Good to Bad: When Industrial Protocol Translation Goes Wrong

Marco Balduzzi[1], Charles Perine[1], Philippe Lin[1], Ryan Flores[1], Rainer Vosseler[1], and Luca Bongiorni[2]

[1] Trend Micro Inc.
[2] Independent Researcher

Introduction to Protocol Gateways

Protocol gateways are embedded devices used in industrial installations to facilitate the communication between production units like control servers, PLCs or machinery; and for the integration of IT and OT networks. These gateways translate ICS protocols – e.g. Modbus, Profibus or BACnet – to enable legacy devices like on serial buses to communicate and interface with modern TCP/IP networks. For example, in a typical Modbus installation, a gateway translates the requests originating from a control server located in a TCP/IP control network (and acting as a master node) to a PLC on RS232 (i.e., a slave node).

Given the importance of protocol gateways in the operation of modern industrial networks, we conducted a security evaluation aimed at understanding how industrial protocols are translated, and at discovering potential risks of abuse. We considered 5 protocol gateway products from well-known, established vendors and observed similar classes of problems across the different vendors. In addition, although our evaluation focused on protocol translation, we also encountered a series of related problems e.g. with authentication and reliability that can facilitate attacks like sabotage or information leakage.

Approach to Security Research

In our evaluation, we adopted a black-box approach in which we compared the traffic generated by a fuzzer (inbound to the tested gateway) with the traffic translated by the gateway (outbound). This strategy was dictated by the fact that the gateways we considered do not publicly disclose information on their design nor implementation. Of course, this approach complicates the analysis because debugging and logs access are not always straightforward.

We made use of an automated system that we designed and implemented to successfully test all gateways under the same conditions, and to cover the largest number of corner cases of the protocol specifications. Our system consists of the following macro components:

- A fuzzer that generates the inbound traffic for the gateway under test. For example, when testing the translation from Modbus TCP to Modbus RTU, the fuzzer generates Modbus TCP test cases.
- A simulator that simulates the receiving unit, e.g. a PLC implementing a Modbus RTU slave. The simulator is needed because protocol gateways may operate incorrectly (or not operating) if not connected to a unit.

– A sniffer that collects information on the outbound traffic (i.e., the translated protocol); an analyzer that collects both inbound and outbound traffic for the analysis. A report is automatically produced for the security analyst.

Findings

We first evaluated the gateways' capabilities in filtering malformed packets[3]. We generated 5,078 and 1,659 invalid Modbus TCP and RTU packets, sent them to the tested devices and calculated their drop rate (i.e., number of discarded packets). The gateways behaved differently with this respect.

We observed that one device implemented poor filtering. In fact, 2,454 packets that were voluntarily constructed to violate the message length specifications were *not* translated, but forwarded *as they are* – i.e, without removing the Modbus TCP header and adding the Modbus RTU checksum. As a result, in a TCP to RTU communication these packets are interpreted with a different semantic by the receiving serial unit. For example, an attacker would be able to trigger a malicious *write multiple coils* request by mean of an innocuous *read input registers* packet[4] – ref. Figure. This issue enables a malicious user to conduct targeted, very difficult to detect attacks e.g. bypassing common protection mechanisms like ICS firewalls and tampering with the production.

Modbus TCP	Transaction ID		ID	Length	Unit ID	Funct Code	Address	Registers
Packet	01	**0F**	0000	*0011*	03	**04**	D1CE	0070
Modbus RTU	Slave ID	Funct Code	Address	Coils	Bytes	Data		CRC

Thanks to the automated analysis that we conducted with our system, we confirmed a number of other translation problems that we summarize here:

– One gateway exposes an out-of-bound vulnerability that allows a malicious user to tamper with the gateway's internal routing table (known as I/O mapping table). For example, to trigger a write register request (like raising the alarm threshold of a temperature sensor from 200 to 2,000°C) by mean of a less evident write coil message (e.g. turn on the air conditioning fan);
– One gateway incorrectly handles read requests having a number of coils to be read equal to zero. These packets trigger a remote reboot (DoS);
– Two gateways incorrectly implement the encryption in the translation from OT to the cloud, permitting a malicious user to access confidential information, e.g. sensor data from a production plant;
– One gateway exposes a local privilege escalation vulnerability allowing a remote user to dump the device's configuration, and potentially implant a Trojan (reprogram the gateway).

In this research, we explored the problem of protocol translation in industrial networks. Our findings have been communicated to the affected parties via the ZDI[5] responsible disclosure program. To the best of our knowledge, no previous research has been conducted so far in this direction.

[3] Not complying with the protocol specifications
[4] Note the wrong message length field's value (0x11 instead of 0x08) and how the function code changes with the semantic
[5] https://www.zerodayinitiative.com/