Detecting Anomalies in Programmable Logic Controllers through Parameter Modeling

---

Chandni Raghuraman

September 30, 2021

University of Twente

Department of Electrical Engineering, Mathematics and Computer Science (EEMCS)
Services, Cybersecurity & Safety (SCS)

Master Thesis

Detecting Anomalies in Programmable Logic Controllers through Parameter Modeling

Chandni Raghuraman

Supervisor and Committee Chair	Dr.Ir. Andrea Continella Faculty of EEMCS and Services, Cybersecurity & Safety (SCS) University of Twente
Committee Member (External)	Dr. Suzan Bayhan Faculty of EEMCS and Design and Analysis of Communication Systems (DACS) University of Twente
Committee Member	Prof.Dr. Andreas Peter Faculty of EEMCS and Services, Cybersecurity & Safety (SCS) University of Twente
Committee Member	Herson Tobias Esquivel Vargas
Supervisors at Deloitte	Mike van der Boon and Dominika Rusek-Jonkers

September 30, 2021

UNIVERSITY OF TWENTE. Deloitte.

Abstract

Industrial Control Systems (ICS) are used to control and automate critical processes in various industrial sectors. Programmable Logic Controllers (PLCs) are a major component within the ICS infrastructure and are used to control logical operations within the system. With the increase of attacks on ICS in recent years, there is a need for robust security solutions in this domain. ICS have different components that are manufactured by various vendors, and therefore, a security solution for ICS must be compatible across a variety of proprietary hardware and software systems. Most ICS security mechanisms are deployed at the higher levels of the ICS setup, making it difficult to secure critical lower level devices such as the PLCs. This thesis proposes an anomaly detection mechanism for ICS that is based on the input and output values of the PLC. This solution views the PLC as a "black-box", thereby enabling its deployment on any PLC, irrespective of the PLC's manufacturer. One-Class Support Vector Machine (OCSVM) is a semi-supervised machine learning algorithm and is used in this thesis to model the parameters of the PLC during normal or baseline functioning. The OCSVM is trained to be sensitive to changes in the PLC's parameters and it classifies PLC data as inliers or outliers. Outlier detection or anomaly detection by the OCSVM indicates that the PLC data has to be investigated further to determine if said outlier or anomaly is an indication of any malicious activity in the ICS environment. The mechanism proposed in this thesis is tested on two different ICS environments, namely Fortiphyd and Elite Town. Attacks are simulated on both environments to acquire necessary data and the performance of the OCSVM on both ICS setups is analyzed during the training phase and the testing phase. The results from the experiments support the working of the anomaly detection mechanism proposed in this thesis and its advantages and provide some directions for further improvement and future work.

Acknowledgement

I would like to express my sincere gratitude to the following people, who have made this endeavour possible with their constant support and guidance.

To Dr.ir. Andrea Continella, University of Twente,
Your guidance and patience with me throughout this thesis has been invaluable and helped me navigate this complex topic to a point where I could proudly present my work. Thank you for helping me stay motivated during the tough times and giving me constructive feedback when I needed it the most. I have learnt a lot from you, and for that, I will always be grateful.

To the Faculty of the SCS Research Group, University of Twente,
Thank you for your enthusiasm and interest in my work. Your feedback in the research group meetings and seminars helped me find breakthroughs in critical junctures of my thesis. I would like to thank Prof. Andreas Peter for giving me the encouragement to pursue my thesis topic in ICS security. I would also like to thank Herson for being a constant support and guiding me through the challenges that are unique to ICS.

To Mike van der Boon and Dominika Rusek-Jonkers, Deloitte Netherlands,
Thank you for being amazing mentors and guiding me through the various challenges I faced during this thesis. Your enthusiasm in my topic paved the way for me to produce the best possible results for my thesis. Your experience and domain expertise led to various insights without which this thesis would not have been possible. Thank you for being a great source of motivation, whether they were the various conversations we had or the early coffee meetings.

To my colleagues,
I could not have asked for a better set of people to ease me into my first corporate experience. Special thanks to Arjen, Jip, Iris, Sabien and Ivan for making my internship a memorable one.

To my friends,
Be it India or the Netherlands, thank you for tolerating my endless bouts of frustration. Thank you for injecting joy and laughter into what would have otherwise been a tedious and lonely journey. Your support and curiosity never failed to boost my morale.

To my family,
Thank you for your love and patience. This has been a tough journey, but knowing that you were just one phone call away made me believe that I could always get through it.

Contents

Acronyms

CPS Cyber Physical Systems.

DCS Distributed Control Systems.

DDoS Distributed Denial of Service.

DMZ Demilitarised Zone.

DPC Discrete Process Control.

DTMC Discrete Time Markov Chain.

EDR Endpoint Detection and Response.

FBD Functional Block Diagrams.

FN False Negative.

FP False Positive.

FSM Finite State Machine.

HMI Human Machine Interface.

HVAC Heating, Ventilation and Air Conditioning.

ICS Industrial Control Systems.

IDS Intrusion Detection Systems.

IT Information Technology.

LL Ladder Logic.

LSTM Long Short Term Memory.

MBTCP Modbus Transmission Control Protocol.

NID Network Intrusion Detection.

NIDS Network Intrusion Detection Systems.

OCSVM One-Class Support Vector Machine.

OT Operational Technology.

PCA Principal Component Analysis.

PLAT PLC Log-Data Analysis Tool.

PLC Programmable Logic Controller.

RTOS Real-Time Operating Systems.

SCADA Supervisory Control And Data Acquisition.

SNIFU Secure Network Interception for Firmware Updates.

SVM Support Vector Machine.

t-SNE t-distributed Stochastic Neighbourhood Embedding.

TN True Negative.

TP True Positive.

1 Introduction

Operational Technology (OT) refers to computing systems that are used to manage industrial operations such as production line management, mining operations control, medical manufacturing and other industrial critical infrastructures. Industrial Control Systems (ICS) is a major segment within OT that encompasses several types of control systems, including Supervisory Control And Data Acquisition (SCADA) systems and Distributed Control Systems (DCS) [1]. ICS are business-critical applications with high-availability requirements and are managed via Programmable Logic Controller (PLC) or Discrete Process Control (DPC) systems. The controllers are used to operate and automate industrial processes that maintain critical portions of industrial infrastructure. PLCs consist of various components, most importantly firmware, which dictate the device logic for the PLC. Firmware handle PLC inputs and outputs that can influence the behaviour of the PLC, and in turn, the ICS environment in which the PLC operates [2].

As part of 'Industry 4.0', also known as the fourth industrial revolution, many critical industries are integrating Information Technology (IT) with OT, thereby making it possible for ICS devices to communicate with their IT counterparts and become more sophisticated in terms of capability and automation [3]. This marks the rise of IT-OT convergence. A key feature of IT-OT convergence is the introduction of the internet to ICS for functions like relaying firmware updates to ICS devices.

IT-OT convergence has its benefits, but it has also introduced new threats that could disrupt safety-critical processes. ICS attacks lead to loss of integrity and availability of the device resources and communication [4], [5], which is detrimental to the industry as it affects process efficiency, reduces system uptime and also leads to monetary loss. A malicious attack or tampering of the controls could lead to catastrophic outcomes such as a power outage or crippling of safety systems in oil and gas facilities [6]. Attacks meant to weaken IT devices can now propagate to the ICS network and attack ICS devices as well [7]. During such attacks, the device firmware or logic of ICS components can be compromised, which leads to changes in processes, and the behaviour of the device is no longer corresponding to the expected one. For example, in the Stuxnet attack (Section A.1), malware infecting the PLCs caused the centrifuges to spin at a speed 1/3 times higher than the rated speed, disrupting the chemical reaction needed for the nuclear reactor to function smoothly [8].

Analyses of Stuxnet and other ICS attacks like CrashOverride [9] and Trisis [10] have been conducted by reverse engineering the ICS device firmware used in the attack environment [2]. The analyses contribute towards identifying and addressing manufacturing problems with proprietary ICS devices. However, the research does not give enough insight into the behavioural changes of the compromised devices during run-time and their effects on the ICS system. More details on malware attacks targetting ICS is explained in the Appendix (Section A.1). Parameter changes such as changes in device input and output can be critical indicators of a malicious breach as many attacks on ICS

tend to overload the ICS network or PLC memory [11]. Since ICS devices are deployed in settings that remain unchanged or static for decades, it is possible to detect any malicious activity exhibited by these devices if their behaviour in normal conditions can be thoroughly analysed. When a monitoring system recognises behaviour patterns in ICS components that deviate from its normal function, those incidents can be isolated and investigated further. This method can be beneficial in thwarting cyber attacks towards the ICS infrastructure.

1.1 Motivation

The ICS infrastructure consists of various components, making it a complex network of devices that are not as powerful or technologically advanced compared to their IT counterparts. Post IT-OT convergance, critical endpoints of ICS networks are exposed to threats originating from IT networks that they are not capable of managing on their own [12]. To help secure the OT infrastructure, monitoring and threat detection mechanisms are installed in the Demilitarised Zone (DMZ), which acts as a bridge between the IT and OT networks.

Figure 1.1 shows the structure of the Purdue Model (explained in Section A.2.1) and the location of various critical ICS devices. Security measures post the DMZ are scarce due to the complicated setup and volume of devices in the OT environment. The PLC is a critical component in all ICS environments [13] and it is usually present in Level 1 of the IT-OT setup. PLCs control the hardware components and tools used for carrying out processes or monitoring processes in the ICS infrastructure. The role of the PLC is dictated by programmable logic, and it can perform a variety of operations across a vast range of industrial sectors. If security threats bypass the DMZ undetected and tamper with the PLC, it can cause sever damage to the entire ICS setup. Due to the importance of PLCs and its critical location in factory setups, creating a security mechanism by focusing on PLCs is a promising method of improving ICS security.

A key vulnerable area in ICS is the lack of security in communication protocols [14]. Shortcomings in communication protocols are being investigated by many, including the Group for Advanced Information Technology (GAIT) and Cisco Systems [15], giving impetus to research for securing ICS communications. A solution to secure ICS environments with focus on PLCs will require data from PLC parameters such as inputs and output data. The necessary information can be procured by gaining access to the PLC logs [16] or by accessing the ICS network and capturing data from communication protocols. Since communication protocols are exploited extensively during attacks on ICS, this data can be considered more practical and robust than PLC logs to create an anomaly detection tool. Information gathered from PLC logs is limited by what the PLC is programmed to record, however, in communication protocols, all information from the ICS network can be used to understand the role of the PLC in the ICS setup better.

ICS environments function continuously, with very limited downtime. As a result, anomaly detection for ICS requires continuous observation and analysis of data from the devices. Intrusion Detection Systems (IDS) for Cyber Physical Systems (CPS) have gained popularity since the Stuxnet [17] attack and various tools involving machine learning are used to evaluate the security of a critical

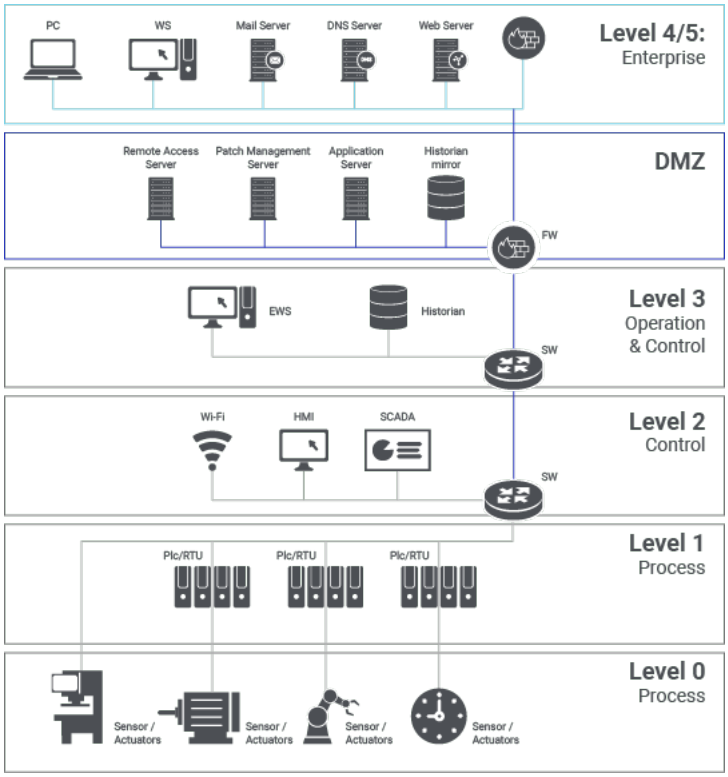


Fig. 1.1.: The Purdue Model describing typical factory setup after IT-OT convergence.

ICS infrastructure. By using algorithms like Naive Bayes, Decision Tree and Gradient Boosting, many safety-related anomalies can be detected in a CPS network [18]. Machine learning allows for changes in the ICS environment or device functioning to be taken into consideration, which will enable the data to be adaptable over a long period of time [19]. Machine learning based security solutions are usually deployed in the higher levels of ICS, and does not focus on device-level data. A solution using machine learning that focuses on learning from PLC data will be novel in ICS and also allow for the processing of large amounts of data over a long period of time, without introducing any additional devices to the ICS infrastructure.

Machine learning relies heavily on the availability of data and its characteristics. Data from communication protocols used by PLCs, that are continuously communicating with other ICS components, can provide a large and reliable database for use with machine learning algorithms. This data, coupled with semi-supervised or unsupervised machine learning methods used for anomaly detection [20], provides promising results for creating an anomaly detection mechanism focusing on PLCs. Machine learning can learn from data spanning decades, and thus, any changes to the normal behaviour of PLCs through firmware patches can also be taken into consideration.

Based on the inferences from literature review, this thesis proposes an anomaly detection mechanism for ICS infrastructures that focuses on PLCs. This mechanism helps strengthen ICS security in the lower levels of the ICS setup. This research makes use of PLC input-output data collected from communication protocols to model the normal or baseline behaviour of PLCs using machine learning. Any change in PLC parameters caused by sending malicious network packets can be detected by identifying PLC behaviour that is deviating from the modeled baseline behaviour. The anomaly

detection mechanism proposed in this research can be deployed in any ICS environment without introducing any new devices to the complex ICS network.

1.2 Problem Statement

This research aims to detect anomalous behaviour in PLC processes which would arise from malicious attacks on PLC device logic. The majority of the literature on PLC focused security mechanisms [2], [5], [7] analyze the kernel and code of PLC firmware. Many ICS vendors build proprietary devices and thus, firmware-based solutions have limited applications. Works such as TABOR [21] and SNIFU [13] focus on securing the ICS network independent of the device features, however, they introduce additional components to the ICS setup for their execution. This thesis proposes an ICS security solution that will focus on the visible process changes exhibited by the PLC as a result of malicious attacks and how they can be detected by analyzing the PLC as a "black-box". The data required to understand the nature of the parameters handled by a PLC can be obtained from the network traffic between PLCs and other ICS components. Using machine learning to learn the characteristics of the PLC parameters under normal conditions, an anomaly detection mechanism can be created specifically for PLCs. This approach is independent of firmware features, and thus can be deployed across various ICS setups. The solution also resolves the drawback of SNIFU and TABOR as it does not require any additional tool to be deployed in the ICS infrastructure.

The following are the Research Questions (RQs) for this thesis:

RQ1. How to study the normal behaviour of a PLC?

RQ2. How to distinguish malicious behaviour exhibited by the PLC from its normal behaviour?

RQ3. How effective is the machine learning model when used for anomaly detection in different ICS environments?

1.3 Approach and Results

In order to collect the relevant data and analyse the performance of anomaly detection using machine learning, I made use of two ICS setups. The first setup was a virtual chemical process simulation called Fortiphyd (explained in Section 5.1.1) and the second setup was a working demo of a hydro-electric power plant called Elite Town (explained in Section 5.1.2). I gained access to the ICS network consisting of the PLC and other ICS components such as the Human Machine Interface (HMI) or other SCADA devices of the two setups. Capturing the communication data between these components helped me obtain data containing PLC parameters such as inputs and outputs to PLC holding registers and coils. I captured data from both ICS setups during the standard operating mode in order to gain baseline behaviour information on the PLC. The data was pre-processed and learnt by a One-Class Support Vector Machine (OCSVM). Post the learning phase, I simulated some

network based attacks in the ICS setups and the data captured during this time was fed to the OCSVM to detect anomalous behaviour (Section **??**). Since the data is encapsulated in network packets, and attacks are simulated through access to the ICS network, the anomaly detection mechanism proposed in this thesis could be categorised as an example of Network Intrusion Detection (NID).

In anomaly detection, inliers and outliers are identified in a given dataset. Since the dataset created for this thesis was collected by simulating attacks during runtime, it is difficult to establish ground truths. In the training phase, which consists of training and validation of the OCSVM, the model is trained using data indicating baseline behaviour of the PLC. Hence, I could conclude that each packet identified as an inlier during the validation phase is a True Positive (TP) and each oulier is a False Negative (FN). However, in the testing phase, the dataset modeled contained PLC data from both normal and attack conditions. Once inliers and outliers were identified by the OCSVM, I verified if every packet identified as an outlier showed any sign of the attack I had simulated. If the signs were present, it was a True Negative (TN), else it was a False Negative (FN). Thus, the anomaly detection model is semi-supervised, as the training phase includes supervised learning and the testing phase uses unsupervised learning performance metrics to evaluate the performance of the OCSVM.

I analysed the similarities and differences in the performance of the OCSVM on the two different ICS setups. I observed the correlation between features present in the datasets from the two setups and the influence they have over the performance of the OCSVM. I also conducted experiments to find the optimal amount of data needed for the OCSVM to learn the characteristics of both setups and to give the best outlier detection results for each setup.

2 Related Work

2.1 Analysis of firmware security

Firmware in ICS literature refers to both Real-Time Operating Systems (RTOS) as well as the logic component of an ICS device. When it comes to firmware security, most literature focuses on the security of RTOS. Andrei et al. [22] unpacked 32 thousand firmware images of PLCs from various vendors and conducted reverse engineering to produce an extensive dataset. This work uncovered vulnerabilities in the firmware that was previously unknown and also helped understand the current state of firmware security for many PLCs commonly used in modern ICS setups. The results were obtained by breaking encryption schemes at the device level as well as network level, authentication methods for certificates, identifying backdoors and manipulating device or network code to taint them [23]. While the work raised several ethical, legal and infrastructural concerns related to PLC as a RTOS, it only focused on the internal processes of the PLC. There exists a research gap in understanding the effect of PLC firmware manipulation on the external ICS network during run-time. Most of the results produced from firmware reverse engineering can be categorised as forensic analysis, which occurs after an attack has been successfully carried out. Firmware reverse engineering helps to make the PLC more secure when it is manufactured, and thus can be applied only to PLCs from specific vendors. In order to improve ICS security as a whole, the PLC must be treated as a "black-box" and focus on the PLCs effects on the entire ICS network. This approach can result in a more general solution that can be deployed across ICS setups in different industrial sectors.

2.2 Deployment of testbed and devices for anomaly detection

To improve ICS security, methods such as the deployment of testbeds and emulation of PLCs have been used. These methods are used in Secure Network Interception for Firmware Updates (SNIFU), which is a tool that makes use of these methods and works best for legacy PLCs [13]. Firmware images of the Wago PLC were reverse engineered to get an accurate understanding of the functioning of a PLC firmware, which was then emulated by the SNIFU device and deployed in the ICS network. Firmware updates passed through the network were first run on this emulation device and when no strange behaviour was exhibited, the update was passed onto the actual PLC. The emulation was successful as SNIFU had access to the firmware signing mechanism that triggered the update to run on its testbed environment. The use of machine learning to automate the classification of code instructions from PLC data and the effective deployment of the device within the ICS network are

some of the key features of this tool. The tool was built using Raspberry Pi, which is an embedded low-level device that can interact with PLCs and also give a programming interface compatible with various programming languages. SNIFU also had ICS network monitoring capabilities, and hence it improved the visibility and monitoring of processes around the PLCs.

Weaselboard [24] is a dedicated device connected to modular PLCs and detects changes in control settings and sensor values in an ICS environment. Weaselboard observes and collects data on the control settings and configuration information of the ICS setup. The tool also analysed firmware and logic information of the ICS devices under its purview. Weaselboard countered the challenge of low-frequency, high-impact attacks from advanced adversaries who use zero-day attacks against PLCs. The device can detect the impact of exploits against PLCs as soon as the state of the PLC changes, rather than after serious damage has occurred. This feature emphasises the need for run-time anomaly detection coupled with forensic investigation. Weaselboard was tested on Allen Bradley Control Logix 5000 Backplane, and it has to be expanded to be used for other PLC vendors and their respective backplane software. Thus, the Weaselboard solution has not yet developed as a large-scale solution providing behaviour-based anomaly detection for PLCs, but the work highlighted some important aspects of such a solution.

While both SNIFU and Weaselboard show signs of strengthening ICS security, they come at the cost of adding complexity to the ICS network. The environment is riddled with many components and hardware, and thus, it could be inconvenient to add testbeds and tools to the network. The processing power required by these additions could also reduce the efficiency of the ICS network, which is very undesirable.

2.3 Anomaly Detection using Machine Learning

Machine Learning has been used extensively in anomaly detection for IT as well as OT infrastructures. The choice of a machine learning algorithm depends on the type of data collected and the features of the dataset. For detecting anomalous behaviour in ICS, the basic distinction that must be made is between normal and abnormal device behaviour. Such a classification would require a one-class classifier. Neural Network implementations like Long Short Term Memory (LSTM) can be used as a one-class classifier to detect fault tolerance and errors in PLC-aided automation systems in ICS [25]. Other machine learning algorithms include Finite State Machine (FSM) in TABOR [21], Discrete Time Markov Chain (DTMC) which can be used as an alternative to FSM [19] and Support Vector Machine (SVM).

A graphical model-based anomaly detection mechanism for ICS called TABOR was designed in 2018, which profiled the normal operational behaviour of a Water Treatment device to detect any abnormalities [21]. Timed automata were learned as a model of everyday behaviours seen in sensor signals such as water level variations in tanks. To discover dependencies between sensors and actuators, Bayesian networks were used. For process anomaly identification, the models were used as a one-class classifier, detecting unusual behaviours and dependencies. Due to the interpretability of the graphical models, the results helped locate suspicious sensors or actuators by tracing back the

state diagrams. This solution made good use of machine learning and also highlighted the use FSMs to extrapolate behavioural characteristics of a device. FSM consists of a defined finite number of states the device can be in and rules for how the device moves from one state to another. Control Logic determines the inputs to the PLC within a state and decides the movement to the next state. The control logic can be programmed using various programming languages such as Functional Block Diagrams (FBD), Ladder Logic (LL), structured text and sequential logic [26]. TABOR is a good example of an ICS security mechanism that uses machine learning to model the behavior of multiple ICS processes. Such a solution would work well in a complex ICS setup, however, it is difficult to replicate this solution to focus on PLCs. Despite this drawback, the methodology to build this solution can be used as an inspiration for a more device focused solution.

PLC Log-Data Analysis Tool (PLAT) is an automated tool that can detect operational faults and behavioural anomalies in PLCs by using the log data of PLC signals [16]. PLAT generates a notional model of the PLC control process automatically and uses a novel hash table-based indexing and searching strategy to achieve anomaly detection. Some advantages of PLAT include fast and real-time anomaly identification. It can be executed with minimal storage and processing requirements, allowing for the use cases to be varied. PLAT also introduces the need for introducing anomaly detection mechanisms in ICS without adding any devices to the already complicated ICS infrastructure. PLAT is a good security solution focused on PLCs, however, it is held back by the limitations present in the PLC log data. A PLC's log data does not give insight into the PLC's parameters with respect to the ICS network, and this could be a critical aspect in creating a robust security solution focused on PLCs.

A novelty detection tool for identifying faulty operations of Heating, Ventilation and Air Conditioning (HVAC) chiller systems was introduced using One-Class Support Vector Machine (OCSVM) [20]. Detecting and mitigating faults in such systems ensured that the energy consumption by the systems remained within acceptable levels and thereby improved energy efficiency. This research introduced a major issue that many face when conducting novelty detection, and that is, the lack of ground truth values for anomalous data. Even in this research, the data from the HVAC systems were unlabelled as the normal functioning of the systems conceal changes in the data related to anomalous conditions. The proposed combination of OCSVMs for classification and Principal Component Analysis (PCA) for discarding the variability related to usual operating conditions changes is effective in the detection of anomalous situations without using labeled data. This solution may not be device-focused, however, it gave insights on how to validate the performance of machine learning when the dataset does not contain ground truths.

3 Threat Landscape

ICS security has gained increased momentum after the discovery of Stuxnet in 2010 [17]. Malicious attacks have often targeted ICS networks unintentionally, as a side effect of attacking an IT infrastructure connected to an ICS environment. Thus, it was important to gain insight not only on the attack vectors for ICS but also on the environment and motivation behind attacking it.

Malware attacks targeting ICS environments has increased in the past decade, as ICS setups are less secure compared to IT devices and an attack with minimal effort can lead to impacts of much greater proportions. Section A.1 in the Chapter A gives details on some of the major ICS attacks like Stuxnet, Triton CrashOverride and BlackEnergy. It gives insight on the motivation behind the malware affecting specific ICS environments and the varied impact they have on ICS security and research to strengthen ICS infrastructure.

Due to the rise in malware attacking ICS, questions were raised on the setup of devices within the ICS network. As part of industry 4.0, some of the key drivers for organizations to evolve towards using modern networking systems to interconnect ICS with business and external networks have been the increasing need to reduce manufacturing and operational costs, improve productivity, and provide access to real-time information [27]. IT-OT convergence has exposed critical infrastructure ICS networks to a variety of external and internal threats, as well as misconfigurations and computing errors [27], [28]. Popular terminologies about ICS setup used in industry are The Purdue Model (Section A.2.1) and the Cyber Kill Chain (Section A.2.2). Modes of propagation of attacks are often explained using one of these two setup conventions, hence we felt it is necessary to take a closer look at what they mean. Learning about these setup terminologies helped to understand the literature that discussed the analysis of ICS firmware threats from the perspective of where the attack is taking place and how. It cemented the notion that ICS infrastructure remain unchanged for years on end and it is difficult to make large scale changes to device setup.

3.1 Current state of ICS Security

ICS plays a role in automating various vertical markets like manufacturing, transportation, energy management and smart city automation [29]. PLCs within such systems are usually dedicated to one particular process or task. For example, a PLC is used to automate the pumping of water from a well to a bucket. Sensors present in the bucket indicate to the PLC how long it needs to pump water so that it does not overflow. If there is an attack on the PLC firmware that leads to denial of service, then the sensor to PLC data will get misinterpreted and the PLC will pump more water than the bucket capacity. While this is a simple example with no dangerous consequences, such

mistakes are not tolerated in more sensitive settings such as a nuclear power plant. PLCs from the same manufacturer may be used in both sensitive and relatively less critical infrastructure, wherein the vulnerabilities due to easy access to the PLC will persist.

Most ICS/SCADA setups today are isolated from the IT infrastructure that is more prone to malicious attacks. However, this setup has not stopped attacks from propagating and causing ICS disruptive events due to IT/OT convergence. While monitoring systems like antivirus and Endpoint Detection and Response (EDR) are actively deployed to secure IT devices, their visibility in OT is comparatively lacking [30]. Although Network Intrusion Detection Systems (NIDS) are present to monitor the ICS network, many attacks like tampering of ICS device logic can go undetected. In malware attacks such as BlackEnergy A.1, the Distributed Denial of Service (DDoS) attacks were carried out using commands that were allowed by the ICS environment [31]. As a result, the commands were not flagged as malicious by NIDS systems. However, the behaviour resulting from the DDoS attack would be different from normal run-time behavior and this change could be an indicator for detecting anomalous and possibly malicious activities occurring in the ICS environment.

A characteristic of ICS environments is that most equipment and devices are installed with the intent to use for decades and the functional requirements do not change drastically over time [11]. Upgrading hardware or software in industrial sites leads to factory downtime, which is not desirable. Thus, updates are rarely performed for ICS sites and the cyber industry often considers ICS as a "static" environment [30]. Inside ICS, cybersecurity choices are guided by sensitivity to process preservation. It is acceptable to use security controls in front of these systems to minimize risk rather than updating and potentially degrading system performance [12]. Due to this, security research must cater to both legacy and relatively new devices spanning across different industrial sectors and device vendors [6].

The increase in awareness regarding ICS security has led to a surge in ICS centric security solutions sold by vendors. Claroty IDS (Section A.3) Dragos Inc. [32] are leading ICS cybersecurity vendors in the market. Their solutions cover OT network monitoring and security deployment in the DMZ. They also backup monitoring data to cloud solutions for forensic investigation, if an attack take takes. While these solutions have good reviews, they can be expensive and may take a long time to be deployed in ICS environments. These solutions may also force industries to change their ICS setups, which can cause disruptions in their process throughput. Industries reliant on ICS have very little tolerance for downtime, and thus, they would find such top-down solutions inconvenient to deploy. Hence, despite the existence of all-encompassing and potentially successful ICS security solutions, they are seen as an inconvenience to deploy. Hence there is a need for small-scale ICS security solutions that are easy to deploy and also strengthen the security of the ICS environment with minimal disruptions.

4 Methodology

4.1 General approach

The approach taken to design and evaluate the proposed anomaly detection tool for ICS is as shown in Figure 4.1. Before collecting PLC parameters, it is necessary to understand the nature of the ICS environment and the PLC's role in the given setup. This knowledge helps to understand the type of data handled by the PLC and the attacks that can be simulated in the ICS environment, as explained in Section 4.2. Once the ICS environment is scoped, data from the network is collected from communication protocols. There are a wide range of protocols used in ICS such as Siemens Step7, Modbus and Profibus [14]. The protocol will determine the PLC registers in which the data is stored and also help determine the attacks that could be deployed on the network (Section 4.3). Once attacks are simulated and the necessary data is collected, the data is processed to get information on the PLC parameters. Post data-processing, OCSVM is used to model the data and determine the inliers and outliers in the network packet dataset. The reasoning behind using OCSVM is explained in Section 4.4.

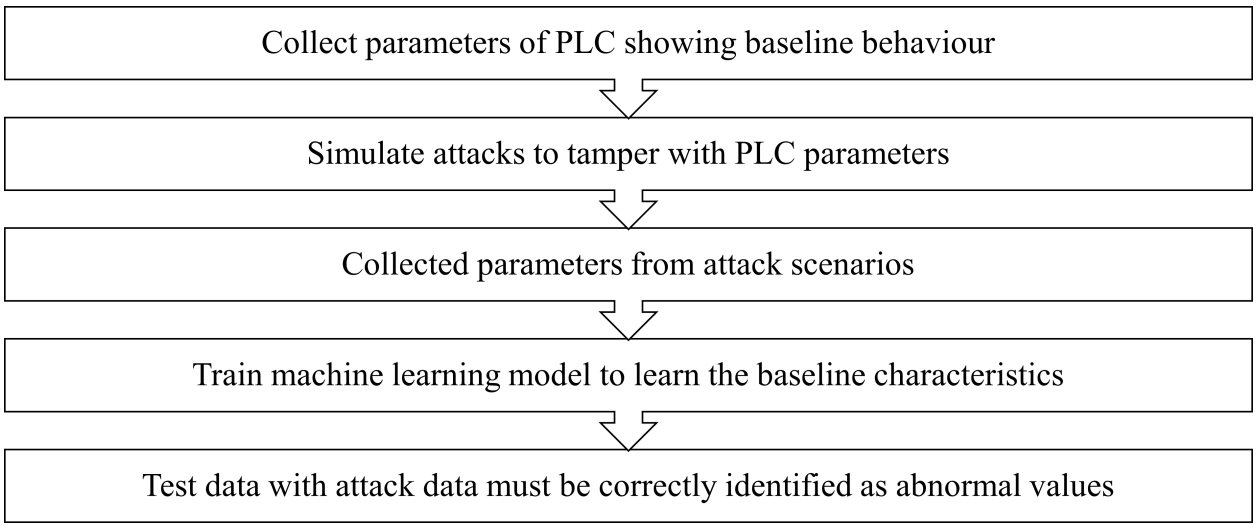


Fig. 4.1.: General approach towards proposed solution

4.2 Understanding PLCs

PLCs handle the logical processing and control of Level 0 actuators in the ICS infrastructure (Section A.2). The use of programming languages such as Ladder Logic, SFC and FBD are stipulated by the IEC 61131-3 standard and it helps bring uniformity to the programming used for PLCs in ICS [33].

PLCs have three types of raw data, they are as follows [34]:

1. Analog data - used to measure continuous signals like temperature fluctuations

2. Logic or Binary data - can be time-dependent or a direct representation of logic operations carried out by the PLC

3. Discrete data - generated from digital components such as sensors or device clocks.

When observing the communication between a PLC and other devices in the ICS network using network traffic analyzer tools, it can be seen that analog data in PLCs is stored in 'Holding Registers', while binary data is stored in 'Register Coils'. Discrete data is stored in holding registers and it is not shared over network communication protocols unless coded otherwise.

To understand the normal behaviour of a PLC and answer RQ1 (Problem Statement RQ1), it is necessary to understand its role in a given ICS infrastructure. The type of data that needs to be handled can be inferred from the responsibilities of the PLC in the given environment. For example, if the PLC logs the pressure inside a pressure controlled environment, the pressure data will be relayed to it in analog form. If the pressure is tampered with as a result of a malicious attack, the PLC will continue to log the undesired pressure values. A monitoring system evaluating the PLC data can alert security mechanisms upon viewing the undesired pressure values from the PLC log.

4.3 Data collection and attack simulation

First, network data is collected from the ICS environment during its baseline or normal functioning. This data will later be used as the training data for the machine learning model to be used for anomaly detection. Once sufficient baseline data is collected, it is time to simulate the attacks. The most common and impactful attack techniques and the tactics used in ICS are explained in the Mitre Att&cks framework for ICS [35]. Some of the attacks from this framework can be simulated in an ICS environment to inject malicious values to the PLC parameters. The network traffic is captured once again for both setups while the ICS network is under attack. Once the attack is simulated sufficient number of times, this capture can be considered as the test data for the machine learning model. The attack was simulated sporadically as ICS attacks occur in rare instances, and in real-world applications [36], attack packets may be observed is a very small proportion compared to packets that indicate normal behaviour. Thus, the dataset generated from network packet captures is imbalanced.

4.4 Use of One-Class Support Vector Machine

Anomaly detection refers to the problem of finding patterns in data that do not conform to the expected behavior. These non-conforming patterns are often referred to as anomalies, outliers or

exceptions. Of these, anomalies and outliers are two terms used most commonly in the context of anomaly detection; sometimes interchangeably in the context of network data. Anomaly detection can be used to detect malicious activity in critical ICS infrastructure, provided it learns from viable ICS data, which can be best obtained from PLCs. The ICS infrastructure remains static and is subjected to minimal updates due to the proprietary nature of the components and high costs of building the setup. As a result, the normal behaviour of ICS systems can be learnt by observation over a long period of time, without the worry of regular changes to the setup or the observations.

Some challenges that persist when using Anomaly Detection in ICS are as follows [37]:

1. Defining a normal region which encompasses every possible normal behavior is very difficult. In addition, the boundary between normal and anomalous behavior is often not precise. Thus an anomalous observation which lies close to the boundary can actually be normal, and vice-versa

2. Anomalies generated from malicious data can be masked if the attack is undertaken in a manner that causes minimal disruption to the ICS network. The more the indicators of change, the more likely an anomaly is correctly detected

3. Availability of labeled data for training/validation of models used by anomaly detection techniques is usually a major issue in ICS

4. Often the data contains noise which tends to be similar to the actual anomalies and is therefore difficult to distinguish and remove.

Machine Learning is used to create a robust anomaly detection tool by learning the PLC parameter data obtained from the ICS setups. The aim of the anomaly detection model is to identify potentially malicious packets in the ICS network, which can then be investigated further to see if they are indeed malicious or not. Since datasets obtained from packet capture is imbalanced (as explained in Section 4.3), there is a much smaller proportion of packets generated by attacks compared to the packets generated during the normal functioning of both ICS environments. Thus, it seems suitable to conduct one-class classification to distinguish the normal packets or inliers from the malicious packets or outliers. Another feature of datasets obtained using the method in Section 4.3 is the lack of availability of ground truth values for each packet. Since the data was obtained by packet captures, and many packets are transferred between components of the ICS network in a short period of time, it is difficult to label every packet with a truth value, hence, the data is unlabelled. Machine learning techniques that operate in a semi- supervised mode assume that the training data has labeled instances for only the normal class. Since they do not require labels for the anomaly class, they are more widely applicable than supervised techniques.

One-class Support Vector Machine (OCSVM) is chosen as a suitable model as it learns the data and identifies soft as well as hard boundaries for distinguishing the inliers from the outliers. As the datasets are imbalanced and unlabelled, only the majority class, which in this case is the data from normal behaviour or inliers, is needed for training the OCSVM. The data for normal behaviour

can be obtained with the clarity that they are all inliers as the attacks are also generated on the same setups as part of the research methodology. If the OCSVM learns the normal behaviour of a given ICS setup correctly, it can detect any new type of packets as outliers and those packets can be investigated further. This ability cannot be assured by a classification model [38].

As the datasets obtained are imbalanced, the regular conditions for assessing the performance of machine learning algorithms that include the confusion matrix cannot be used. The OCSVM is trained using normal or baseline data, or positive values. The testing data introduces attack packets as well as packets exhibiting normal behaviour.

The training data can be divided into a training set and validation set. The training set is usually 70%-80% of the training data while the validation set is the remaining 20%-30% of the training data. In supervised learning methods, the training set is used to train the data while the validation set is used to check if the model has learnt optimally from the training set. Similarly, in a semi-supervised setting, the validation set can be used to check for the performance of the OCSVM using classical parameters like Precision, Recall and F1-score. This method is known as Cross-Validation. Based on the predictions on the validation set, which shows the number of inliers and outliers, the inliers are the TP values and the outliers are the FN values. There will be no instances of FP or TN in this output. The Precision for this result will always remain 1, while Recall or Sensitivity and the F1-score will vary with each execution. Sensitivity evaluates how good the test is at detecting a positive instance, which in this case are the inliers. Thus it is necessary to give importance to the Sensitivity of the OCSVM over other evaluation parameters.

The datasets obtained from network captures may consist of multiple columns, each of which are features, and thus faces the curse of dimensionality. When visualising the predictions and decision boundary of the OCSVM, it is necessary to reduce the features while still retaining the attributes that contribute to sensitive anomaly detection. The data in this research is reduced using t-distributed Stochastic Neighbourhood Embedding (t-SNE). It is used over other reduction methods like PCA as t-SNE is a non-linear dimensionality reduction technique. t-SNE also preserves the local structure of the data better than PCA while also handling outliers in the data caused due to human or machine error. Once the data is visualized, the soft and hard decision boundary for the data become apparent. Based on the assumption that training samples are representative, an ideal decision boundary of OCSVMs should be neither tight to ensure the generalization of classifiers, nor loose to ensure the sensitivity to outliers [39].

5 Experimental Setup

5.1 ICS systems

The aim of this research is to create an anomaly detection method for PLCs using machine learning and analyze the performance of the proposed mechanism in different ICS setups. In order to do this efficiently, two ICS systems were chosen for this thesis. The ICS components in both setups consist of the HMI, PLC, and I/O modules which are all connected using standard ICS network protocols, including the Modbus protocol [40]. Table 5.1 shows the role of some of the function codes used by Modbus.

5.1.1 Fortiphyd

Fortiphyd is a Graphical Realism Framework for Industrial Control Simulations (GRFICS) developed to help entry-level ICS security professionals [41]. It is a simulation of an ICS environment that enables beginners to understand the expensive, proprietary hardware and software used in ICS and the inherent dangers of manipulating real physical processes. Fortiphyd virtualizes ICS networks, from the operator interface to realistic simulations of physical processes rendered in a 3D gaming engine. Using this framework, users can attack typical ICS vulnerabilities and witness the physical impact in the process visualization. Users can also program methods to strengthen the network against such attacks after acquiring a deeper understanding of the strong interaction between the cyber and physical worlds in ICS networks. This open-source platform is utilized for our research as one of the two ICS setups to understand the functioning of PLCs and build an appropriate anomaly detection mechanism.

The Fortiphyd simulation visualizes a chemical process control network that follows the Cyber Kill Chain (Section A.2.2) ICS setup. Figure 5.1(a) shows the baseline running of the Fortiphyd simulation. Modbus shows register values of the PLC from the network data of the simulation. Since Modbus, and virtually all other ICS network protocols, are unauthenticated [41], it is possible to inject malicious commands directly to the I/O modules, launching Man-in-the-Middle (MITM)

Tab. 5.1.: Modbus protocol function code description.

Function Code	Function Name	Function
1	Read Coils	Reads binary data from PLC
3	Read Holding Register	Reads analog data from one PLC register
4	Read Multiple Holding Register	Reads analog data from all PLC registers at once
5	Write Coils	Writes binary data into PLC
6	Write Holding Registers	Writes analog data into one PLC register

(a) Baseline operation **(b)** Pressure reduced below normal

Fig. 5.1.: Fortiphyd simulation.

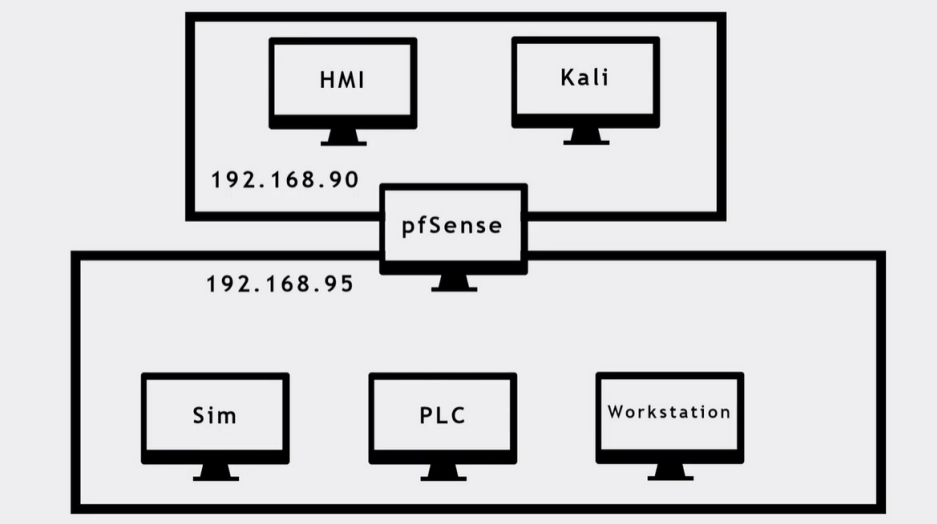


Fig. 5.2.: Fortiphyd Network Setup.

attacks to report false data, and designing firewall rules and intrusion detection rules to protect the insecure-by-design I/O. Figure 5.1(b) shows the simulation when it is under attack due to a MITM malicious injection attack.

Figure 5.2 shows the network setup that makes the simulation run with multiple Virtual Machines (VM). The setup consists of two network subnets 192.168.90 and 192.168.95. The communication between these subnets is facilitated through the "pfSense" hub. The 192.168.90 subnet consists of the HMI and the Kali Linux attacker VM. The 192.168.95 subnet consists of the workstation VM which contains the OpenPLC [42] program that is used to code the PLC's programming logic. The PLC is also a component present in the same subnet as the workstation, and changes from the OpenPLC code are directly relayed to the PLC within this subnet. The Fortiphyd simulation is also hosted within this subnet. By tapping into this network using the attacker VM, the network data for Fortiphyd is collected and the parameters of the PLC are analysed.

The network data from the Fortiphyd setup shows the following Modbus function codes:

1. Function code 1 - In this use case, the coils contain 1 value in bit 40 which indicates whether the system status is shutdown or not.

2. Function code 4 - In this setup, 13 registers from Register 0 to 12 are used to store various values pertaining to the chemical PCN. For example, Register 4-6 contain the pressure value of the main tank conducting the chemical process. If the pressure value is not within the desirable range, the tank can either cease the chemical reaction or burst due to high pressure.

3. Function code 5 - This function code is seen in the Fortiphyd network when an attack to activate Coil bit 40 is done, triggering the shutdown of the setup.

4. Function code 6 - For Fortiphyd, the pressure values can be set to exceed the pressure limit or reduce the pressure to 0 by writing the undesired values into the registers. This can be done by sending malicious data packets to the network using PyModbus or by gaining access to the Ladder Logic program in the OpenPLC program which is present in the Workstation VM.

5.1.2 Elite Town

Elite Town is an ICS demo custom-built by Deloitte Nederland to demonstrate the need for securing ICS setups to clients. Elite Town is a model consisting of two separate parts - a hydro-electric power plant beneath an acrylic base and a village setting beside it. The scenario depicted in this setup is that the hydro-electric power plant fulfills the power demand of the village. To obtain the correct amount of power from the plant, the PLC senses the power requirements of the village, simulated by the number of houses that are turned on, and regulates the amount of water that flows through the generator. This makes this demo unique in the fact that the PLC is performing a task that is also seen in industrial environments and not a simulation of its behavior. The village part of the model is controlled by a Raspberry Pi Zero (RPI) and the PLC in the power plant communicates with the HMI via the network hub. The OT Laptop hosts the monitoring software to check the voltage and setpoint values that the PLC senses. The OT Laptop gets this information via a connection to the hub.

The Attacker Laptop is introduced in this subnet to simulate MITM attacks within the Elite Town setup. Similar to the attacks simulated in Fortiphyd, a network analyser tool is used to sniff the data from the Elite Town network components. The Modbus protocol is used for communication between the ICS components, and thus, the register values of the PLC is obtained using the sniffed network data. Using the PyModbus Python Library [43], a malicious Python code is made to inject abnormal register values into the PLC via Modbus and the attack is simulated on the Elite Town setup. The network setup of this ICS model is shown in Figure 5.3.

The following Modbus function codes can be observed from the Elite Town network data:

1. Function code 1 - In this use case, the coils contain 2 status bits, one indicates whether the system needs to shutdown and the second indicates if the RPI is running or not.

2. Function code 3 - In this setup, values can be read across 4 registers and they give information about the Voltage I/O and the setpoint value compared to the actuator value that the PLC senses.

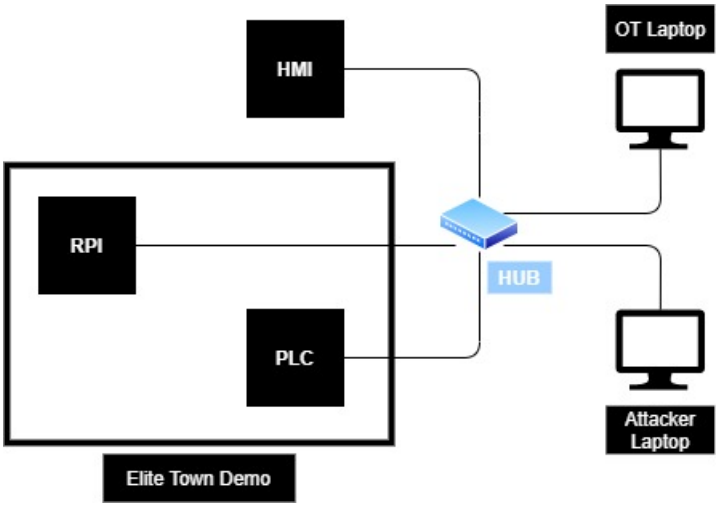


Fig. 5.3.: Elite Town network setup.

3. Function code 5 - A shutdown can be triggered or the RPI can be stopped if the bit values of the coil are tampered with, and this function code will be used to carry out such an attack.

4. Function code 6 - Voltage values and incorrect setpoint values can be fed to the PLC using malicious Modbus command packets that indicate this function code.

5.2 Simulating attacks on the ICS setups

Although the PLC has very different roles in Fortiphyd (Section 5.1.1) and Elite Town (Section 5.1.2), obtaining the parameters of the PLC and also attacking the ICS network by feeding malicious PLC parameter values can be done using the same approach. First, the network data for both setups are collected when they function normally. Due to both setups using the Modbus protocol, the PyModbus Python library was used in order to send malicious Modbus commands with incorrect parameter values. Table 5.2 shows the attacks that I simulated in for this research on the two ICS setups. The network traffic is captured once again for both setups while incorrect values are injected to the PLC through the ICS network. These malicious packets can be indicated by the corresponding function codes that have the "WRITE" operation.

Tab. 5.2.: Techniques simulated in this research mapped to Mitre Att&cks for ICS.

Technique	Tactic	Explanation
Drive-by Compromise	Initial Access	PLC data is captured using network packets by gaining access to the ICS network
Engineering Workstation Compromise	Initial Access	The PLC values are manually changed to abnormal values
Automated Collection	Collection	Network traffic from both ICS setups is collected using Wireshark
I/O Image	Collection	Input and Output parameters for the PLC are obtained via Automated Collection
Man in the Middle	Collection	Automated Collection and I/O Image is possible due to access to the ICS networks
Detect Operating Mode, Monitor Process State	Collection	The status of both ICS setups is indicated by the PLC coil bit value
Change Operating Mode	Execution, Evasion	Both ICS setups can be shut down by changing the PLC coil bit value
Modify Parameter	Impair Process Control	Holding register values can be changed through packet injections to the network
Denial of Service	Inhibit Response Function	Changing Operating Mode and Modifying Parameters leads to this attack

5.3 Deployment of OCSVM

OCSVM is implemented for this research by using the Scikit Learn Machine Learning tool for Python [44] coupled with the Anaconda Python Jupyter Notebook [45] framework. The ICS network data is captured using the Wireshark Network analyzer tool [46] and the Modbus packets are exported as a JSON File. From this file, the PLC parameter data is retrieved and converted into a spreadsheet with each column corresponding to a feature for the OCSVM. Data retrieval and conversion to a tabular format is custom coded using Python using Pandas, a data management library [47]. Figure 5.4 shows the process of data conversion before it is used in the OCSVM.

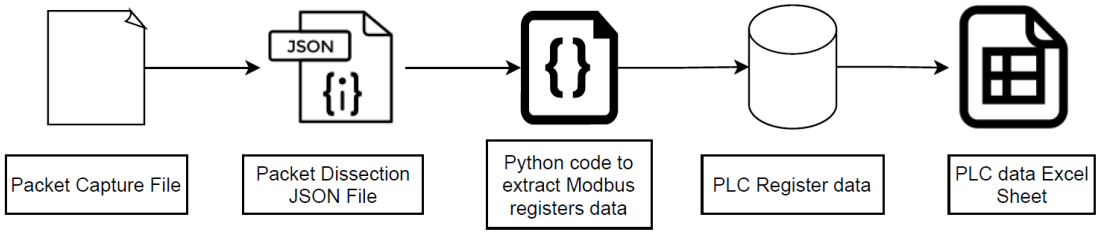


Fig. 5.4.: Network Capture to Spreadsheet Conversion.

Features present in the dataset parsed to the OCSVM include MBTCP Identifier (MBTCP ID), Modbus Function Code, Modbus Reference number, Modbus Word Count, Modbus Byte Count, Modbus Request Frame, Modbus Response Time and the register and coil values that differ between the two setups.

Since the data includes packet transmission, there is always a pair of Query and Response packets for each MBTCP ID. The query packets contain the MBTCP ID and the Function Code, while the response data contains the values of the register or coil, which is decided by the Function code sent by the Query. The reason for including the function code is to allow the OCSVM to learn the relationship between a query-packet pair. This way, packet integrity can be checked, and any failed or delayed transmissions can also be accounted.

The data to be used for training and testing the OCSVM must contain either integers or floating point values. Therefore, the Query/Response tag is label encoded where '1' denotes a query packet and '0' denotes a response packet. There are many values that may not exist in the Modbus encapsulated data depending on whether it is a query or a response packet. For example, there are no register values in query packets. Thus these "NaN" or "NA" values are replaced with '0'.

6 Results

The results obtained on applying the OCSVM (Section 4.4) on the two ICS setups Fortiphyd (Section 5.1.1) and Elite Town (Section 5.1.2) gave some interesting results. They will be analysed individually and together to understand the performance of anomaly detection using OCSVM in different ICS applications.

6.1 Fortiphyd

The PLC data from the Fortiphyd setup includes 13 register values and 1 coil value which when triggered can shut down the virtualization (Section 5.1.1). These values, coupled with the Modbus packet information (Section 5.3) form the features for the dataset pertaining to this model. The data in the training set only includes function codes 1 and 4, as WRITE operations are only performed during attack simulations.

Fortiphyd has a complex setup of ICS components. Thus, the number of packets transmitted across the various components in a short amount of time is very large. However, only 20% of all the transmitted packets are Modbus packets. As a result, the data collection time for Fortiphyd is on the higher side compared to other ICS setups. This is because, the number of components and intranet communication between traditional ICS components may be low.

Table 6.1 shows the recall and f1-score values as the training and validation dataset is increased. The predicted values of the validation set are used to check the recall and f1-score for the OCSVM with respect to the Fortiphyd data. The f1-score was in the range of 0.90 to 0.99 depending on the sensitivity or recall value which was 0.82 to 0.99. The sensitivity values were best generated when

Tab. 6.1.: Fortiphyd Validation set performance

Data collection hours	Training data	Validation set	Inliers	Outliers	Recall	F1-Score	% Outliers
7	19590	5893	4846	1047	0.82	0.901	21.60
7	19590	5893	4846	1047	0.82	0.901	21.60
8	20569	8839	7259	1580	0.82	0.902	21.76
8	21549	9428	7741	1687	0.82	0.902	21.79
10	24487	10607	8732	1875	0.82	0.903	21.47
11	27426	13021	11774	1247	0.90	0.949	10.59
12	30364	12964	11966	998	0.92	0.959	8.34
13	32323	14732	13990	742	0.94	0.974	5.30
14	34282	15910	15685	225	0.98	0.992	1.43
16	39180	17678	17660	18	0.99	0.999	0.10
19	47141	11785	11774	11	0.99	0.999	0.09
21	53386	17433	16981	452	0.97	0.98	2.59

the predicted values had the least number of outliers or false negatives. A better F1-score indicates ta better trained model. The OCSVM was run for multiple iterations while increasing the training data, thereby increasing the number of sets for training and cross-validation. Figure 6.1 shows how the percentage of outliers predicted reduces as the amount of training data increases, and then reaches a constant percentage. Based on the experiments, the optimal amount of training data is 40000 to 48000 training data packets which is obtained when data is collected for 16 - 19 hours. The number of outliers predicted in this range was 18 to 11 packets, which gave a percentage range of 0.10 to 0.09.

When the outlier packets were assessed individually, some patterns could be inferred as to why they were classified as outliers. Some packets had only a query packet or a response packet for a given MBTCP. The second packet was lost either due them being cut at the start and end of the data capture or due to delay in packet retransmission. Usually 3-5 packets have this issue in a pool of 400000 packets, so the probability of the outlier having this issue was quite minimal. A second commonly seen reason among the packets labelled as outliers was the large response time. Response times were sometimes abnormally high due to increase in network traffic when most components are communicating simultaneously or when packet retransmission occurs. As the packet captures are saved in the Wireshark tool, it was beneficial to view the packet capture timestamp to see what was happening in the overall network at that time. The packet capture file gives insight into activities of network protocols other than Modbus, hence increase in network traffic could be detected.

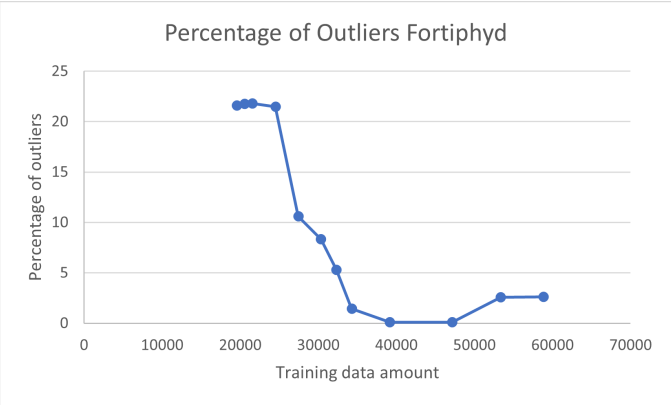
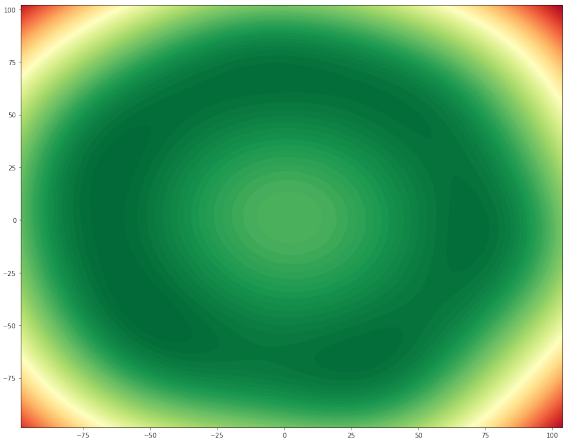


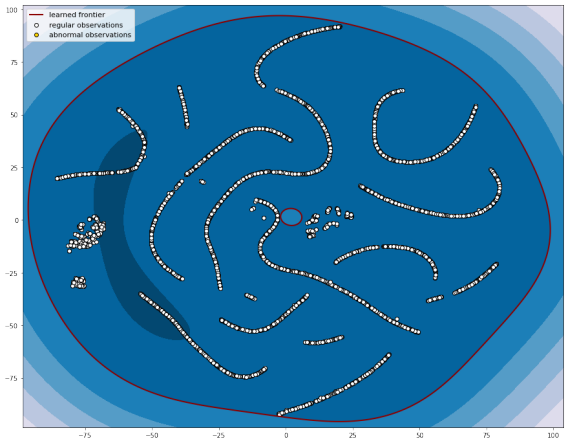
Fig. 6.1.: Percentage of outliers with respect to Training data for Fortiphyd.

In order to visualize the data, the features needed to be reduced to two dimensions, which was achieved through t-SNE reduction. The contour of the decision boundary and the distribution of the inliers and outliers as per this decision boundary can be seen in Figures 6.2(a) and 6.2(b) respectively. This is an example of point-based anomaly detection and the points form a "worm-like" structure as many of the values increase linearly, for example, MBTCP ID and Word Count.

The dataset obtained has many features that are correlated. In order to understand the correlation between them, a heat map was generated for the features, with a coloured gradient indicating the negative to positive correlation between features. Negative correlations are shown in red, neutral correlations in white and positive correlations in blue. The heat map for the Fortiphyd setup is as shown in Figure 6.3.



(a) Decision Boundary Contour for all features



(b) Inlier-Outlier distribution for all features

Fig. 6.2.: OCSVM performance on Fortiphyd.

Tab. 6.2.: F1-score comparison table for Fortiphyd.

Training set data	F1-Score before removing features	F1-Score after removing features
19590	0.902	0.949
20569	0.901	0.948
21549	0.901	0.948
24487	0.903	0.949
27426	0.903	0.949
30364	0.923	0.960
32323	0.954	0.975
34282	0.972	0.986
39180	0.999	0.999
47141	0.999	0.999
53386	0.903	0.949

Based on the correlations shown in the heat map, it can be concluded that features like ID, MBTCP ID, MB Reference Number and MB Word Count can be dropped from the training data. These changes are made and the data is fit into the OCSVM. The performance of the OCSVM without these features is reexamined to see if the performance is improved in any way. Table 6.2 shows the f1-scores as the dataset is increased before and after dimensionality reduction using t-SNE. Figure 6.4 shows that the F1 score has increased with this change, as the sensitivity value lies between 0.90 to 0.99, with the best performing number of training samples remaining the same. This shows that, even after removing some of the badly correlated features, the best number of training samples for the OCSVM with regards to the Fortiphyd data lies between 40000 to 48000 packets.

The data with the removed columns is once again reduced to two dimensions using t-SNE in order to visualise the decision boundary and distribution of inliers as well as anomalies. The new visualization for the decision boundary contour, as seen in Figure 6.5(a), shows a slightly tighter decision boundary at the top. On seeing the distribution of inliers and outliers in Figure 6.5(b), the data points are distributed into clusters and the "worm-like" distribution is lesser compared to

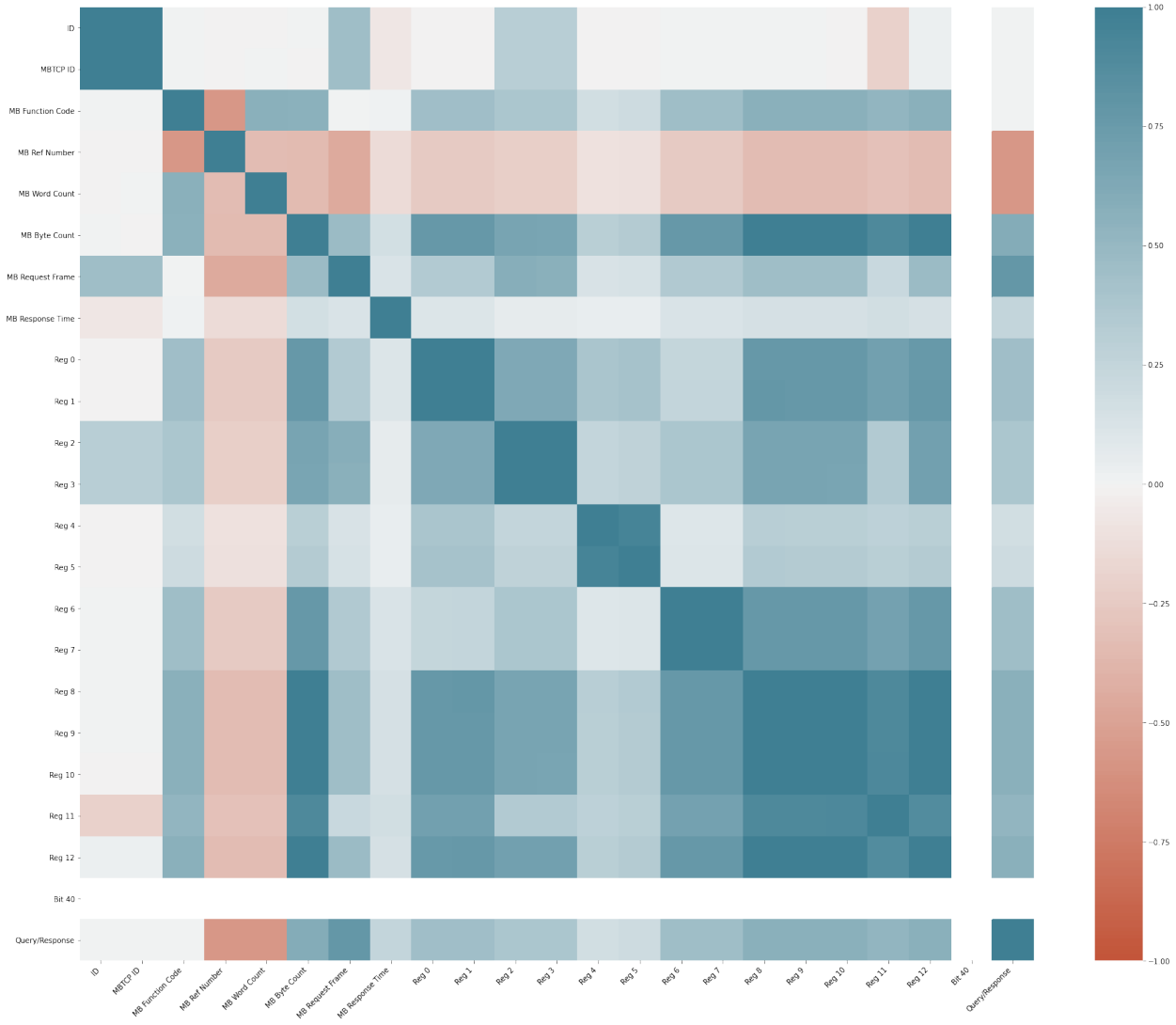


Fig. 6.3.: Heat Map of Fortiphyd features.

Figure 6.2(b). The slightly linear distribution is still seen due to the presence of the feature "MB Byte Count" which increases exponentially as the packet number increases.

The performance of the OCSVM is improved when the badly correlated data is removed from the dataset. However, when it comes to studying the outlier packets to see why they may be predicted incorrectly, it becomes very difficult to find the packet in the network capture without the information present in the removed features. As the performance of the OCSVM remains the same for the optimal range of training data, it can be inferred that there is no disadvantage in training the OCSVM with the badly correlated data. This is because, doing so, does not have a great negative impact on the result and allows for a smoother forensic analysis of the packets that are false negatives.

6.2 Elite Town

The PLC data from the Elite Town model includes 4 Register values and 2 Coil values, along with the Modbus packet information (Section 5.3). Coil Bit 0 is set to FALSE, so that it does not trigger

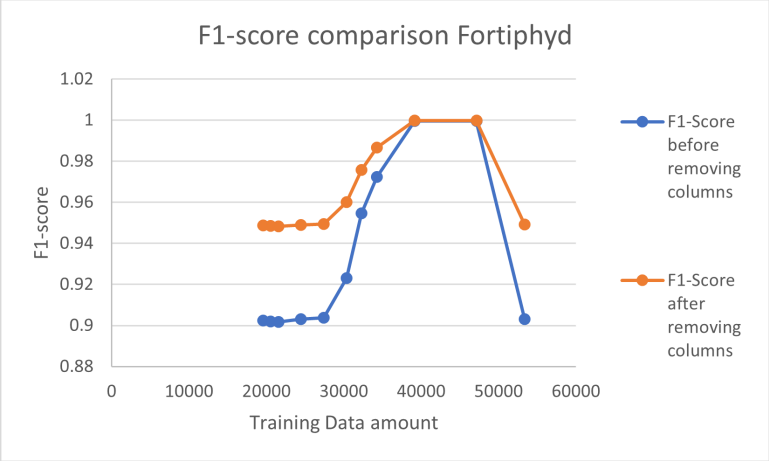
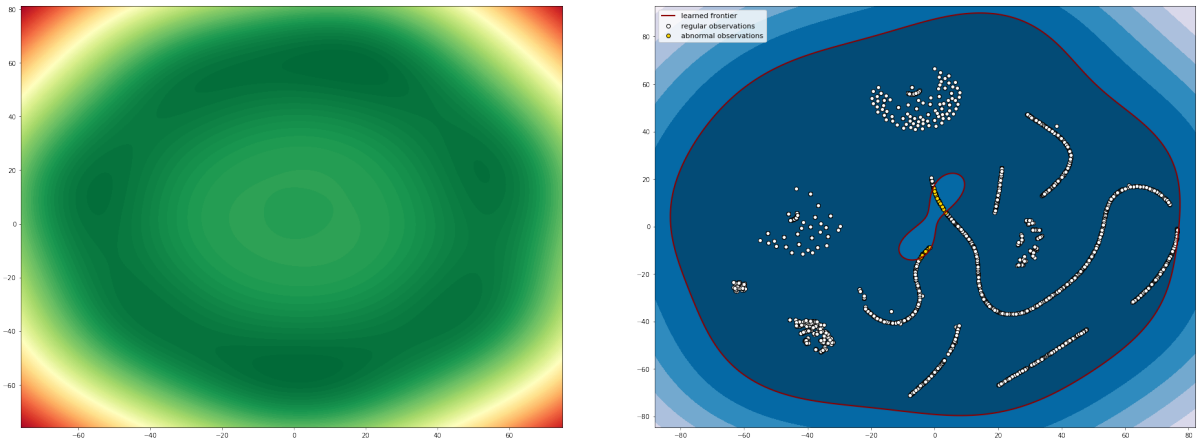


Fig. 6.4.: F1-score comparison for Fortiphyd.



(a) Decision Boundary Contour after removing features **(b)** Inlier-Outlier distribution after removing features

Fig. 6.5.: OCSVM performance on Fortiphyd after feature removal.

a shutdown, and Bit 1 is set to TRUE, to show that the RPI in the village setup is functioning (Section 5.1.2). The data in the training set only include function codes 1 and 3, as information is read by the register one at a time.

Elite Town has a relatively simple setup of ICS components. As a result, the number of packets transmitted across the various components in a short amount is large and more than 70% of all the transmitted packets are Modbus packets. Although this is a good indication for reducing the data collection time compared to Fortiphyd, there is one more difference between the two setups. In Elite Town, the Modbus protocol queries the values of each register in a separate network packet, contrary to its functioning in the Fortiphyd setup, where one network packet can query the values in all 13 Holding Registers. Because of this difference in functioning, it takes 8 network packets, that is, 4 query-response pairs of Function code 3 in order to gain the values present in all the Holding registers. Similarly, 4 network packets or 2 query-response pairs are needed to receive the values in the Register Coils. Thus, despite having a simpler network infrastructure, the Elite Town setup has a high data collection time period as it needs more packets capture to receive all the necessary PLC data.

Tab. 6.3.: Elite Town Validation set performance

Data collection hours	Training data	Validation set	Inliers	Outliers	Recall	F1-Score	% Outliers
2	7740	2322	1179	1143	0.50	0.67	49.22
4	16016	4805	2768	2037	0.57	0.73	42.39
5	21573	6472	3963	2509	0.61	0.75	38.76
6.5	26074	7822	4855	2967	0.62	0.76	37.93
7	28917	8675	5447	3228	0.62	0.77	37.21
8	31087	9326	6217	3109	0.66	0.79	33.33
8	33707	10112	6916	3196	0.68	0.81	31.60
10	41147	12344	8454	3890	0.67	0.79	31.52
12	49793	14937	2541	4788	0.67	0.78	32.06

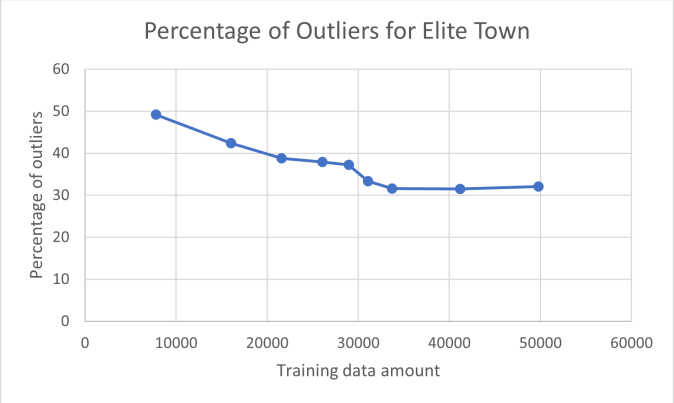
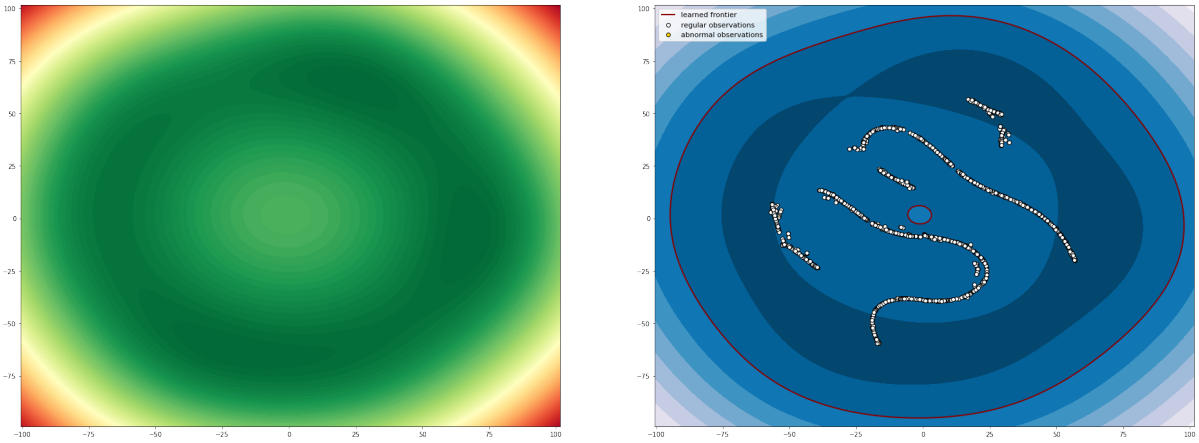


Fig. 6.6.: Percentage of outliers in Elite Town data with respect to Training data.

Table 6.3 shows the recall and f1-score values as the training and validation dataset is increased. The predicted values of the validation set are used to check the recall or sensitivity and F1-score for the OCSVM with respect to the Elite Town data. The F1-score was in the range of 0.67 to 0.81 depending on the Sensitivity value which was 0.50 to 0.68. The sensitivity values were best generated when the predicted values had the least number of outliers or false negatives. A better F1-score indicates that the model was trained better. The OCSVM was run for multiple iterations while increasing the training data, thereby increasing the number of sets for training and cross-validation. Figure 6.6 shows how the percentage of outliers predicted reduces as the amount of training data increases. Based on the experiments, the optimal number of data packets is 31000 to 34000 training data packets which is obtained when data is collected for 8-10 hours. If all the register values were being queried in the same query-response pair like in Fortiphyd, the training and testing data packets needed for optimal learning would be much lesser, due to the simplicity of the Elite Town ICS network.

The number of outliers predicted in the optimal range of training data was 3100 to 3300 packets, that gave a percentage range of 0.31 to 0.33 of the total data used for training and cross-validation. This performance is not desirable, compared to the results of Fortiphyd. When the outlier packets were assessed individually, some patterns could be inferred as to why they were classified as outliers. Similar to Fortiphyd, query-response pairs with missing packets were predicted by the OCSVM as outliers. Another reason why this case was unique to the Elite Town setup was because sometimes, the query packet requests for register coil values of both bit 0 and bit 1 were queried in the same packet. It also received the response with both values in the same packet. While such a query occurs



(a) Decision Boundary Contour for all features

(b) Inlier-Outlier distribution for all features

Fig. 6.7.: OCSVM performance on Elite Town.

infrequently in the network, it is labelled as an outlier as it departs from the normal approach of querying a single value in a single query-response pair. While on inspection it is clear that this is not a malicious case, it does show that the OCSVM is able to learn the unique characteristics of this ICS setup and shows the potential of this mechanism to be adaptable across vastly different ICS network configurations.

The performance of anomaly detection is improved with t-SNE reduction, while also making it possible to visualize the data. The contour of the decision boundary and the distribution of the inliers and outliers as per this decision boundary can be seen in Figures 6.7(a) and 6.7(b) respectively. This is an example of point-based anomaly detection. The points form a "worm-like" structure as many of the value, such as "MBTCP ID" and "Word Count", increase linearly.

In order to determine the correlated features in the Elite Town dataset, a heat map is once again generated, as shown in Figure 6.8.

Based on the correlations shown in the heat map, it can be concluded that features like ID, MBTCP ID, MB Reference Number and MB Word Count can be dropped from the training data, similar to Fortiphyd. It can also be seen that Reg1 and Bit1 have neutral correlation, as they have the same setpoint value and TRUE value respectively, to indicate the setpoint and the positive status of the RPM. In the validation set they may not play a role, but these values are crucial for test data, so we will continue to keep these features. Thus, these changes are made and the data is fit into the OCSVM. The performance of the OCSVM without these features is reexamined to see if the performance is improved in any way. Figure 6.9 shows that the F1 score has increased with this change, as the sensitivity value lies between 0.90 to 0.99. The best performing number of training samples remaining the same. Compared to Fortiphyd, this shows the importance of correlation in the Elite Town data, and removing badly correlated features improved the OCSVM performance to a great extent.

The data with the removed columns is once again reduced to two dimensions using t-SNE in order to visualise the decision boundary and distribution of inliers as well as anomalies. The new

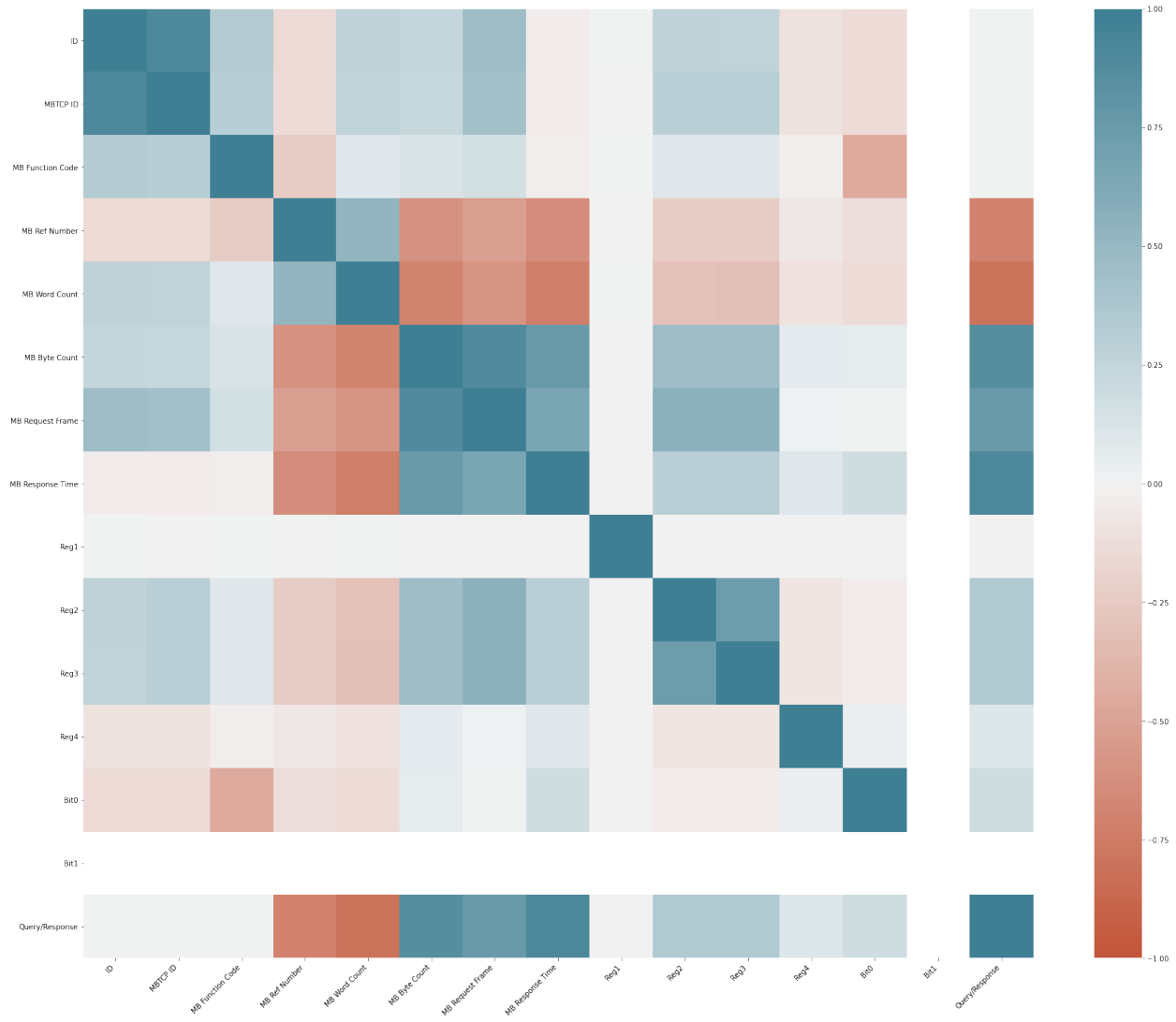


Fig. 6.8.: Heat Map of Elite Town features.

visualization for the decision boundary contour, as seen in Figure 6.10(a), shows a slightly tighter decision boundary to the left. On seeing the distribution of inliers and outliers in Figure 6.10(b), the data points are distributed into clusters and the "worm-like" distribution is lesser compared to Figure 6.7(b).The slightly linear distribution is still seen due to the presence of the feature "MB Byte Count." The MB Byte Count increases linearly as the packet number increases.

Table 6.4 shows the F1-score for Elite Town before and after t-SNE reduction. As the performance of the OCSVM remains the same for the best range of training data, it can be inferred that there is no disadvantage in training the OCSVM with the badly correlated data, since it does not have a great negative impact on the result and allows for a smoother forensic analysis of the packets that are false negatives.

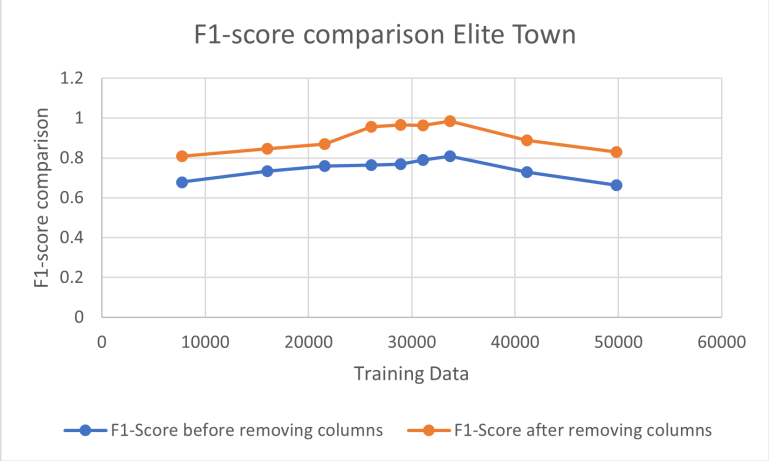
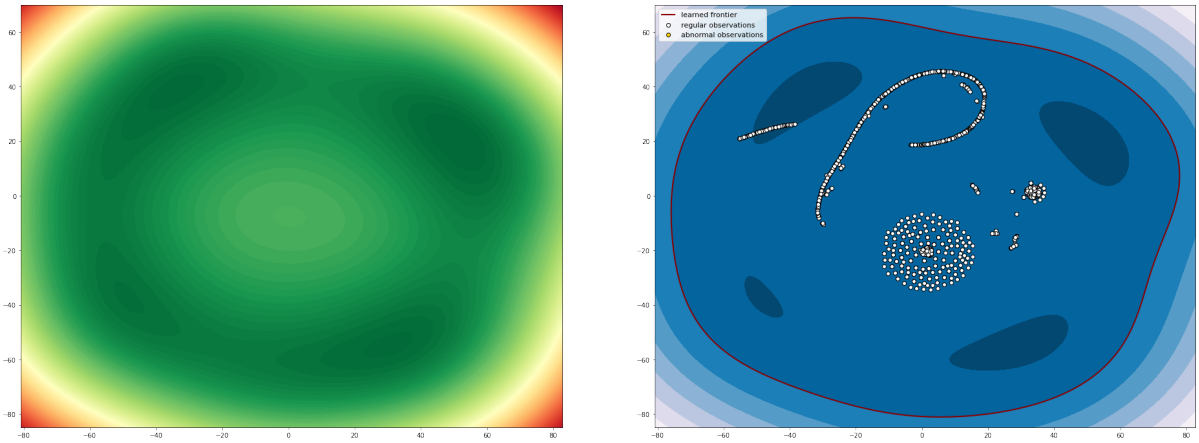


Fig. 6.9.: F1-score comparison for Elite Town.



(a) Decision Boundary Contour after removing features **(b)** Inlier-Outlier distribution after removing features

Fig. 6.10.: OCSVM performance on Elite Town after feature removal.

Tab. 6.4.: F1-score comparison table Elite Town

Training set data	F1-Score before removing features	F1-Score after removing features
7740	0.673	0.808
16016	0.731	0.845
21573	0.759	0.870
26074	0.765	0.955
28917	0.771	0.965
31087	0.799	0.962
33707	0.812	0.984
41147	0.761	0.889
49793	0.674	0.829

6.3 Performance of OCSVM on test data

So far, the results obtained from both Fortiphyd and Elite Town are based on the OCSVM predictions on the validation sets as part of the cross-validation technique. After finding the optimal training data amount for both datasets, the OCSVM is trained with a training dataset in this optimal range. Upon predicting the values of the test data, which contains of both normal as well as malicious data packets, the resulting performance is as shown in Table 6.5. The performance is measured post t-SNE reduction for both setups as it either greatly improves the sensitivity of the OCSVM or it gives similar result pre-reduction.

From the test data, once inliers and outliers are identified by the OCSVM, the packets classified as outliers are analysed individually. If the packet in question originated as part of one of the simulated attacks, it is indeed an outlier, else it is a "false negative". Distinguishing "true positive" and "false positive" packets from inliers is difficult as the proportion of packets to be analysed individually is much higher than outliers. With TN and FN values, the the Negative Predictive Value (NPV) can be found, to determine the performance of OCSVM on test data from Fortiphyd and Elite Town. The closer the value of NPV to 1, the better the performance of the OCSVM.

Although Fortiphyd performed better than Elite Town in the training phase, as it produced a lesser outlier percentage, the OCSVM is able to identify the outliers more accurately for Elite Town. This could be because the parameters of the PLC in Elite Town show greater deviation from its baseline parameters during attack simulations. It could also be due to Elite Town being a more simple setup than Fortiphyd, and thus, it may be easier for the OCSVM to identify deviating features in the PLC parameters of Elite Town.

Tab. 6.5.: Performance of OCSVM on test data.

Setup	Training data	Test data	Inliers	Outliers	TN	FN	NPV
Fortiphyd	39180	11291	8641	2650	2338	312	0.88
Elite Town	33029	10543	8979	1564	1537	27	0.98

7 Evaluation

7.1 Fortiphyd

The Fortiphyd setup is more complex compared to the Elite Town setup and the data collection time needed for Fortiphyd is also longer. The OCSVM performed well despite these challenges, and produced a very high sensitivity rate. Gathering more information on the values stored in each of the 13 Holding registers could give more insight into the correlation between these register values. It is possible that reducing more features before training the model can lead to a better performance. The attacks simulated on Fortiphyd affected only 4 out of the 13 Holding registers and the Coil bit value. If more attacks could be simulated to affect the other register values, based on what those values indicate, more in-depth results could be obtained and the OCSVM performance could be better analysed.

7.2 Elite Town

The Elite Town setup has the PLC performing a simple operation of sensing the actuator value, sending the signal to adjust the value to the setpoint and triggering the HMI to send the correct voltage value to provide electricity for the village. This use case requires less communication over Modbus compared to what was witnessed in the Fortiphyd setup. For its minimal use case, the OCSVM successfully identifies potential outliers.

7.3 Discussion on the proposed solution

The proposed anomaly detection mechanism shows positive results upon deployment in two different ICS environments. While I have made some positive inferences from the results obtained, there are also some challenges posed by this solution, which are being addressed in this section.

It must be noted that outliers detected are not necessarily malicious, they could be benign packets generated due to some other issues in the setup other than a malicious attack. Thus all outliers predicted by the OCSVM are not necessarily malicious, as without the ground truth labels, it is difficult to ascertain the accuracy of this model. This issue can still be managed as from a practical standpoint. When there are 10,000 packets, it is better to investigate 300 packets that are flagged malicious rather than investigate all the packets. In such a scenario, this solution can save computational power and the cost of continuous analysis in ICS setups.

Due to the use of semi-supervised machine learning methods while deploying the OCSVM, it is difficult to infer the results from the training and testing phase as a whole. It is important to distinguish the two results as the results of the training phase indicate how sensitive the OCSVM is towards changes in the PLC parameters while the testing phase analyses the performance of the OCSVM in terms of accurate prediction of outliers. Sensitivity is with respect to the changes in the features present in the dataset obtained from each of the two ICS setups. With this understanding, I can infer that the OCSVM is more sensitive to changes in the features of Fortiphyd compared to Elite Town. However, it gives a more accurate outlier prediction for Elite Town than for Fortiphyd, as seen in the testing phase.

There are other research efforts that have implemented anomaly detection mechanisms focusing on PLCs [48]. However, the approach taken is either theoretical, or tested only on one ICS setup. The research in this thesis is tested on two different ICS setups across the physical and virtual medium. The attacks simulated in this research are realistic and conforming to commonly observed ICS attack techniques.

8 Conclusion

This thesis provides a mechanism for anomaly detection in ICS that is based on changes observed in PLC parameters. The methodology involves understanding the role of the PLC in a given ICS setup and the baseline functioning of the ICS environment. The ICS network is accessed and data pertaining to PLC parameters like input and output is collected by capturing network data from ICS communication protocols. OCSVM is used to model the data obtained, so that it can learn the features of the data obtained in normal run-time conditions. The OCSVM is then able to identify data generated during attacks as they would contain features deviating from what it has modeled during the training period. This mechanism is tested by deploying the OCSVM on two different ICS setups. Attacks involving modifying parameter values of the PLC are simulated by sending malicious Modbus packets and the data is collected from the network for testing. The performance of the OCSVM on both setups is evaluated during the training and testing phase, with conclusions being made on the sensitivity and accuracy of the OCSVM.

8.1 Importance of this mechanism

The deployment of the OCSVM outlier detection mechanism does not introduce any additional tools or devices to the ICS setups. The machine learning algorithm can be set up in any Level 4 or Level 5 workstation within the ICS infrastructure or in monitoring devices in the DMZ (Section A.2.1) to carry out data capturing activities to gain the necessary data. Due to the use of network data, a large amount of training data can be generated or collected from the data historian in ICS infrastructures to train the OCSVM with optimal number of data packets. Procuring reliable data is always a challenge in machine learning. Since this solution is semi-supervised, the burden of procuring labelled data is also managed. The use of SVM as opposed to other classification models has its advantages as it can focus on learning only the normal behaviour without the need to classify them into further categories. Although SVMs require greater computational power, it is the most optimal solution for this particular use case of handling PLC data from ICS networks.

Novelty detection or outlier detection is not uncommon in ICS. However, the main focus of the mechanism proposed in this thesis in anomaly detection for PLC parameters. The role of the PLC is extremely crucial in ICS setups and this solution makes full use of the data collected by the PLC. As the PLC is one of the last programmable devices in the ICS environment, it also acts as the last line of defense against any malicious attack to the ICS infrastructure. On the occasion that a malware attack goes unnoticed and injects malicious commands to the PLC though the network, a solution such as the one proposed here can detect the malicious attack and alert monitoring devices about a potential threat. The implementation of this OCSVM anomaly detection mechanism is independent

of the role the PLC plays in different ICS environments. As seen by the good performance of the OCSVM for Fortiphyd and Elite Town data, this mechanism can be deployed in any ICS environment involving a PLC and a network for communication among ICS components.

8.2 Future Work

The anomaly detection solution proposed in this thesis using OCSVM is a form of novelty detection, where a semi-supervised learning model of OCSVM is used to train and test the performance of the outlier detection. While this model showcases good sensitivity towards feature changes and good accuracy towards outlier detection, it is unable to process the network data in real-time. The time taken to capture the packets and then analyse it can be a critical period in an ICS setup to identify a possible malicious attack. Thus, there is a need in the industry to make use of real-time anomaly detection solutions. A future scope could be integrating OCSVM with a real-time detection mechanism, as the optimal performance and simple execution of such an anomaly detection mechanism cannot be ignored.

Deep learning algorithms like neural networks and timed automata can be used to learn the state of the PLC from its parameters to create a more robust anomaly detection model. The information necessary for such implementations can be availed from the network data, and some potentially better solutions or answers complementing the OCSVM anomaly detection mechanism can be made. While the solution proposed here focuses on the parameters of the PLC, the same approach can also be taken to secure other components of the ICS network such as the HMI.

One of the limitations of the mechanism proposed in this research is that the ICS communication protocol used must involve Modbus. However, the solution can be extended to handle communication from other ICS communication protocols such as the Step 7 protocol used by Siemens proprietary devices and the Profibus protocol. By expanding the solution to include these protocols, the mechanism proposed using OCSVM can be used across a larger range of ICS environments.

Bibliography

[1] *Supplemental Information for the Interagency Report on Strategic U.S. Government Engagement in International Standardization to Achieve U.S. Objectives for Cybersecurity*, 2015 (cit. on p. 3).

[2] A. Garcia, *Firmware Modification Analysis in Programmable Logic Controllers*, 2014 (cit. on pp. 3, 6).

[3] B. C. Ervural and B. Ervural, „Overview of Cyber Security in the Industry 4.0 Era," in *Industry 4.0: Managing The Digital Transformation*, Cham: Springer International Publishing, 2018, pp. 267–284 (cit. on p. 3).

[4] Dragos Inc., „TRISIS Malware," pp. 1–19, 2017 (cit. on pp. 3, 41).

[5] N. Falliere, L. O. Murchu, and E. Chien, „W32. stuxnet dossier," *Symantec Security Response*, vol. 14, no. February, pp. 1–69, 2011 (cit. on pp. 3, 6).

[6] J. Slowik, „Evolution of ICS Attacks and the Prospects for Future Disruptive Events," Ph.D. dissertation, 2019, pp. 1–15 (cit. on pp. 3, 12, 45).

[7] Z. Basnight, J. Butts, J. Lopez, and T. Dube, „Firmware modification attacks on programmable logic controllers," *International Journal of Critical Infrastructure Protection*, vol. 6, pp. 76–84, 2013 (cit. on pp. 3, 6).

[8] R. Langner and B. Schneier, „To Kill a Centrifuge," *The Langner Group*, no. November, pp. 1–37, 2013 (cit. on pp. 3, 40, 41).

[9] Dragos Inc., „CRASHOVERRIDE: Analysis of the Threat to Electric Grid Operations," pp. 1–35, 2017 (cit. on pp. 3, 41, 42).

[10] A. D. Pinto, Y. Dragoni, and A. Carcano, „TRITON: The first ICS cyber attack on safety instrument systems," *Black Hat USA*, pp. 1–26, 2018 (cit. on pp. 3, 41).

[11] K. E. Hemsley and D. R. E. Fisher, „History of Industrial Control System Cyber Incidents," *INL/CON-18-44411-Revision-2*, no. December, pp. 1–37, 2018 (cit. on pp. 4, 12, 40).

[12] E. Habibi and S. Hollis, „ICS cybersecurity: Protecting the industrial endpoints that matter most," *American Fuel and Petrochemical Manufacturers, AFPM - AFPM Annual Meeting 2017*, pp. 568–581, 2017 (cit. on pp. 4, 12).

[13] H. Benkraouda, M. A. Chakkantakath, A. Keliris, and M. Maniatakos, „SNIFU: Secure Network Interception for Firmware Updates in legacy PLCs," *Proceedings of the IEEE VLSI Test Symposium*, vol. 2020-April, 2020 (cit. on pp. 4, 6, 8).

[14] G. P. Sandaruwan, P. S. Ranaweera, and V. A. Oleshchuk, „PLC security and critical infrastructure protection," *2013 IEEE 8th International Conference on Industrial and Information Systems, ICIIS 2013 - Conference Proceedings*, pp. 81–85, 2013 (cit. on pp. 4, 13).

[15]E. J. Byres, D. Huffman, and N. Kube, „On shaky ground - A study of security vulnerabilities in control protocols,“ *5th International Topical Meeting on Nuclear Plant Instrumentation Controls, and Human Machine Interface Technology (NPIC and HMIT 2006)*, vol. 2006, pp. 782–788, 2006 (cit. on p. 4).

[16]A. Ghosh, S. Qin, J. Lee, and G. N. Wang, „PLAT: An Automated Fault and Behavioural Anomaly Detection Tool for PLC Controlled Manufacturing Systems,“ *Computational Intelligence and Neuroscience*, vol. 2016, 2016 (cit. on pp. 4, 10).

[17]P. Robin and M. Baezner, „CSS CYBER DEFENSE PROJECT Hotspot Analysis : Stuxnet,“ no. October, pp. 1–16, 2017 (cit. on pp. 4, 11, 40).

[18]M. Gaiceanu, M. Stanculescu, P. C. Andrei, *et al.*, „Intrusion Detection on ICS and SCADA Networks,“ in *Recent Developments on Industrial Control Systems Resilience*, E. Pricop, J. Fattahi, N. Dutta, and M. Ibrahim, Eds., Cham: Springer International Publishing, 2020, pp. 197–262 (cit. on p. 5).

[19]A. Boukema and R. Lahaye, „Advantages of Anomaly Detection Between the Controlling Unit and the Process Devices of an Industrial Control System,“ 2017 (cit. on pp. 5, 9).

[20]A. Beghi, L. Cecchinato, C. Corazzol, *et al.*, „A one-class SVM based tool for machine learning novelty detection in HVAC chiller systems,“ *IFAC Proceedings Volumes (IFAC-PapersOnline)*, vol. 19, pp. 1953–1958, 2014 (cit. on pp. 5, 10).

[21]Q. Lin, S. Adepu, S. Verwer, and A. Mathur, „TABOR: A Graphical Model-based Approach for Anomaly Detection in Industrial Control Systems,“ *ACM Reference Format*, vol. 12, 2018 (cit. on pp. 6, 9).

[22]A. Costin, J. Zaddach, A. Francillon, *et al.*, *A Large-Scale Analysis of the Security of Embedded Firmwares A Large-Scale Analysis of the Security of Embedded Firmwares*, 2014 (cit. on p. 8).

[23]C. Dphil, M. Saleem, M. Evanglopoulou, *et al.*, *Defending Against Firmware Cyber Attacks on Safety-Critical Systems* (cit. on p. 8).

[24]J. Mulder, M. Schwartz, M. Berg, *et al.*, „WeaselBoard: Zero-Day Exploit Detection for Programmable Logic Controllers,“ 2013 (cit. on p. 9).

[25]B. Joshi, *Application Of Neural Network On PLC-based Automation Systems For Better Fault Tolerance And Error Detection*, 2019 (cit. on p. 9).

[26]C. Cohenour, „Teaching Finite State Machines (FSMs) as part of a Programmable Logic control (PLC) course,“ *ASEE Annual Conference and Exposition, Conference Proceedings*, vol. 2017-June, 2017 (cit. on p. 10).

[27]L. Obregon, „Information Security Reading Room Secure Architecture for Industrial Control Systems,“ 2020 (cit. on pp. 11, 43).

[28]D. A. V. D. Wouw, „Knowledge needed to develop malware to infect and impact Industrial Control Systems,“ 2013 (cit. on p. 11).

[29]Liao Zhang, „An Implementation of SCADA Network Security Testbed,“ Ph.D. dissertation, University of Victoria, 2015, pp. 10–17 (cit. on p. 11).

[30]C. Press, „Rail Transportation,“ *Washington Information Directory 2011–2012*, pp. 675–675, 2015 (cit. on p. 12).

[31]R. Khan, P. Maynard, K. McLaughlin, D. M. Laverty, and S. Sezer, „Threat Analysis of BlackEnergy Malware for Synchrophasor based Real-time Control and Monitoring in Smart Grid,“ pp. 53–63, 2016 (cit. on pp. 12, 42).

[32]B. Productive and S. I. Systems, „Blending Resilience and Protection to Achieve Greatest Security for Business-Viable Industrial Systems,“ 2018 (cit. on p. 12).

[33]R. Ramanathan, „The IEC 61131-3 programming languages features for industrial control systems,“ *World Automation Congress Proceedings*, pp. 598–603, 2014 (cit. on p. 13).

[34]P. Prjevara and D. V. D. Wouw, „Improving Machine Learning based Intrusion and Anomaly Detection on SCADA and DCS using Case Specific Information,“ 2018 (cit. on p. 14).

[35]O. Alexander, M. Belisle, and J. Steele, „MITRE ATT&CK for Industrial Control Systems : Design and Philosophy,“ no. March, 2020 (cit. on p. 14).

[36]C. Wheelus, E. Bou-Harb, and X. Zhu, „Tackling class imbalance in cyber security datasets,“ *Proceedings - 2018 IEEE 19th International Conference on Information Reuse and Integration for Data Science, IRI 2018*, pp. 229–232, 2018 (cit. on p. 14).

[37]V. Chandola, A. BANERJEE, and V. KUMAR, „Survey of Anomaly Detection,“ *ACM Computing Survey (CSUR)*, vol. 41, no. 3, pp. 1–72, 2009 (cit. on p. 15).

[38]Published, P. H. Gaskell, P. K. Jimack, M. Sellier, and H. M. Thompson, „A survey of outlier detection methodologies,“ 2006 (cit. on p. 16).

[39]H. Wang, L. Zhang, Y. Xiao, and W. Xu, „An Approach to Choosing Gaussian Kernel Parameter for One-Class SVMs via Tightness Detecting,“ in *2012 4th International Conference on Intelligent Human-Machine Systems and Cybernetics*, vol. 2, 2012, pp. 318–323 (cit. on p. 16).

[40]„Modicon Modbus Protocol Reference Guide Modicon Modbus Protocol Reference Guide,“ (cit. on p. 17).

[41]D. Formby, M. Rad, and R. Beyah, „Lowering the barriers to industrial control system security with GRFICS,“ *ASE 2018 - 2018 USENIX Workshop on Advances in Security Education, co-located with USENIX Security 2018*, no. Vm, 2018 (cit. on p. 17).

[42]T. R. Alves, M. Buratto, F. M. De Souza, and T. V. Rodrigues, „OpenPLC: An open source alternative to automation,“ *Proceedings of the 4th IEEE Global Humanitarian Technology Conference, GHTC 2014*, pp. 585–589, 2014 (cit. on p. 18).

[43]G. Collins, „Pymodbus Documentation,“ 2017 (cit. on p. 19).

[44]F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, „Scikit-learn: Machine Learning in Python,“ *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011 (cit. on p. 21).

[45]W. Spotz, „Jupyter Notebook - HTML w/BibTeX Citations.,“ (cit. on p. 21).

[46]A. Orebaugh, G. Ramirez, J. Beale, and J. Wright, *Wireshark - Ethereal Network Protocol Analyzer Toolkit*. Syngress Publishing, 2007 (cit. on p. 21).

[47]W. McKinney, „Data Structures for Statistical Computing in Python,“ in *Proceedings of the 9th Python in Science Conference*, S. van der Walt and J. Millman, Eds., 2010, pp. 56–61 (cit. on p. 21).

[48]K. Yau, K. P. Chow, S. M. Yiu, and C. F. Chan, „Detecting anomalous behavior of PLC using semi-supervised machine learning,“ *2017 IEEE Conference on Communications and Network Security, CNS 2017*, vol. 2017-Janua, pp. 580–585, 2017 (cit. on p. 33).

[49]C. Bodungen, B. Singer, A. Shbeeb, S. Hilt, and K. Wilhoit, *Hacking Exposed Industrial Control Systems*, 2017 (cit. on p. 40).

[50]K. Kruglov, „Threats posed by using RATs in ICS,“ 2018 (cit. on p. 41).

[51]J. Rrushi, H. Farhangi, C. Howey, K. Carmichael, and J. Dabell, *A Quantitative Evaluation of the Target Selection of Havex ICS Malware Plugin*, 2015 (cit. on p. 42).

[52]Andrew Ginter, „The Top 20 Cyberattacks on Industrial Control Systems,“ *Waterfall Security Solutions*, no. May, pp. 1–28, 2018 (cit. on pp. 42, 43).

[53]M. Assante and R. Lee, „Information Security Reading Room The Industrial Control System Cyber Kill Chain,“ *Sans Institute*, pp. 1–22, 2015 (cit. on p. 44).

[54]G. Industry, „MIDSTREAM AND DOWNSTREAM OIL & GAS How Claroty Supports the Midstream and Downstream Sectors of the Oil & Gas Industry,“ 2017 (cit. on p. 45).

Appendix

A

A.1 Malware attacks on ICS

Over the past decade, malware attacks have grown both in numbers as well as sophistication. Most of these events are not reported to the public, and the threats and incidents to ICS are not as well-known as enterprise cyber threats and incidents [11].

A.1.1 Stuxnet

The discovery of Stuxnet increased the global understanding of cybersecurity issues. Nation-states realized that vital infrastructures were vulnerable to cyber attacks due to this type of malware and that the consequences could be catastrophic [17]. The code attacked centrifuges in various Iranian nuclear facilities, causing them to function incorrectly and resulted in significant damage to the processing capabilities of the Iranian nuclear program. Most analyses on Stuxnet are done to understand the sophisticated programming behind the malware code, the use of zero-day exploits and the political consequences that resulted from this perpetrated attack.

A cyber-physical attack makes use of attack engineering techniques and manipulates algorithms, communication protocols or device bits to carry out its activities. Such an attack traverses through three layers, each with its own set of vulnerabilities [8]:

- IT layer - used to spread malware;
- Control system layer - used to exploit process control and not disrupt it;
- Physical layer - responsible for the actual damage.

Stuxnet is a textbook example of how the interaction of these layers can be used to use a cyberattack to cause physical damage. The physical layer weakness in the cyberattack against Natanz, the Iranian nuclear facility, was the fragility of the fast-spinning centrifuge rotors, which was exploited by manipulating process pressure and rotor speed. The centrifuges were controlled using ICS infrastructure and monitored using Human Machine Interfaces (HMI). The malware consists of two different attack routines - one to change the speed of the centrifuges and the other to perform a Man-in-the-Middle (MITM) attack within the PLC [49] to hide PLC code and change the logic. The code attacked specific PLC models used in Natanz at the time. They were the Siemens S7-315 and S7-417 PLCs, which were communicating with pressure controller devices via PROFIBUS commands.

A deep understanding of these devices was necessary to carry out the attacks. Antivirus software identifies and blocks malware listed in their signature database. For custom-built malware that is used in ICS attacks, the antivirus software will most likely not identify them as threats in their initial variants. The first Stuxnet variant has been active since 2007, but it only six years later that it was discovered as malware, due to the insight acquired from studying later variants. This first version of the malware was indistinguishable from a legitimate application software package, allowing it to slip past anti-virus detection [8].

A.1.2 TRISIS (TRITON or HatMan)

A Middle Eastern oil and gas petrochemical facility was confirmed to have had its safety system shut down as a result of a malware attack in December 2017 [10]. The malware showcased a trend of communicating directly with a Safety Instrumented System (SIS). SIS is the final line of automated safety protection for industrial facilities, designed to avoid equipment failure and catastrophic events like explosions or fire. Due to the involvement of SIS and its use of undocumented proprietary devices, there has been significant research conducted on TRISIS. The malware attacked SIS controllers, causing them to enter a failed state that resulted in the shut down of processes. The 'Triconex SIS controller' and other related proprietary devices and software were the main targets of this malware. The malware took advantage of internet-connected ICS devices and the increased connectivity of ICS devices with IT, thus exploiting a greater attack surface than its predecessors.

A major drawback of ICS security is the lack of visibility of processes within the OT environment. It would be better to visualise the device processes and their inherent firmware logic as it would give a better and understanding of the ICS environment and how the PLC should behave under normal circumstances [4]

A.1.3 CrashOverride (Industroyer)

CrashOverride is a sophisticated multi-component malware designed to disrupt the operation of ICS in electrical substations. It is the first malware explicitly built to target power grids and the first reported Industroyer assault took place in December 2016 [9]. Unlike Stuxnet and TRISIS, this malware does not operate exclusive to any one vendor or configuration, but rather makes use of awareness of grid operations and network communications to make an impact. The malware was dropped using a backdoor, injected into the ICS system via the payload module and masked its operations by triggering supporting payload modules. The malware mainly caused a denial of service (DoS) by causing a shut down to the electric grid. CrashOverride could be used in any electric grid and the malware could also be re-purposed to be used in other industrial infrastructure if the programmers have the basic knowledge in its operations. Solutions to the issues caused by this malware, especially about the behaviour of Remote Access Tools (RAT) has been deployed as a precautionary measure in most ICS networks today [50].

A.1.4 Havex

Havex is similar to CrashOverride A.1.3, wherein the malware exploits electric grids and behaves as a RAT [51]. The difference lies in the goal of the malware, it collects data from the compromised sites or Operation Process Control (OPC) servers and forwards them to a remote server and the information could then be used as a ransom or for espionage purposes [9].The malware uses network services and phishing e-mails or malicious links to exploit web browsers and enter the ICS environment via internet-connected devices. It is the first malware to make use of trojanized installers to take over the system. The malware selects targets based on a variety of validations, one of which is to ensure that a compromised computer is not a trap designed to help a forensics analyst discover the malware operation. Analysis of the Havex malware shows that the malware itself is not a sophisticated piece of code, but its target systems were mostly legacy devices that needed constant firmware or software updates from vendor browsers. There were very minimal or no checks on whether a file from the trusted vendor site is tampered with or not, and as a result, the Havex malware plugin could download malicious files and compromise the ICS system. The study of this malware shows the gap between legacy devices and modern equipment, the need to conduct checks like signature verification of update files within the device before it tampers with the firmware or software it updates. The ransomware like capabilities of Havex also shows how cyber-physical threats could lead to major ransom events along with safety risks.

A.1.5 BlackEnergy

BlackEnergy, BlackEnergy 2 and BlackEnergy3 are trojans developed between 2007 and 2014 that were found in ICS in Russia and Ukraine. They mainly conducted DDoS attacks, cyber espionage and information destruction activities. The initial variants of this malware were DDoS attacks that unintentionally caused shutdowns in the ICS network, but the later versions were more advanced, falling under the category of Advanced Persistent Threats (APTs) [31]. BlackEnerygy 2 and 3 use a plugin-based architecture similar to Havex that exposes the Dynamic Link Library (DLL) files in windows based software or ICS firmware. BlackEnergy 3 has the capabilities to evade anti-virus software present in ICS by using dynamic signatures and can also target more than one IP address per hostname. These developments were not present in the older versions of the malware. BlackEnergy code can be re-purposed for more Windows-based devices across different industrial sectors. It also showed progress in the types of plugins used and masking capabilities between the versions. The first successful cyberattack on a power grid was caused by BlackEnergy 3 in Ukraine in 2015 [52]. Thus, this shows the consequences of ICS attacks on the geopolitical world, and the need to secure them.

A.2 ICS setup framework

A.2.1 Purdue Model

The Purdue Model for Control Hierarchy logical framework was developed by the International Society of Automation (ISA) Committee for Manufacturing and Control Systems Security. It identifies four zones and six levels of operations [27]:

1. Safety Zone
2. Cell/Area Zone
    - Level 0: Process
    - Level 1: Basic Control
    - Level 2: Area Supervisory Control
3. Manufacturing Zone
    - Level 3: Site Manufacturing Operations and Control
4. Enterprise Zone
    - Level 4: Site Business Planning and Logistics
    - Level 5: Enterprise

Level 0,1,2,3 form the OT environment and Level 4,5 form the IT part of the industry setup. Between Level 4 and 3, that is, between the Enterprise zone and the Manufacturing zone, exists the Demilitarised Zone (DMZ) that manages IT-OT converging operations. The Purdue Model suggests that systems containing ICS data that must be accessed by systems or users on the enterprise network should be stored in a DMZ, and the links between the enterprise network and the DMZ should be inspected by a stateful inspection firewall. ICS applications that need to connect with the business network can do so with the DMZ as well. A stateful inspection firewall must also audit these links. Thus, by using a firewall and isolating activities necessitating IT-OT convergence, malware attacks propagating from IT to OT can be detected if proper security mechanisms are in place in the DMZ.

Antivirus is a recommended component of the DMZ. However, since the model was proposed in 2004, the component now seems less important as it is unable to detect new threats. While antivirus remains an integral part of 21st century ICS environments, its significance has reduced to be a part of a multi-layer security approach and no longer operates as the sole security check for the system. The reason it is placed in the DMZ is so that it can analyse data traffic from IT propagating to OT and it can identify or block known threats from the IT domain, like IT malware targetting ICS systems [52].

The drawback of heavily relying on the DMZ is that recent ICS malware like BlackEnergy 3 and TRISIS have shown masking capabilities that can bypass the security mechanisms in the DMZ. This is possible due to their use of robust encryption and understanding of commands that go from Level 3 devices to Level 4 devices. Since this model was proposed in 2004, most ICS specific cyberattacks have been well aware of the architecture proposed by this model. If the malware passes through the DMZ, then detecting the malware before it executes its goal will rely on the capabilities of NIDS in Level 0-3, which are not very sophisticated. Their anomalous behaviour also depends on the ICS environment and the malware end goal, so a static setup such as this will not account for any behaviour based detection.

A.2.2 Cyber Kill Chain

The cyber kill chain is a collection of measures that follow the progression of a cyberattack from reconnaissance to data exfiltration. The kill chain aids in the understanding and prevention of malware, security breaches, and advanced persistent threats (APTs). The Cyber Kill Chain architecture, developed by Lockheed Martin, is part of the Intelligence Driven Defense paradigm for detecting and preventing cyber intrusions. The model describes the tasks that the adversaries must accomplish to achieve their goal [53]. The ICS Cyber Kill Chain breaks down the steps taken by an adversary into two stages, where stage 1 includes activities in IT and stage 2 covers OT operations. Malware injection, setup and execution take place mainly in stage 1, and the repercussions of the execution results are discerned in stage 2.

The seven steps in Stage 1 of the Cyber Kill Chain are as follows:

1. Reconnaissance
2. Weaponization
3. Delivery
4. Exploitation
5. Installation
6. Command and Control (C2)
7. Actions on Objectives

The five steps in Stage 2 of the Cyber Kill Chain are as follows:

1. Develop
2. Test

3. Deliver

4. Install/Modify

5. Execute ICS Attack

By adopting and understanding a “complete kill-chain” approach to ICS attack methods, sophisticated malware can move from stage 1 to stage 2 of the Cyber Kill Chain and bring down critical ICS infrastructure [6]. Similar to the Purdue Model, the Cyber Kill Chain also has terminology that is divided into stages and used to describe how the malicious code reaches the ICS devices. If the security measures are not stringent enough to detect malicious code in either stage, it becomes invisible within devices that make up these two stages and the attack is carried out.

A.3 The Claroty Platform

The Claroty Platform is a top-down ICS centric IDS approach that improves the visibility of OT operations from Level 3 to Level 0 of the Purdue Model [54]. The proposal for the platform cites the example of the Trisis malware and how this platform is a result of the malware threats that ICS faces. The platform provides the following functionalities that guarantee a more secure ICS environment that can face ICS attacks.

1. Visibility - It includes asset discovery, management and insight on the asset setup within the ICS environment;

2. Threat Detection - It provides continuous security monitoring and uses antivirus based measures such as proprietary threat signatures to detect malicious activity within the ICS environment;

3. Vulnerability Management - Risk assessment and identifying vulnerabilities in the ICS setup is carried out periodically;

4. Triage and Mitigation - The platform carries out Root Cause Analysis to link different concurrent malicious activities in the ICS network to locate the point or cause of attack that led to the undesired activities to occur in the first place;

5. Secure Remote Access (SRA) - The platform has a secure remote access solution for OT which is similar to the more commonly seen SRA setups for IT devices.

The Claroty platform uses many IT security features to device an OT specific top-down solution. However, some drawbacks of this solution are as follows:

- The solution requires the ICS setup to be up to the standards of Industry 4.0, which is difficult to achieve in the relatively static ICS environments;

- The cost of implementing this solution will be heavy and industries may not have the financial backing for a large-scale security overhaul

- The platform focuses on the entire ICS network of an industry, which can be complicated and proprietary in nature. Thus the solutions suggested are customised as per the environment.

- Different security measures are needed for the different ICS levels as per the Purdue Model, and thus the setup is complicated.

The Claroty Platform's shortcomings highlight the need for a simple approach that focuses on one device or layer of devices to solve the large scale security issue within ICS.

Colophon

This thesis was typeset with LaTeX 2_ε. It uses the *Clean Thesis* style developed by Ricardo Langner. The design of the *Clean Thesis* style is inspired by user guide documents from Apple Inc.

Download the *Clean Thesis* style at http://cleanthesis.der-ric.de/.