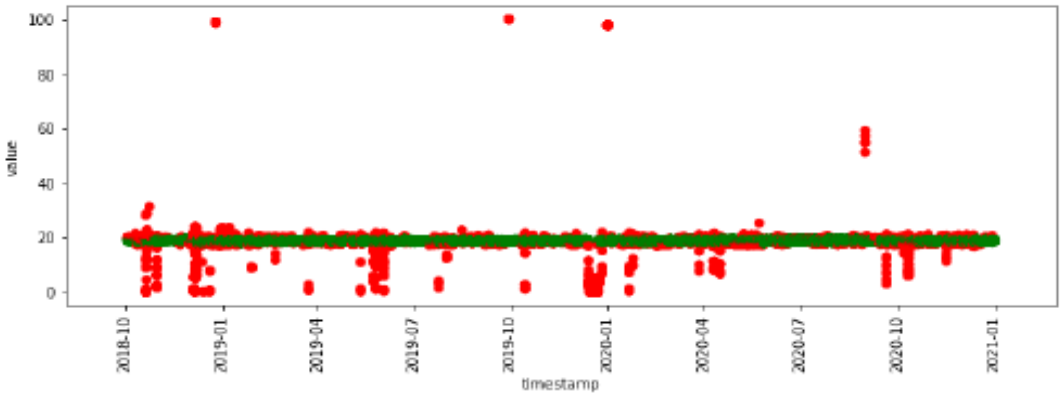
```
Lower threshold: 17.701954000000004,
Upper threshold: 19.560705999999996
```



Isolation Forest

In [13]:
```
def scale_anomaly_scores(s):
    """Changing (-0.5, 0.5) to (0, 1)"""
    a, b = (-0.5, 0.5), (1, 0)
    (a1, a2), (b1, b2) = a, b
    return (b1 + ((s - a1) * (b2 - b1) / (a2 - a1))) * 100
```

In [47]:
```
iso_forest = IsolationForest(n_estimators=125)
iso_df = fit_model(iso_forest, Data)
iso_df['Predictions'] = iso_df['Predictions'].map(lambda x: 1 if x==-1 else 0)
plot_anomalies(iso_df)
```



In [15]:
```
iso_df.head()
```

Out[15]:

	timestamp	value	Predictions
0	2018-10-04 00:00:00	19.19495	0
1	2018-10-04 00:27:00	19.04453	0
2	2018-10-04 00:28:00	19.26260	0
3	2018-10-04 00:37:00	19.05637	0
4	2018-10-04 00:46:00	19.25317	0

In [16]:
```
univariate_data = Data['value'].to_numpy().reshape(-1, 1)
# These values range from -0.5 to 0.5
anomaly_scores = iso_forest.decision_function(univariate_data)
anomaly_scores
```

Out[16]:
```
array([0.04886386, 0.10295042, 0.00900373, ..., 0.10285085, 0.11731228,
       0.11731228])
```

In [29]:
```
iso_df['Anomaly Scores'] = anomaly_scores
iso_df.head()
```

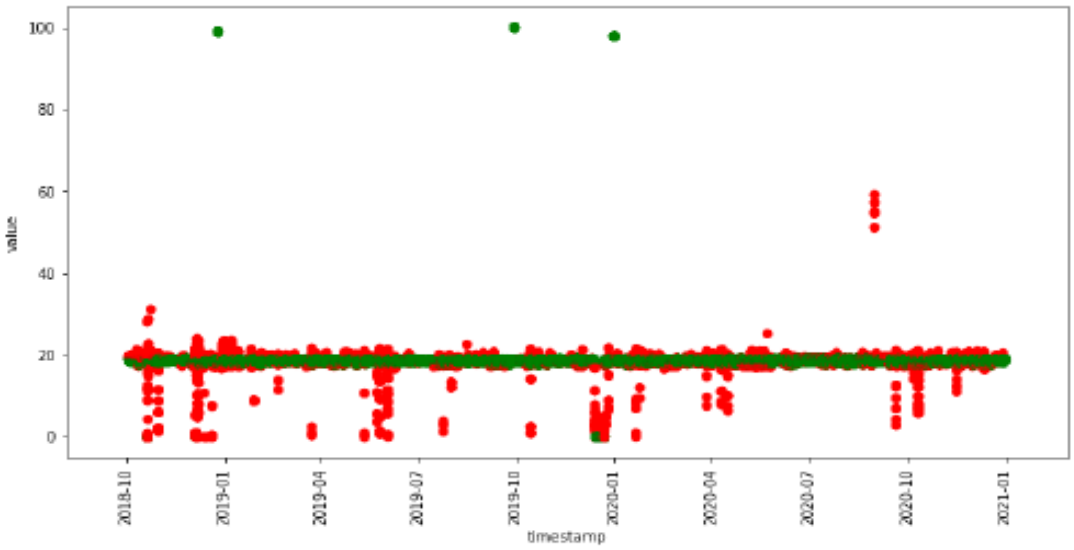
Out[29]:

	timestamp	value	Predictions	Anomaly Scores
0	2018-10-04 00:00:00	19.19495	0	0.048864
1	2018-10-04 00:27:00	19.04453	0	0.102950
2	2018-10-04 00:28:00	19.26260	0	0.009004
3	2018-10-04 00:37:00	19.05637	0	0.093530
4	2018-10-04 00:46:00	19.25317	0	0.019190

KNN model

In [50]:
```
"""KNN Based Outlier Detection"""
knn_model = KNN()
```

```
knn_df = fit_model(knn_model, Data)
```



In [32]:
```
def get_anomaly_scores(model):
  anomaly_scores = model.decision_scores_
  threshold = model.threshold_
  return anomaly_scores, threshold
```

In [33]:
```
anomaly_scores, threshold = get_anomaly_scores(knn_model)
print(f"Anomaly Scores: {anomaly_scores}, \nThreshold: {threshold}")
```

```
Anomaly Scores: [8.0e-05 2.0e-05 1.3e-04 ... 5.0e-05 3.0e-05 3.0e-05],
Threshold: 0.0002099999999991553
```

In [34]:
```
knn_df['Anomaly Scores'] = anomaly_scores
knn_df.head()
```

Out[34]:

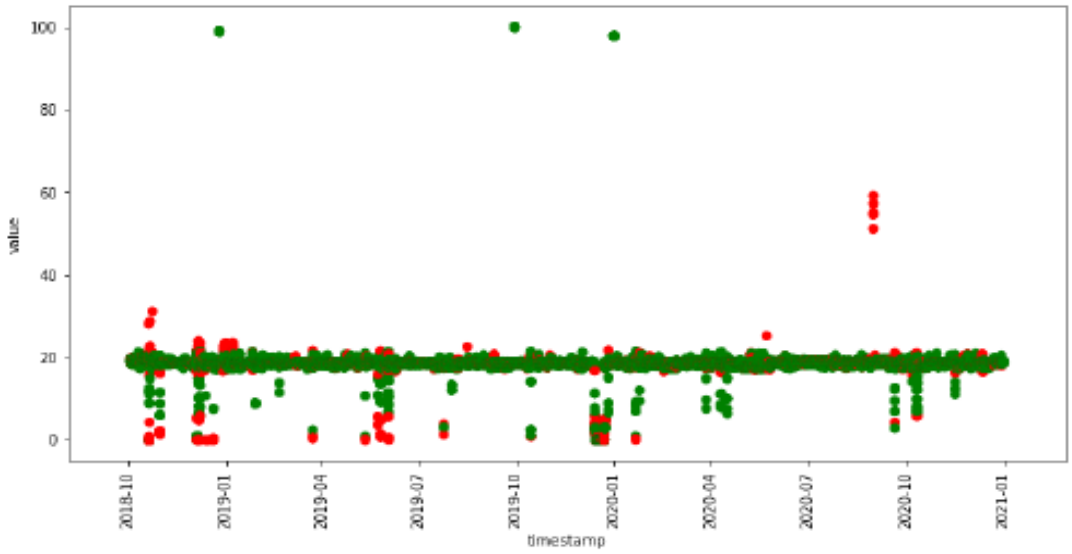
	timestamp	value	Predictions	Anomaly Scores
0	2018-10-04 00:00:00	19.19495	0	0.00008
1	2018-10-04 00:27:00	19.04453	0	0.00002
2	2018-10-04 00:28:00	19.26260	0	0.00013
3	2018-10-04 00:37:00	19.05637	0	0.00006
4	2018-10-04 00:46:00	19.25317	0	0.00013

In [35]:
```
# Threshold computation
pd.Series(anomaly_scores).quantile(.90)
```

Out[35]:
```
0.0002099999999991553
```

Local Outlier Factor Model

In [51]:
```
lof_model = LOF()
lof_df = fit_model(lof_model, Data)
plot_anomalies(lof_df)
```



In [37]:
```
anomaly_scores, threshold = get_anomaly_scores(lof_model)
print(f"Anomaly Scores: {anomaly_scores}, \nThreshold: {threshold}")
```

```
Anomaly Scores: [1.00666953 1.04217189 0.9899425  ... 0.98774249 0.97775098 0.97775098],
Threshold: 1.0797704251094205
```

In [38]:
```
# Threshold computation
pd.Series(anomaly_scores).quantile(.90)
```

Out[38]:
```
1.0797704251094205
```

In [ ]:
```

```