Report for the CPS Final Project

Andrea Auletta
andrea.auletta@studenti.unipd.it
Niccolò Zenaro
niccolo.zenaro@studenti.unipd.it

February 16, 2025

Contents

1 Introduction and objectives **3**
- 1.1 Modbus 3
- 1.2 Attack identification 4
- 1.3 Objectives 4

2 System setup **5**

3 Experiments **5**

4 Results and Discussion **5**

Abstract

Modbus is a commonly used protocol in Supervisory Control And Data Acquisition (*SCADA*) environments for monitoring, control and data acquisition. Despite its wide popularity, Modbus is not secure because when it was developed and adopted (1979) security was not considered to be a concern in isolated Industrial Control Systems (*ICS*), thus is not designed to be secure like modern IT networks. Among the various attacks, 4 different taxonomies can be identified to facilitate formal risk analysis efforts for clarifying the nature and the scope of the security threats on Modbus systems and networks.

1 Introduction and objectives

1.1 Modbus

The Modicon Commuication Bus (*Modbus*) protocolo operates in a master-slave or server-client based model. The master devices initiates the queries while the slave devices respond to all such queries. Masters can either send a broadcast message to all the slaves or indivdually poll a specific device. All the experiments run in this work, like in the original paper [1] are focused on TCP/IP implementation, while Modbus protocol can be implemented also on top of several communication networks like serial or UDP.

Modbus TCP messages are wrapped in TCP/IP header and transmitted over an ethernet-based Modbus network between different devices. The *Modbus Slaves* listen for incoming TCP connections on port 503, that has been selected by us, and once a connection has been established the Protocol Data Unit (*PDUs*) are exchanged and encapsulated in TCP messages. The protocol itself can be broken down into six sections:

- Transaction Identifier: a 2-byte field that is used to correlate request and responses; it is easily predictable due to poor randomization.
- Protocol identifier: a 2-byte field that for Modbus is always set to 0.
- Length: a 2-byte field that indicates the length of remaining bytes in the payload.
- Unit Identifier: 1 byte that is used to identify the specific slave at an IP address.

- Function Code: is a 1-byte field that indicates the action requested my master. For example, reading and writing coils, registers or holding registers.

- Data: this field has variable length, with values associated with the various function codes.

1.2 Attack identification

In general attacks on Modbus systems and networks can exploit protocol's specifications, i.e. they are common to all Modbus systems/networks that are conform to the protocol specifications. The attack identification methodology used by the authors [1] involves an analysis of each protocol, that leads to four groups or threat categories: *interception*, *interruption*, *modification*, *fabrication*. The main targets for the Modbus protocol are the master, the field device, the serial communication links and messages. Attacks were implemented based on the possibility of the system to have a Modbus sniffer and a packet injector, that also could block, modify or fabricate arbitrary Modbus messages or sequences of messages.

Fifteen attacks that exploit TCP protocols have been recognized, and as said require access to the master device, network communication path or field device. The most serious attacks are those that disable or bypass the master unit and seize control of field devices, this type of attacks affect the integrity and the availability of the messages or of the network; on the other hand attacks on confidentiality involve obtaining information on the network or on slave devices by simply reading messages.

1.3 Objectives

In this work we tried to emulate the various attacks defined by the authors, precisely we emulate one attack for each taxonomy identified in the paper. We built our own Modbus simulator with python libraries and then we produced python scripts for each attack. More precisely, *interception*, *interruption*, *modification* and *fabrication* identified in the original paper are described and implemented as:

- *Passive reconnaissance* involves passively reading Modbus messages or network traffic, intercepting the messages and reading field device data.

- *TCP FIN flood* is an interruption attack that aims to launch spoofed TCP packets with the FIN flag set after a legitimate message from Modbus server to Modbus client, in order to close the TCP connection or cause important delays.
- *Rogue Interloper* is a sort of man-in-the-middle attack where a MITM device can sniff and fabricate messages.
- *Response Delay* involves delaying a response message so that the master receives out-of-date information from slaves, and is done sniffing and modifying field device, sending the modified packet with a delay.

2 System setup

This work was entirely done with python scripts, crafting master and slaves with `PyModbus` and working on TCP level with `Scapy`. Although these are two of the most famous python labraries for dealing with TCP Modbus packets, we found out that their documentation is often incomplete and not accurate, with most of the informations difficult to retrieve.

3 Experiments

4 Results and Discussion

References

[1] Peter Huitsing et al. "Attack taxonomies for the Modbus protocols". In: *International Journal of Critical Infrastructure Protection* 1 (2008), pp. 37–44.