Hes·so VALAIS WALLIS

School of Engineering

Hes·so
Haute Ecole Spécialisée de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences and Arts Western Switzerland

Degree Programme

Systems Engineering

Major Infotronics

BACHELOR'S THESIS

DIPLOMA 2024

Rémi Heredero

OT Security

PEN-testing and security about embedded devices

Professor
Prof. Medard Rieder, medard.reider@hevs.ch

Expert
Rico Steiner, rico.steiner@hooc.ch

Submission date of the report
30 August 2024

Valais excellence CERTIFIED

HES-SO Valais

SYND	ETE	TEVI
X	X	X

Données du travail de bachelor
Aufgabenstellung der Bachelorarbeit

FO 1.2.02.07.FB
che/13.03.2024

Filière / Studiengang **SYND**	Année académique / *Studienjahr* **2023-24**	No TB / *Nr. BA* **IT/2024/78**
Mandant / *Auftraggeber* ☒ HES—SO Valais-Wallis ☐ Industrie ☐ Etablissement partenaire *Partnerinstitution*	Etudiant·e / *Student/in* **Heredero Rémi** Professeur·e / *Dozent/in* **Rieder Medard**	Lieu d'exécution / *Ausführungsort* ☒ HES—SO Valais-Wallis ☐ Industrie ☐ Etablissement partenaire *Partnerinstitution*
Travail confidentiel / *vertrauliche Arbeit* ☐ oui / ja ☒ non / nein	Expert·e / *Experte/Expertin* *(nom, prénom, E-mail/Name, Vorname, E-Mail)* **Steiner Rico** - rico.steiner@hooc.ch HOOC AG, Visp	

Titre / *Titel* **OT Security labs development**

Description

The goal of this Bachelor thesis is to prepare several security related experiments in an Operational Technology (OT) environment. The environment is simulated, either using a platform named "Winter Resort Simulator Pro" or using Minecraft. In the latter case, the simulated environment still must be defined and developed. The scenarios that will be developed include wired communication by MODBUS, wireless communication using a 862 MHz radio, process control using sensors and PLC's.

Tasks

- Create an environment and select tools permitting to intercept, modify and reinject MODBUS packets, assuming the intruder has access to the communication infrastructure. Propose solutions to monitor the communication infrastructure and therefor detect and prevent such activity.
- Create an environment and select tools permitting to scan a wireless network. Intercept packets and reinject them into the network. Propose solution to harden wireless protocols against such actions.
- Create an environment and select tools to intercept sensor data to compromise the function of the controller due to invalid data. Propose approaches permitting hardening the controller software against such action.
- Prepare a setup and tools permitting to break into a PLC. Propose solutions hardening PLCs against such action.

Deliverables

- A report
- A working demonstrator
- All source code and tools
- A presentation

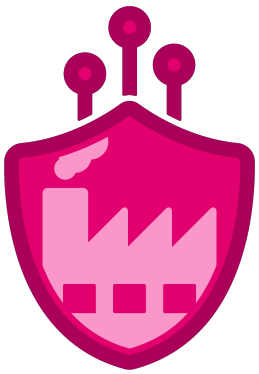
Signature ou visa / *Unterschrift oder Visum*	Délais / *Termine*
Responsable de l'orientation / *Leiter/in der Vertiefungsrichtung:* ..... [1] Etudiant·e / Student/in:	Attribution du thème / *Ausgabe des Auftrags:* **27.05.2024** Présentation intermédiaire / *Zwischenpräsentation*: **Semaine/Woche 27** (01-05.07.2024) Remise du rapport final / *Abgabe des Schlussberichts:* **30.08.2024, 12:00** Expositions / *Ausstellungen der Bachelorarbeiten:* **23.08.2024** – HEI **26.08.2024** – Monthey **29.08.2024** – Visp Défense orale / *Mündliche Verfechtung:* **Semaine/Woche 38** (16-20.09.2024)

[1] *Par sa signature, l'étudiant·e s'engage à respecter strictement la directive DI.1.2.02.07 « Travail de bachelor ».*
Durch seine Unterschrift verpflichtet sich der/die Student/in, sich an die Richtlinie DI.1.2.02.07 „Bachelorarbeit" zu halten.

Hes·so VALAIS WALLIS

School of Engineering

OT Security



Graduate Rémi Heredero

Objective

Produce attack scenarios where each includes a vulnerability, an attack using the vulnerability, and a solution to avoid the attack. These scenarios must be usable as the basis for a laboratory experiment for students or industrial partners.

Methods | Experiences | Results

This thesis is divided into 2 scenarios. The first one, a Man in the Middle attack, consists of intercepting the communication between two devices and modifying the packets exchanged. The second scenario is a replay attack on a wireless transmission medium performed with a Flipper Zero.

The first scenario (Man in the Middle Attack) is implemented on a Modbus/TCP communication between a controller and a house's security system. An attacker (a Kali Linux laptop) redirects communication to him to modify the sensor values sent to the controller. It is also possible to sent fake data. This Thesis also shows how securing the communication with certificates (Modbus/TLS) but without certificate validity check can be subject to an attack too. The attacker can intercept the communication in the same way. The thesis concludes that it is important to check certificates to guarantee the interlocutor.

The second scenario (Replay) is implemented on a 433MHz basic wireless communication. A Flipper Zero can record the transmission and replay it later to trigger the same effect. Securing such a communication can be simply implemented by using a rolling code or a signature. If each message is unique, an attacker cannot replay it.

Bachelor's Thesis
| 2024 |

Degree programme
Systems Engineering

Field of application
Infotronics

Supervision professor
Prof. Medard Rieder
medard.reider@hevs.ch

Information about this report

Contact Information

Author:	Rémi Heredero
	Bachelor Student
	HEI-Vs
Email:	remi.heredero@students.hevs.ch

Declaration of honor

I, undersigned, Rémi Heredero, hereby declare that the work submitted is the result of a personal work. I certify that I have not resorted to plagiarism or other forms of fraud. All sources of information used and the author quotes were clearly mentioned.

Place, date: Sion, 29.08.2024

Signature: 

Contents

Figures

Listings

Acknowledgements

As this thesis marks the completion of my bachelor's degree at the University of Applied Sciences Western Switzerland, HES-SO Valais Wallis, I would like to express my deepest gratitude to several individuals who have supported a guided me throughout this journey.

First and foremost, I would like to thank my thesis supervisor, Prof. Medard Rieder, for his guidance not only this Thesis but also throughout my studies. His expertise and knowledge have been invaluable to me, and I am grateful for the time and effort he has dedicated to helping me succeed. I am also deeply grateful to Michael Clausen, who followed my progress closely and offered continuous support and insightful feedback.

I would also like to thank Prof. Silvan Zahno, who introduced me to Typst and crafted a wonderful template and work for this new, fresh tool. My appreciation also goes to Prof. Christopher Metrailler, who presented me a lot of tools, including my new favourite code editor, ZED.

My heartfelt thanks go to my family for their support, and to my friend Yann for his unwavering support and for his proofreading of my report. Finally, I am profoundly grateful to my brother Louis, whose thorough review and incredible Typst packages played a crucial role in the final presentation of this thesis.

Thank you all for your contributions and support. This thesis would not have been possible without you.

Abstract

This thesis examines the security of embedded systems in a world where the distinctions between IT and OT have increasingly blurred. Traditionally, OT systems were physically isolated to ensure security. However, with the advent of the IoT, these systems are becoming more interconnected, making them more susceptible to cyberattacks.

In response to these evolving challenges, University of Applied Sciences Western Switzerland, HES-SO Valais Wallis overhauled his teaching laboratory dedicated to OT security. This lab will focus on practical, hands-on exercises to help students to understand the unique challenges of securing embedded systems.

The primary objective of this thesis was to create realistic attack scenarios for use in these laboratories, with a focus on Modbus communication protocols and wireless systems. These scenarios are designed to help students identify vulnerabilities in OT systems and learn how to secure them effectively.

The thesis centres on two key attacks scenarios. The first is a MitM attack on Modbus/TCP communication, demonstrating how an attacker can intercept and alter unencrypted messages. This scenario underscores the importance of implementing TLS and verifying digital certificates to mitigate such threats. The second scenario involves a replay attack on a wireless communication system, using the Flipper Zero device to capture and retransmit signals.

Keywords: *OT, security, man in the middle, modbus, replay, attack, flipper*

1 Introduction

This thesis is set in a world that is becoming increasingly interconnected. Not long ago, the IT world and industrial Programmable Logic Controller (PLC) systems were distinct and separate, with clear boundaries between them. However, the rise of the IoT has blurred these lines, breaking down the barriers that once existed between these two domains.

From the outset, the IT world has faced threats from malicious individuals wanting to gain control of systems. In response, various security measures have been developed to fend off these attackers. In contrast, the OT world, which includes industrial systems and PLCs, has traditionally focused more on robustness and reliability rather than on security. The prevailing strategy for industrial systems was to isolate them from external networks, assuming that physical separation would suffice for security.

Additionally, embedded systems in the OT world lacked the processing power needed to perform cryptographic operations, making security implementations challenging. As a result, the OT domain remained largely disconnected from best practices in cybersecurity for many years.

With the advent of IoT, the desire to interconnect everything has become a driving force. Industries now seek to push real-time production data to the cloud, making OT systems more vulnerable to attacks. This shift has forced the OT world to incorporate security measures into its systems. Meanwhile, advancements in technology have made embedded systems more powerful, enabling them to handle cryptographic operations that were previously out of reach.

While some industries are beginning to secure their production lines, many still employ poor practices and maintain unsecured systems. In this context, the University of Applied Sciences Western Switzerland, HES-SO Valais Wallis has introduced a security course in the 6th semester for Infotronics students. With the upcoming changes in the study plan, this course will evolve to focus more specifically on the security of embedded systems, also known as OT Security.

I also chose this thesis subject due to my keen interest in Security and PEN-testing. Additionally, I already have some basic ethical hacking background and would like to specialize in the security of embedded systems during the rest of my studies. This thesis presents a valuable opportunity to delve into this field, and I have already enrolled for a master's degree to continue on this path.

1.1 Objectives

As the first step in overhauling the security course, this thesis plays a crucial role. It aims to develop scenario cases that can be used in the laboratory. These scenarios must be both realistic and feasible for students to perform during lab exercises, with an additional goal of making them engaging and fun.

This thesis will produce between three and four attack scenarios. Each scenario will start with an unsecured situation, either physical or simulated. The scenario will then include an attack that exploits this vulnerability, followed by a solution to secure the system. Ide-

ally, students should be able to carry out the attack or implement the security measures in future lab sessions.

The scenarios will cover at least one Modbus communication and one wireless communication. Whenever possible, the scenarios should involve a Wago PLC and a Flipper Zero device.

1.2 Methodology and structure

An important aspect of a bachelor's thesis is that it also needs to serves as a valuable learning experience for author. Therefore, I chose to approach this project with as much independence as possible. In my view, by working autonomously and asking fewer questions, one can gain a deeper understanding and truly benefit from the experience, even if it means the project takes longer to complete.

The thesis begins with an analysis of various attacks that can be carried out on embedded systems, with a particular focus on communication vulnerabilities. This section also explores the different media that can be used to execute these attacks and examines the simulation environments suitable for testing them.

The thesis then develops two selected scenarios in greater detail. It presents the environments and tools needed for each scenario, explains how to perform the attacks, and finally, proposes solutions to secure the systems.

The main repository for this thesis, which also serves as the entry point for all code and documentation, can be found at the following address:

https://github.com/Klagarge/BachelorThesis-OTSecurity

Some modern tools like AI were used to assist in the reformulation and paraphrasing of this final document. However, all code and text in this thesis were originally written by the author, Rémi Heredero.

2 Impact on Sustainability

This section explores the impact of this thesis on sustainability, with a specific focus on the United Nations' Sustainable Development Goals (SDGs) . By examining the intersection of OT security and sustainability, this section demonstrates how securing industrial and home automation systems contributes to achieving global sustainability targets.

The SDGs provide a blueprint for achieving a better and more sustainable future. This thesis aligns particularly with the following goals:

Goal 9: Industry, Innovation, and Infrastructure
Industry 4.0 relies heavily on interconnected OT systems. By addressing vulnerabilities and enhancing the security of these systems, this thesis promotes the development of robust and resilient infrastructure. Secure industrial processes foster innovation and sustainable industrialization. This thesis contributes to building infrastructure that supports economic development and human well-being, with a focus on sustainable industrialization and fostering innovation.

Goal 11: Sustainable Cities and Communities
Home automation systems are integral to the development of smart cities. This thesis examines security measures for these systems, ensuring they are protected against cyber threats. Secure home automation contributes to the safety, efficiency, and sustainability of urban environments. By protecting the systems that manage energy use, water distribution, and waste management, this research supports the development of cities and human settlements that are inclusive, safe, resilient, and sustainable.

Goal 12: Responsible Consumption and Production
Efficient resource management is a key aspect of responsible consumption and production. This thesis enhances the security of systems that monitor and control resource usage, such as smart meters and automated manufacturing processes. By preventing tampering and ensuring accurate data collection, this research helps optimize resource consumption and reduce waste. This aligns with the goal of ensuring sustainable consumption and production patterns.

Conclusion
This bachelor thesis on OT security helps to advance on sustainability goals. By enhancing the security and reliability of industrial and home automation systems, this research supports the UN's Sustainable Development Goals, promoting a more sustainable and secure future. Integrating security into these systems is crucial for sustainable development, highlighting the need for interdisciplinary approaches in addressing global challenges.

3 Analysis

This section discusses various attacks, communication media and simulation environments that could be used in the laboratory. It aids in selecting the appropriate attack on the right medium and simulation environment, essential for the future laboratory. The requirements of this thesis include the use of Modbus and an attack with the Flipper Zero device.

Contents

3.1 Attacks

Numerous attacks can occur in the context of OT security and can be classified into different categories. This thesis covers some attack as following, though many others exist.

3.1.1 Sniffing Attack

This attack consists in listening to the communication between two devices [1]. It can be performed on every communication medium with varying levels of difficulty. Wireless communication is particularly vulnerable because anyone can intercept the signals. For example, in Figure 1, Alice sends a message to Bob over the air without encryption, allowing Eve to listen to and read the message.

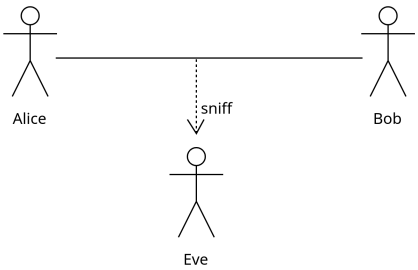


Figure 1: Sniffing attack

A sniffing attack can be performed to get secret information or understand a chemical recipe, for example. It can also be used as part of other more complex schemes.

Security Measures

To protect against sniffing attacks, the communication must be encrypted. The encryption must be strong enough to resist the attacker. A simple symmetrical encryption is enough. Particular attention must be paid to the key exchange. Protocols such as Diffie-Hellman with symmetrical encryption are recommanded to ensure protection against sniffing attacks.

3.1.2 Spoofing

In network security, spoofing involves impersonating another device. This attack is often used in combination to other attacks, such as creating a fake Wi-Fi hotspot. In the context of OT security, spoofing is less relevant and will not be discussed further in this thesis.

3.1.3 Denial of Service

An attack by DoS aims to render a service unavailable by overloading a device or network with messages [2]. In OT environments, a capable computer can execute a DoS attack effectively. To reach this goal, a device sends a large number of messages to surcharge a device or a network. In Figure 2 we can see that Mallory wants to overload Bob with messages. Bob is hence unable to answer to all messages and becomes unavailable. In IT world, it is more often a Distributed Denial of Service (DDoS) attack because usually servers are more powerful than a computer, and thus can handle more concurrent messages. The attacker distributes the messages on multiple devices to make the offense more difficult to block. In OT world, it is useless to perform a DDoS to make a device unavailable since a DoS with a capable computer is generally enough. Another perspective is that OT world is typically in closed loop networks and thus not accessible from the outside world.

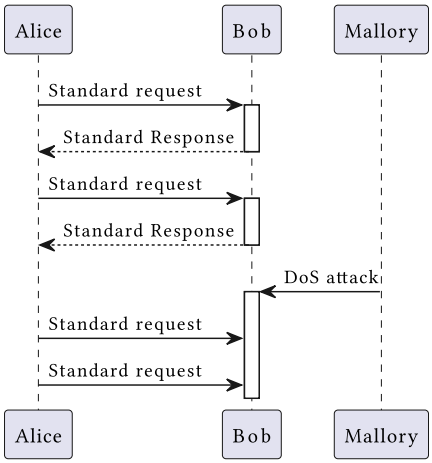


Figure 2: DoS attack

There are two primary types of DoS attacks in OT systems:

- **DoS on the communication medium**: This type of attack focuses on disrupting the data flow between devices by flooding the network with excessive traffic. This leads to network congestion and delays or blocks legitimate communication.
- **DoS on the controller**: This attack targets the processing capabilities of the controllers, such as PLCs, by sending numerous commands. This overcharge of the PLC can lead to system failure rendering it unresponsive.

> Did you know that when the Apollo 11 mission landed on the moon, the navigation system was overloaded because of a faulty sensor [3], forcing Neil Armstrong to take manual control of the landing?
> This is an example of an unintentional DoS.

Security Measures

Several ways exist to protect against DoS. At this stage, the best is to avoid doing an action at the reception of a message. Unfortunately, it is not always possible. Another way is to limit the number of messages that can be received in a particular timespan. To achieve that, it is often necessary to use another device to filter the messages. This device can be a firewall for example. However, in the case of a DDoS, it is almost impossible to block the attack. The only way is to have a very powerful filtering device. Still, even with such a device, the attack can be successful.

3.1.4 Replay

A replay attack involves resending a previously intercepted message as if it were from the original sender [4]. As we can see in Figure 3 Mallory sniffs the message between Alice and Bob. Mallory can then send the message to Bob as if Alice had sent it. This is particularly relevant in OT environment with wireless communication, such as the typical example: replaying a command to open a garage door. The attacker sniffs the command and replays it to open the door.

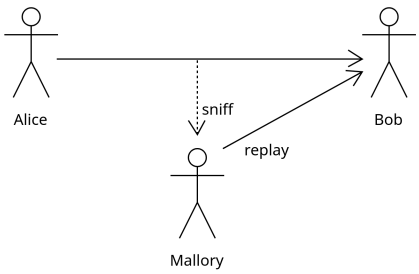


Figure 3: Replay attack

Security Measures

Two main ways exist to protect against replay attacks. It depends on whether the communication is broadcast or bidirectional.

When both devices communicate together in a bidirectional transmission, it is possible to add a timestamp to the message and sign the hash.

When communication is broadcast, the sender device is often not directly connected to other devices. In this case, it is not possible to have a timestamp. Using rolling codes is a good way to secure against replay attacks. The rolling code is a value that changes at each message. Both devices use a pseudo-random number generator. The receiver device can check whether it is a valid subsequent code.

3.1.5 Man in the Middle - Connected

A MitM attack occurs when a third party can actively intercept, modify or send packets on a network [5]. Usually, this involves connecting a new device to a star or bus network topology. Once connected to the network, Mallory (Figure 4) can perform a sniffing attack or send a message. The aim is often to intercept an message from Alice, modify it and send it to Bob.

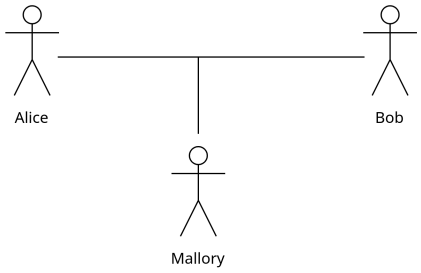


Figure 4: MitM on a connected network

Security Measures

Encryption with a symmetrical key can be used to protect the message from being intercepted, modified or impersonated. The key can be exchanged with the standard Diffie-Hellman algorithm.

3.1.6 Man in the Middle - Full interception

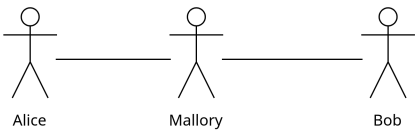


Figure 5: MitM intercept everything

When Mallory is on the gateway or between Alice and Bob, as in Figure 5, Mallory can intercept all messages and neither Alice nor Bob can be sure that they send and receive messages to the right person. This is the most dangerous attack because Mallory can impersonate Alice and Bob and send a message to the other person. Even with the security measures seen in Section 3.1.5, Mallory was able to impersonate Alice and Bob and create its own key with each other. This is why a MitM attack is so dangerous.

Security Measures

When an attacker can intercept all messages exchanged since the beginning of the communication, it is impossible to be completely sure that the messages are from the right person. We need to trust someone before. Certificates are made for this exact purpose. They are signed by a trusted third party and can be used to verify the identity of the person or device with which we are communicating. The most common type certificate is X.509 certificate.

3.2 Communication media

Different communication media are vulnerable to these attacks, highlighting the critical distinction between IT and OT security. In OT security, communication is a highly sensitive aspect, and historically, security measures were minimal or non-existent [6].

3.2.1 Modbus

Modbus is a communication protocol developed by Modicon in 1979 [7]. It involves a Modbus Master requesting data from a Modbus Slave. The client (master) sends a request to read from or write data to a server (slave). Modbus was originally designed for serial communication (called Modbus RTU). It has since been adapted for use over TCP/IP (called Modbus TCP).

Modbus RTU

Modbus with RTU is a serial, compacted, binary representation of the data. It is transported on the physical layer RS232 or RS485. Modbus RTU includes a Cyclic Redundancy Check (CRC)–16 bits checksum for error detection.

A frame is composed of:
- Address: 1 byte
- Function code: 1 byte
- Data: 0-252 bytes
- CRC: 2 bytes

Each byte of the frame is sent as 11 bits:
- 1 start bit
- 8 data bits
- 1 parity bit
- 1 stop bit

Modbus TCP

Modbus over TCP is a modern adaptation of Modbus. It is also a binary protocol, but transported over TCP. This adaptation eliminates the need for CRC due to inherent error detection in TCP. The rest of the frame is composed like Modbus RTU and the default port for Modbus TCP is 502.

3.2.2 wM-Bus

wM-Bus [8, part. 4] is a wireless version of the Meter-Bus (M-Bus) [8] protocol, used primarily in Europe for metering applications. It adheres to the ISO layer model [9] but implements only specific layers :

- Layer 1: Physical layer ([8, part. 2] for wired and [8, part. 4] for wireless)
- Layer 2: Data link layer ([8, part. 2] for wired and [8, part. 4] for wireless)
- Layer 7: Application layer ([8, part. 3])

The wireless specification has several modes of operation, to work on several frequency bands. The most common are:

Mode S

This mode [8, part. 4, p. 16] works on 868 MHz with 2-Level Frequency-Shift Keying (2FSK) modulation on a single channel. Meters send data many times a day to a stationary collector. The collector can be used in a power-saving function and is awakened up by the long heading of the frame. This mode has a one-directional (S1) or bidirectional (S2) sub-mode.

Mode T

This mode [8, part. 4, p. 19] works on 868 MHz with 2FSK modulation. The one-directional sub-mode T1 has a single-channel, but the bidirectional sub-mode T2 can use 2 channels. Meters are frequently sending data to a collector. This collector can be mobile.

Mode R2

This mode [8, part. 4, p. 24] works on 868 MHz with 2FSK modulation. This mode is bidirectional and has 10 channels. This mode can use frequency hopping for a duty cycle higher than with other modes.

Mode C

This mode [8, part. 4, p. 27] works on 868 MHz with 2FSK modulation. The one-directional sub-mode C1 has a single channel and the bidirectional sub-mode C2 has 2 channels with 2 different bandwidths. It can be used for stationary or mobile collectors.

Mode N

This mode [8, part. 4, p. 30] works on 169 MHz with 4-Level Gaussian Frequency-Shift Keying (4GFSK) modulation. It can be one- or bidirectional and has 13 channels. This mode is used for long-range communication with a stationary collector.

Mode F

This mode [8, part. 4, p. 35] work on 433 MHz with 2FSK modulation. This mode has only a bidirectional sub-mode. It is used for long-range communication with a stationary or mobile collector.

3.3 Simulation environments

As this thesis is part of the preparation of a new laboratory, the simulation environment must extend beyond abstract communication. The objective is to have a real physical controller interfaced with a simulated process. This simulated process remains necessary because a fully physical setup is prohibitively expensive and far less flexible than a simulated one.

3.3.1 Factory I/O

Factory I/O, developed by Real Games, is a realistic simulation software designed to emulate a factory environment. It allows for the creation of custom scenes, providing flexibility for various industrial scenarios. However, the software comes with an expensive licence, which must be considered when selecting the simulation environment. Factory I/O could also be beneficial for the Power & Control specialization. This software can interface with modbus over TCP, but an additional third-party software is required to implement a security layer. While Factory I/O is only available on Windows, it operates effectively on Linux using Wine.

Figure 6: Factory I/O palletizer scene

Scenario idea

In this scenario, a PLC could control the palletizer scene (). A wireless sensor could indicate the presence of a truck to be loaded. The wireless replay attack could target this sensor. The DoS attack could be executed on the same sensor. The MitM attack could be conducted on the Modbus/TCP communication between the PLC and the palletizer, with the objective of gaining control over the clamp.

3.3.2 Home I/O

Home I/O is also developed by Real Games and simulates a House () equipped with extensive home automation features. This software is available at a lower organizational licence cost and could be of interest to ETE students. Home I/O offers a REST API for interfacing with all sensors and actuators. Similar to Factory I/O, it runs well on Linux using Wine.

Figure 7: Home I/O scene

Scenario idea

In this scenario, a PLC manages the alarm and access systems, including the main door and garage. The garage door can be opened using a wireless remote, which could be the target of a wireless replay attack. An external presence detector on the main door could be used for the DoS attack. MitM attacks could be executed on the Modbus/TCP communication between the PLC and the alarm system, aiming to deactivate the alarm system.

3.3.3 Minecraft

Another suitable approach would be to use the Electrical Age world in Minecraft () which was previously featured in the Telecommunication course. The goal of this lab was to control a factory and energy system to maximize production. Continuing this laboratory could provide valuable opportunities for students to explore and secure communication systems. In Minecraft, security can be implemented using the Open Computers mod or by creating an extension mod for Modbus over TLS, which might be simpler than using Lua with Open Computers.

Figure 8: Minecraft Electrical Age scene

Scenario idea

The scenario involves reusing the PLC from the previous lab, which controls various operations. A physical Human-Machine Interface (HMI) could be constructed to manage the factory and coal production, similar to the HTML interface used in the previous lab. A wind wireless sensor could be added to the setup. Wireless replay and DoS attacks could target this sensor. MitM attacks could be executed on the Modbus/TCP communication between the PLC and the factory, with the goal of gaining control over the factory's operations.

3.4 Conclusion

This section presented various attacks that can be performed on OT systems. The communication media discussed during the preliminary phase of this work were also outlined. Additionally, potential simulation environments for laboratory use were evaluated.

Based on this information, M.Rieder and M. Clausen have decided on the simulation environment. The chosen platform is Home I/O, as it could benefit ETE students.

The planned attacks include a replay attack on a wireless control or sensor, a DoS attack on an external sensor with valid data (overloading the controller rather than the communication medium) and a MitM attack on the Modbus/TCP communication. The MitM attack will be conducted in two phases. The first phase will involve an unencrypted Modbus/TCP communication while the second phase will involve encrypted Modbus/TCP communication with a symmetrical key exchanged by Diffie-Hellman.

Wired communication will be carried out using Modbus, a widely used protocol in OT systems, which aligns with the brief for this thesis.

For the replay attack on a wireless control or sensor, the idea is to use the Flipper Zero to record and replay a message. This attack will be performed only on the physical layer. The Flipper Zero can only execute such an attack on basic wireless protocols, as it cannot perform a replay attack on frequency hopping protocols. The wireless protocol must employ On-Off Keying (OOK), Amplitude-Shift Keying (ASK) (with 270 or 650 kHz Bandwidth) or 2FSK modulation. Consequently, protocols like Zigbee or DigiMesh are not suitable for this attack and were not explored in depth.

wM-Bus is a single canal 2-Level Gaussian Frequency-Shift Keying (2-GFSK) protocol and is well-suited for replay attacks using the Flipper Zero. Given that typical wM-Bus T-mode application include electricity or water meters, incorporating such meters into the simulation would be relevant with a replay attack targeting these devices.
If the wM-Bus implementation proves too challenging, a contingency plan involves using a simple OOK modulation at 433 MHz.

4 Man in the Middle Scenario

The Man-in-the-Middle (MitM) scenario focuses on intercepting, modifying, and forwarding packets to gain control over a Modbus/TCP installation. This protocol, commonly used in industrial settings, was selected for this thesis because it is widely adopted and fulfils the requirement to demonstrate an attack on it. The MitM attack was chosen due to its prevalence and potential for significant impact. It is a comprehensive attack that encompasses several other techniques, such as sniffing and spoofing through ARP poisoning. This scenario assumes that the attacker has already gained access to the network, enabling them to intercept and manipulate the data packets.

Contents

4.1 Simulation Environment

Home I/O is a smart home simulation platform that allows interaction with a wide range of sensors and actuators, as detailed in Section 3.3.2. For the Man-in-the-Middle attack scenario, the entrance hall within this simulation is particularly relevant. As shown on Figure 9 the entrance hall has two doors with their sensor, one motion sensor and the house alarm.

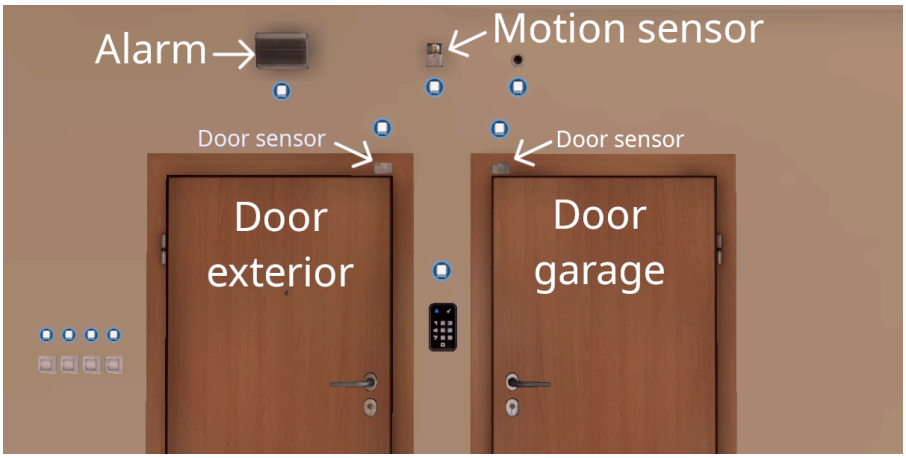


Figure 9: Home I/O scene

Since Home I/O does not natively support Modbus communication, a custom Go software was developed by Michael Clausen to convert Modbus requests into the REST requests required by the simulation. This software simulates a Modbus server for each room. In this thesis, the architecture consists of one computer running the simulation and another acting as the house controller, as shown in Figure 10. The controller, functioning as a Modbus client, is implemented using another Go program that could be deployed on a Wago CC100 PLC in a real-world scenario. For the purposes of this thesis, the controller is intentionally kept simple.

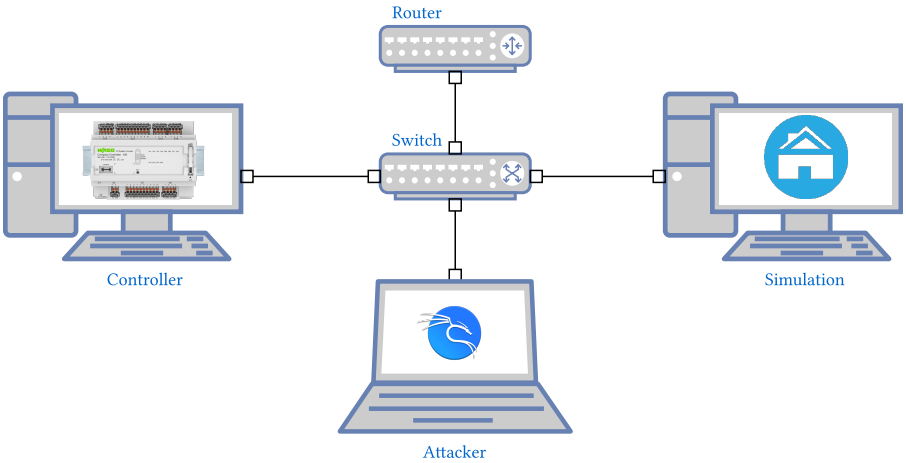


Figure 10: MitM scenario implementation

In this demonstration, the controller sends requests to check the status of both door sensors and the motion sensor. If either the door is open or motion is detected, the controller sends a request to the alarm to activate it. If not irregular activity is detected, the controller sends a request to deactivate the alarm.

The controller, the simulation and an attacker are all connected to the same network via a switch. This thesis uses a Kali Linux (Icon 1) laptop, although the software can be execute on any device with the appropriate tools.

4.2 Requirements

This scenario centres on a MitM attack, where the attacker must position themselves between the communication parties, ensuring that all packets pass through their system. Since Modbus/TCP is an IP-based protocol, the attacker can utilize a powerful tool called **Ettercap** (Icon 2). Ettercap simplifies the execution of an ARP poisoning attack.

An Address Resolution Protocol (ARP) poisoning attack involves sending fraudulentARP messages across the network, tricking the target into associating the IP address of another party with the attackers Media Access Control (MAC) address, as illustrated in Figure 11. As a result, the target's packets are sent to the attacker, who can them manipulate these packets before forwarding them (or not) to the intended recipient. This attack occurs at layer 3 of the OSI model [10].

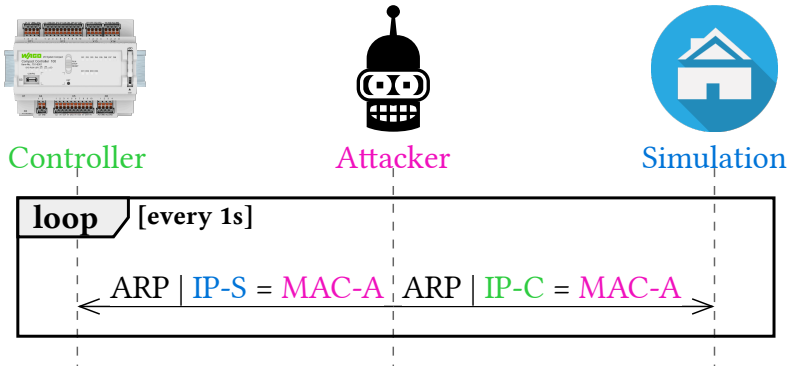


Figure 11: ARP poisoning

To use Ettercap, the attacker needs to know the IP address of the controller and the Home I/O simulation. This information can be obtained by sniffing the network using Ettercap itself or other tools such as **`Wireshark`**, **`nmap`**, or **`hping3`**. Once the IP addresses are identified, the attacker can initiate the ARP poisoning attack with the following command:

```
ettercap -T -i eth0 -M arp /IP_CONTROLLER// /IP_HOMEIO//
```

Listing 1: Start an ARP poisoning with Ettercap

This command, shown in Listing 1 starta Ettercap in text mode (**`-T`**) on the **`eth0`** network interface (**`-i eth0`**) and performs an ARP poisoning attack (**`-M arp`**). The IP addresses of the controller and the Home I/O simulation are specified by **`/IP_CONTROLLER//`** and **`/IP_HOMEIO//`**. A graphical user interface is also available but has to be installed as an extra package.

To execute this attack, **iptables** (Icon 3) will be used to redirect the packet to a python (Icon 4) script that modifies them in real-time. This script employs the **scapy** (Icon 5) library, a powerful tool for crafting, decoding and encoding packets from / to a wide range of protocols. The second part of this attack scenario (see Section 4.4) will use a **go** (Icon 6) script.

4.2.1 Tools

Here are all the tools that are used for this scenario :


Icon 1
Kali Linux


Icon 2
Ettercap


Icon 3
Iptables


Icon 4
Python


Icon 5
Scapy


Icon 6
Golang

Kali Linux (Icon 1) is an operating system for Pen-Testing.
Ettercap (Icon 2) is a suite tool to perform MitM attacks.
Iptables (Icon 3) is a firewall used to create redirection rules.
Python (Icon 4) is a programming language useful for scripting.
Scapy (Icon 5) is a packet manipulation Python package.
Golang (Icon 6) is a statically typed, compiled programming language.

4.2.2 Closer look on Modbus

As discussed in Section 3.2.1, Modbus is a straightforward protocol that was initially developed for serial communication via RTU but is now commonly implemented over TCP/IP. This protocol operates in request-response mode, as illustrated in Figure 12. In this setup, the controller sends a request to the Home I/O simulation to retrieve a value or alter an output. Home I/O then replies with the requested value or an acknowledgment message, after which the controller can issue another request or close the connection.

This request-response mechanism means that the attacker must capture traffic in both directions to modify packets in real-time effectively. The attacker needs to know the nature of the request to determine if the response needs alteration. Without this knowledge, it would be impossible to decide if a response needs to be modified.

In this thesis simulation, the controller is tasked with checking the status of two door sensors and a motion detector, all of which are connected to the same Modbus slave, the **Entrance Hall** (UnitID=5). The alarm system is represented as a coil on the same slave. A summary of these registers can be found in Table 1.

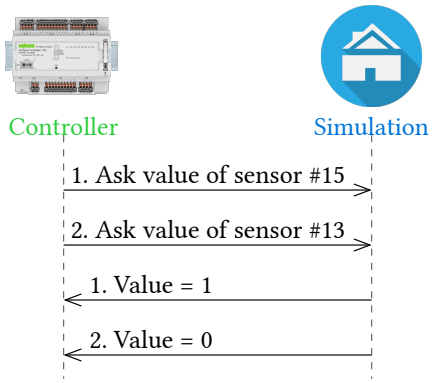


Figure 12: Home IO details

,

Sensor / Actuator	Unit ID	Function	Address
Door exterior	5	Discrete Inputs	13
Door garage	5	Discrete Inputs	14
Motion detector	5	Discrete Inputs	15
Alarm	5	Coils	5

Table 1: Modbus Registers

,

4.3 Attack on Modbus/TCP

To modify a packet during a Modbus/TCP attack, the first step is to establish a MitM position. This is achieved using ARP poisoning as described in Section 4.2. Once the attacker is able to intercepts all the packets, they need to redirect them to a **Python** (Icon 4) script for real-time modification. This redirection can be accomplished by configuring **iptables** (Icon 3) to add rules to the attacker's firewall. The idea is to place all the packets into a queue, enabling the Python script to retrieve and analyse them sequentially. Listing 2 demonstrates how to use **iptables** (Icon 3) to enqueue all packets adressed to **`192.168.0.0/16`** sub-network into queue **`1`**.

```
iptables -I OUTPUT -d 192.168.0.0/16 -j NFQUEUE --queue-num 1
```

Listing 2: Put packets on queue 1 with Iptables

4.3.1 Closer look on TCP

Transmission Control Protocol (TCP) operates at layer 4 of the OSI model [10] and is responsible for establishing a reliable connection between two devices. In this thesis, the primary concern is TCP's mechanism for ensuring packet integrity via checksums. The checksum ensures that the packet has not been corrupted during transmission; i.e. if the checksum does not match, the packet is discarded. This aspect is crucial for the attacker, who must modify the packet on the fly. If the modified packet has an incorrect checksum, it will be discarded, causing the attack to fail. An acknowledgment (ACK) is sent back when the packet is received correctly, as illustrated in Figure 13. Within the TCP segment, a Modbus packet is encapsulated as shown in Table 2.

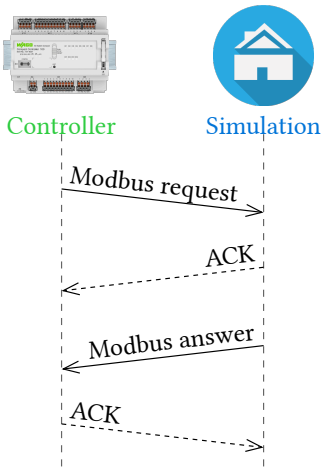


Figure 13: TCP exchange

0	1	2	3	4	5	6	7	8	9	10	...
Transaction ID		Protocol identifier		Length		Unit ID	Func. code	Reference number		Modbus data	

Table 2: Modbus TCP packet structure

4.3.2 Modify packet on the flys

Packet modifications are carried out using a **Python** (Icon 4) script, leveraging the **scapy** (Icon 5) library. Scapy is particularly useful for on-the-fly modification. With Scapy it is straightforward to dissect the different layers of a packet. To extract the IP layer, one can use the statement `scapy_packet = IP(pck.get_payload())`. To check if the packet is TCP and retrieve its payload, the function `payload = bytes(scapy_packet.payload.payload)` can be used.

This binary payload contains the Modbus message, which can be inspected and altered as needed. If the destination port is 1502, the packet is from the controller heading to the server. In this case, the attacker should check if the request concerns a door sensor or the motion sensor and then save the transaction ID of this request.

If the source port is 1502, the packet is from the server and is destined for the controller. The attacker should then verify whether the response corresponds to a previously saved transaction ID and, if so, modify the response as necessary.

4.3.3 Summary

The entire process is summarized in Figure 14.

(1) Initially, under normal conditions, the controller sends a request to the server, the server responds, and the controller triggers the alarm if necessary.

(2) However, when the attacker initiates an ARP poisoning attack, all packets are routed through the attacker.

(3) This allows the attacker to perform a MitM attack and modify the packets in real-time. In this scenario, the controller sends a request to the server, the attacker intercepts and alters the response, and the controller proceeds as if the modified response was legitimate.

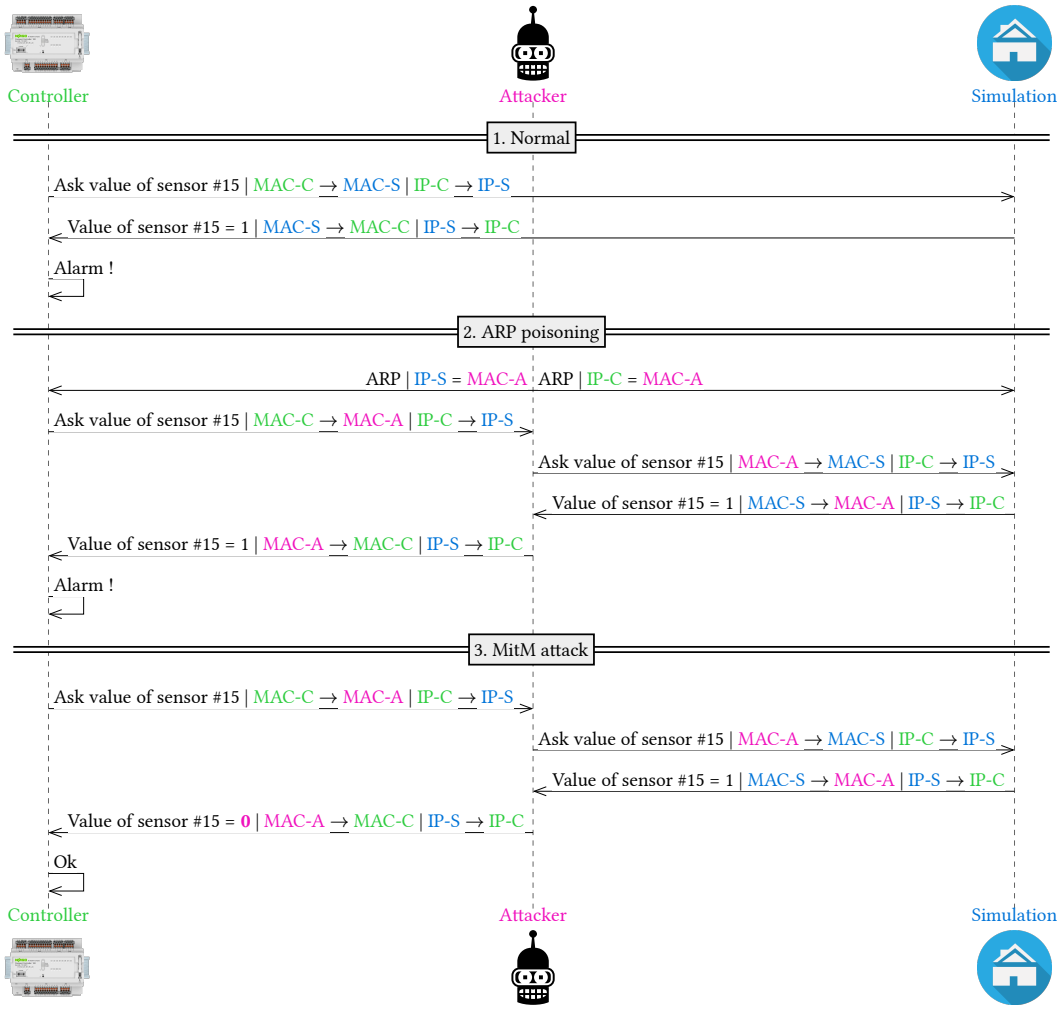


Figure 14: Modbus/TCP attack scenario

4.4 Implement Modbus/TLS

While clear communication can work, it falls short when it comes to security. To safeguard Modbus/TCP communication, it is essential to add an encryption layer. Modbus over TLS is a secure version of Modbus/TCP, encrypting the communication between the controller and the server. With Modbus/TLS in place, the attacker is unable to intercept and modify the packets in real-time, as they are protected by encryption.

4.4.1 Closer look on TLS

Transport Layer Security (TLS) is a cryptographic protocol that operates over TCP, primarily associated with layers 5 and 6 of the OSI model [10]. TLS is commonly considered a layer 6 protocol. Technically, it spans both layers 5 and 6. It establishes an encrypted session at layer 5 and handles the encryption itself at layer 6.

TLS provides secure communication between two parties, typically a client and a server. The process begins with a handshake, as illustrated in Figure 15, where the client and server exchange a series of messages to establish a secure connection. This handshake involves the exchange of random numbers, a certificate, and a seed. The Diffie-Hellman key exchange is used to generate a master key, a symmetric key that will be used to encrypt all communications within that session.

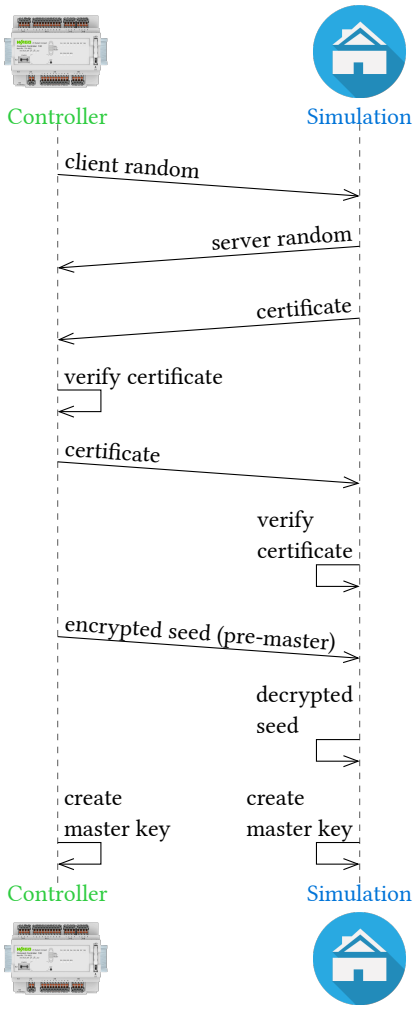


Figure 15: TLS handshake

During the handshake, the client, and server also exchange X.509 certificates to verify each other's identity. An X.509 certificate contains information about the certificate's owner, such as their name and expiration date. For server certificates, it also includes the server's IP address or domain name used for TLS communication. The certificate also includes the owner's public key, which is used to authenticate their message later on. X.509 certificates rely on a chain of trust provided by Certificate Authoritys (CAs). A CA is a trusted third party that verifies the identity of the certificate's owner and signs the certificate to validate it. The client and server can then verify the validity of each other's certificates by checking the CA's signature. If the signature is valid, the certificate is considered trustworthy, forming the basis of the TLS handshake.

4.5 Attack on Modbus/TLS

This thesis demonstrated how easily Modbus/TCP packets can be intercepted and modified. However, when Modbus/TLS is employed, the packets are encrypted, making it seemingly impossible to perform a MitM attack to take control of the system. This is only true if every step of the TLS implementation has been executed correctly.

The thesis also reveals that control can still be compromised if certificates are not properly verified. In test or debugging environments, it is common for certificates not to be signed by a CA, rendering them invalid. As a result, certificate verification is often bypassed in such environments. However, it is crucial to ensure that certificates are signed and verified in a production environment. In the OT world, it may be advisable for a company to maintain its own internal CA to sign all certificates. This makes it easier to install the CA on all company devices and ensure proper certificate verification.

To perform a MitM attack on a Modbus/TLS installation that does not check certificates, the attacker must first establish a MitM position. This can be done using the same ARP poisoning attack described in Section 4.2. During this thesis, considerable time was spent trying to modify the random values and certificates on the fly. This approach proved difficult because the TLS handshake (Figure 15) must be fully implemented, and there are no tools specifically designed to modify only the TLS layer in real-time. While many tools exist for performing MitM attacks in the IT world, such as **`Burp suite`**, **`mitmproxy`**, or **`bettercap`**, they are typically focused on the Hypertext Transfer Protocol Secure (HTTPS) protocol.

A more effective approach would be to handle the entire connection rather than attempting to modify packets on the fly. The attacker can redirect the TLS traffic from both targets to their own server, where the traffic can be decrypted. To redirect the traffic, **`iptables`** (Icon 3) can be used, as demonstrated in Listing 3.

```
iptables -t nat -A PREROUTING -p tcp --dport 5802 -j REDIRECT --to-port 5803
```

Listing 3: Redirect traffic to another port with Iptables

This command redirects all TCP traffic destined for port **`5802`** (used for TLS communication between the controller and Home I/O) to port **`5803`** (the port where the attacker's server is running).

The attacker can then set up their own Modbus/TLS server to decrypt the traffic and forward it to the Home I/O simulation. If desired, the attacker can modify the response before forwarding it back to the controller. It is easy to see a trace of such an attack because the attacker has to use a dummy certificate to create the symmetric encryption key. This certificate can be seen with tools like **`Wireshark`**

4.6 Conclusion

In summary, this chapter has outlined the simulation environment, detailed the tools required for this scenario, and demonstrated how to implement a MitM attack on both Modbus/TCP and Modbus/TLS communications.

The thesis focused on a simple system composed of two door sensors and a motion sensor controlling an alarm. The attacker's primary goal was to gain control over this system to open the doors without triggering the alarm. The findings illustrate how easily this can be achieved when no security measures are in place. The attacker only needs access to the local network, which could potentially be obtained by compromising an external sensor.

Additionally, this thesis has shown how implementing Modbus/TLS can secure the system and highlighted the critical importance of verifying certificates to ensure robust protection.

Finally, with this scenario, many layers of the OSI model have been tackled. Layer 2 and 3 with ARP poisoning, layer 4 with the attack on Modbus/TCP and layer 5 and 6 with the attack on Modbus/TLS.

5 Replay Scenario

The replay attack scenario involves intercepting and resending a message on a wireless connection to trigger the same effect as the original message, like, for example, a garage door opening remote. This scenario is particularly engaging because it can be easily implemented with the Flipper Zero device, making it more interactive and enjoyable for students to witness a physical attack in action. Additionally, it highlights the significance of wireless attacks, a critical topic in the OT world. This scenario operates at the physical layer, providing a complementary perspective to the MitM scenario discussed in Section 4.

Contents

5.1 Simulation Environement

The orginial idea was to do a replay attack on a wM-Bus system as shown on Figure 16. A real world situation could be a malicious person recording the data emitted by an electrical meter during two weeks of holidays. Then they could replay thoses weeks to reduce their electrical bill.

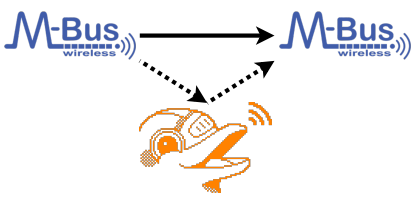


Figure 16: Replay scenario implementation

5.2 Requirements

This scenario focuses on a replay attack on a wM-Bus system, where understanding the content of the message is unnecessary. The attacker only needs to capture the physical layer of the message and retransmit it.

The Flipper Zero device (Icon 7) is used to execute this attack. The device features a “read raw” mode in the “sub-ghz” section, which allows it to listen to a specific frequency with a particular modulation and record the data at the physical layer (layer 1 of the OSI model [10]).

For transmitting messages on wM-Bus, a **RC1180** transceiver from Radiocraft is employed. This device can both transmit and receive messages via a serial connection. A Go (Icon 8) program can be used to send and receive messages through the RC1180.

5.2.1 Tools

Here are all the tools that are used for this scenario :

Icon 7
Flipper Zero

Icon 8
Golang

Flipper Zero (Icon 7) is a portable multi-tool for pentesters.
Golang (Icon 8) is a statically typed, compiled programming language.

5.2.2 Closer look on RC1180

The **RC1180** is a wM-Bus transceiver interfaced via a serial connection. To function correctly, this chip must be configured to the appropriate mode. In the context of this thesis, T-mode is utilized. The transceiver also needs to be set up as either a transmitter or a receiver. Configuration is achieved by sending specific serial commands to the device, which will respond with a confirmation message. The following commands are used for configuration :

- **`0x00`**: to enter configuration mode
- **`0x47`**: to change the wM-Bus mode
- **`0x01`**: to configure to T-mode
- **`0x46`**: to change the C-field of all futur messages
- **`0x44`**/**`0x04`**: to set master (**`0x44`**) or slave (**`0x04`**)
- **`0x00`**: to exit configuration mode

After each command, a confirmation message 0x3E is expected.

Once the configuration is complete, a message can be sent to the device to initiate transmission. The message format is shown in Table 3.

L	CI	Data

Table 3
RC1180 uart sending message format

L: Length of the message (1 byte)
CI: Control Information (1 byte)
Data: Data to send (L bytes)

For this thesis, the **`CI`** is set to **`0x80`** to send a general message, as this is a demonstration setup rather than a real wM-Bus meter.

5.3 Attack on Wireless M-Bus

With the setup complete, the goal is to perform a replay attack on the wM-Bus system using the Flipper Zero device. As a reminder, the wM-Bus in T-Mode operates at 868.95 MHz with 2GFSK modulation and a 50 kHz deviation.

5.3.1 Flipper Zero

For this thesis, the "sub-1GHz" feature of the Flipper Zero is utilized. The Flipper Zero contains a **`CC1101`** chip from **`Texas Instruments (TI)`**, designed to operate within the 300-348 MHz, 387-464 MHz, and 779-928 MHz frequency bands. It supports various modulation schemes, including 2(G)FSK, 4(G)FSK, ASK, and OOK.

In read-raw mode, the Flipper Zero needs to be configured to operate at 868.95 MHz for the wM-Bus system. The modulation is set to **`FM476`**, which corresponds to frequency modulation with a 47.6 kHz deviation. Based on the documentation, the Flipper should ideally detect the specific type of frequency modulation used by the signal.

5.3.2 Modulation FSK vs GFSK

Initial attempts at replaying the signal were unsatisfactory. While the Flipper Zero successfully detected and recorded a signal, the replayed signal was received by the RC1180 but not understood. This indicated that the Flipper Zero was not correctly recording or

replaying the signal. To diagnose the issue, the wireless signal was analysed using an Agilent spectrum analyser. Although the measurements were not conducted to obtain precise quantitative results, they provided a qualitative understanding of the signal.

A measurement of the original signal emitted by the wM-Bus transceiver is shown in Figure 17. The signal is centred around 868.95 MHz with a 50 kHz bandwidth and uses 2-Level Gaussian Frequency-Shift Keying (2-GFSK) modulation.

In contrast, the signal replayed by the Flipper Zero, shown in Figure 18, also centres around 868.95 MHz with a 47,6 kHz bandwidth, but the modulation appears to be 2-Level Frequency-Shift Keying (2FSK) rather than 2-GFSK. This difference in modulation explains why the RC1180 does not recognize the replayed signal.

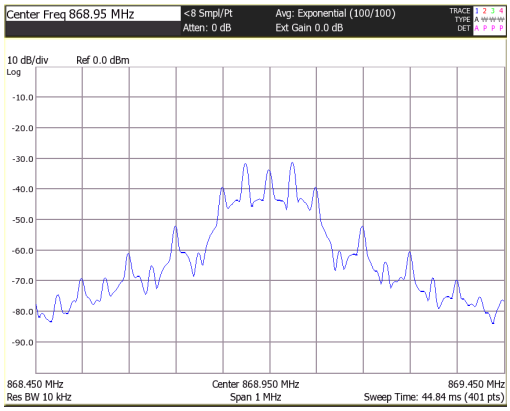


Figure 17
Spectrum analyzer of the original wM-Bus signal

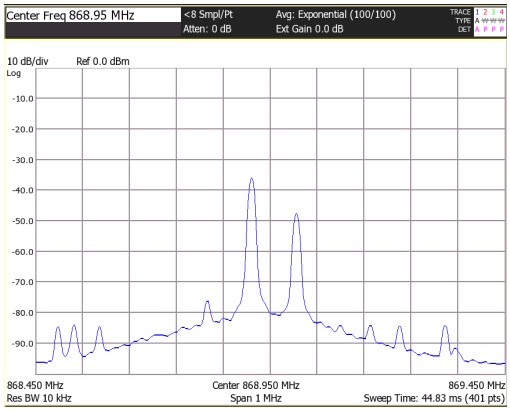


Figure 18
Spectrum analyzer of the replayed signal in FM476 modulation

The Flipper Zero seems unable to detect the correct type of frequency modulation. Upon closer examination of the firmware, it was discovered that the **`FM476`** preset is configured only for 2FSK modulation. While a brief effort was made to add a configuration setting for 2-GFSK with a 50 kHz deviation, the limited time remaining for the thesis made it impractical to proceed. Configuring this would require a detailed understanding of the CC1101 chip's register settings, a complex task requiring in-depth knowledge of the chip's datasheet.

Given these challenges, it was decided to proceed with Plan B.

5.4 Attack on basic 433 MHz transceiver

Instead of using a complete protocol like wM-Bus, a basic 433 MHz transceiver can be employed for the replay attack scenario. This approach is simpler and effectively simulates a garage door and its remote control. In this thesis, a basic OOK transceiver from **`M5Stack`** is used. The setup consists of two chips: a transmitter (**`SYN115`**) and a receiver (**`SYN513R`**). OOK modulation, the simplest form of ASK, represents digital data by the presence or absence of a carrier wave.

The transceiver emits a signal at 433.92 MHz when a signal is present on its data pin and nothing when the data pin is low. A challenge with the receiver is detecting a varying

signal; if no signal is detected, it amplifies the noise until it perceives sufficient variation to consider it as a signal. Similarly, if the signal is present for too long, the receiver reduces amplification, mistaking noise variations for the signal.

To overcome this issue and maintain simplicity, the strategy involves transmitting a serial message with a long preamble to stabilize the signal's amplification, followed by start characters. This enables the receiver to correctly detect both the signal and the message.

Given the simplicity of this modulation, the Flipper Zero device has no trouble reading and replaying the signal. It have to be configurate on the right frequency and an **`AM650`** modulation preset. The demonstration conducted in this thesis involves a simple message sent by the transmitter and received by the receiver. The Flipper Zero is used to record the message and replay it, allowing the receiver to detect the message and trigger the same action as the original signal.

5.5 Security in wireless broadcast isolated devices

A replay attack is always possible if two identical messages trigger the same action. To prevent such attacks, each message must be unique.

Typically, this is achieved by adding a timestamp to the message. However, in this scenario, it's not feasible due to the isolation between the receiver and transmitter, as well as the unidirectional nature of the communication. Without the ability to perform a challenge-response, alternative methods must be considered.

5.5.1 Closer look on rolling code

For this thesis, a rolling code was chosen as the security measure. Rolling code security involves appending a short, changing code to each message. This code is generated using a simple pseudo-random number generator. The receiver stores the last code received and only accepts one of the next codes in the sequence.

A pseudo-random number generator relies on a seed and a function that generates a new, seemingly random number. The seed is the last code received, and both the transmitter and receiver use the same function. The receiver must be able to compare with more than one subsequent code; otherwise, if a code is transmitted but not received, the receiver may become desynchronized and unable to decode future codes.

This method is straightforward and allows for easy addition of new remotes. To add a new remote, the receiver must enter learning mode, and the transmitter sends a code that the receiver adopts as the seed for the new remote.

This method is not perfect for security, as if the attacker knows how works the pseudo-random number generator, he can predict the next code. But it is enough for a garage door remote.

If an attacker understands how the pseudo-random number generator of this system works, they can predict future codes. While this method is not foolproof, it provides sufficient security for a garage door remote.

5.5.2 Closer look on signature

An alternative method involves signing the message with a private key, which the receiver then verifies using a public key. This approach offers greater security but requires more complex hardware and software capable of performing cryptographic operations. Additionally, adding a new remote becomes more complicated, as the receiver must be configured to recognize the new remote's public key.

To sign a message, a hash of the message is created and encrypted with the private key. The receiver decrypts the signature with the public key and compares the hash with the hash of the received message. If they match, the message is considered authentic. To prevent replay attacks, a simple counter can be included in the message, as it is also authenticated.

5.6 Conclusion

In summary, this chapter has presented the replay attack using the Flipper Zero. This attack is relatively simple, as the Flipper Zero has built-in functions designed to execute such attacks on unsecured systems. Initially, the plan was to conduct this attack on a wM-Bus system, but due to the Flipper Zero's lack of preset support for 2-GFSK modulation, the attack was instead performed on a basic 433 MHz transceiver.

Since a wireless replay attack targets layer 1 of the OSI model [10], the system must implement security at a higher layer. In this thesis, a rolling code was used to secure the system. While this method is simple and effective for a garage door remote, more secure systems may benefit from a message signature approach

6 Conclusion

This final chapter summarizes the work conducted in this thesis, addressing the challenges encountered and considering the future potential of this work, particularly in the context of laboratory use.

6.1 Project summary

In this thesis, we have explored various types of attacks, communication media, and simulation environments that could be employed in a laboratory setting. This analysis was crucial in selecting the appropriate attacks, mediums, and simulation environments essential for future laboratory exercises. A key requirement of this thesis was the inclusion of Modbus communication and the use of the Flipper device in the attack scenarios.

The Home I/O simulation environment was chosen for its relevance to ETE students. This environment was used to simulate the Modbus communication protocol, forming the basis of the MitM attack scenario, which was divided into two parts.

The first part involved an attack on Modbus/TCP communication, starting with an ARP poisoning attack to position the attacker in a MitM position. This part demonstrated how easily an attacker can intercept and modify unsecured communication, emphasizing the importance of securing such communication, specifically in this case, to prevent disabling the house alarm system. The scenario highlighted the necessity of using a secure communication protocol like TLS to prevent such attacks.

The second part focused on an attack on Modbus/TLS communication, again using ARP poisoning to establish a MitM position. This part illustrated the critical importance of verifying TLS certificates. Without proper verification, an attacker can replace certificates and establish their own encrypted communication with both the server and client. The scenario stressed the need for certificate verification, a practice that might seem obvious to those in the IT world but is a significant vulnerability in the IoT world, as highlighted by Aapo Oksman at DEF CON 31.

> Even if you have the best encryption, if you connect to the bad guys, it doesn't matter how well is your encrypted connection between you and the bad guys. They would still be able to see your connection. That's why you really need to authenticate the server first before establishing the encrypted connection.
>
> — Aapo Oksman on DEF CON 31

6.2 Encountered difficulties

Opting to work with minimal guidance posed its set of challenges. This approach led to significant time spent on problems that, in hindsight, might have been resolved more quickly with some assistance. However, by tackling with these issues independently, I gained a more profound understanding of the thesis subject.

The first major challenge was the lack of tools for performing MitM attacks below layer 7 (application layer). Most existing tools are designed for HTTPS and are not applicable to this thesis, where the layer 7 is Modbus for this thesis. I spent a considerable amount of time trying to perform a MitM attack on TLS on the fly. After much effort, I eventually realized that setting up a server-client bridge and providing a fake certificate was a better approach. Something that now seems obvious in hindsight. Ultimately, this is the solution I implemented.

The second significant difficulty was related to the differences in modulation between wM-Bus and the Flipper Zero device. It took me a long time to recognize that the issue was not due to poor transceiver configuration on my part. After extensive attempts to measure and analyse the radio frequency signal, I eventually realized that my misunderstanding of the Flipper Zero documentation was the root cause. I then spent additional time attempting to configure the TI chip in the Flipper Zero. Moving on to Plan B after investing so much time and effort in the initial approach was difficult.

These challenges meant that I did not have an opportunity to implement the DoS scenario. However, this was the least critical scenario, as it is relatively straightforward to understand. In exchange, the MitM scenario was developed in two parts, with one focusing on TCP and the other on TLS, which, I believe, are more interesting and relevant.

6.3 Future perspectives

Finally, what can be learned from this Bachelor's thesis, and how can it be useful in the creation of the course and the laboratory?

This thesis provides a comprehensive environment for conducting a MitM attack. For the laboratory, it would be beneficial to encapsulate the Home I/O simulation and its **Go** interface in a Docker container, making it more portable and easier for students to use. The controller could be implemented on the Wago CC100, with its functionality extended to cover the entire house for a more realistic scenario. Additionally, the Kali attacker could be deployed on a Raspberry Pi, demonstrating to students that even a small device can carry out significant attacks.

For the replay scenario, it could be interesting to connect it to the Wago CC100 to trigger the garage door in the Home I/O simulation, making the scenario more realistic and engaging for students. However, an interface between the 433 MHz transceiver and the Wago CC100 would need to be developed, along with a remote control for student use. This scenario remains more of a proof of concept, as it currently relies on serial communication with a **Go** software.

In any case, this thesis provides a guide for performing MitM and replay attacks. Parts of the thesis could be used as support material for the course. Additionally, this work led to the creation of Typst diagrams and, indirectly, Typst packages that could be useful for both course and laboratory support

7 Glossary

Flipper Zero: Flipper Zero is a portable multi-tool for pentesters and geeks in a toy-like body. 2, 4, 6, 14, 25, 26, 27, 28, 29, 30, 32

***HMI* – Human-Machine Interface**: A human-machine interface (HMI) is a user interface or dashboard that connects a person to a machine, system, or device. 14

***IoT* – Internet of Things**: The Internet of Things (IoT) is a system of interrelated computing devices, mechanical and digital machines provided with unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to computer interaction. 2, 3, 31

***IT* – Information Technology**: Information Technology (IT) is the use of computers to store, retrieve, transmit, and manipulate data or information. 2, 3, 8, 10, 23, 31

***OT* – Operational Technology**: Operational Technology (OT) is hardware and software that detects or causes a change through the direct monitoring and/or control of physical devices, processes, and events in the enterprise. 2, 3, 7, 8, 9, 10, 14, 23, 25

***Pen-Testing* – Penetration Testing**: Penetration testing, also known as pen testing, is a simulated cyber attack against your computer system to check for exploitable vulnerabilities. 18

***PLC* – Programmable Logic Controller**: A programmable logic controller (PLC) is an industrial digital computer that has been ruggedized and adapted for the control of manufacturing processes, such as assembly lines, robotic devices, or any activity that requires high-reliability control and ease of programming. 3, 8, 12, 13, 14

***SDG* – Sustainable Development Goal** 5

7.1 Attacks

***DDoS* – Distributed Denial of Service**: A distributed DoS is basically the same as a DoS attack, but the attack comes from multiple sources. 8

***DoS* – Denial of Service**: A denial-of-service (DoS) attack is a cyber-attack in which the perpetrator seeks to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of a host connected to the Internet. 6, 8, 12, 13, 14, 32, 33

***MitM* – Man in the Middle**: A Man in the Middle (MitM) attack is a form of eavesdropping in which communication between two users is monitored and modified by an unauthorized party. 6, 2, 9, 10, 12, 13, 14, 15, 16, 17, 18, 19, 21, 23, 24, 25, 31, 32

7.2 Communications

***2FSK* – 2-Level Frequency-Shift Keying**: 2-Level Frequency-Shift Keying (2-FSK) is a form of Frequency-Shift Keying (FSK) modulation that uses two levels of frequency to encode digital data. 11, 14, 28

***2-GFSK* – 2-Level Gaussian Frequency-Shift Keying**: 2-Level Gaussian Frequency-Shift Keying (2-GFSK) is a form of FSK modulation that uses two levels of Gaussian filtering to encode digital data. 14, 28, 30

***4GFSK* – 4-Level Gaussian Frequency-Shift Keying**: 4-Level Gaussian Frequency-Shift Keying (4-GFSK) is a form of FSK modulation that uses four levels of Gaussian filtering to encode digital data. 11

***ARP* – Address Resolution Protocol**: The Address Resolution Protocol (ARP) is a communication protocol used for discovering the link layer address, such as a MAC address, associated with a given internet layer address, typically an IPv4 address. 6, 15, 17, 19, 21, 23, 31

***ASK* – Amplitude-Shift Keying**: Amplitude-shift keying (ASK) is a form of modulation in which the amplitude of a carrier wave is varied proportionally to that of a modulating signal. 14, 28, 34

***CRC* – Cyclic Redundancy Check** 10

***FSK* – Frequency-Shift Keying**: Frequency-shift keying (FSK) is a frequency modulation scheme in which digital information is transmitted through discrete frequency changes of a carrier wave. 33, 34

***HTTPS* – Hypertext Transfer Protocol Secure** 23, 32

***IP* – Internet Protocol** 10, 17, 18, 20

***MAC* – Media Access Control**: A media access control address (MAC address) is a unique identifier assigned to a network interface controller (NIC) for use as a network address in communications within a network segment. 17

***M-Bus* – Meter-Bus** 11

***OOK* – On-Off Keying**: On-Off Keying (OOK) denotes the simplest form of ASK modulation that represents digital data as the presence or absence of a carrier wave. 14, 28

***RTU* – Remote Terminal Unit** 10, 11, 18

***TCP* – Transmission Control Protocol** 2, 10, 11, 12, 13, 14, 15, 17, 18, 19, 20, 22, 23, 24, 31, 32

***wM-Bus* – Wireless M-Bus** 5, 6, 6, 11, 14, 26, 27, 28, 30, 32

7.3 Cryptography

***CA* – Certificate Authority**: A certificate authority (CA) is an entity that issues digital certificates. 22, 23

Diffie-Hellman: Diffie-Hellman key exchange is a method of securely exchanging cryptographic keys over a public channel. 9, 14, 22

***TLS* – Transport Layer Security**: Transport Layer Security (TLS) is a cryptographic protocol that provides communications security over a computer network. 2, 13, 22, 23, 24, 31, 32

X.509: X.509 is a standard that defines the format of public key certificates. 10, 22

Bibliography

[1] "Sniffing Attack." Jun. 04, 2023. Accessed: Jun. 24, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Sniffing_attack&oldid=1158437808

[2] "Denial-of-Service Attack." May 27, 2024. Accessed: May 28, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Denial-of-service_attack&oldid=1225934024#DDoS_extortion

[3] "Lunar - Missions - Apollo 11 Mission." Accessed: Jun. 24, 2024. [Online]. Available: https://www.lpi.usra.edu/lunar/missions/apollo/apollo_11/

[4] "Replay Attack." Jan. 04, 2024. Accessed: May 28, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Replay_attack&oldid=1193515559

[5] "Man-in-the-Middle Attack." Jun. 20, 2024. Accessed: Jun. 25, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Man-in-the-middle_attack&oldid=1230046662

[6] Fahmida Y. Rashid, "The Old Ways Aren't Working: Let's Rethink OT Security," Nov. 2021, [Online]. Available: https://www.darkreading.com/cyber-risk/the-old-ways-aren-t-working-let-s-rethink-ot-security

[7] "Modbus." May 23, 2024. Accessed: May 27, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Modbus&oldid=1225269451

[8] "EN 13757 - Communication Systems for Meters." Swiss Association for Standardization (SNV).

[9] "ISO/IEC 7498-1:1994 - Information Technology – Open Systems Interconnection – Basic Reference Model: The Basic Model." Swiss Association for Standardization (SNV).

[10] "ISO/IEC 7498-1:1994." Accessed: May 29, 2024. [Online]. Available: https://connect.snv.ch/en/isoiec-7498-1-1994