Kristian Tørseth

Precision Time Protocol under Synchronization Attack

Master's thesis in MIS4900 - Information Security
Supervisor: Professor Stephen Wolthusen
June 2022

Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

Kristian Tørseth

Precision Time Protocol under Synchronization Attack

Master's thesis in MIS4900 - Information Security
Supervisor: Professor Stephen Wolthusen
June 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

NTNU
Kunnskap for en bedre verden

Acknowledgments

I want to thank my supervisor, Professor Stephen Wolthusen, for all the guidance and assistance when working on my master thesis. I would also like to thank my close friends who have been there throughout the two years of my master's studying and a global pandemic. Finally, I would like to thank my family for all the support, and for checking in on me every week.

Thanks,
-KT

Abstract

Precision Time Protocol IEEE 1588 is an industry standard used for time synchronization and is much used in the industrial control system (ICS). It was made a standard in 2002 and has been extended in version 2 (2008) and version 2.1 (2019). PTP ensures synchronization for a system that requires the synchronization accuracy to be less than one nanosecond (< 0.000001 seconds). It was initially not designed with security in mind. With smart manufacturing, newer technology is starting to be used in the industry, which means that the traditional Operation Technology (OT) is being mixed with Information Technology (IT). Making industrial plants a cyber target for Advanced Persistent Threats (APTs). In this thesis, Precision Time Protocol has been examined, and its vulnerabilities have been reviewed. Topologies have been created based on attack scenarios that have been created based on current literature on PTP vulnerabilities. Looking how PTP timestamps can be altered during an attack has been simulated using Omnet++ using current state-of-the-art attacks that can be used. The attacks have shown what will happen with PTP synchronization during an active attack. The types of attacks have varied from noisy attacks to stealthy attacks on the protocol. The attacks conducted in this were all successful in showing that an attacker can add substantial delay to the PTP devices in the network by performing attacks such as Denial Of Service, Packet Delay, and Best Master Clock Algorithm attacks.

Sammendrag

Precision Time Protocol IEEE 1588 (PTP) er en synkroniserings protokoll som brukes mye i industrielle kontroll systemer hvor man ønsker en synkroniserings presisjon under et nanosekund($<$ 0.000001 sekunder). Denne protokollen ble først designet og gjort som standard i 2002, og siden det har det kommet to nye versjoner av den versjon 2 (2008) og versjon 2.1 (2019). PTP ble originalt ikke laget med sikkerhet i tankene og lider av at protokollen er mottakelig mot ulike angrep som utnytter sårbarheter i protokollen. Nå som Operasjonell Teknologi (OT) starter å bruke smartere systemer så blandes dette med Informasjons Teknologi (IT) så kan en Advanced Persistent Threats (APTs) få nye veier inn i et system. Ved å se på svakhetene til PTP så har det blitt sett på eksisterende litteratur tilgjengelig, og kjente svakheter i PTP er blitt tatt opp i denne master oppgaven. Det er laget flere forskjellige topologier basert på state-of-the-art litteratur, og det er laget angreps scenarioer som vil angripe tidsstemplingen til PTP. Disse angrepene er blitt utført ved å bruke software simulerings verktøyet Omnet++. Ulike angrep har blitt gjennomført mot PTP og det er utført angrep som lager mye bråk og som prøver å ikke bli oppdaget. Angrep som ble utført i denne master oppgaven var et pakke forsinkelse angrep, et Denial of Service (DOS) angrep og et angrep på algoritmen som bestemmer hvilken grandmaster klokke som er den beste i nettverket.

Contents

Figures

Tables

Chapter 1

Introduction

1.1 Keywords

Industrial Control Systems, Precision Time Protocol, IEEE 1588, Network Simulation, PTP

1.2 Problem Description

Critical Cyber-Physical Systems (C-CPS) are systems that are becoming a more attractive place for cybercriminals and nation sponsored actors to go after and attack. C-CPS is a set of industrial control systems (ICS) that is used for controlling operations. The focus of these systems is to be available and up and running more than 99% of the time, given that Availability has been the most important focus in these systems and security has often been neglected and forgotten. C-CPS is defined as systems that run critical infrastructure such as the Norwegian energy sector with hydroelectrical power together with oil and gas from the North Sea making up a substantial portion of Norwegian critical infrastructure, but healthcare and telecommunication are also part of the Norwegian critical infrastructure. With these systems starting to take advantage of recent technology such as Internet of Things (IoT) technology and other IT technology, the ICS opens to new threats to the systems that were not previously available. With this more focus on Confidentiality and Integrity (from the CIA triangle) is needed to ensure that malicious threat actors are detected and stopped as there have been devastating attacks on the Ukrainian energy sector [1], as well as very known attacks such as the famous Stuxnet and Triton attacks [2] [3].

Looking at the digital twin to be used for security assurance inside an ICS, it is first essential to look at how data will flow into the digital twin and how it can execute using the data. The digital twin would need to run in real-time, mimicking the existing system. To ensure that the digital twin is in sync with the actual system, the often-used ICS synchronization protocol precision time protocol (PTP) is used for synchronization in ICS. The protocol can be used to synchronize the digital twin with the existing system that runs in real-time. The goal of the master thesis is to understand synchronization issues that can happen using a digital twin, and the focus of the master thesis will be to look at what kind of vulnerabilities and attackers can use to target synchronization in PTP.

1.3 Motivation

Industrial control systems are found in all critical cyber-physical systems. Countries use this, especially in the power sectors, to ensure that the population has access to essential utilities such as power, water, telecommunication, and similar. Attacks on these facilities will have significant consequences on the affected people's daily lives. Knowing the impact, this can have put the C-CPS of a nation in the "crosshairs" of hostile nation-state actors and cybercriminals. The threat to C-CPS increases with the current situation going on as the invasion in Ukraine increases the threat landscape as Russian energy resources are currently sanctioned.

In this master thesis, a look at the synchronization protocol PTP will be looked at and its vulnerabilities. It will also be done using software simulation tools. If an attacker were to leverage vulnerabilities that PTP has, that could have significant implications on ordinary people's everyday lives. This thesis will hopefully be a springboard for other information security students to use software simulation tools to research other Operational Technology (OT) protocols and fields where ICS resides.

1.4 Research Questions

The research questions that have been crafted after a pre-study done in the autumn semester in IMT4205 Forprosjekt and based on problem description and motivation have been conformed into these research questions and will be investigated and answered throughout the master thesis:

1. What are the current threats and vulnerabilities that PTP (IEEE 1588) face?
2. What current attack scenarios can be done with PTP?
3. How will an ICS using PTP be affected by a network attack?
4. How is it possible to detect an attack on PTP?

1.5 Thesis Contribution

In this thesis, we have investigated what known state-of-the-art vulnerabilities that can target the synchronization protocol Precision Time Protocol. Building on previous research papers published on the topic, three example topologies were created, and four attack scenarios were built to examine PTP synchronization timestamps under Denial of Service, Packet Modification, and Best Master Clock Algorithm attacks. The three attacks were simulated using Omnet++, and the results generated gave a clear indication of how PTP timestamps can be altered. Under these attacks, the BMCA attack was the stealthiest attack that can be the most difficult to detect.

1.6 Structure of the Thesis

Introduction includes an introduction to the thesis, keywords, problem description, motivation, research questions, and planned contributions to the thesis.
Background information on ICS devices that can use PTP for synchronization. A background on how PTP works is given with information on different clocks and messaging protocols PTP uses.
Related work includes previous and state-of-the-art research on PTP vulnerabilities and extensions to PTP. It will also look at why synchronization is needed for a digital twin.
Methods will hold essential methods that were used to answer the research question identified for the thesis. The chapter will include information on how literature review, simulation tools, and the development of attack scenarios used in the simulations to generate results.
Results will include the preformance of the designed attack scenarios, and results on their impact will be presented.
Discussion will include a discussion of the gathered results and compare them with state-of-the-art research for validation.Discussion about obstacles along the way will be addressed here for future readers to learn.
Conclusion will enlighten the key contributions generated by this thesis. It will also contain Future work that will include recommendations and tips for future readers to extend the research done.

Chapter 2

Background

2.1 Industrial Control System (ICS)

An industrial control system is often found in an industrial network which is a network that is operating an automated control system. An industrial network might have components and assets which is considered critical infrastructure, or the whole industrial network can be part of the critical infrastructure. Critical infrastructure often will have an industrial control system operating them. Part of the HSPD-7 list of the United States' critical infrastructure includes industries such as nuclear energy, chemical plants, pharmaceutical manufacturing, energy generation (hydro, oil, and gas), energy transmission, telecom, finance, and health care. In this thesis, critical cyber-physical systems that reside within the energy generation and energy transmission will be mostly mentioned. However, the context can be applied to other industries not mentioned (nuclear, chemical, finance), as PTP is a widely used industry standard for high precision cyber-physical systems.

Energy generation where electricity is generated, e.g., water, gas, and oil, is considered a critical infrastructure that is reliant on industrial control systems for easy operations. Securing energy generation for cyber threats is important as disruption to the operation of this kind of system can have major implications for society. Energy transmission is linked to energy generation as the power will be needed to be distributed. An industrial control system ensures that the distribution is done correctly and in an automated way. A smart electrical grid can be a part of a modern energy transmission system as a smart grid builds upon the already old legacy electrical system and implements better monitoring and automation. A smart grid is heavily automated using industrial control systems meaning that a smart grid is at high risk of cyber security attacks. These systems also need to have a high precision synchronization protocol such as PTP to have reliable timestamps for measurement equipment such as a synchrophasor. A synchrophasor can be used in a smart grid to provide measurements of the quality of electricity across a power system. IEEEC37.118[4] defines that a synchrophasor should achieve $>$ 1% error PTP is used as the standard to timestamp [5].

2.1.1 Field Devices

Within an industrial network, many different field devices control local operations. These devices feed data into an industrial control system. A brief introduction will be given to essential devices, followed by an explanation of two crucial control systems.

Remote Terminal Device (RTU)
Remote terminal units have redundant communication capabilities. These devices are often used at substations or at remote locations where the remote terminal unit monitors parameters from a facility or pipe and transmits them back to a central monitoring facility. A remote terminal unit feeds data back to a master terminal unit that is located at a populated facility. Usually, this is done using communication methods such as Ethernet (fiber), 5G, and radio. These communications use a design pattern called publish/subscribe [6] pattern, which is a software design pattern that is used to limit congestion.

Intelligent Electronic Device (IED)
An Intelligent Electronic Device is used in the electrical industry to manage substations. IEDs are used to manage electrical transmission lines and are used to manage different loads on the electrical grid. IED's functionality is like the RTUs, but the IEDs are newer and more specialized RTUs integrated with the system they control. An IED does logical operations on power equipment such as circuit breakers and transformers on a power grid[7].

Programmable Logic Controller (PLC)
A programmable control unit (PLC) is a component that controls real-time processes in a control network. A PLC is a custom configured PC that runs specialized operating systems and is specialized for inputs and output. A PLC uses software that is specialized in automation. A PLC is programmed to do actions when an input, e.g., a sensor value, reaches a specific value. The PLC will output a value to open or close a valve. PLC uses ladder logic [8] [9] which is like legacy logic used by electromechanics relays and checks if specific inputs are either true or false. If all inputs meet the programmed desired logic, the PLC will execute the desired output[7].

Human Machine Interface (HMI)
Human-machine interfaces (HMI) are software-based components that give an operator a graphical overview and control over PLCs, RTUs, and IEDs found in an industrial network. HMIs are used to monitor components' behavior and can hold information about temperature and pressure, position and current state, and positioning. HMI's primary purpose inside an industrial control system is to centralize all information in one place and replaces legacy control rooms where physical dials and switches were used to operate. An HMI can be set up to monitor and control

one single system containing multiple IEDs and PLCs, or it can be a centralized HMI that controls multiple systems. There can be multiple HMIs in an operator's control room, each different controlling system. HMIs will also take input from an operator; an example is when the operator raises the pressure to pump more water into a machine for cooling. Here the HMI takes the input from an operator, and sends this to a PLCs input where the PLC opens a valve[10].

HMI must be accessible and ready for operation all the time, meaning that anyone with access to the location of an HMI can operate it. This is because an HMI is needed in emergencies for the operator to read information to make safe decisions on the HMI. A nonsafe situation can happen if the HMI password is typed wrong too many times, resulting in password lockout, which will mean that the system the HMI is monitoring and controlling is unavailable. This can result in plant shutdowns and substantial financial loss, worst-case injury to human personnel.

Data Historian
A data historian reads data from the HMI and writes this to disks on a data historian. Data historian takes serialized data and stores them in a human-readable way in a read-only environment within an industrial network. This data can then be used to show trends and historical data. It is a place where logs for the industrial control systems are stored. A data historian will store a configured data set that will contain alarm events, specific values that a sensor outputs, or other events of interest for the specific industrial plant. Data from the data historian can be used for historical reviews of events and periodic representations of tendencies in data.

2.1.2 Supervisory Control And Data Acquisition (SCADA)

SCADA gives an operator control on a supervisory level. It is composed of RTUs and PLCs that are distributed in various locations throughout the production pipeline. A SCADA system acquires data at a centralized location, and an operator will use the SCADA system for monitoring and controlling different inputs and outputs throughout the production pipeline. Operators can interact with a SCADA system using its HMI or access the SCADA server directly from a designated host machine. SCADA systems will also generate logs that a data historian stores and historical event data can also be accessed from the SCADA. SCADA will alert an operator if a process shows a high increase in errors. An operator can then from the SCADA system shut off production or initiate mitigating actions from the SCADA system. SCADA systems can automate different processes, e.g., if the temperature in a pipe reaches critical levels, then the SCADA system can be automated to send commands to a PLC to open an exhaust valve to release heat. SCADA will send simple commands, but it is designed for monitoring purposes. For more control, a Distributed Control System is preferred[11].

2.1.3 Distributed Control System (DCS)

Distributed Control System (DCS) combines PLC operations with SCADA monitoring. The DCS functions as an interface for all PLCs in a facility and allows an operator to control 20+ PLCs from a single point. The DCS sends values and commands downstream to PLCs stationed at different locations inside the facility and gives the operator direct control over PLCs. This makes controlling different PLCs easier as the DCS helps different manufactured PLCs talk with each other. A DCS is set to automate PLCs to do actions if specific values reach a set point. This can be to open or shut all valves in a safety emergency or open exhaust valves inside a facility.

Within ICS control, there are many overlaps as PLCs were initially made to have control over a single machine. SCADA was made to have supervisory supervision over data and information coming from all PLCs inside a facility, and the DCS was designed to have complete control of all PLCs in a facility. Nowadays, SCADA has evolved to have monitoring and control of a facility (PLC/SCADA system). All these systems are interconnected with each other, often on an industrial network using Ethernet nowadays for communication.

2.2 IEEE 1588, Precision Time Protocol

For synchronization, we already have a widely used network synchronization protocol called Network time Protocol (NTP). A vast area of consumer devices used NTP. Devices such as laptops and phones use NTP for synchronization, and the protocol assures synchronization that is within 10ms according to NTP standard [12].

Precision Time Protocol (PTP) is made for systems that require synchronization to happen in less than one µs (< 0.000001 seconds). PTP is used in systems that require high time precision. Our power grids use PTP to synchronize the opening and closing of relays to, e.g., deliver power to a new home. Industrial Control Systems (ICS) use PTP for actions that should happen without delay. PTP is an official IEEE standard and has been standardized in IEEE 1588. There are two versions published on IEEE 1588. Version 1 (v.1) was published in 2002, and IEEE 1588 version 2 (v.2) was published in 2008. IEEE 1588v.2 replaces IEEE 1588v.1, and IEEE 1588v.2 is not backwards compatible with IEEE 1588v.1[13][14].

PTP has been designed to work for Local Access Networks (LANs) and is not usable on Wide Area Networks (WANs). The protocol is designed to work on top of several types of already established protocols (e.g., Ethernet, UDP/IPv4, UDP/IPv6). When designed, PTP was made to synchronize different types of clocks of various types of quality (drift and stability See section 2.2.3 on clocks). PTP was designed to be used in ICS and has low resource requirements. This makes it so that PTP can be used on small embedded systems and other microcontrollers with limited resources. This also makes PTP use a small number of bandwidth resources on a network. In IEEE1588v.1, messages were sent every 30 seconds, but a higher messaging rate is needed in systems with higher timing requirements. A message rate of 16 messages per second is used as a starting point in telecom. PTP has three different industrial profiles that are used. The industry can have different requirements on needs and functionality. The most used is the IEEE 1588 v2, which is found in industrial automation. IEEE 802.1AS [15] is a generalized version of PTP often called General Precision Time Protocol (gPTP), and this is used in systems that will have little to no additions or deductions in connected PTP devices. It was created for audio and video synchronization and has been adopted into new electrical vehicles. IEEE 802.1AS is widely used in new types of cars that have been issued with different sensors and cameras (video streams). These types of systems also need to use PTP to make decisions based on sensor information.

Profile	**Default**	**Power**	**802.1AS**
IEEE Standard	IEEE 1588v2	IEEE C37.238	IEEE 802.1AS
Path delay	End to end delay	Peer-to-peer delay	Peer-to-peer delay
Non-PTP device allowed	Yes	No	No
Transport	UDP over IP	L2 Multicast	L2 Multicast
Transparent Clock	End to end	Peer-to-peer	Peer-to-peer

Table 2.1: Table showing the profile standards for PTP.

2.2.1 PTP Terminology

For PTP synchronization, a LAN is needed for it to function. This has a set of computers that are connected to a network. Within the network, there are several types of devices (routers, switches) that connects the network. The computers are, for simplicity, connected to the network with a single link to communicate on are called *terminal devices*. Devices that are positioned within the network and connect terminal devices to other devices are called *network devices*. Terminal devices that want to use PTP for synchronization will have to have a *clock* clock that will execute PTP functionality. These devices are all ordinary clocks. Ordinary clocks are either a *master clock* or a *slave clock*. Network devices that have PTP functionality will either be a *boundary clock* or a *transparent clock*. Terminal devices and network devices that do not have PTP functionality do not have a specific name assigned to them in IEEE 1588.

2.2.2 PTP Clock Error

Clock error is found in all clocks that live within a device that uses PTP. An ideal scenario would be that $clock_1$ and $clock_2$ were in synchronization the whole time meaning that at $t = 1s$ $clock_1 = 1s$ and $clock_2 = 1s$. This is not the case as the oscillators within a clock will be different depending on factors like manufacturing and temperature. If $clock_1$ and $clock_2$ were started at $t = 0s$, then over time $clock_1 \neq clock_2$ because of clock drift. The drift is the clock either running faster or slower. This clock error concept is represented with equation 2.1 where $C_{frequency}$ is our clocks frequency, $C_{perfect}$ is our ideal time perfect time for a clock and $C_{clockerror}$ is the time error of the clock.

$$C_{frequency} = C_{perfect} + C_{clockerror} \tag{2.1}$$

If we let clocks run for over longer time without using protocols such as PTP then clock difference will continue to accumulate with time.

2.2.3 PTP Clocks

In IEEE 1588, devices that perform PTP functionality will have a PTP clock used for timestamping for setting the time. This is the device that will be used for PTP synchronization with a master clock. PTP uses a hierarchical master-slave architecture for the clocks, Ordinary clocks are devices with PTP functionality that are terminal devices. This means that the device has a single network connection. The clocks are either a Grandmaster/ master clock or a slave clock. A slave clock is a recipient (destination) device that will have its time set to the master's time.

Master clocks will be the central reference point (root) as it transmits synchronization information to clocks that live in their network segment. A master clock fetches the time from an external radio clock or a most used GPS receiver. At any current time, there is one and only one master clock that is active in the network. There can be multiple redundant master clocks in a network. However, in an ICS type of network, it is good practice to have redundant master clocks that are inactive until one fails. However, in the event of a connection loss to the active master clock, the redundant clock will become active and the new main reference point for time. A boundary clock is a bridge that has multiple PTP network connections. The boundary clock can be used to bridge time synchronization between network segments. Boundary clocks receive on its ingress port (slave port) and send out on its egress port (master port). A boundary clock also functions as a source of time, but it is not the central reference point as the master clock. The functionality is that the boundary clock is a slave for a master higher in the hierarchy and also a master for slaves lower in the hierarchy. Inserting boundary clocks allows for PTP segmentation. Boundary clocks function as ordinary clocks as they will also ask the grandmaster for the time.

Transparent clocks are connectors within a network and do not work as an ordinary clock but more as a traditional network switch. It forwards PTP requests and calculates their delay.

As displayed in figure 2.1 we see a PTP topology with a parallel redundant master clock. Connections are made to and from the ports, but these are displayed alongside the connection for simplicity. Connections flow from the master port to the slave port as a full duplex. The inactive port is an inactive master port in figure 2.1 because, at any time, there will be only one active master clock, hence the inactive link to the redundant clock.

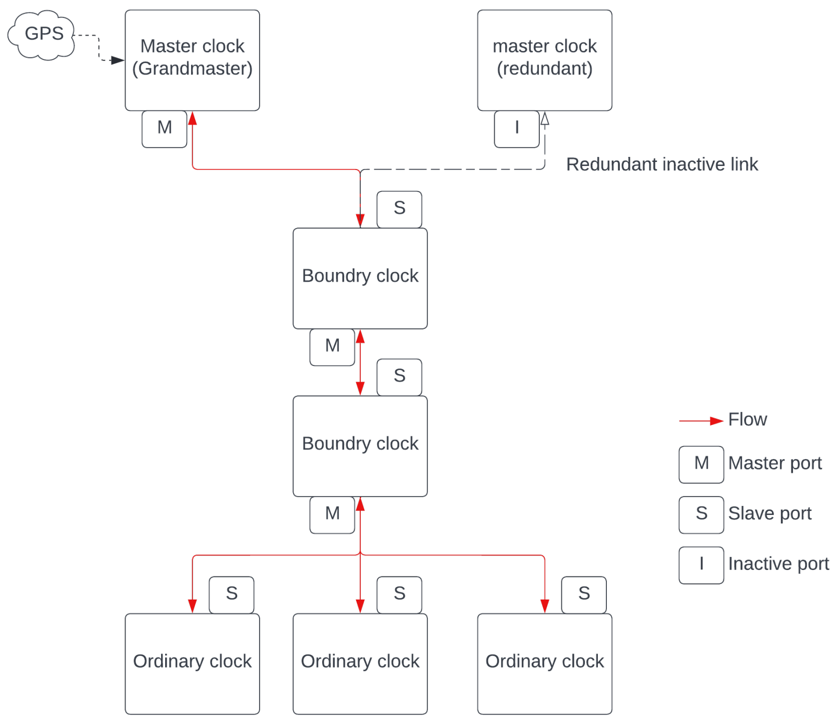


Figure 2.1: Displays a PTP topology with a parallel redundant master clock. Ports are displayed beside the connection for simplicity

2.2.4 PTP Message Synchronization, End-to-End (E2E)

In a PTP synchronization message exchange, ordinary clocks are synchronized with the grandmaster clock. The master will at t_1 send out a *Sync* message, which holds the timestamp of when the grandmaster read the time. This timestamp is stored locally on the grandmaster and is at the same time sent as a *Sync* message to the ordinary clock. If the message exchange is set up to have a two-step method, then a *Follow_up* message will be sent after the *Sync* message as the *Follow_up* message will hold the timestamp from the master clock when t_1 happened, as there can be a delay between reading time and send time. The delay is caused by the grandmaster's internal forwarding time, which results in a delay measured in nanoseconds. The *Follow_up* message contains the precise time when the *Sync* was initially sent. The *Sync* arrives at the ordinary clock at t_2. At time t_2 the ordinary clock stores its local clock time at t_2. The ordinary clock will then await the *Follow_up* message with the original timestamp on when the *Sync* was sent and store this locally on the ordinary clock. The ordinary clock now holds two values. The first is the local time on the ordinary clock when the *Sync* message arrived, and the second is the master clock time with the correct timestamp when the *Sync* message was sent. This is received by the ordinary clock in the *Follow_up*.

The ordinary clock will with these two values. At time t_3, the ordinary clock will send a *Delay_request* to the master clock. The ordinary clock will store the local timestamp on the ordinary clock at t_3. The master clock receives the request at time t_4 and stores the master clock local time at t4. This timestamp is stored by the master clock and sent with a *Delay_response* back to the ordinary clock. Now the ordinary clock has four values $t_1, t_2, t_3, and t_4$ that are needed to synchronize itself with the master clock. The ordinary clock calculates the mean path delay as shown in equation 2.2 and the offset is shown calculated in equation 2.3. The offset shows how much the ordinary clock will have to correct itself to be synchronized with the master clock.

In a real-world system, this is a little bit different as the ordinary clock will need to make sure that the propagation delay on the wires stays stable. This means that the ordinary clock would try to determine the propagation delay between t_1 and t_2, meaning that it would repeat this until satisfied. This is done to try and determine the frequency ratio to the master clock's frequency for better accuracy.

$$MeanPathDelay = \frac{(t_2 - t_1) + (t_4 - t_3)}{2} \tag{2.2}$$

$$Offset = t_2 - t_1 - MeanPathDelay \tag{2.3}$$

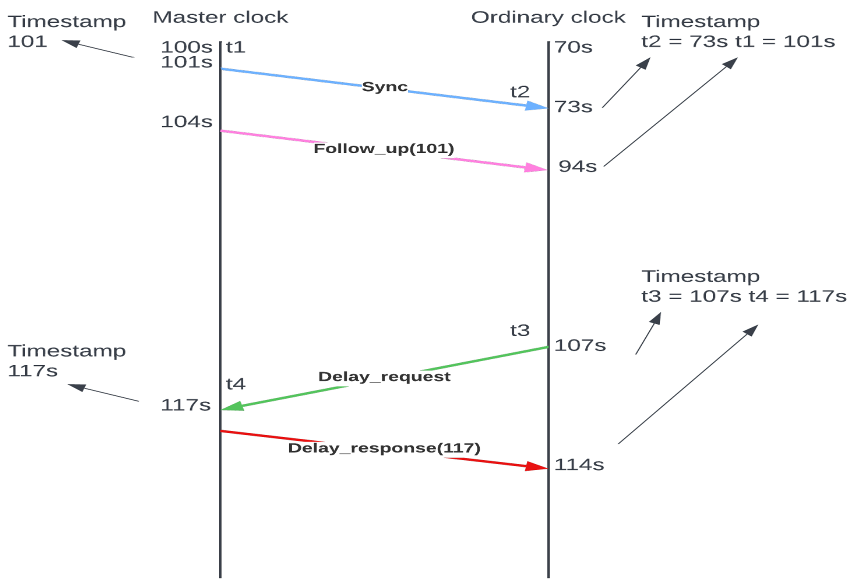


Figure 2.2: Message exchange using PTP end-to-end e2e

In figure 2.3 the message exchange from shown in figure 2.2 goes through a transparent clock (see section 2.2.3). This is similar to the message exchange described between the master and the ordinary clock. Notably, the PTP transparent clock will function as a regular network switch, and messages will be forwarded over the transparent clock (e.g; propagation delay will become longer between *Sync* and *Follow_up* (t_1 and t_2), and the *Delay_request* and Delay_response (t_3 and t_4)) This residence time on the transparent clock is added to the respective message going through the transparent clock. An example is the Sync message going through the transparent clock. This message will contain the time t_1, and the transparent clock will append the residence time which is given by $residecetime = t_{transparentout} - t_{transparnetin}$ to the *Sync* message that the *Follow_up* holds. A PTP message can pass through multiple transparent clocks. The same is done on the *Delay_reponse*. This gives us two new equations for t_2 equation 2.4 and t_4 equation 2.5. For all is added as a PTP message can have multiple hops in a network.

$$t_2 = t_2 + \sum residencetime \forall t_1 \tag{2.4}$$

$$t_4 = t_4 + \sum residencetime \forall t_3 \tag{2.5}$$

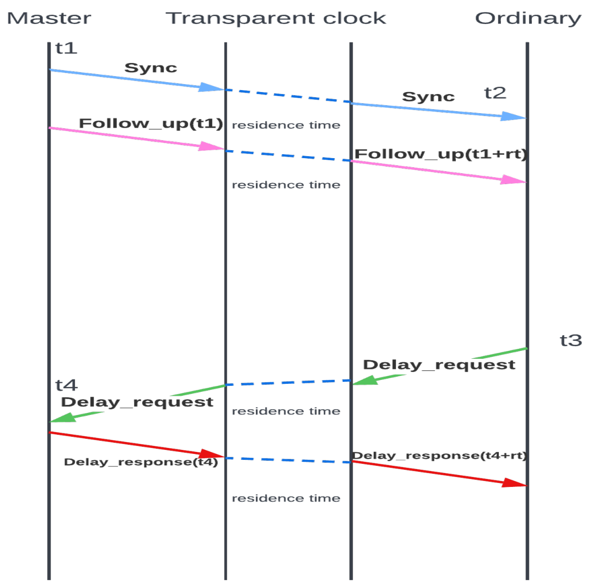


Figure 2.3: Message exchange using end-to-end over a transparent clock.

2.2.5 PTP Message Synchronization, Peer-to-peer (P2P)

In peer-to-peer message exchange, the Sync messaging will work the same way as in the end to end. A Sync message passes through the transparent clock on its way to the ordinary clock as shown in figure 2.4 for simplicity, the *Follow_up* message is not shown, but the same principles apply as in the model. There is the same residence time through the transparent clock as in section 2.2.4 where end to end behavior is discussed. The Sync message will continue towards the ordinary clock through the transparent clock. In peer-to-peer, the t_1 is the same as in end to end t_1, but in peer-to-peer, all network devices (transparent clocks and Boundary clocks) will calculate their own delay. This means that in a PTP network set up to use peer-to-peer messaging, a transparent clock and ordinary clocks (Boundary clocks or slaves) will calculate their own delay. Clocks will calculate their own mean path delay.

$$t_{p2p-meanpathdelay} = \frac{(pt_2 - pt_1) + (pt_4 - pt_3)}{2} \tag{2.6}$$

Furthermore, it will update the Sync messages timestamp for each hoop it makes, including the mean path delay the end clock has as well. This correction to the timestamp is added between each hoop. This makes the time arrival of t_2 to be equal to equation 2.7.

$$t_2 = t1 + (\sum PD + \sum RT) \tag{2.7}$$

Here the sum of all nodes' path delay (PD) is combined with the sum of all nodes' residence time (RT). This is then used to calculate the correct time for t_2.

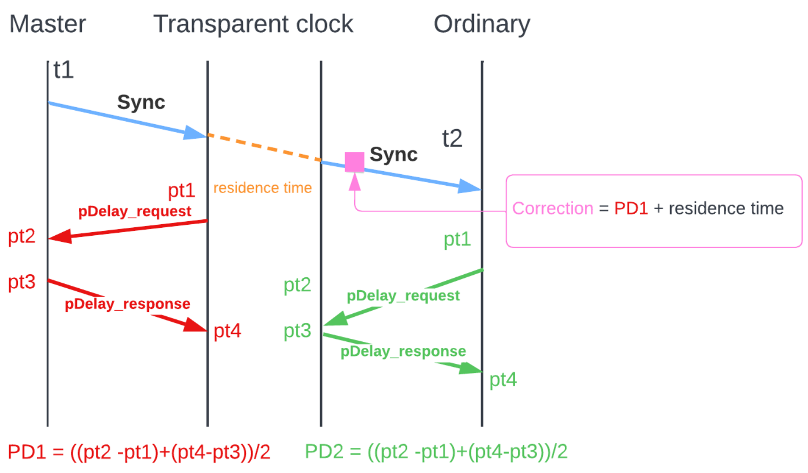


Figure 2.4: p2p message exchange with two hops

2.3 PTP Message Header

The PTP header message is common for all PTP messages, whether these are end-to-end or peer-to-peer messages, and the header holds the message type of whether the PTP message is a Sync, Delay_req, Follow_up, or delay_resp. Message length defines the full length of the PTP message. Domain number holds information on what clocks our PTP message is for, as this says what PTP group the message belongs to. Flag gold parameters for different flags set. The correction field will hold the correction set in nanoseconds and will hold the path delay in a peer-to-peer message exchange. Source port identity identifies what port the message came from. Sequence ID holds a sequence number that can be used to identify PTP messages, this can also be used as mitigation against replay attacks. The Control field says what type of version is used, and the log message interval is set by the type of message[16].

PTP Message Header									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
transportSpecific				messageType				1	0
Reserved				versionPTP				1	1
messageLength								2	2
domainNumber								1	4
Reserved								1	5
Flags								2	6
correctionField								8	8
Reserved								4	16
sourcePortIdentity								10	20
sequenceID								2	30
controlField								1	32
logMessageInterval								1	32

Table 2.2: Table shows PTP header that is sent with all PTP messages.

Chapter 3

Related work

3.1 Synchronization for Digital Twin in ICS

Digital twin technology is a hot topic in smart manufacturing, and in the paper by F. Akbarian, E. Fitzgerald, and M. Kihil, digital twin synchronization architecture issues. The authors address an issue that can happen in an actual facility system. The digital twin needs to follow the behavior of the actual system in real-time if abnormal events happen in the existing system. The authors proposed three different network architectures to tap (replicate the data and send it to the digital twin network) input and output data that will/has flown through the controller and system in the real system. This architecture requires no synchronization to be in place. This architecture is shown later to be unstable and not an architecture that would work. Architecture two is tapping data flow on input before the controller and between the controller and system. This is then fed into a Kalman Filter which is an observer that can predict estimated signals by removing noise from its inputs. The third architecture proposed uses an architecture like the first architecture. They design their controller for the digital twin a proportion, integral, derivative (PID). The Kalman filter architecture and the PID architecture give good results in synchronizing the digital twin. Although they are precise, they are not as precise as if PTP were to be integrated to match timestamps for the architecture. Results from the paper discuss essential aspects of how the synchronization can be done and gives a good point on what approaches can be built upon using high precision synchronization protocol such as PTP. The results highlight the importance of good synchronization is to have devices run in real-time.

3.2 PTP Vulnerabilities

There have been long known flaws within the PTP synchronization protocol that an Advanced Persistent Threat (APT) can use to manipulate time on a system using PTP for synchronization. In their paper, Waleed Alghamid and Michael Schukat present known weaknesses in PTP IEEE1588 that an APT can use to manipulate

PTP synchronization. In the paper, they categorize the types of attacks into two categories. A Man-in-the-middle type of attack is where a third-party malicious actor asserts themselves in a position where all network traffic flows through a compromised node where the attacks have an insight of all traffic flowing through the node. The second category for types of attacks is a packet injector attack. This type of traffic is where an attacker inserts themselves inside a trusted PTP network with a node that can start to generate types of network traffic. A prerequisite for the types of attacks was that they were "inside" types of attacks. An inside type of attack will mean that the attacker has already gained a foothold inside the network, meaning that they would have found other methods to breach the network.

- Packet Content Manipulation Attack
- Packet Removal Attack
- Packet Delay Attack
- Source Degradation Attack
- Master Spoofing Attack
- Slave Spoofing Attack
- Replay Attack
- BMCA Attack
- DOS Attack

The types of attacks that are thoroughly explained in W. Alghamid and M. Schukat paper [17] and in M. Han and P. Crossley in their paper [18], where they conduct similar types of attack on IEEE 1588 PTP. It is essential for the thesis that the reader understands the different kinds of vulnerabilities that PTP is acceptable for exploitation, and there are nine known vulnerabilities.

While it is possible to do external types of attacks, it is the internal once that will have the most impact on a PTP network. PTP is vulnerable to Man-in-the-middle types of attacks where a malicious third party can start to manipulate PTP packets. The attacker needs to have access to the network security keys. These types of attacks will intercept Sync and Follow_up messages and can change the version number from version 2 (IEEE 1588 v2 2008) to version 1 (IEEE v1 2002). This is because PTP version 2 is not backward compatible. This type of attack is called a packet content manipulation attack.

PTP packet modifications attacks are made by a malicious node where the node intercepts PTP traffic and forwards an altered PTP message. This is a man-in-the-middle type of attack

PTP packet removal attacks are made by a malicious node where the node intercepts PTP traffic and sinks/removes PTP packets. This can be done externally and internally on the network. This type of attack will make the downstream terminal clocks go into a free-running mode, where the clocks will continue to use their local timestamps and not be in sync with the master clock.

Time source degradation attacks are an attack that targets the master clocks or a boundary clock. An attacker can initiate an attack on the master clock by

Attack	**Cryptographic**	**Multiple path**	**Redundant GM**	**NTP**
Packet content manipulation	y*	y	n	n
Packet removal	n	y*	n	n
Packet delay	n	y*	n	n*
Time source	n	n	n	n*
Master spoofing	n	y*	n	y*
Slave spoofing	n	n*	n	y*
Replay	y	n	n	n
BMCA	n*	n	n	y*
DOS	n	n	n	n

Table 3.1: Table shows mitigation techniques that can be used against different PTP attacks.

jamming GPS signals making the clock synchronization from the master out of sync with its main source (GPS).

An attacker can insert themselves into the PTP network and start to imitate master clock behavior. This is a type of injection attack where the attack's master clock will inject illegitimate PTP synchronization messages. The attack is spoofing itself as the master clock.

Slave spoofing attacks are possible to do when the attacker has gained control over a slave node. Here the attacker will alter the timestamps that the slave node sends to the master clock. This will make it so that timestamps will be altered from the real delay time. An attacker can also manipulate sequence numbers going back to the master. This can lead to the messages being dropped as sequence numbers do not match what was expected to come.

A BMCA attack is when an attacker inserts themselves as the master clock. They modify parameters to trick the algorithm into choosing the attacker's clock. This leads to the attacker having complete control of timestamps that are being distributed in the PTP network.

PTP is vulnerable to replay attacks where the attacker holds PTP synchronization messages for some time before continuing to transmit them at a later stage. This will make the timestamps in PTP messages inaccurate. DOS and DDOS types of attacks are possible to do on PTP nodes, and the attacks that PTP is vulnerable to are, e.g., MAC flooding and IP spoofing. An attacker who has access to the cryptographic key used in the network may also send MACsec and IPsec packets that can cause CPU loads on receivers to spike and make them unavailable.

Some of these attacks are advanced internal attacks, which means an attacker has a foothold within the network and can manipulate more than just PTP. Mitigation techniques against these types of attacks are sparse. The paper analyses four different mitigation techniques, and in table 3.1 an overview of what mitigation that is effective is shown.

Cryptographic mitigation in packet content manipulation is effective on simple attacks where the attacker does not have the cryptographic key but fails to mitigate against advanced types of attacks.

Packet removal attacks can be mitigated unless the attacker has complete control over all nodes where traffic flows through and can view the contents of network packets. Packet delay manipulation can be mitigated by using multiple paths to each PTP node unless the attacks delay all types of traffic. NTP redundancy against packet delay attacks can be effective except when the attacks delay all network traffic. Source Degregation attacks (GPS jamming) can be mitigated using NTP redundancy. If not, the attacker also manipulates NTP timestamps.

Master spoofing attacks can be mitigated using NTP unless the attacker manipulates this protocol, same goes for mitigation of Slave spoofing attacks.

A replay attack can be mitigated by cryptographic security as the sequence numbers would identify that a replay attack is ongoing.

BMCA attacks are not detected able if the attacker has the cryptographic key. This is an example of a stealth attack where the attacker will have complete control of all PTP timestamps, and NTP can be used as a redundant way of checking the correctness of the timestamps. This is not effective if the attacker has compromised NTP and PTP. Although a good configured PTP network has and uses redundant clocks such as an atomic clock or NTP, their attacks on these and their vulnerabilities are out of the scope for this master thesis.

3.3 PTP Security

PTP version 2.1 is hardening its security capabilities, and the paper by K. Rösel, M. Helm, J. Zirngibl, and H. Stubbe looks at PTP version 2.1 AUTHENTICATION TLV [19]. This is a cryptographic integrity check value (ICV) which is a hash that is appended to PTP messages. This is to ensure the integrity of the message. This ICV is added to ensure that no modifications have been made to the PTP message upon arrival. The end node can calculate the IVC and check the hash sum, and if they do not match, then there has been some change to the message during sending. This mechanic functions that all nodes need to know a secret key. It has been proposed that this is a symmetrical key from paper x. The key is used to generate the ICV, and without the correct key, the ICV will be wrong and will give an indication of an ongoing Man-in-the-middle type of attack and can raise alerts. The AUTHENTICATION TLV noticeably contains a security parameter pointer that tells the node which node to contact. Each unique node will have its own set of parameters, as a PTP node can exchange messages with multiple other PTP nodes. It will contain a security parameter index where different flags are set to indicate what the TLV should contain and the identification of what key has been used. Disclosed key field holds previously used keys. These keys are sent in clear text as the keys are no longer in use. The TLV will also hold a sequence number that can be used to mitigate replay attacks.

When receiving a PTP message on a node, the node can do some checks to clarify if the message is authentic or not. The first step would be to look for the AUTHENTICATION TLV. If it is not present, then drop the message and/or raise an alarm. If it is present, then calculate the ICV. The second step is if the ICV does not match, then drop and raise an alarm. In the third step, if the sequence number is not different, then drop it as this indicates a possible replay attack. If it is not seen before, then process the PTP message[20].

In a letter written by Moussa, Robillard, Zugenmaier, Kassouf, Debbabi, and Assi [21] for IEEE Communication in 2019, A PTP security model is suggested to be added to check for compromised PTP communication. The authors describe current weaknesses in using a shared symmetric key authentication when adding Integrity Check Values (ICV) to PTP messages and raise concerns that if an attacker were to gain the symmetric key, all PTP messages within the network could be compromised. The authors suggest having an SNMPv3 host machine also be a monitoring machine for compromised PTP Sync messages. The authors suggest PTP hardening with messages sent to the monitoring machine to be timestamped and checked at the monitoring machine (SNMP manager). The authors made a simple algorithm to check if timestamps match, and if timestamps do not match alert events will trigger as there is a manipulated Sync message that one or more PTP slave devices have received. The authors suggest using SNMPv3 as this has implemented security and is already speaking with all devices on the network. The network will take little to no noticeable hit on traffic overhead, and the authors suggest adding check PTP messages every 5 seconds. Research done in this letter helps harden PTP vulnerabilities and could be effective if an attacker does not have complete control over a network. The method suggested will be able to detect and notify if false PTP messages were to be received at an end node (slave).

3.4 Software Simulation for Validation

There have been conducted few simulations using software simulation tools on PTP vulnerabilities. A good paper where PTP vulnerabilities were validated using the software simulation tool Omnet++ was a paper by B. Moussa, M. Kassouf, R. Hadjidj, and C. Assi [22], where they have spent time customizing the code to be able to detect on PTP slaves when a possible attack is taking place. The authors of the paper conducted three different attack scenarios. They used a compromised grandmaster clock, a delay attack on a connection line, and a transparent clock attack where they inserted a malicious transparent clock that altered the timestamps. Their model introduces their Network Time Reference (NTP) that intercepts the delayed reply messages sent back from the slave to the grandmaster. Their model is then validated using Omnet++, where they created the attack scenario and inserted their Network Time Reference model by modifying and using Wolfgang Wallner [23] PTP library. Their results show promising results for using Omnet++ and software simulations as a method for validating PTP network behavior and test their NTP model.

This is one of few PTP simulations that takes advantage of software simulation tools and is of great relevance in helping determine what software solution is suitable and on libraries that can be used for the master thesis.

3.5 White Rabbit - An Extention to PTP

Another time synchronization that is starting to mature is the PTP extension called White Rabbit [24]. White Rabbit has been developed as an open-source project and is developed by the European Organization for Nuclear Research (CERN), Helmholtz Center for Heavy Ion Research (GSI), and Fermilab with support from Seven Solutions who is making hardware for White Rabbit. White Rabbit technology ensures sub-nanosecond time synchronization. White Rabbit ensures precision better than 50 picoseconds (10^{-12}). White Rabbit is built upon fiber optical networks and is based on Ethernet and PTP technology. It was developed to improve timing without requiring an overhaul of existing fiber infrastructure. White Rabbit works the same way as PTP IEEE 1588 as it has a master and slave model. For configurations, White Rabbit used an analog 1 Pulse Per Second and 10MHz clock signals to obtain the time and uses NTP for information about the Time of Day information. White Rabbit uses Layer 1 syntonization to distribute clock references and, on the receiving end, create a copy of the shared clock. White Rabbit uses PTP packets with specific signaling messages that hold calibration parameters. This is used with hardware to create timestamps. Phase measurements are used to take measurements down to the picosecond, and White Rabbit uses syntonization for precise measurements and is used to set the offset of the clocks. White Rabbit is still in the maturing phase but is already being used in financial institutions such as Deutsche Börse uses White Rabbit to synchronize their customer's trading clocks [25]. Challenges that ICS still faces are asymmetric paths and changes in temperature. But with the technology maturing White Rabbit is a good candidate for sub PTP precision [26][19] [27].

3.6 Concluding Related Work

The majority of literature presented in this related work section has been on PTP vulnerability and PTP security. PTP vulnerability has first been presented thoroughly, and the papers used here describe the know state-of-the-art attacks on PTP. Most of the papers use a hardware simulation approach for generating results and analyzing from the results what implications different PTP vulnerabilities will have on PTP timestamping. The approach presented in 3.4, who has successfully simulated PTP together with their security solution to address three attack vulnerabilities. Using a software simulation approach as this paper will be the best approach for this paper as no expensive equipment is needed to conduct experiments.

Chapter 4

Method

Several different tools were used to answer the identified research questions during the planning, executing, and writing of this master thesis. The project has been executed using a modified scrum approach. The ability to gain knowledge on the issue with little to no pre-knowledge has been divided into three phases that have been worked on iteratively using a scrum approach. The Discovery phase was the initial phase of working with thesis-specific issues. In this phase, much reading was conducted, and an extensive literature review was done to narrow down the project's scope. This phase has also been the phase where tools have been tried out. After the discovery phase is the modeling phase, where knowledge gained from literature is modeled and experiments are conducted, this is the phase of the master thesis process where network models are tried, and simulations are conducted and verified. Then is the analysis phase of the project, where writeups on experiments and verification of data were done. These three phases have also been applied to all small tasks and have been worked on iteratively. Some of the phases have been conducted multiple times as it was natural to test a concept first and then add pieces and build upon proof of concept.

For tracking progress, tasks were identified and kept track of using a task management tool online. Scrum sprints were also used to keep small deadlines for progress tracking and having frequent meetings with the supervisor to help keep the thesis on track. These scrum tools have ensured a good overview of the challenges and have been helpful in identifying issues that would need more time and help.

4.1 Literature Review

To answer some of the research questions presented in 1.4 a thorough literature review of known literature on PTP had to be consumed. A literature review is a standard method often used to gain knowledge on research that supports the solving of research questions identified. Doing a good research question can also identify new research questions. A literature review helped put together the pieces of the puzzle [28], which is the thesis, and while the literature review was initially planned to be done in the starting phase of the thesis, it has continuously been used as an effective method to gather new knowledge when problems with lack of knowledge have been identified.

For conducting a successful literature review, search engine tools have been used. To gain a foothold within the research field, a wide google search was done to “poke” around and look for ideal keywords to look for. IEEE search engines have also been effective in finding relevant literature as the PTP protocol is IEEE 1588 standard. After reading for a while, a “shopping list” of keywords starts to form, and the literature review becomes more selective based upon keywords, abstracts, and conclusions. Creating a “shopping list” of keywords to look for was one good piece of advice that the supervisor came with during the writing of this thesis and is highly advised to use for future studies and readers of this thesis. NTNU also has access to Scopus [29] which is a search engine for scientific papers that is very powerful that was used extensively in the later phases of the projects. Combining the shopping list method with common methods of reading research papers meant that the literature review effectiveness grew exponentially as the time to identify papers was more than cut in half at the end versus the time spent in the beginning. When a paper was identified, it was thoroughly read through, and notes were an effective way of summarizing papers to later be used to make a simulation scenario and to help support building the simulation. Tools used for the literature review were:

- Google
- IEEE search engine
- Scopus
- Common literature review techniques
- "Shopping list" keyword method

4.2 Simulation Tools

The main method for generating results for the thesis has been the usage of a network simulation tool named Omnet++[30]. Omnet++ is a modular-based C++ simulation framework that is made for network simulations. Omnet++ is supported by INET [31] library, which gives pre-built modules for a vast majority of all known network communications protocols. Omnet++, together with INET, are the main network simulation tools used for the thesis for simulating PTP protocol attacks and vulnerabilities. Omnet++ is used to model different PTP network topologies created from the scenario drafting. Omnet++ has its main community on google groups, and this is used for troubleshooting and getting help.

INET is an open-source model library that is commonly used when making Omnet++ simulations. INET framework contains models and examples of internet stacks such as TCP, UDP, IPv4, IPv6, OSPF, BGP, and many more). The framework support wired as well as wireless link layer protocols such as Ethernet and IEEE 802.11. INET is used considerably through the modeling and for validation and has pre-built classes (modules) that assign nodes with, e.g., IP and Mac addresses, and pre-built UDP classes that are used by all libraries and for message sending. It contains pre-build classes for network wiring and gives the user full customization on network cables that transmits network packages. The same principle applies to wireless communication. INET has pre-built classes for almost all known network functionality and is used in most cases when creating an Omnet++ simulation, as building from scratch can take time. INET is the core component in all attack scenarios that are created using Omnet++Using the available resources gathered, three different attack scenarios were created. These types of attacks were then designed and tested. For creating the attacks, all three scenarios used a version of Omnet++ and INET. Other dependencies will be discussed in their respective section.

4.3 Development of PTP Attack Scenarios

Targeted scenarios for a PTP attack were drafted based on information gathered for the literature review. Attack scenarios were created based on some prerequisites:

- An attacker will already be inside the network.
- An attacker has mapped out the network (reconnaissance).
- An attacker can do any type of PTP attack.
- Little to no mitigation or hardening on the PTP network is present.

These requirements were initially thought of based on literature review and fitted well to be able to carry out attacks that would generate results. The attacker has already done reconnaissance of the targeted network, and this would not be needed to be done in the experiment. For the sake of simplicity, the attacker has full insight into the network. The attack can also do any type of attack, and this means that the attacker has gained access to cryptographic keys used for PTP network messaging. The network targeted has little to no hardening against PTP attacks.

The attack scenarios were made to try and simulate results for both End to end and Peer to peer types of PTP message exchanges. Attack scenarios were worked on iteratively, and the figure 4.1 represents the workflow for creating scenarios built on knowledge actively collected in pre-studies and literature review. After this, modeling and testing inside of simulation environment are done. Minimal Valuable Product (MVPs) are made to validate that the model is feasible. Here validation of results is done in iteration number one of a model. The cycle then continues in iteration number two, with the same steps, and builds upon iteration number one. This is done *n* iterations until results are within acceptable criteria. This is an agile problem-solving approach.

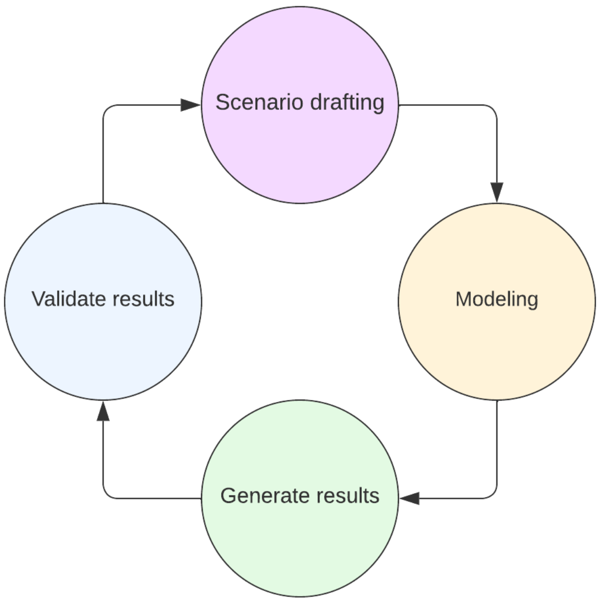


Figure 4.1: Workflow for creating feasible attack scenario

4.3.1 Deciding on Attack Strategy

In identifying types of attacks that can be done inside the simulation tool extensive testing of supportive PTP libraries was tried out. Omnet++ does not have any PTP libraries included, and the INET library did not look to support PTP protocol at the current testing stages of the thesis. From the Omnet++ library list, three community created open-source libraries were found:

- IEEE 802.1AS gPTP for Clock Synchronization
  - Authors Henning Puttnies, Peter Danielis, Enkhtuvshin Janchivnyambuu, Dirk Timmermann.
  - Created for Omnet++ version 5.2.
  - Created for INET version 3.6.3.
  - Available at Gitlabs [32].
- Precision Time Protocol for INET
  - Author Martin Levesque
  - Created for Omnet++ version 4.6 or greater.
  - Created for INET version 2.6.
  - Available at Github [33].
- libPTP - Library to simulate the Precision Time Protocol (PTP, IEEE 1588) in OMNeT++

- Author Wolfgang Wallner
- Created for Omnet++ version 4.6.
- Created for INET version 2.6.
- Available at Github [34], and official website [35].

What is common for all libraries is that the PTP messaging is done using User Datagram Protocol (UDP) protocol for message sending. UDP is used for host-to-host communication and is a connection-less type of communication. This means that a UDP sender will send packets regardless of if there is a receiver or not. PTP uses UDP as the transport protocol for IP communication and can utilize unicast and multicast modes. This is important to know for continuing building attack scenarios. The results that are outputted by successfully building the attack scenarios within Omnet++ will help validate and answer what types of attacks are possible and if the attacks can be detected. These libraries are used to build three different types of attacks.

4.3.2 Planning the Attack Models

Using the available libraries that were examined, three different attack scenarios were created. These types of attacks were then designed and tested. For creating the attacks, all three scenarios used a version of Omnet++ and INET. Other dependencies will be discussed in their respective section.

1. **Packet Delay Attack p2p**
   Using IEEE 802.1AS gPTP for Clock Synchronization [36] a packet delay attack was identified to suit the package well. This package is a general PTP IEEE 802.1AS and is found in video and audio synchronization and used in newer types of vehicles that use sensors and cameras, e.g., Tesla and newer types of electric motor vehicles. This package simulates only Peer to peer type messages, and the experiment will be a packet delay attack on Peer-to-peer message exchange in PTP. The package required Omnet++ version 5.2 and INET version 3.6.3. Finding a suitable topology to go with the type of attack was built upon the existing example topology that came along with the library. The library was studied, and examples were validated to give expected results. Then the creation of scenario 1. Packet delay attack was created based on example topology.
   The PTP package came with an example simulation that has been modified to build the packet delay attack. Modifications within the network transmission in Omnet++ made it possible to change the transmission rate within the network. This is then used to create the attack scenario: PTP packet delay attack, which is identified as a good test experiment to showcase vulnerabilities in PTP.

2. **Packet Delay Attack e2e**
   For making a packet delay attack for end-to-end message exchange in PTP the Omnet++ library and Omnet++ library, Precision Time Protocol for

INET was found. Here modifications can be done to add delay to packets. The library requires Omnet++ version 4.6 with INET 2.6. Modeling the network was done based on an example topology consisting of 10 slave nodes with one master node. The topology is the same as will be used in the DOS attack.

3. **PTP DOS Attack**
   When examining the Omnet++ library Precision Time Protocol for INET, the thought of making a DOS attack became clear as one of the examples had split downstream traffic and upstream traffic. Given this, two-host in-between grandmaster PTP clock and slave PTP clocks could be set up. Precision Time Protocol for INET uses Omnet++ version 4.6 with INET version 2.6, meaning that the simulation environment is different from the Packet delay attack. This library will simulate a DOS attack that is done on an End-to-end PTP network. The name of the attack was given in scenario 2. DOS attack and a suitable topology were used to simulate results. Within Omnet++ and the library, configurations were changed to be able to send a high capacity of network packets. By changing the number of packets sent, the results can be analyzed to see the impact a DOS attack has on a PTP network. In the simulation, UDP packets with a higher priority are sent between two hosts in the network. This congests the lines that the PTP message exchange is done over. In a real work system, the UDP messages would be GOOSE type of messages. GOOSE IEC 61850 (Generic Object-Oriented Substation Event) works as a publisher that will send messages when a new event happens. This is then sent to all machines that are subscribed to the publisher. GOOSE messages are used to send urgent messages between IEDs and are messages that are sent over the link layer to deliver messages as quickly as possible (< 4ms). Goose messages are given priority and will be sent ahead of other message protocols such as PTP or UDP.

4. **BMCA Attack**
   BMCA attack was found to be a suitable attack scenario where library libPTP - Library to simulate the Precision Time Protocol (PTP, IEEE 1588) in OMNeT++ was a good choice. This library comes with four different clock types where the gold star clock is a perfect clock, the silver star clock is of good quality but not perfect as the gold star, bronze star clocks are slightly off but not terrible. Simulating the result between the gold star clock and the bronze star clock could give a good understanding of how a BMCA attack may influence the precision and synchronization within a PTP network. This library is of the more outdated versions available for Omnet++ and has some more dependencies than the other two packets. The library needs to use Ubuntu 16.04 with Omnet++ version 4.6 and INET 2.6, and the package also has an additional library that adds noise to the oscillators inside the clocks. The addition of noise on clocks was decided not to be used for

the simulations.

4.4 Attack Scenarios

Attack scenarios are described in this section. All attacks use the UDP protocol for message sending. All attacks have been modeled and simulated using Omnet++ together with INET. Attack scenarios use their respective PTP libraries that are described in 4.3.2. The scenario results are later presented in the results chapter.

4.4.1 Scenario 1.1 Packet Delay Attack p2p

Scenario: 1.1 Packet delay attack using a peer-to-peer type of message exchange in PTP. In the scenario that the attacker has gained access to the PTP network and has done the necessary reconnaissance on the network. The scenario does not speculate on how the attacker has gained access or whether this is an insider helping (insider threat is discussed in discussion).

The attacker will have control over one or more bridge nodes within the PTP network. In this scenario, the attacker has put a delay on all connections to and from the switches. PTP packets are sent using IEEE 802.1AS gPTP for Clock Synchronization, and this utilizes Peer-to-peer message exchanging. Packages and imports are specified in table 4.1. The topology is created using the example "networkDaisyChain.ned" with a few modifications. Topology is displayed in figure 4.2. The .ned file has been modified to apply the desired delay. For this attack, the delay was simulated to be 10000ms. Adding this value is a arbitrary value and was chosen to get good values to read from the graph. The simulation time is set to be 100s where the attack is happening from 0s to 100s. The delay iss modified in the .ned file shown in figure 4.3. Here modifications to the ethernet connections connecting each device have been modified to simulate a packet delay attack. INET Network configurator has its documentation hosted on INETS official sites [37].

Package	**Description**
ieee8021as.GPtpBridge;	gPTP bridge node set up for p2p message exchange
ieee8021as.GPtpMaster;	gPTP master node with configurations set up by author (p2p)
ieee8021as.GPtpSlave	gPTP slave node class (p2p)
inet.networklayer.configurator. ipv4.IPv4NetworkConfigurator;	IPv4 network configurator by INET. Internal data structure for storing IP addresses, support automatic and manual address assignments

Table 4.1: Table shows Package description used in attack.

4.4.2 Scenario 1.2 Packet Delay Attack e2e

Scenario 1.2 Packet delay attack using an end-to-end type of message exchange in PTP. In the scenario that the attacker has gained access to the PTP network

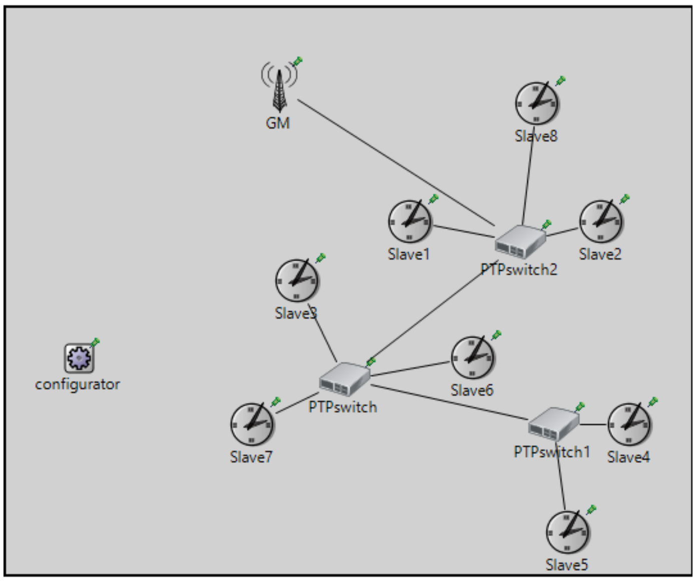
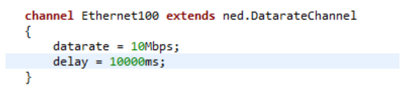


Figure 4.2: Attack scenario 1.1 Packet delay attack topology based on example file.



```
channel Ethernet100 extends ned.DatarateChannel
{
    datarate = 10Mbps;
    delay = 10000ms;
}
```

Figure 4.3: Modification done to the ethernetcables.

and has done the necessary reconnaissance on the network. The scenario does not speculate on how the attacker has gained access or whether this is an insider helping (insider threat is discussed in discussion).

The attacker will have control over one or more nodes within the network and will start to add delay to the packages. For this scenario, the "Precision Time Protocol for INET" library is used, and the topology where the attack takes place can be seen in figure 4.4. For simulating the attack, an Omnet++ environment has been created in Ubuntu 16.04, building upon the example simulation that came together with the PTP library Precision Time Protocol for INET. In this attack scenario, the PTP message exchange is configured to use an end-to-end type of message exchange, and the attack scenario will show the result of how this effect the targeted PTP network. The scenario used a topology that was suitable for both a packet delay attack and a DOS type of attack, which is described in 4.4.3 next. For this scenario, some traffic is present between node trafSource and trafDest, but this is set to 1 package. Modifications have mainly been done to the .ned file where delay on every connecting connection has been added 0.050s delay to.

4.4.3 Scenario 2. PTP DOS Attack

Scenario 2. DOS attack using an end-to-end type of message exchange in PTP. In the scenario that the attacker has gained access to the PTP network and has done the necessary reconnaissance on the network. The scenario does not speculate on how the attacker has gained access or whether this is an insider helping (insider threat is discussed in discussion).

The attack will have control over one or more network nodes within the network and will start to send massive amounts of network data over the network. The packets sent from host 1 to host 2 are UDP packets in the simulation, but in the scenario, these are said to be GOOSE IEC 61850 types of messages. In the scenario, the two hosts will congest the network line, and the higher priority messages sent between them will make it difficult for PTP messages to come through. The messages sent for this scenario are PTP End-to-end message exchanges (section 2.2.4). The scenarios are heavily influenced by Precision Time Protocol for INET by Martin Levesque, and the scenario topology is built up from the example file that came with the package. The amount of data that is sent between the hosts has massively increased from 89 packets to more than 8000 packets. This is to simulate a malicious attack. This can also be an example of a miss configured device.

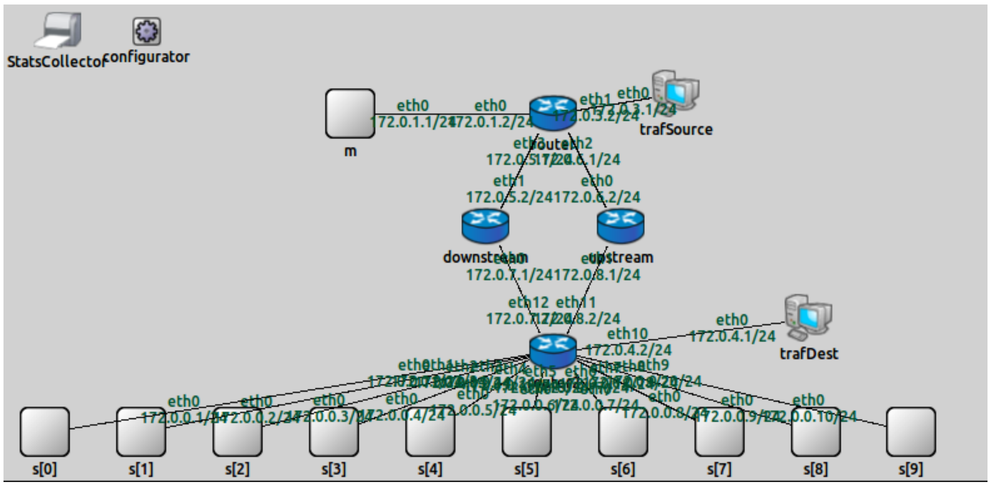


Figure 4.4: Example topology used for atack scenario 1.2 and 2.

4.4.4 Scenario 3. BMCA Attack

Scenario 3. BMCA (Best Master Clock Algorithm) attacks a peer-to-peer type of message exchange in PTP. In the scenario that the attacker has gained access to the PTP network and has done the necessary reconnaissance on the network. The scenario does not speculate on how the attacker has gained access or whether this is an insider helping (insider threat is discussed in discussion).

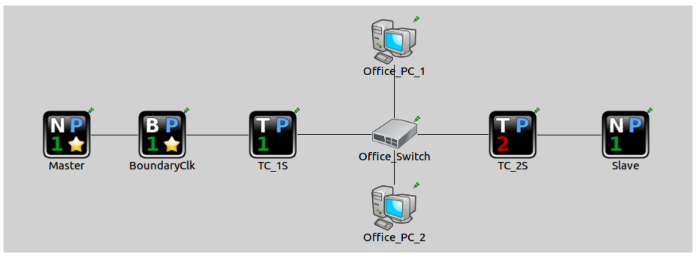


Figure 4.5: Topology where BMCA has chosen grandmaster that is not under control by the attacker.

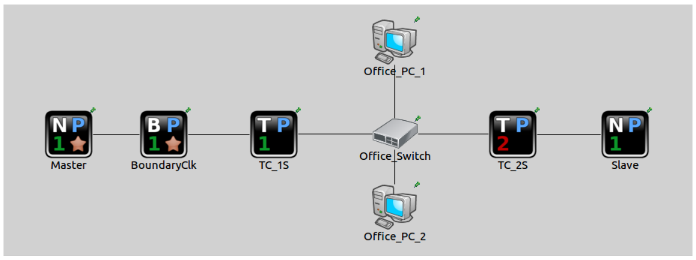


Figure 4.6: Topology where BMCA has chosen grandmaster that is under control by the attacker.

In the scenario, the attacker will announce a "better" grandmaster clock by falsely advertising to the BMCA algorithm that the attacker's master clock is the best in the network. BMCA is an algorithm that always will try and pick the "best" grandmaster. This algorithm always runs and is responsible for picking a new grandmaster clock if network connections are lost to the original grandmaster or if any other disruption were to happen on the network. In this scenario, the attacker will insert its own grandmaster PTP clock and effectively have full control over all PTP timestamps. Figure 4.5 shows an ideal topology where the correct (yellow start) grandmaster is present, this is the topology when running normally, and in figure 4.6 the attacker has taken over control of the grandmaster clock and the boundary clock, this is the topology used to simulate BMCA attack. The attack will only simulate the performance the attacker can have on PTP timestamps and will not simulate the necessary exploitation and reconnaissance done before the attack starts.

Chapter 5

Results

5.1 Scenario 1.1 Packet Delay Attack p2p

Simulation results for attack scenario 1.1. are shown in figure 5.1. The figure shows the effect of the introduced time delay on every network connection (10 000ms (10s)) and figure 5.1 shows the mean results of the type of attack. This attack is done on PTP Peer-to-peer message exchange and the results are presented as a graph. The active simulation time is set to end at $t = 100s$. As the graph shows the delay time will be higher depending on the number of hops that are needed in the network. As there are more connections the delay only grows depending on how many hops are present in the network between the grandmaster clock and the node. The graph indicates that after all nodes have received a Sync message, the calculated delay looks to have been already calculated as the graph will start to converge towards $0s$ delay starting after the $40s$ mark. This is due to the peer-to-peer message exchange mechanism, and due to the attacker using a static value for the delay, the PTP protocol using peer to peer will manage to adjust its clocks inside the network. The PTP clocks were given the following drifts as seen in tabel 5.1 based on what was used in paper [36].

M	Psw2	Psw1	Psw	S1	S2	S3	S4	S5	S6	S7	S8
0	-20	10	5	15	-10	-15	-20	15	10	20	-10

Table 5.1: M: master clock, Psw*: PTP switch, S: slave

Color	Devices	Hop(s)	First Sync after
Red	PTPswitch2	1	10s
Blue	Slave1, Slave2, Slave8, PTPswitch	2	20s
Orange	Slave3, Slave6, Slave7, PTPswitch1	3	30s
Green	Slave5, Slave4	4	40s

Table 5.2: Table describes what devices are present at one hop from topology from Figure 4.2

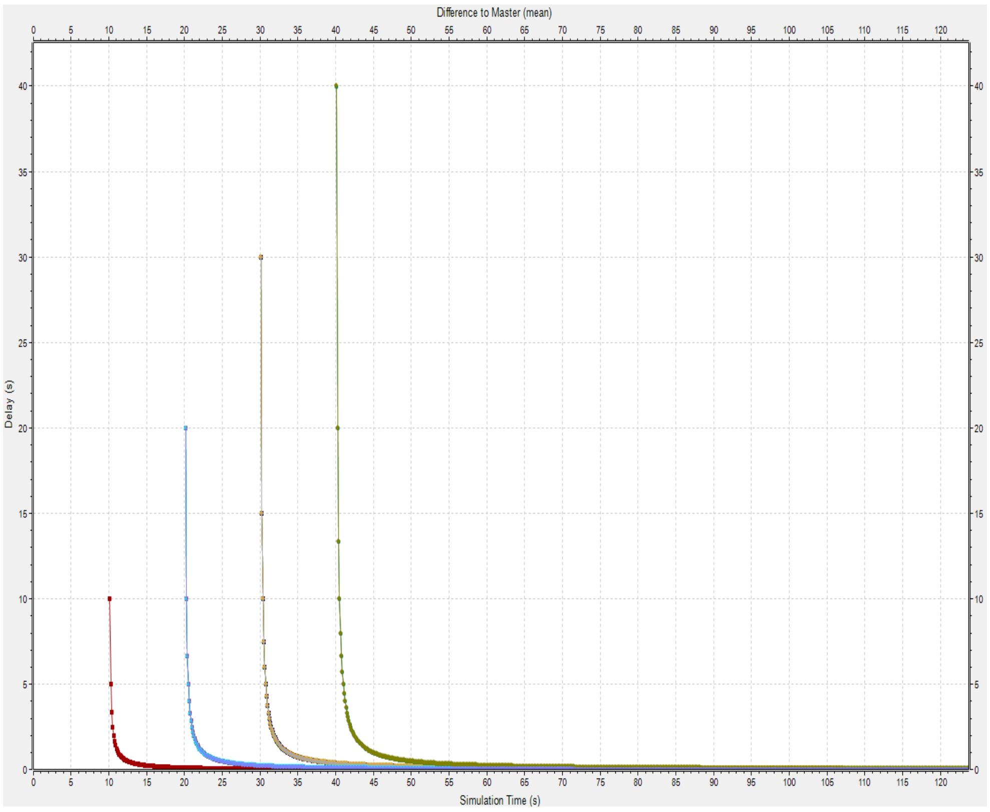


Figure 5.1: Mean delay using peer-to-peer message exchange.

5.2 Attack scenario 1.2 Packet Delay Attack e2e

In this delay attack End to end PTP message exchange has been delayed by 0.050s the results are presented in figure 5.2 and are simulated for $t = 100s$. In the graph below the initial 0.050s is first registered and then cut in half, before staying stable at 0.025s delay throughout the attack. The delay looks to be cut to approximately half as described in equation x in section y. This is due to the end-to-end message mechanism. The abnormalities at $t = 0s$ on the horizontal axis look like a result of some hiccup in timestamp update within the simulation environment. The type of attack looks to have a good effect on an attack on PTP devices and this will alter the PTP timestamps. The PTP protocol does not manage to adjust to the sudden delay added by the attacker in the attack scenario.

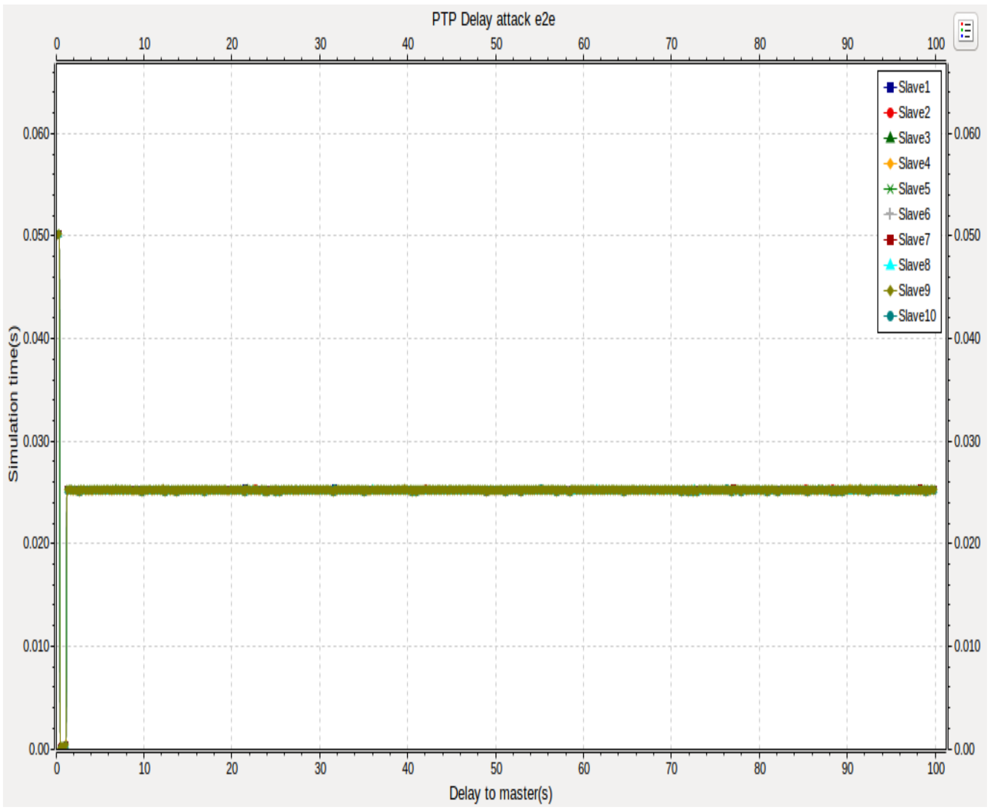


Figure 5.2: Packet delay attack on e2e PTP message exchange. Figure stops at 45s to preserve image quality

5.3 Attack scenario 2. PTP DOS Attack e2e

The simulation results for scenario 2. PTP DOS attack can be seen in figure 5.3. The attack commences at $t = 0s$ and ends at $t = 100s$. From the graph, we can see that the attacker is inserting an abnormal amount of network packages that are transmitted between a malicious host and a receiving host. The traffic looks to have some effect, but as shown in figure 5.3 there are some noticeable delays when the packet amount increases compared to what normal/little traffic looks like in figure 5.4. Based on the packets sent the amount of delay added to PTP clocks is about $0.00015s$, meaning that the type of attack is successful in adding delay to PTP synchronization. In this scenario, PTP message exchange is done using end to end messaging. Comparing figure 5.3 and figure 5.4 there is a less stable network in figure 5.3 where the attacker has started to flood the network with packets, and the attacker achieves to disrupt clock synchronization in the PTP network with a DOS attack.

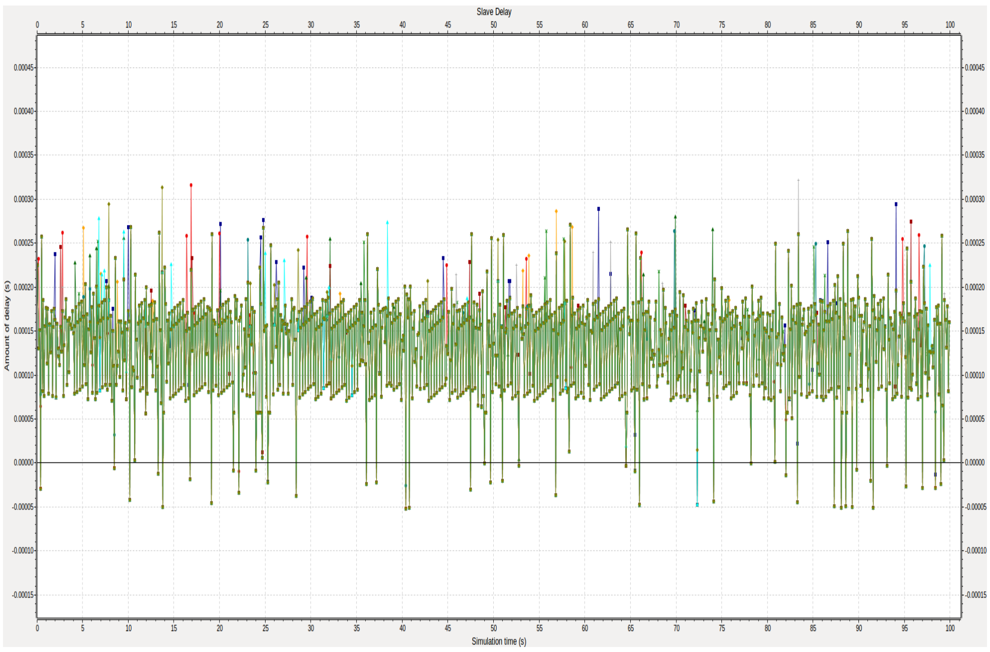


Figure 5.3: DOS attack effect on PTP network

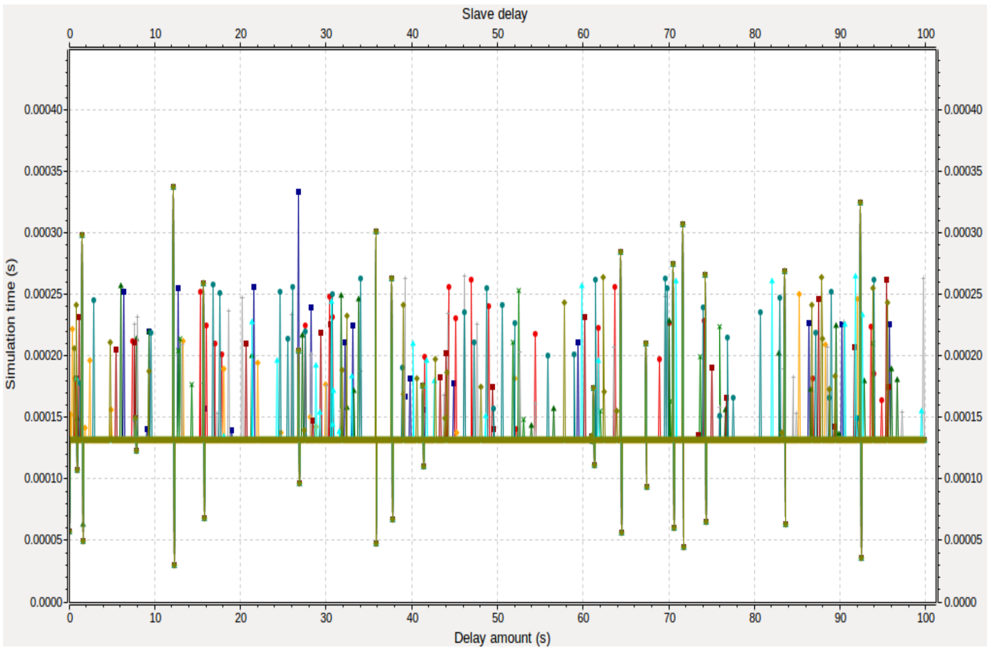


Figure 5.4: PTP network with normal traffic

5.4 Attack Scenario 3. PTP BMCA Attack

The simulation results for scenario 3. BMCA attack can be seen in figure 5.5. Here the attacker has advertised their malicious grandmaster clock to be the best in the network, and with this, the attacker can have full control over timestamps inside the PTP network. Here the attacker has started an attack to lower the quality of the timestamps, for comparison a figure of a good quality grandmaster clock can be seen in figure 5.6. Comparing the two figures the attacker can gain control over timestamps by exploiting the BMCA vulnerability as shown in the two figures. The attack starts at $t = 0s$ and ends at $t = 13s$. The PTP message exchange is done using peer to peer mechanism.

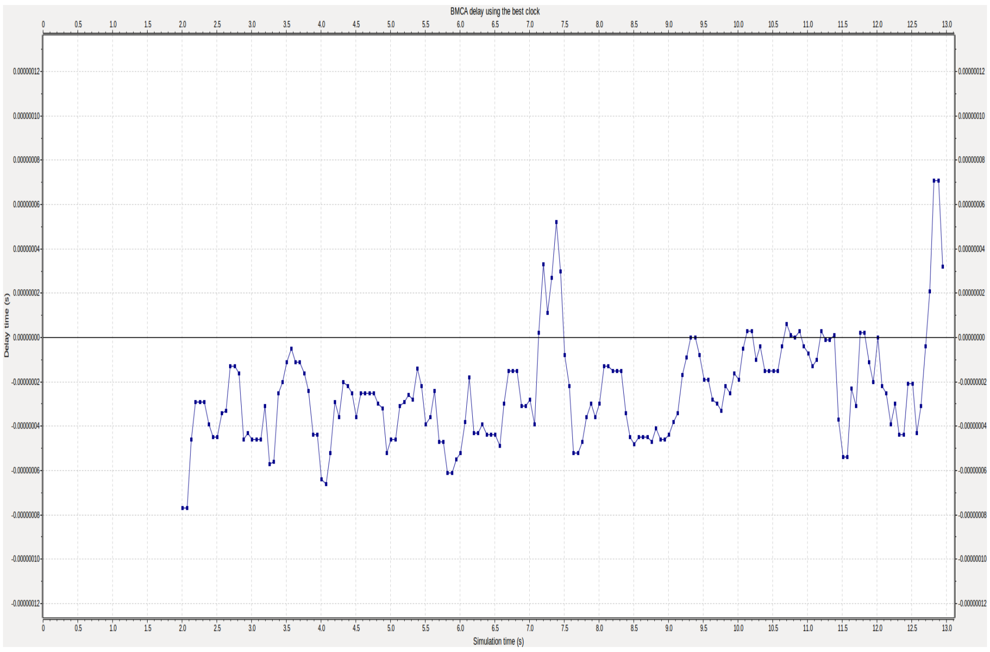


Figure 5.5: PTP network where the BMCA has chosen the correct best grandmaster clock

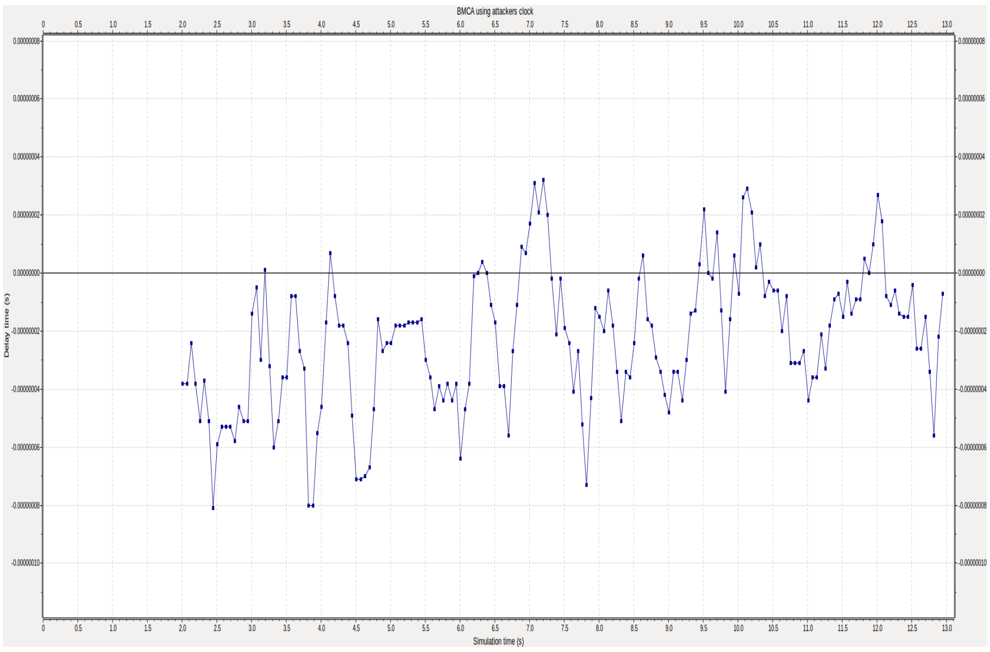


Figure 5.6: PTP network where BMCA has here chosen the malicious grandmaster by exploiting BMCA selection.

Chapter 6

Discussion

During the thesis a review has been conducted to the possible known vulnerabilities to PTP. In the literature review it was referred to papers who had discovered and experimented on the impact PTP vulnerabilities could have to PTP timestamping. This included for both messaging mechanism (e2e and p2p). For all known vulnerabilities it is necessary for the attacker to penetrate through a good amount of cyber security defenses if the attacker is attacking from an external locating. It is not impossible as recent types of attacks and comptonizations found in industrial type of networks can be retraced back to highly capable types of APTs that will have the technical skills, founds and motivation to be able to breach any kind of defenses given enough time.

6.1 Scenario Results

Results from scenario 1.2 have some slight hiccups at the beginning of the simulation time. This is discussed a bit in the results and looks to be the frequency of the Omnet++ observer that works in faster simulation time than the actual simulation and is due to the observer overachieving. The results generated have displayed that PTP is acceptable for a delay attack. This is done using topology presented in figure 4.4, where the delay was added to the upstream connection going to the master. Here the delay is added on one path making the connection asymmetrical, and the PTP messages rate will take effect from this. The value added was picked based on parameters from paper [17] and [18] that was presented in section 3.2. By using the same values as in paper [18], the paper could be used for validation together with the equation 2.2 from the background section 2.2.4. Based on the results, the simulation results show to be a success as it has clearly shown weaknesses in PTP. In scenario 1.1, the delay is added to both upstream and downstream traffic, and this makes the path delay symmetrical, and PTP can work with a symmetrical path delay. Here the delay added could be any value if the value is for upstream and downstream network packets.

The results for scenario 2 generated excellent results showing how PTP timestamps are affected when there are vast amounts of background network data. Initially, it was proposed to send GOOSE messages, but problems in setting priority in data packets took too much time, so the scenario had to be altered in production to include a DOS attack done by overflowing with UDP packets. The results generated from the simulation is running first with no background traffic over 100s, and this is represented in figure 5.4 and then with a lot of background traffic in figure 5.3. For injecting more traffic modifications to how many UDP packets were sent was needed to be configured for the omnet++ .ini file. The .ini file holds configurations to nodes and traffic in the simulation. The traffic generation parameter was changed from little to no traffic 80 packets, to vast amounts of background traffic: 8035 packets. The results from the scenario can be validated using research reviewed in related work [18].

The results for scenario 3. BMCA attack shows that an attacker will have complete control of PTP timestamps and the quality of the timestamps as it will insert itself as the master clock by falsely advertising it as the best time source, making the BMCA pick the attacker's "malicious" clock. In the attack scenario, small changes have been made from figure 5.5 where the original "good" master clock is the main source for timestamping. When the attack started, figure 5.6 shows the "malicious" clock doing the timestamping for PTP, and figure 5.5 gives more uneven timestamps with delay times varying more than in figure 5.5. The type of attack is harder to detect as the variation in delay increases by a small margin, but ultimately this is the most stealthy way an APT can gain full control over a systems synchronization where the goal is not to be detected but to add some stress to the system that can degrade it faster over time than with a good grandmaster clock.

6.2 How to Detect

By using and studying the results from the attack scenarios created there are good indications of what types of attacks are detectable and not. To be able to detect types of attacks there needs to be an understanding that there must be some sensor or intrusion detection system present in the network. With no cyber resilience put onto the network, PTP attacks can be hard to detect, but not impossible. Taking advantage of using an intrusion detection system that has been in the network and can detect anomalies can be able to detect PTP types of attacks. The intrusion detection system will look for abnormalities in network traffic or hosts alert if unknown behaviour is seen. Anomaly detection is known to alert on a lot of false positives but is a useful tool. Of the three attacks anomaly detecting will be able to detect scenarios 1.1, 1.2, 2 and scenario 3. Of the scenarios, 1.1 will alert as a connection went down and operations are back to normal (when slaves start to receive PTP messages again). This can then be compared to historic data that will be compared to identify If the delay has become higher or lower. Scenario 1.2 is also able to be detected by anomaly detection as the sudden addition of delay will be observed. Scenario 2 can be detected by any kind of sensor that can look at packets and the number of packets that flow through. A sensor that can inspect packets will be more efficient in filtering away false positives as it can look for bogus data inside the packets. Anomaly detection can possibly detect a BMCA attack where the attacker's master has been picked. This depends on how each PTP node is configured with parameters. The IDS can detect if there were inserted new nodes with different configuration parameters to already known nodes. It is detectable if the alterations are big enough to cause a change in the network, but this depends on how sensitive the anomaly detection is configured to be. Of the attacks covered in the thesis, BMCA is the most stealth kind of attack where an attacker can have full control of PTP timestamps. A method for correctly identifying if there is a BMCA attack is to compare other grandmaster clocks used for PTP synchronization in different segregations of the network. By comparing the GPS clock from different masters an abnormal value on one master can be identified and can be investigated.

6.3 Attack Scenarios Continued

What can be devastating for many types of industrial control systems and not only for PTP is an insider threat. Systems used in these types of environments will have their focus on availability and uptime, and some systems do not have any security as this can be "unsafe" in some situations. Control systems should be kept behind cyber defenses such as firewalls, intrusion detection systems and network segregations, as well as physical defenses such as keypad doors, facial recognition etc. Given an insider threat all these defense assurances can easily be bypassed, and vulnerable systems be accessed and deployed malware on. An insider can also give very specific details about machine types and firmware specific specifications for a malware creator to make tailor made malware for that specific system.

6.4 Hardware Versus Software Simulations

The thesis has successfully shown that a software simulation approach is viable to use to detect and plan out topologies. Previous research show to a hardware-based solution [17] is often used to simulate what effect PTP vulnerabilities will have on timestamping, and with this thesis, proof that software is just as viable and more cost-efficient has been proven. The software simulation approach can open up for more research to be done on PTP as all libraries and tools used in the thesis have been open source and some libraries are continued supported with newer versions of Omnet++ and INET. The software simulation solution can also be extended to include types of detection such as the paper [22] showed. With the cost benefits of using software simulation and the modifications that can be done to it, it is a good choice for researchers and students who can use the software simulation for discovery and proof of concept before buying hardware. The software simulation can also complement already built PTP networks for simulating results when adding new configurations to an already built network. The software will also continually be patched and added with new features.

6.5 Simulation Tool

The choice of working with Omnet++ has been difficult as Omnet++ modelling is done in C++, and requires good pre-knowledge on how classes work in Object-Oriented Programming (OOP). Review of each packet was also challenging and time-consuming as it was necessary to know how the code worked before implementing attacks. Omnet++ has on their main website advertised two different libraries that has cybersecurity in mind. These are NETA [38] and SEA++[39]. NETA library is created by Network Engineering & Security Group[40]. The library comes with sinkhole, delay and dropping attacks for Omnet++, and looked promising to use for the thesis, but was not compatible with any of the PTP libraries that had to be used as NETA required INET version 2.1 that was not compatible with the rest of the project. SEA++ had the same issues as NETA as it required an older version of Ubuntu just to run, as well as an older INET version. SEA++ is an Omnet++ library that lets the user write a custom scenario using XML. It comes with many different modules that could be utilized to create attack scenarios. Some modules were Destroy, Drop, Create etc.

Newer versions of INET did look promising for creating attack scenarios, as new versions of INET had gained prebuilt modules for dropping, delaying and replay packets in a newer version of INET. Unfortunately, it was not possible at the time for this thesis to port one of the PTP libraries to a newer INET version. It was attempted to create interfaces and adapters for PTP libraries, but it was quickly found out that this would take more time than was available.

Omnet++ also has enormous amounts of documentation and at the same time a small community. This caused it difficult to look for tutorials using YouTube or blogs. It was a community based on Google Groups, and this was the go-to place to ask for help. If there were coding specific questions StackOverflow could be used, but it was difficult for both places to get answers. The Google Groups community had other authors asking for help making attack scenarios and if they were answered it was either with an answer that it

6.6 Current Threat Landscape

While writing this thesis war between one of Europe main energy supplier of gas and oil started a war with Ukraine. With sanctions on energy resources from this country to EU becoming a factor, it is not unlikely that major APTs will and have started to target energy facilitations where PTP is used as synchronization protocol. It is therefor likely that industrial control systems will face a higher risk in the current threat landscape. The current threat landscape can be a trigger for new types of PTP vulnerabilities to be discovered as major energy contributors in

Europe will need to harden their cyber security. With this it is no unlikely that similar attack such as the Triton attack can be deployed within Europe energy sector.

Chapter 7

Conclusion

The master thesis focuses on the synchronization protocol used in ICS named Precision Time Protocol (PTP). PTP offers nanosecond precision timestamp accuracy that is used in ICS environments. PTP is a well know synchronization protocol that has little to no security features, as PTP is made with availability in mind. PTP is vulnerable to known attacks such as Denial of Service, Replay attacks, and Packet delay attacks. PTP is also vulnerable to PTP-related attacks such as BMCA, master and slave spoofing, packet modifications, and attacks on GPS as PTP gets its time reference using GPS signals. To examine the effects that different vulnerabilities have on PTP, four different scenarios were made to look at the effect different attacks would have on PTP. The attack scenarios were created based on three known vulnerabilities: Packet Delay Attack, Denial of Service attack, and Best Master Clock Algorithm (BMCA) attack. The different scenarios were crafted based on previous research presented in the related work section of the thesis.

For generating results, a software simulation approach was done using Omnet++ with INET and three different PTP libraries that had the capability to simulate peer-to-peer and end-to-end PTP message exchange. The simulations were modeled based on attack scenarios, and three example topologies were created to run the attack simulations. The three different topologies were different from one another as they were used for scenario-specific attacks. Three different attacks on two unique topologies were successful using attacks such as Packet delay on end-to-end communication, Denial of Service on end-to-end messaging, and a BMCA attack where an attacker inserter their malicious master clock all generated expected results where the attacker managed to successfully alter PTP timestamps. Of the three attacks,attacks, it was the BMCA attack that was concluded to be the stealthiest. This can mimic the the original master clock (good clock) and initiate alterations to timestamps whenever the attacker wantswants to start the attack. From the simulations, the difference between a good clock and a “worse” clock, where the worse clock added a more unstable delay swing in the graph.

The thesis's main findings are to examine PTP vulnerabilities using simulation software Omnet++. Three ideal topologies have been used based on knowledge gained from studying state-of-the-art vulnerabilities to PTP. The results generated are successful as they can be validated using previously available research.

7.1 Future Work

The choice of simulating the attacks using software simulations has left room for improvements that can be studied further. What can be continued on is looking at how the different attacks would perform on the same topology instead of using three different. The topology used for future work should also be a more advanced topology that can better resemble what an actual industrial network would be designed. By using a more advanced topology, different attacks that were not conducted in this master thesis could be done and record how they would affect PTP timestamping. The kinds of attacks should also be looked to extend to advanced stealth types of attacks on PTP as it is reasonable to guess that an APT would utilize such attacks. By expanding the attacks, it is also possible to validate the simulation results by configuring own hardware-based PTP simulation.

In one, only one of the four scenarios where background traffic was introduced, and this should be a key factor in extending the research done by observing and adding valid background traffic that would be present within an industrial network where ICS devices reside.

One could also implement own security features as modules in Omnet++ and create observers that can look for active attacks. This would be similar to implementing an intrusion detection system module to Omnet++, and results can be generated based on how good the security features are at detecting attacks on PTP. How a digital twin would behave under different PTP attacks will also show what PTP hardening would need to be addressed to ensure that synchronization issues cannot arise if the digital twin is used together with ICS devices.

Bibliography

[1] *Cyber-Attack Against Ukrainian Critical Infrastructure | CISA*, Feb. 2016. [Online]. Available: `https://www.cisa.gov/uscert/ics/alerts/IR-ALERT-H-16-056-01`.

[2] B. Johnson, D. Caban, M. Krotofil, D. Scali, N. Brubaker and C. Glyer, *Attackers Deploy New ICS Attack Framework "TRITON" and Cause Operational Disruption to Critical Infrastructure | Mandiant*, Dec. 2017. [Online]. Available: `https://www.mandiant.com/resources/attackers-deploy-new-ics-attack-framework-triton`.

[3] R. Langner and B. Schneier, 'To Kill a Centrifuge A Technical Analysis of What Stuxnet's Creators Tried to Achieve The Langner Group "The definitive analysis of Stuxnet",' 2013. [Online]. Available: `www.langner.com`.

[4] A. S. Rana, N. Parveen, S. Rasheed and M. S. Thomas, 'Exploring IEEE standard for synchrophasor C37.118 with practical implementation,' *12th IEEE International Conference Electronics, Energy, Environment, Communication, Computer, Control: (E3-C3), INDICON 2015*, Mar. 2016. DOI: `10.1109/INDICON.2015.7443664`.

[5] *Smart power grid synchronization using IEEE 1588 PTP • Qulsar*. [Online]. Available: `https://qulsar.com/Applications/IoT/Smart_Power_Grid.html`.

[6] M. O'Riordan, *Everything You Need To Know About Publish/Subscribe | Ably Realtime*, Jul. 2020. [Online]. Available: `https://ably.com/topic/pub-sub`.

[7] E. D. Knapp and J. T. Langill, *Industrial Network Security Securing Critical Infrastructure Networks for Smart Grid, SCADA, and Other Industrial Control Systems Second Edition*, second, R. Samani, Ed. Elsevier Inc., 2015.

[8] R. Ramanathan, 'The IEC 61131-3 programming languages features for industrial control systems,' *World Automation Congress Proceedings*, pp. 598–603, Oct. 2014, ISSN: 21544832. DOI: `10.1109/WAC.2014.6936062`.

[9] M. Maslar, 'PLC standard programming languages: IEC 1131-3,' *IEEE Conference Record of Annual Pulp and Paper Industry Technical Conference*, pp. 26–31, 1996, ISSN: 01902172. DOI: `10.1109/PAPCON.1996.535979`.

[10] *What is HMI? | Inductive Automation*, Aug. 2018. [Online]. Available: https://www.inductiveautomation.com/resources/article/what-is-hmi.

[11] *Industrial Control System - Definition*. [Online]. Available: https://www.trendmicro.com/vinfo/us/security/definition/industrial-control-system.

[12] D. L. Mills, 'Internet Time Synchronization: The Network Time Protocol,' *IEEE Transactions on Communications*, vol. 39, no. 10, pp. 1482–1493, 1991, ISSN: 00906778. DOI: 10.1109/26.103043.

[13] *Documents - IEEE P1588 Working Group*. [Online]. Available: https://sagroups.ieee.org/1588/public-documents/.

[14] '1588-2008 IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems.,'

[15] M. D. Johas Teener, G. M. Garner, B. Hur and D. Huang, 'Overview and timing performance of IEEE 802.1AS,' *2008 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication, ISPCS 2008, Proceedings*, pp. 49–53, 2008. DOI: 10.1109/ISPCS.2008.4659212.

[16] *Implementing IEEE 1588v2 for use in the mobile backhaul*. [Online]. Available: http://www.butlergroup.ie/wp-content/uploads/wpsc/product-files/880_Calnex_Technical_Brief___IEEE_1588v2_PTP_Mar10.pdf.

[17] W. Alghamdi and M. Schukat, 'Precision time protocol attack strategies and their resistance to existing security extensions,' *Cybersecurity*, vol. 4, no. 1, pp. 1–17, Dec. 2021, ISSN: 25233246. DOI: 10.1186/S42400-021-00080-Y/FIGURES/9. [Online]. Available: https://cybersecurity.springeropen.com/articles/10.1186/s42400-021-00080-y.

[18] M. Han and P. Crossley, 'Vulnerability of IEEE 1588 under Time Synchronization Attacks,' *IEEE Power and Energy Society General Meeting*, vol. 2019-August, Aug. 2019, ISSN: 19449933. DOI: 10.1109/PESGM40551.2019.8973494.

[19] K. Rösel, M. Helm, J. Zirngibl and H. Stubbe, 'Current Developments of IEEE 1588 (Precision Time Protocol),' DOI: 10.2313/NET-2021-05-1{_}04.

[20] D. Arnold, *The PTP AUTHENTICATION TLV*, Jun. 2020. [Online]. Available: https://blog.meinbergglobal.com/2020/06/04/the-ptp-authentication-tlv/.

[21] B. Moussa, C. Robillard, A. Zugenmaier, M. Kassouf, M. Debbabi and C. Assi, 'Securing the Precision Time Protocol (PTP) Against Fake Timestamps,' *IEEE Communications Letters*, 2018, ISSN: 15582558. DOI: 10.1109/LCOMM.2018.2883287.

[22] B. Moussa, M. Kassouf, R. Hadjidj, M. Debbabi and C. Assi, 'An extension to the precision time protocol (PTP) to enable the detection of cyber attacks,' *IEEE Transactions on Industrial Informatics*, vol. 16, no. 1, pp. 18–27, Jan. 2020, ISSN: 19410050. DOI: 10.1109/TII.2019.2943913.

[23] W. Wallner, 'Simulation of the IEEE 1588 Precision Time Protocol in OMNeT++,' Sep. 2016. DOI: 10.48550/arxiv.1609.06771. [Online]. Available: https://arxiv.org/abs/1609.06771v1.

[24] *White Rabbit Official CERN website*. [Online]. Available: https://white-rabbit.web.cern.ch/.

[25] *Deutsche Börse Group - Time services*. [Online]. Available: https://www.deutsche-boerse.com/dbg-en/products-services/ps-technology/ps-connectivity-services/ps-connectivity-services-time-services.

[26] F. Girela-Lopez, J. Lopez-Jimenez, M. Jimenez-Lopez, R. Rodriguez, E. Ros and J. Diaz, 'IEEE 1588 High Accuracy Default Profile: Applications and Challenges,' *IEEE Access*, vol. 8, pp. 45 211–45 220, 2020, ISSN: 21693536. DOI: 10.1109/ACCESS.2020.2978337.

[27] 'White Rabbit training Seven Solutions,' [Online]. Available: www.sevensols.com.

[28] W. Sydney University Library Study Smart, 'Library Study Smart Literature review purpose,' 2017. [Online]. Available: http://services.unimelb.edu.au/academicskills/all_resources/writing-resources.

[29] *Scopus preview - Scopus - Welcome to Scopus*. [Online]. Available: https://www.scopus.com/standard/marketing.uri#basic.

[30] *OMNeT++ Discrete Event Simulator*. [Online]. Available: https://omnetpp.org/.

[31] *Releases · inet-framework/inet · GitHub*. [Online]. Available: https://github.com/inet-framework/inet/releases.

[32] *IEEE8021AS · master · Peter Danielis / gPtp implementation · GitLab*. [Online]. Available: https://gitlab.amd.e-technik.uni-rostock.de/peter.danielis/gptp-implementation/-/tree/master/IEEE8021AS.

[33] *GitHub - martinlevesque/ptp-plusplus: Precision Time Protocol (PTP) module for OMNeT++ / INET 2.6*. [Online]. Available: https://github.com/martinlevesque/ptp-plusplus.

[34] *GitHub - ptp-sim/libPTP: Library to simulate the Precision Time Protocol (PTP, IEEE 1588) in OMNeT++*. [Online]. Available: https://github.com/ptp-sim/libPTP.

[35] *Simulation of PTP (IEEE 1588) | ptp-sim.github.io*. [Online]. Available: https://ptp-sim.github.io/.

[36] H. Puttnies, P. Danielis, E. Janchivnyambuu and D. Timmermann, 'EPiC Series in Computing A Simulation Model of IEEE 802.1AS gPTP for Clock Synchronization in OMNeT++,' vol. 56, 2018. [Online]. Available: `https://gitlab.amd.e-technik.uni-rostock.de/peter.`.

[37] *Ipv4NetworkConfigurator*. [Online]. Available: `https://doc.omnetpp.org/inet/api-current/neddoc/inet.networklayer.configurator.ipv4.Ipv4NetworkConfigurator.html`.

[38] 'NETA: A NETwork Attacks Framework Architecture and Usage developed by NESG (Network Engineering and Security Group) members,' 2013.

[39] *seapp/seapp_stable: SEA++ sources (stable)*. [Online]. Available: `https://github.com/seapp/seapp_stable`.

[40] NETA and Network Engineering and Security Group, 'NETA: A NETwork Attacks Framework Architecture and Usage developed by NESG (Network Engineering and Security Group) members,' 2013.