Computers & Security 124 (2023) 103007

Contents lists available at ScienceDirect

Computers & Security

journal homepage: www.elsevier.com/locate/cose

Detection and mitigation of field flooding attacks on oil and gas critical infrastructure communication

Abubakar Sadiq Mohammed*, Eirini Anthi, Omer Rana, Neetesh Saxena, Pete Burnap

School of Computer Science, Cardiff University, UK

ARTICLE INFO

Article history:
Received 31 July 2022
Revised 4 October 2022
Accepted 6 November 2022
Available online 9 November 2022

Keywords:
ModbusTCP
Cybersecurity
Operational technology
Cyber-physical systems
Machine learning

ABSTRACT

Industrial Cyber-Physical Systems (ICPS) are highly dependent on Supervisory Control and Data Acquisition (SCADA) for process monitoring and control. Such SCADA systems are known to communicate using various insecure protocols such as Modbus, DNP3, and Open Platform Communication (OPC) Data Access standards (providing access to real-time automation data), which are vulnerable to a range of attacks. This leads to increased cyber risks faced in critical infrastructures, especially in the Oil and Gas sector. One of the most popular and critical attacks deployed against such infrastructure is Denial of Service (DoS), as it can have severe consequences that range from financial loss to loss of life. Such attacks can disrupt the ability of an operator to control hazardous operations leading to potentially unsafe scenarios. A novel Field Flooding attack is described which takes advantage of the packet memory structure of the Modbus protocol to perform a DoS attack. This attack can cause overflowing of the memory bank allocated in the Programmable Logic Controller (PLC) for Modbus operations. The attack is deployed and evaluated on a real industrial testbed and its impact against the Mitre ATT&CK framework is assessed, in order to identify which tactics an adversary could use to compromise the system. A novel mechanism that utilises supervised machine learning to detect this attack in industrial control system networks is also described. Experimental results show that the proposed mechanism, using the XGBoost algorithm, can identify this attack with 99% accuracy.

1. Introduction

The Modbus protocol and its variants are the most widely used communications protocols in the oil and gas (OG) industry especially for pipeline operations (Huitsing et al., 2008) and for monitoring remote offshore operations. The protocol was extended to allow control messages to be transported over TCP (He et al., 2019), creating the ModbusTCP variant. This hastened the wide adoption by the OG industry as communication could be integrated seamlessly within existing systems. Similar to other industrial protocols like DNP3 and OPC DA, the ModbusTCP protocol is insecure, lacking authentication or encryption, which makes it susceptible to cyber attacks (e.g. Man-in-the-Middle, Denial of Service, command injection, etc). The nature of OG operations, especially offshore production, requires remote monitoring of the production of highly volatile hydrocarbons from subsea to surface. This requirement, together with the ease of deployment of ModbusTCP to transmit sensor readings and actuator states has increased the widespread use of the protocol in the OG industry, and as a result increased the attack surface of the Operational Technology (OT) being deployed.

Consequently, there has been an increase in the number of cyber attacks carried out on critical infrastructure using Supervisory Control and Data Acquisition (SCADA) systems in general, and even more so in the OG industry. The Colonial pipeline cyber attack in May 2021 and the more recent incident in February 2022 where three European oil transport and storage companies, namely Oiltanking in Germany, SEA-Invest in Belgium and Evos in Netherlands (Tidy, 2022), were targeted. Recent studies have also shown that theft of operational information and Denial of Service (DoS) are the most frequent impacts of documented cybersecurity incidents in the OG industry (Mohammed et al., 2022).

These incidents have led to a corresponding increase in security research focused on OT and critical infrastructure communications. However, due to the high cost of OT equipment, most research is carried out in simulated environments which may not represent exact OT system behaviour during cyber attacks. Consequently, not much is known about attack impact across different industrial environments. Would the same attack behave differ-

* Corresponding author.
E-mail addresses: mohammedas@cardiff.ac.uk (A.S. Mohammed), anthies@cardiff.ac.uk (E. Anthi), ranaof@cardiff.ac.uk (O. Rana), saxenaN4@cardiff.ac.uk (N. Saxena), burnapp@cardiff.ac.uk (P. Burnap).

https://doi.org/10.1016/j.cose.2022.103007
0167-4048/

ently in a different industrial environment? These factors have motivated the study presented in this paper with a focus on (1) the ModbusTCP protocol, (2) Denial of Service attacks, and (3) implementation of attacks on different real industrial systems to analyse behaviour/response to attacks.

More specifically, we present a novel Field Flooding attack which alters the structure of the ModbusTCP packet with additional malicious fields to target the PLC controlling critical processes. The attack involves sniffing network packets (Man-in-the-Middle) for ModbusTCP communications and injecting the malicious packets to the PLC to cause a denial of service. The Field Flooding attack is unique from most Man-in-the-Middle (MitM) and DoS attacks studied in the literature in the following ways:

- Does not require ARP poisoning as an initial step so would not be mitigated with standard measures capable of detecting ARP poisoning - a typical defence against MitM attacks.
- Does not increase the rate of packet transmission to the PLC (e.g. SYN Flood - a popular type of DoS attack widely studied). Rather, with much fewer, carefully crafted packets, can overwhelm the PLC which could prevent response to requests. This results in a behaviour that requires a different approach for detection/mitigation besides known measures (e.g. packet rate limiting).

Our main contributions are:

1. The identification of a novel “Field Flooding” attack on the ModbusTCP protocol which can lead to a severe Denial of Service (DoS) attack;
2. A novel Intrusion Detection System to effectively detect the Field Flooding attack on industrial control networks using a supervised machine learning approach; and
3. A labelled dataset collected from three industrial testbeds containing benign and malicious activity that can further support security research surrounding attack detection on ICS systems.

The remainder of this paper is structured as follows: Section 2 discusses the background and related work in this research area. Section 3 describes the attack methodology, attacker model, and tools used including the testbeds utilised in the study. In Section 4 the results of the experiments are provided, while Section 5 analyses these results in more detail. In Section 6, supervised machine learning techniques are applied to detect the Field Flooding attack, and the performance of these techniques is evaluated. Key lessons learnt and a summary is included in Section 7.

2. Background and related work

In this section both the context of the proposed work and related literature are described. An overview of the ModbusTCP protocol is provided, followed by vulnerabilities in this protocol.

2.1. Structure of the ModbusTCP protocol

The ModbusTCP protocol communicates using a simple request/ reply mechanism between a control centre and field devices (Huitsing et al., 2008). The control centre(s) are the clients (formerly called ‘Master’), while the field devices are the servers (formerly called ‘Slaves’). This variant of the Modbus protocol uses TCP/IP as a transport mechanism for Modbus messages. There are four data storage modes in Modbus servers to store analog and digital input/output (I/O) which are highlighted in Table 1.

A function code (FC) included in a Modbus message describes the purpose of the message (Gonzalez and Papa, 2007). Table 2 describes the most used public FCs by vendors while Fig. 1 shows the basic structure and size allocated to each header. The Modbus Application Data Unit (ADU) has a total size of 260 bytes. This is shared by the Modbus Application (MBAP) header and the Protocol Data Unit (PDU) in the order of 7 bytes and 253 bytes respectively. The fields in the MBAP header are explained as follows:

Table 1
Modbus addressing format for data storage.

I/O Range	Description
00,001–10,000	Read/Write discrete output or coils
10,001–20,000	Read discrete inputs
30,001–40,000	Read input registers (16-bit registers for analog inputs)
40,001–50,000	Read/Write holding registers (16-bit storage)

Table 2
Most used public Modbus function codes.

Function	Code	Hex	Type	Size (Bits)
Read Discrete Inputs	2	0x02	Read Only	1
Read Coils	1	0x01	Read/Write	1
Write Single Coil	5	0x05	Read/Write	1
Write Multiple Coils	15	0x0F	Read/Write	1
Read Input Registers	4	0x04	Read Only	16
Write Single Register	6	0x06	Read/Write	16
Read Holding Registers	3	0x03	Read/Write	16
Write Multiple Registers	16	0x10	Read/Write	16

- **Transaction ID:** This is a number that matches the Modbus server [Programmable Logic Controller (PLC)] response to its corresponding query from the Modbus client [Human Machine Interface (HMI)] and is incremented by one for consecutive queries.
- **Protocol ID:** This is usually set to “0” to indicate ModbusTCP protocol.
- **Length:** The length field indicates the size of the data (in bytes) in the rest of the packet (i.e. size of Unit ID, Function Code, and Data fields) so the receiving party knows what to expect from the packet.
- **Unit ID:** This is set to the Unit ID of the Modbus server the client wishes to communicate with. For the ModbusTCP protocol, the Unit ID is not relevant as the IP address of the server dictates the destination of the packet.
- **Function Code:** The function code identifies the action the Modbus server should take.
- **Data:** The Data field contains the data to write/ read and the address of the data store on the Modbus server.

The client-server query-response cycle Queries from Modbus clients (e.g. HMI) and the corresponding response from Modbus servers (e.g PLCs) are sent in loops that are milliseconds apart. The query from the client contains the FC that tells the server what action to perform (RS, 2002). The “Data” field contains the address information that should be read or written to and specifies how many addresses to consider.

The corresponding response from the Modbus server (e.g. PLC) is usually an echo of the FC in the query (RS, 2002), unless an error occurs. The data returned by the server indicates process status (in the case of a read request) or confirmation of data written (in the case of a write request). The packet structure of read and write queries/responses is shown in Fig. 2.

2.2. Related work

Vulnerabilities in the Modbus protocol have been widely considered, primarily due to lack of authentication and ease of deployment of this protocol. This section focuses on presenting relevant work that focuses on: (a) vulnerabilities reported in the ModbusTCP protocol – these studies have been carried out mostly on simulated testbeds, and (b) studies focusing on Intrusion Detection Systems (IDS) for the ModbusTCP protocol.

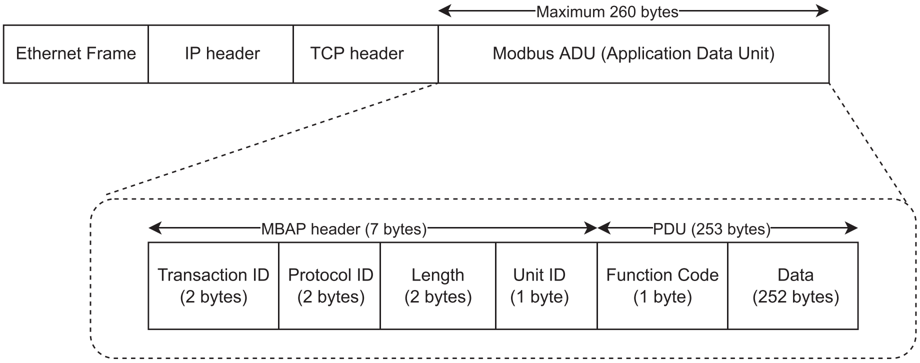


Fig. 1. ModbusTCP packet structure.

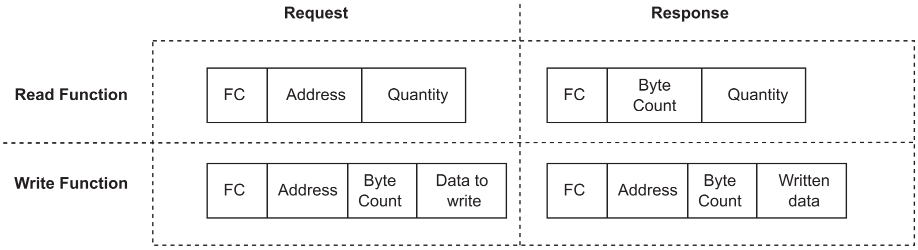


Fig. 2. ModbusTCP message structure for memory access operations.

ModbusTCP vulnerabilities Chattha et al. (2021) presented an implementation of cyber-physical systems with ModbusTCP communication for real-time security testing. Their study used two simulated case studies (i.e. Automatic Voltage Regulation and DC motor position control) using MATLAB Simulink, OpenPLC, and ScadaBR to understand the effects of attacks launched on the system. The authors of Luswata et al. (2018) used a penetration testing approach to identify attacks on SCADA systems specifically focusing on the ModbusTCP protocol. Their study combined three simulation tools (i.e. Qmod master, Modbuspal and, Conpot server) that were utilised for attacking the ModbusTCP protocol and developing countermeasures. Similarly, Parian et al. (2020) carried out two attacks on the ModbusTCP protocol comprising of a MitM and malware attacks where the latter involved modifying requests made by the Modbus client, ensuring that the response from the server is reversed. They utilised Scapy (tool discussed further in Section 3.1) to manipulate the Modbus server response by changing the value of the requested coil. Our attack approach in this study however utilises Scapy differently to alter the ModbusTCP packet structure rather than change the value of the Modbus command/response. Their experimental setup was based on virtualisation technology, with the client, server, and attacker machines all hosted within Virtual Machines. However, a key limitation of the aforementioned studies is that they are all based on simulated environments which do not fully reflect real system usage (Satyanarayana et al., 2021). These studies, therefore, did not consider attacks that alter the ModbusTCP packet structure and did not evaluate the impact of the attacks on a physical industrial testbed.

Furthermore, Bashendy et al. (2020) presented a formal attack tree for representative explored attacks against the ModbusTCP protocol that models the attack steps in detail with different attributes. They categorised the attacks using the CIA triad (Confidentiality, Integrity, and Availability) where various modifications of the packets are made. Modifications included changing the FC to an unsupported one, injecting a replayed payload, or changing a specific value in the payload (Bashendy et al., 2020). Similarly, in Stranahan et al. (2019), the authors also highlight the vulnerability of the ModbusTCP protocol to malicious attacks using standard attack tools utilised in penetration testing. The attacks carried out in their study which impacted the system were limited to data manipulation (writing coils), MitM, and DoS. These studies, however, did not consider attack vectors dealing with protocol mutation by altering the ModbusTCP packet structure. Finally, Alcaraz et al. (2019) explored security issues related to covert channels applied to ModbusTCP in industrial networks using a testbed comprising of various equipment including a Raspberry Pi 3 board simulating the logic of a PLC. They presented two approaches based on (1) timing - where insignificant delays are injected in the TCP/IP channels, and (2) storage - by the inclusion of hidden data in specific fields of the ModbusTCP packets. While the attacks presented in these studies leverage on manipulating the values in various fields (e.g. Unit ID, FC, Data) being transmitted or stored in some way using the ModbusTCP protocol, they all work within the existing structure of the ModbusTCP packet. In our Field Flooding attack, the ModbusTCP packet structure itself is manipulated, compromising the controller (PLC/RTU), resulting in adverse behaviour outside the intended response as designed.

Intrusion detection systems (IDS) for modbusTCP Radoglou Grammatikis et al. (2020b) developed a novel anomaly-based IDS called ARIES which adopted a set of machine learning (ML) methods, consisting of three detection layers: (a) network flow-based detection, (b) packet-based detection, and (c) operational data-based detection. Particularly, the second layer of their model inspects ModbusTCP packets and their attributes to detect anomalies such as unauthorised ModbusTCP commands and function code enumeration attacks. Specifically, they used real datasets originating from a power plant in Greece containing operational data which was used to detect anomalies. Their proposed method is suitable for a specific domain (i.e. power plant) and not for general industrial usecase. Satyanarayana et al. (2021) also examined the vulnerability of ModbusTCP to false command injection, false access injection, and replay attacks. Their proposed IDS involved using a frame filtering module that will send only authorized commands and Modbus re-

Table 3
Summary of related work. *FF* = *Field Flooding*, Hybrid testbed = ◐.

Author / Reference	Simulation	Physical industrial testbed	Multiple vendor hardware	Alter ModbusTCP packet structure	Can detect FF attack
Chattha et al. (2021)	•				
Luswata et al. (2018)	•				
Parian et al. (2020)	•				
Bashendy et al. (2020)		•			
Stranahan et al. (2019)		•			
Alcaraz et al. (2019)	◐				
Radoglou Grammatikis et al. (2020b)		•			
Satyanarayana et al. (2021)		•			
Saharkhizan et al. (2020)	◐				
Katulić et al. (2022)		•			
Morris et al. (2013)		•			
This study		•	•	•	•

quests to the PLC by checking the IP address and port of the Modbus client, allowed function codes, and allowed register addresses. Furthermore, Saharkhizan et al. (2020) designed an IDS using Deep Learning (DL) long short-term memory (LSTM) modules into an ensemble of detectors which was trained and evaluated on a simulated Modbus network traffic dataset. The dataset was categorised into MitM attacks, ping DDoS (Distributed Denial of Service) flood attacks, Modbus query flood attacks, and TCP SYN DDoS flood attacks which are mostly “high-rate” attacks. These attacks typically work by sending a series of packets to a target device at a hyper-increased rate, exhausting the capacity for a timely response, if any. The authors focused on detecting mostly “high-rate” attacks, which can be easier to detect based on the high packet flow. However, they have not evaluated their system against attacks that may be more sophisticated and disguised like the one presented herein (i.e. Field Flooding attack). Therefore, there is no evidence that the proposed IDS could be utilised for detecting such attacks. Also, the attacks used in Saharkhizan et al. (2020) did not alter the ModbusTCP packet structure.

Finally, the authors in Morris et al. (2013) and Katulić et al. (2022) describe a comprehensive set of rules that could be combined with popular signature-based IDS (e.g. Snort, Suricata) to prevent exploitation of the Modbus protocol. In Katulić et al. (2022), the authors carried out DoS (SYN Flood), MitM (spoofing), and reconnaissance attacks on a cyber-physical system via ModbusTCP and created custom rules focusing on the Modbus data field, which is plant-specific. A limitation of their work is that these rules would not apply to any other industrial network and is therefore not an adaptable solution. The advantage of an ML-based IDS over this system is its adaptability (ability to learn features of multiple industrial environments) and that it could detect a wider variety of attacks. Both studies - Morris et al. (2013) and Katulić et al. (2022) - examined rules that preserve the integrity of the Modbus packet, but did not consider manipulation attacks where malicious fields are appended to the packet while the parameters within each field remain valid. Also, deploying these rules to adequately protect OT networks requires an in-depth knowledge of various thresholds and set points. Since each OT network has its own unique parameters, thresholds that adequately protect one network may not work as efficiently on another. This solution is not scalable or adaptable across several industrial networks. To summarise, these studies did not consider attacks that abuse the memory allocation of the PLC while preserving the integrity of the Modbus frame. Subsequently, the field flooding attack described in this paper demonstrates the ability to bypass these preventive techniques by ensuring that the malicious packet is coming from an authorised IP address/port and probing using legitimate function codes and allowed register addresses. Table 3 summarises the studies discussed in this sub-section.

3. Attacking the ModbusTCP protocol

3.1. Attacker model and capabilities

The attacks presented in this paper consider the following basic assumptions to form the attacker model. OT networks can often include remote access for vendors to maintain their systems remotely. An attacker could perform a phishing attack against a supplier or an integrator/ vendor’s remote access link to the OT network (Assante and Lee, 2015). In order to effectively troubleshoot, upgrade or modify system parameters (e.g. PLC logic, proprietary software, hardware configuration files, firmware updates, etc.) during scheduled or emergency maintenance activities, vendors would require administrative privileges on the remote workstations they connect into. This is usually the case, especially in oil and gas offshore platforms located thousands of miles away from shore. It is assumed that our attacker has gained access to the OT network and has the following capabilities: (i) network sniffing; (ii) command injection through scripting; (iii) modification of operational parameters. These capabilities will be further mapped out using the Mitre ATT&CK framework in Section 5. The attacker’s objectives/ motivation are:

- To compromise an operator’s ability to control processes on the remote system (i.e. impair process control).
- Collect information about operational processes including sensor readings and process state.
- Disrupt a process to damage equipment (potentially leading to loss of life and damage to the environment).

The tools used in these attacks are:

1. **Smod**: `Smod` is the most widely known pen-testing tool related to ModbusTCP (Radoglou-Grammatikis et al., 2020a). It aggregates a set of diagnostic and offensive features that can be used in pen-testing the ModbusTCP protocol.
2. **Scapy:** `Scapy` is an interactive packet manipulation program written in Python. It is capable of forging or decoding packets for a wide number of protocols, sending them on the wire, capturing them, matching requests and replies, and much more. Rohith et al. (2018).
3. **Wireshark:** `Wireshark` is a widely-used network protocol analyzer (Combs, 2022).
4. **Tshark:** `Tshark` is the terminal version of `Wireshark`
5. **Nmap:** A widely used network discovery tool.

3.2. Description of attacks

As discussed in 3.1, the attack scenario assumes the attacker has gained entry into the OT network by gaining user credentials

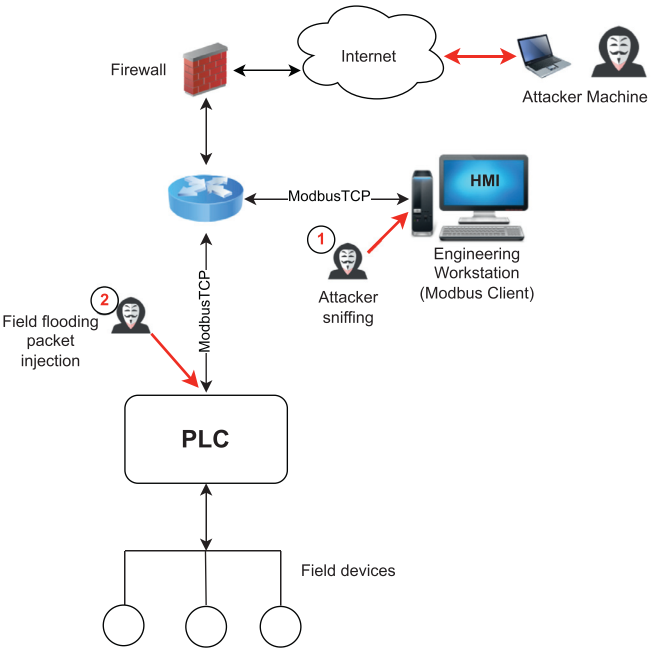


Fig. 3. Attacker's target points within the OT network; 1 = Field Flooding step 1, 2 = Field Flooding step 2.

of a third-party vendor from spear phishing activities, then using the stolen credentials to access a dedicated workstation with privileges to carry out maintenance activities. The workstation is running an HMI that constantly polls the PLC for process state and displays the status for the operator in real-time. The attacker's targets are highlighted in Fig. 3

In the initial phase of the attack, the attacker used Nmap and Smod tools to carry out reconnaissance of the network. Nmap was used to discover devices with port 502 (default port used by ModbusTCP protocol) open, while Smod was used to scan the PLC for allowed function codes. Due to recent attacks, most vendors no longer allow diagnostic function codes to be sent to PLCs as they can easily be used to discover details about the system, shut it down, or force it into a limited service mode (e.g. Force listen-only mode). The Smod enumeration on Modbus function codes confirmed that diagnostic function codes are disabled by the vendor on the PLCs. However, all the public FCs shown in Table 2 were accessible for exploitation.

The next phase of the attack involved using Scapy to sniff network traffic between the HMI and PLC which was analysed with Wireshark. ModbusTCP communication between HMI and PLC is usually in a continuous loop. The communication loops in the case of the experimental setups used in this study are described as follows (testbeds are described in detail in 3.3):

- **Testbed 1:** one query (to read 2 holding register addresses), its corresponding response (from Modbus server - PLC 1), and finally an acknowledgement (ACK) from HMI - 3 packets.
- **Testbed 2:** two queries (HMI polling PLC for data/status) and two responses (PLC sending requested data/status to HMI). Each query (from HMI) is followed by a corresponding response (from PLC) and an acknowledgement of receipt of data by the PLC - 6 packets.
- **Testbed 3:** one query (to read 1 coil address), its corresponding response (from Modbus server - PLC 3), and an acknowledgement (ACK) from HMI - 3 packets.

In all experiments, the critical metric was the communication time, which was approximately 7 ms (milliseconds) between a query-response-ack loop, and 100 ms between loops. This gave an initial indication of when malicious packets can be injected into the stream as shown in Fig. 4. The longer the communication time, the easier it is for Scapy to craft a packet and inject. From the Wireshark analysis, the time window most favourable for a successful packet injection was the 100 ms between PLC acknowledgement for receiving holding register data and HMI requesting input register data in the case of testbed 2. For both testbeds 1 and 3, the packet injection window was after the ACK of the loop, but before the next query from the HMI which also was approximately 100 ms. To craft a packet that will be accepted by the PLC, it needs to:

Table 4
Common industry use-cases for the 3 PLCs used in our experiments.

PLC	Common industry use-case
PLC 1	Oil and gas industry
PLC 2	Manufacturing, smart buildings, general automation
PLC 3	Smart grid, manufacturing

- conform with the ModbusTCP standard format (contain function code, transaction and protocol identifiers, unit ID, length and register starting address);
- utilise sequence (SEQ) and ACK numbers in the previous packet (ACK packet transmitted from HMI) to use as its own SEQ and ACK numbers.

Secondly, in order to generate a malicious packet targeting the PLC with a Field Flooding attack, the following techniques were used: (i) alteration of the length field in the MBAP header; (ii) alteration of the number of fields in the PDU header. Recall that the maximum memory allocated for ModbusTCP ADU header is 260 bytes. By altering the length field in the MBAP header and increasing the number of fields in the PDU layer, this limit is exceeded which can potentially disrupt the communication between the HMI and PLC. The following experiments were carried out with varying parameters:

- Create ModbusTCP read packet (FC 01/03/04) similar to communication loop packets and inject (packet replay attack).
- Modify ModbusTCP write packet (FC 05/15/06/16) with increased length field in MBAP header and inject (altered length attack).
- Modify ModbusTCP write packet (FC 05/15/06/16) with 1 additional field (2bytes) in PDU layer and inject (Field Flooding attack).
- Modify ModbusTCP write packet (FC 05/15/06/16) with 2 additional fields (4bytes) in PDU layer and inject (Field Flooding attack).

A summary of the Field Flooding attack sequence steps and corresponding stages on the cyber kill chain is shown in Fig. 5.

3.3. Experimental setup

To carry out these experiments, three testbeds with relevant hardware from real industrial network communications in critical infrastructure were used. Three different testbeds were used in order to evaluate and investigate the impact of the attack on different industrial environments. The main features of the testbeds, such as the PLCs (acting as Modbus servers) and their common industry use cases are listed in Table 4. The PLC brands have been concealed for security reasons.

Testbed 1 (oil and gas) This testbed emulates a gas wellhead production monitoring system using compressed air flowing through the pipes. The PLC (PLC 1) is commonly deployed in oil and gas platforms because of its numerous control functions and ability

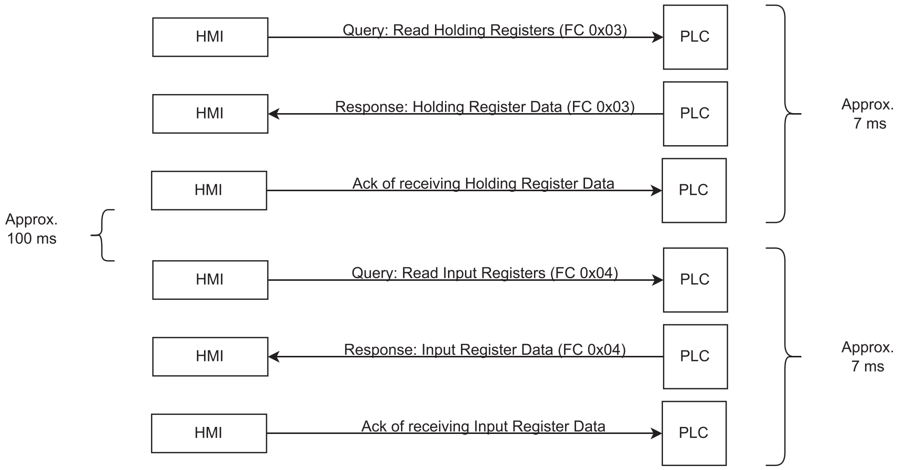

HMI
Query: Read Holding Registers (FC 0x03)
PLC
HMI
Response: Holding Register Data (FC 0x03)
PLC
HMI
Ack of receiving Holding Register Data
PLC
Approx.
7 ms
Approx.
100 ms
HMI
Query: Read Input Registers (FC 0x04)
PLC
HMI
Response: Input Register Data (FC 0x04)
PLC
HMI
Ack of receiving Input Register Data
PLC
Approx.
7 ms

Fig. 4. PLC-HMI Communication loop showing timings in milli-seconds.

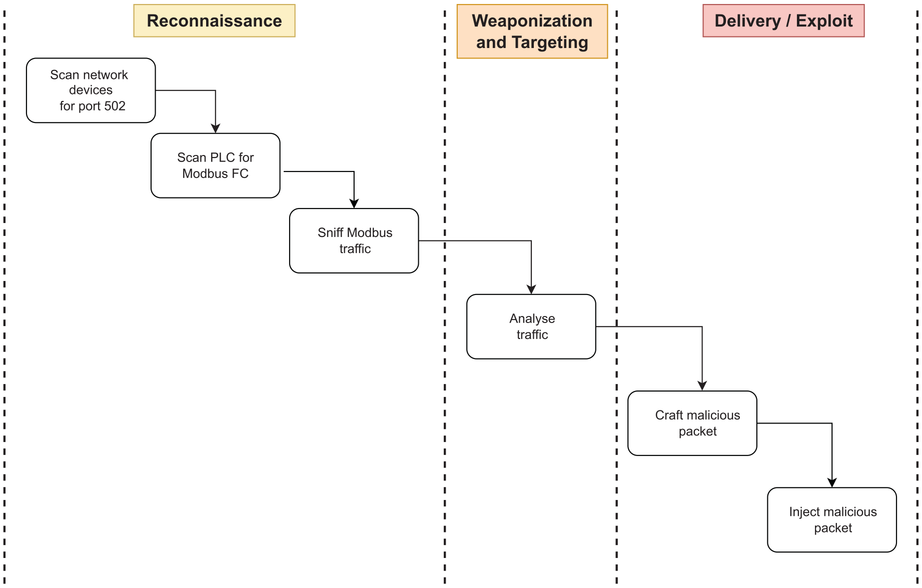
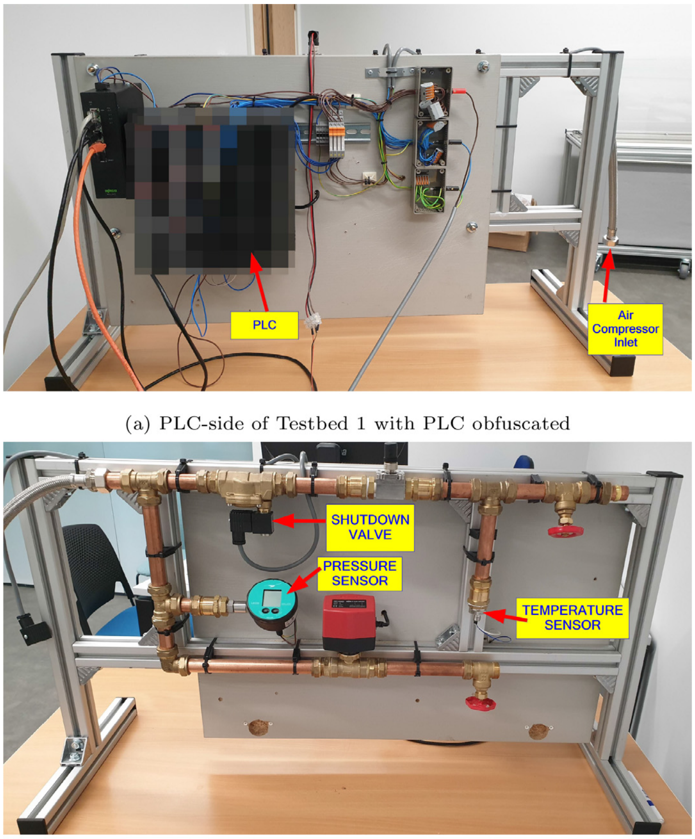


Fig. 5. Field Flooding attack sequence and phases on the cyber kill chain. FC = Function Code.

to withstand operations in harsh environments like offshore platforms. An air compressor is connected to the pipe inlet (Fig. 6(a)) which pumps compressed air through the system. Monitoring equipment (shown in Fig. 6(b)) includes pressure and temperature sensors and a shutdown valve. The values of the sensor readings are stored in the PLC holding register addresses 40,099 and 40,199. To control the testbed and monitor sensor values, an HMI software, AdvancedHMI (2022) provides a Graphical User Interface (GUI) which accesses the stored values in the PLC holding registers and displays the sensor readings (i.e. pressure and temperature). The HMI was programmed to periodically poll the PLC for data representing sensor readings stored in the holding registers using the function code 0x03 (read holding registers). All communication is via ModbusTCP. There is also a shutdown valve to provide the operator ability to shut off airflow emulating an emergency shutdown scenario. This can be controlled via the HMI “on/off” buttons using FC 0x06 (write holding register).

Testbed 2 (manufacturing) This testbed represents a simple setup that monitors the temperature and humidity readings of an assembly line to ensure the quality of production. The setup was provided by the National Digital Exploitation Centre (NDEC) and included an encrypted VPN (Virtual Private Network) tunnel to access the testbed remotely. This was to emulate a remote workstation monitoring system process. The hardware comprises a PLC, a temperature sensor, and a humidity sensor. The sensors are hardwired to the PLC, which communicates the values in real-time to the HMI (Fig. 7(a)) using ModbusTCP. For demonstration purposes, both sensors are only reading the temperature and humidity of the room where the testbed is located. These sensor readings are constantly polled and displayed in real-time on the HMI – a feature that allows the operator to keep track of production quality. The temperature value is stored in a holding register while the humidity value is stored in an input register. The HMI periodically polls the PLC for the temperature and humidity values using the Modbus function codes 0x03 (read holding registers) and 0x04 (read input registers) respectively. The testbed setup is shown in Fig. 7(b).

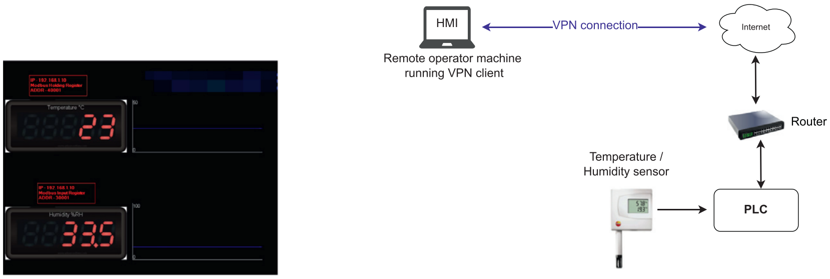
Testbed 3 (smart city) This is a SCADA testbed consisting of two critical infrastructure systems (a) smart city buildings and (b) a train system looping around the city. These two systems are controlled separately by two different PLCs. Our study focused on the PLC controlling the smart city buildings. Within the building models (shown in Fig. 8), there are LED (Light Emitting Diode) lights wired to connect each building to a power source, provided by the PLC. When energised, all the buildings are powered up and



(a) PLC-side of Testbed 1 with PLC obfuscated

(b) Sensor-side of Testbed 1

Fig. 6. Setup of testbed 1 showing sensors, valves and setup arrangement.



(a) HMI used to poll PLC for sensor readings

(b) Setup of Testbed 2

Fig. 7. Setup of testbed 2 showing remote operator access and HMI used (security details obfuscated).

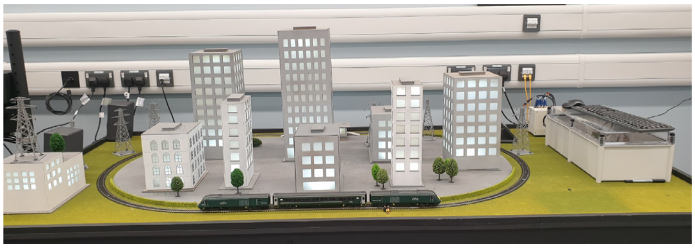


Fig. 8. Complete setup of Testbed 3.

illuminated. This is controlled by binary coil values stored in the PLC indicating status as “on” or “off” – indicating when lights in the building can be turned on/off. Auxiliary power lines are included on the surface of the testbed as an aesthetic feature. The HMI tracks the power status of the smart city buildings by accessing the values stored at coil address 0001 using FC 0x01 (read coil) and gives the operator the ability to turn on the power, or power down (using FC 0x05 - write single coil) for maintenance activities.

In the next section, the results of these attacks on all three testbeds are described.

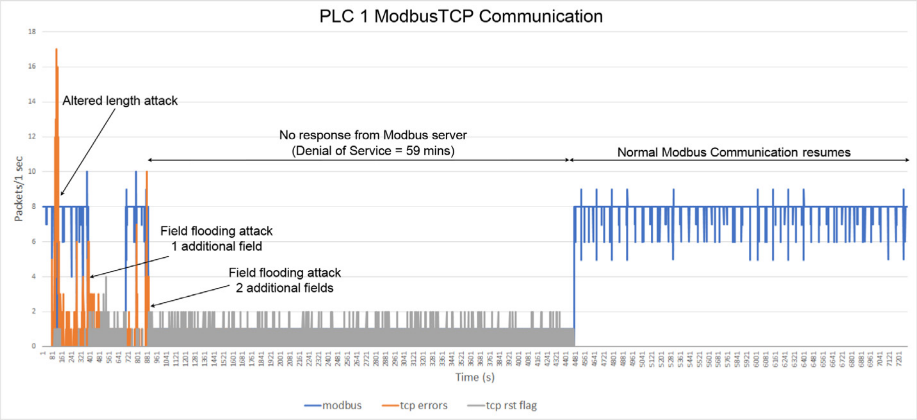


Fig. 9. Disruption of ModbusTCP communication from field flooding attack on Testbed 1.

4. Results

Malicious packets were successfully injected into the ModbusTCP communication for all the testbeds with each packet altered according to the experiments listed in 3.2. In each testbed, sniffing network traffic and injecting the exact same ModbusTCP read packets in the communication loop resulted in the corruption of the TCP session, however, the TCP protocol session management was able to self-correct with minimal disruption - approximately 1 s (i.e. spurious retransmissions were discovered and the TCP three-way handshake was re-initiated to re-establish communications).

The next type of malicious packets that were injected was the altered length field in the MBAP header. Again, for all three PLCs in the various testbeds, the results were similar. The injected packet with an increased length field corrupted the TCP session and the session management self-corrected the communication. However, in this case, the communication loop was restored after a RST ACK packet which triggered the re-initiation of the TCP three-way handshake. This also took approximately 1 s to correct and all 3 PLCs handled this error adequately. It's also worthy to note that rule No. 3 in Morris et al. (2013) will effectively block this attack. The aim of this experiment was to establish a baseline for the PLCs' error handling capabilities.

Finally, malicious packets with additional fields (field flooding attack) to the PDU header were injected and all three PLCs behaved differently in handling this attack. Each malicious Field Flooding Packet injected (disguised as a Modbus client query) triggered an initial response to the sent query from all three PLCs, which confirmed a successful packet injection and enabled a continuation of the attack (maximum of 4 packets injected) until the PLC is unable to respond to further legitimate requests for varying periods. The impact on each testbed is further described as follows:

Field Flood Attack on Testbed 1: Two types of malicious ModbusTCP packets with additional fields in the PDU header were injected to cause a field flood attack on PLC 1. The first packet was injected with only 1 additional field (2 bytes) while the second packet had 2 additional fields (4 bytes). The first packet (additional 2 bytes) caused a denial of service for up to 5 min where the PLC (modbus server) did not respond to queries from the HMI (modbus client). The second field flood attack (2 additional fields - 4 bytes) had a more damaging impact on the modbus server as the PLC was continuously responding to queries from HMI with RST ACK packets in an attempt to reset the TCP session. The field flooding attack effectively forced the PLC into a listen-only mode for approximately 59 min leading to a denial of service. This is shown in Fig. 9.

Field Flood Attack on Testbed 2: The field flooding attack also showed adverse behaviour on PLC 2. Although the injected packet with only 1 additional field in the PDU header resulted in a corruption of the TCP session for 9 s, when repeated with a malicious field flooding packet containing 2 additional fields, it resulted in a denial of service. The additional 4 bytes appended to the PDU header made the PLC non-responsive to HMI queries by sending RST ACK packets for approximately 7 min (Fig. 10).

Field Flood Attack on Testbed 3: For PLC 3 (smart city testbed), the field flooding attack was also carried out by injecting malicious ModbusTCP packets with 1 additional field and 2 additional fields. The field flood attack with 1 additional field to the PDU header corrupted the TCP session for about 20 s, while that of 2 additional fields forced the PLC to restart as shown in Fig. 11. This also caused a denial of service scenario as, during the period of the restart, the PLC would no longer be responsive to commands or report process state.

The summary of all the attacks carried out on the testbeds and their corresponding impact on the behaviour of the PLCs is shown in Table 5.

5. Analysis of field flooding attack impact

From the results shown in Section 4, it can be deduced that different PLCs behave uniquely to the field flooding attack. This is another advantage that real systems have over simulated environments as this difference in PLC behaviour cannot be accounted for in simulated experiments. Our experiments show that PLC 1, which is predominantly used in the oil and gas industry, is the most vulnerable to the field flooding attack in comparison to PLCs 2 and 3. This could potentially have serious implications on process safety in such a volatile, critical industry. For example, in oil and gas production platforms, where SCADA is used to control the heating and separation of volatile hydrocarbons, operators monitor and ensure safe operations via HMI equipped with override functions for emergency shutdowns. This attack has the potential to impair process control leading to pipeline explosions, loss of lives and damage to the environment.

One of the dangers of the field flooding attack is that a low-skilled adversary can execute this attack and cause huge damage. Its relative ease of execution can be demonstrated by mapping the attack pattern on the Mitre ATT&CK for ICS framework. This framework is a curated knowledge base for cyber adversary behavior in the ICS technology domain (Alexander et al., 2020). It comprises a taxonomy that describes adversarial tactics and techniques.

Using the Mitre ATT&CK for ICS framework the field flooding attack was mapped to show the tactics and techniques utilised by

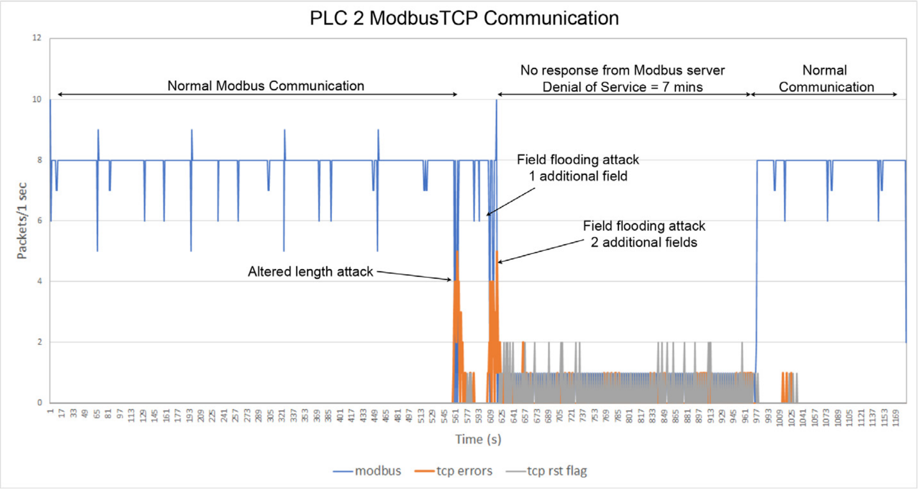


Fig. 10. Disruption of ModbusTCP communication from field flooding attack on Testbed 2.

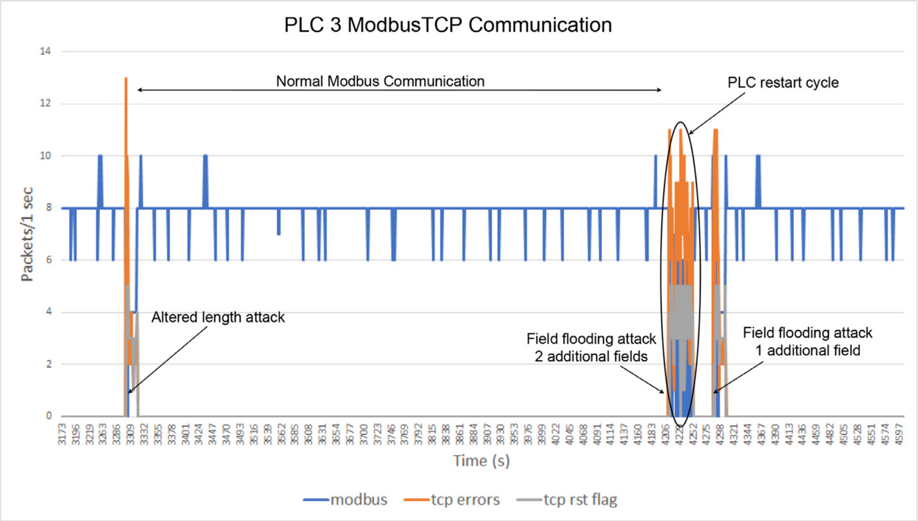


Fig. 11. Disruption of ModbusTCP communication from field flooding attack on Testbed 3.

Table 5
Summary of impact of attacks carried out on all three testbeds.

		Attack impact		
Testbed	PLC/RTU	Altered length attack	Field flooding attack (1 field)	Field flooding attack (2 fields)
1	PLC 1	Spurious retransmissions (1 s)	Denial of service (5 min)	Denial of service (59 min)
2	PLC 2	Spurious retransmissions (1 s)	TCP session corruption (9 s)	Denial of service (7 min)
3	PLC 3	Spurious retransmissions (1 s)	TCP session corruption (20 s)	PLC forced restart

the attacker. Out of 12 available tactics, only 6 were required to achieve the attacker's goal of Denial of Control (T0813) and Denial of View (T0815). The fewer tactics used to reach the desired impact goal, the easier it is to carry out an attack on live production systems. This is summarised in Table 6.

Table 6
Summary of Mitre ATT&CK tactics and techniques used in field flooding attack.

Tactic	Technique	Technique ID
Initial Access	Internet accessible device	T0883
Execution	Command-line interface, scripting	T0807, T0853
Discovery	Network Sniffing	T0842
Inhibit Response Function	Block reporting message, denial of service	T0804, T0814
Impair Process Control	Modify parameter, unauthorised command message	T0836, T0855
Impact	Denial of control, denial of view	T0813, T0815

6. Detection of field flooding attack: supervised machine learning

6.1. Dataset

The dataset was created by collecting a combined 4 h worth of network pcap traffic from all three testbeds using `Wireshark`. During the capture, the PLCs had malicious packets injected into the stream as described in 3.2 and the data was saved into three separate pcap files (i.e. one from each testbed). These pcap files were converted into a csv file format using `Tshark`, and subsequently combined into a single file to make a total of 127,758

Table 7
Summary of dataset.

Total data points	127,758
Benign data points	114,700
Attack data points	13,058
Total capture duration	3.8 h

Table 8
Summary of attacks in dataset (AL = Altered length, FF = Field Flooding).

Attack type	Packets injected	Attack duration (s)
Packet replay	3	4.6
AL Injection	3	6.7
FF + 1 Field	6	13.1
FF + 2 Fields	12	35.7
Total	**24**	**60.1**

data points containing 29 features (114,700 = benign and 13,058 = malicious) – comparable in size to datasets used in other similar studies (e.g. Anthi et al., 2021b; Injadat et al., 2018). To label the dataset, it was ensured that every malicious packet injected successfully had the same transaction ID (e.g. 8000). By filtering the field `mbtcp.trans_id` == 8000, the start of each field flooding attack was identified and labelled along with its impact. Combining the datasets from the 3 testbeds enabled the development of a more robust model that would generalise better when using data from similar ICS networks. The total attack duration of the experiments carried out was approximately 60 s and a summary of the dataset description is shown in Tables 7 and 8.

6.2. Feature selection

To train a supervised machine learning model effectively, it is important to identify features that best describe the dataset (Anthi et al., 2021a). As the focus of our study is the ModbusTCP protocol, features from the TCP/IP layers and the Modbus layer (embedded within the TCP layer) form the key selected features for our model training. Features from the ethernet layer (e.g. mac addresses, src and dst addresses) were not considered because they include properties which may lead to overfitting of the machine learning model. At the same time, temporal features from the Frame header (e.g. `frame.time_delta`) to capture packet inter-arrival times were also selected. This created an initial dataset with 30 features with the labelled target variable inclusive.

To further reduce the risk of overfitting, features that represent identifying properties (e.g. IP/mac addresses) were also removed from the feature set (Anthi et al., 2021a). In this case, the `mbtcp.trans_id` feature was also removed as all the attacks had the same transaction ID. Furthermore, features that had only one unique value within the dataset were not considered as these would have no effect on the target variable and would increase computational overhead. This resulted in pruning the number of selected features to 24.

Additionally, to understand the worth of each feature for the target variable, a selection filter – InfoGainAttributeEval – using `Weka` (Hall et al., 2009) was applied to the remaining 24 features. The InfoGainAttributeEval evaluates the *worth* of an attribute by measuring information gain with respect to the class (Anthi et al., 2021a; Mahfouz et al., 2020). This identifies features more significant for detecting an attack.

The result of the filter, shown in Table 9 indicates that the raw SEQ, ACK, and delta time attributes are ranked as the most important. This could be attributed to the way the field flooding attack is executed as it measures SEQ and ACK numbers and uses them as the seed to generate a malicious packet. Also, the delta time attribute indicates the time difference between consecutive packets in a capture. This would make sense as the Field Flooding attack exploited the gap of 100ms between loops in the ModbusTCP transmission to inject the malicious payload, which would invariably lead to distortion of the regular benign delta time packet transmission.

Table 9
Information gain ranking filter for features.

Rank	Rank score	Attribute
1	0.42723	tcp.seq_raw
2	0.391132	tcp.ack_raw
3	0.322357	tcp.time_delta
4	0.317954	tcp.analysis.initial_rtt
5	0.305935	frame.time_delta
6	0.27151	tcp.srcport
7	0.243965	tcp.window_size_value
8	0.1808	tcp.flags
9	0.169913	tcp.dstport
10	0.166015	frame.len
11	0.097374	ip.len
12	0.069722	modbus.func_code
13	0.067809	tcp.pdu.size
14	0.066313	tcp.len
15	0.062963	mbtcp.len
16	0.046628	modbus.byte_cnt
17	0.040866	mbtcp.unit_id
18	0.029992	tcp.checksum
19	0.019521	tcp.analysis
20	0.014819	ip.ttl
21	0.006837	ip.proto
22	0.003641	modbus.word_cnt
23	0.000474	modbus.reference_num
24	0.00042	ip.flags

Table 10
Classification metrics results.

Classifier	Precision	Recall	F1-score
Logistic Regression	0.953	0.99	0.971
Random Forest	0.998	0.998	0.998
Naòve Bayes	0.993	0.406	0.577
Decision Tree	0.998	0.998	0.998
XGBoost	**0.999**	**0.999**	**0.999**
K-NN	0.997	0.996	0.997
Kernel SVM	0.995	0.993	0.994
SVM	0.975	0.973	0.974

6.3. Model training and analysis

All machine learning experiments were carried out on a Windows 10 PC with Intel(R) Core(TM) i7-8665U CPU at 1.90 GHz processor and 16 gb RAM. The final dataset with 24 features selected as discussed in Section 6.2 went through data pre-processing (i.e. data normalisation and label encoding) before model training. The dataset was randomly split into 60% for training and 40% for testing and evaluation on unseen data. The choice of an appropriate algorithm is based on model performance for a particular problem and the properties of data that characterise the problem (Anthi et al., 2021a). Eight classifiers were considered based on other relevant work (Leevy et al., 2021; Luan and Dong, 2018); and based on how they operate. In more detail, the models included algorithms that function based on conditional dependencies in the dataset or assume conditional independence (e.g., Bayesian Network and naive Bayes), discriminative models that aim to maximize information gain without modeling any underlying probability or structure of the data (e.g., J48 decision tree and support vector machine), and ensemble models that utilise multiple ML algorithms to produce higher predictive performance than could be obtained from a single ML classifier (Mahfouz et al., 2020; Ryu et al., 2010) (e.g. Random Forest, XGBoost).

Table 11
Confusion matrices for XGBoost and random forest classifiers.

		Predicted				Predicted	
		Malicious	Benign			Malicious	Benign
	(a) XGBoost				(b) Random Forest		
Actual	Malicious	5189	65	Actual	Malicious	5181	73
	Benign	49	45,801		Benign	75	45,775

Before discussing the metrics to be used in evaluating the classifiers, the following terms shall be explained:

- True Positives (TP): Number of actual positives correctly predicted.
- True Negative (TN): Number of actual negatives correctly predicted.
- False Positive (FP): Number of actual negatives predicted incorrectly as positive.
- False Negative (FN): Number of actual positives predicted incorrectly as negative.

In evaluating the performance of our classifiers, it is recommended to use precision, recall, and F1-scores (Timčenko and Gajin, 2018) defined as:

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2{*}Precision{*}Recall}{Precision + Recall} = \frac{2{*}TP}{2{*}TP + FP + FN}$$

The best performing classifier was XGBoost with an F1-score of 99.9% while the second best performing classifier was the Random Forest with an F1-score of 99.8%. Both XGBoost and Random Forest are ensemble algorithms that use decision trees as their meta-classifier and generally perform well on non-linear problems as in our case. Table 10 shows the precision, recall, and F1-scores of all evaluated classifiers. The confusion matrices of both XGBoost and Random Forest reveal that the XGBoost classifier predicted marginally less FN/FP than the Random Forest classifier as shown in Table 11.

7. Conclusions

With the increase in cyber attacks on Industrial Control Systems and the frequency of those attacks leading to DoS scenarios, this study identifies a pathway to attacking these systems to deny legitimate service using the ModbusTCP protocol. Previous work has focused on protecting ModbusTCP packets by ensuring the size allocated to a particular field in the MBAP and PDU headers are within set limits. In this study, a novel field flooding attack capable of bypassing these protection mechanisms was demonstrated, keeping the fields within their data size (in bytes) limit, but increasing the number of fields by 2, resulting in an additional 4 bytes of fields to the PDU header.

The impact of the field flooding attack was evaluated on three physical industrial testbeds with different configurations. The results show that the PLC usually deployed in the oil and gas (OG) industry was the most vulnerable to this attack as one malicious packet resulted in a denial of service of approximately 59 min. In OG operations this could have significant implications as it could potentially lead to unsafe conditions which could damage the environment due to the hazardous nature of hydrocarbons. Although this attack has been shown to be capable of disrupting OG operations, it could also potentially disrupt critical ICS communications in other sectors. This work also shows that PLCs may behave differently to the same cyber attack, which highlights a clear advantage of using real industrial testbeds for security research and the limitations of simulated cyber-physical testbeds – as these simulated experiments are unable to account for the difference in PLC behaviour in a real system.

To effectively detect the Field Flooding attack, our initial machine learning experiments demonstrated that the best performing classifier was XGBoost – an ensemble algorithm based on a Decision Tree meta-classifier. This paper presents the initial experiments for automatically detecting attacks using machine learning algorithms by utilising signatures from pcap files. Given the preliminary stage of this investigation, this analysis has been conducted offline in order to examine its feasibility. Following the positive findings of this initial study, the next step is to implement this system in real-time, so that it can be deployed in a real and larger environment. This will allow the system to be further evaluated on more complex and more sophisticated attacks to study further the IDS response times and overall system impact.

A key limitation of this work, and a basis for our future work, is the lack of access to larger-scale testbeds with multiple industrial protocols. This study does not include scenarios that demonstrate how field flooding attacks could propagate within an industrial network with multiple brands of PLCs and protocols.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Abubakar Sadiq Mohammed: Conceptualization, Methodology, Software, Writing – original draft. **Eirini Anthi:** Supervision, Writing – review & editing. **Omer Rana:** Project administration, Supervision, Writing – review & editing. **Neetesh Saxena:** Supervision, Writing – review & editing. **Pete Burnap:** Funding acquisition, Supervision.

Data Availability

Data will be made available on request.

Acknowledgement

This work was carried out in collaboration with the Petroleum Technology Development Fund, Nigeria (PTDF) and the National Digital Exploitation Centre (NDEC), a part of the Thales Group.

References

AdvancedHMI. HMI software by AdvancedHMI, the industry's most flexible HMI. 2022. https://www.advancedhmi.com/.

Alcaraz, C., Bernieri, G., Pascucci, F., Lopez, J., Setola, R., 2019. Covert channels-based stealth attacks in industry 4.0. IEEE Syst. J. 13 (4), 3980–3988.

Alexander, O., Belisle, M., Steele, J., 2020. Mitre Att&ck® for Industrial Control Systems: Design and Philosophy. The MITRE Corporation, Bedford, MA, USA.

Anthi, E., Williams, L., Burnap, P., Jones, K., 2021a. A three-tiered intrusion detection system for industrial control systems. J. Cybersecur. 7 (1), tyab006.

Anthi, E., Williams, L., Javed, A., Burnap, P., 2021b. Hardening machine learning denial of service (DOS) defences against adversarial attacks in IoT smart home networks. Comput. Secur. 108, 102352.

Assante M.J., Lee R.M.. The industrial control system cyber kill chain. SANS Institute InfoSec Reading Room2015; 1.

Bashendy, M., Eltanbouly, S., Tantawy, A., Erradi, A., 2020. Design and implementation of cyber-physical attacks on modbus/TCP protocol. World Congress on Industrial Control Systems Security (WCICSS-2020).

Chattha, H.A., Rehman, M.M.U., Mustafa, G., Khan, A.Q., Abid, M., Haq, E.U., 2021. Implementation of cyber-physical systems with modbus communication for security studies. In: 2021 International Conference on Cyber Warfare and Security (ICCWS). IEEE, pp. 45–50.

Combs G.. Wireshark. 2022. https://www.wireshark.org/.

Gonzalez, J., Papa, M., 2007. Passive scanning in modbus networks. In: International Conference on Critical Infrastructure Protection. Springer, pp. 175–187.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H., 2009. The WEKA data mining software: an update. ACM SIGKDD Explor. Newsl. 11 (1), 10–18.

He, X., Robards, E., Gamble, R., Papa, M., 2019. Anomaly detection sensors for a modbus-based oil and gas well-monitoring system. In: 2019 2nd International Conference on Data Intelligence and Security (ICDIS). IEEE, pp. 1–8.

Huitsing, P., Chandia, R., Papa, M., Shenoi, S., 2008. Attack taxonomies for the modbus protocols. Int. J. Crit. Infrastruct. Prot. 1, 37–44.

Injadat, M., Salo, F., Nassif, A.B., Essex, A., Shami, A., 2018. Bayesian optimization with machine learning algorithms towards anomaly detection. In: 2018 IEEE Global Communications Conference (GLOBECOM). IEEE, pp. 1–6.

Katulić, F., Sumina, D., Erceg, I., Groš, S., 2022. Enhancing modbus/TCP-based industrial automation and control systems cybersecurity using a misuse-based intrusion detection system. In: 2022 International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM). IEEE, pp. 964–969.

Leevy, J.L., Hancock, J., Khoshgoftaar, T.M., Peterson, J., 2021. Detecting information theft attacks in the bot-IoTdataset. In: 2021 20th IEEE International Conference On Machine Learning And Applications (ICMLA). IEEE, pp. 807–812.

Luan, C., Dong, G., 2018. Experimental identification of hard data sets for classification and feature selection methods with insights on method selection. Data Knowl. Eng. 118, 41–51.

Luswata, J., Zavarsky, P., Swar, B., Zvabva, D., 2018. Analysis of SCADA security using penetration testing: a case study on modbus TCP protocol. In: 2018 29th Biennial Symposium on Communications (BSC). IEEE, pp. 1–5.

Mahfouz, A., Abuhussein, A., Venugopal, D., Shiva, S., 2020. Ensemble classifiers for network intrusion detection using a novel network attack dataset. Future Internet 12 (11), 180.

Mohammed, A.S., Reinecke, P., Burnap, P., Rana, O., Anthi, E., 2022. Cybersecurity challenges in the offshore oil and gas industry: an industrial cyber-physical systems (ICPS) perspective. ACM Trans. Cyber-Phys Syst. doi:10.1145/3548691.

Morris, T.H., Jones, B.A., Vaughn, R.B., Dandass, Y.S., 2013. Deterministic intrusion detection rules for MODBUS protocols. In: 2013 46th Hawaii International Conference on System Sciences. IEEE, pp. 1773–1781.

Parian, C., Guldimann, T., Bhatia, S., 2020. Fooling the master: exploiting weaknesses in the modbus protocol. Procedia Comput. Sci. 171, 2453–2458.

Radoglou-Grammatikis, P., Siniosoglou, I., Liatifis, T., Kourouniadis, A., Rompolos, K., Sarigiannidis, P., 2020a. Implementation and detection of modbus cyberattacks. In: 2020 9th International Conference on Modern Circuits and Systems Technologies (MOCAST). IEEE, pp. 1–4.

Radoglou Grammatikis, P., Sarigiannidis, P., Efstathopoulos, G., Panaousis, E., 2020b. Aries: a novel multivariate intrusion detection system for smart grid. Sensors 20 (18), 5305.

Rajesh, L., Satyanarayana, P., 2021. Detection and blocking of replay, false command, and false access injection commands in SCADA systems with modbus protocol. Secur. Commun. Netw. 2021, 15. doi:10.1155/2021/8887666, Article ID 8887666.

Rohith, R., Moharir, M., Shobha, G., et al., 2018. SCAPY—A powerful interactive packet manipulation program. In: 2018 International Conference on Networking, Embedded and Wireless Systems (ICNEWS). IEEE, pp. 1–5.

R. RS. 485 specification, modicon modbus protocol reference guide PI-MBUS-300 rev. 2002.

Ryu, J.W., Kantardzic, M., Walgampaya, C., 2010. Ensemble classifier based on misclassified streaming data. In: Proc. of the 10th IASTED Int. Conf. on Artificial Intelligence and Applications, Austria, pp. 347–354.

Saharkhizan, M., Azmoodeh, A., Dehghantanha, A., Choo, K.K.R., Parizi, R.M., 2020. An ensemble of deep recurrent neural networks for detecting IoT cyber attacks using network traffic. IEEE Internet Things J. 7 (9), 8852–8859.

Stranahan, J., Soni, T., Heydari, V., 2019. Supervisory control and data acquisition testbed vulnerabilities and attacks. In: 2019 SoutheastCon. IEEE, pp. 1–5.

Timčenko, V., Gajin, S., 2018. Machine learning based network anomaly detection for IoT environments. In: ICIST-2018 Conference.

Tidy J.. European oil facilities hit by cyber-attacks. 2022. https://www.bbc.co.uk/news/technology-60250956.

Abubakar Sadiq Mohammed is currently carrying out Ph.D. research in cybersecurity at Cardiff University. He received a B.Eng. Degree in Mechanical Engineering from the Federal University of Technology, Minna, Nigeria, and an M.Sc. in Petroleum and Gas Engineering from the University of Salford, U.K. In addition to his qualifications, he has gained over 14 years of engineering experience working in the oil and gas industry. He brings his engineering background to industrial cybersecurity to help gain valuable insights on how to secure industrial control systems. His research interests include cybersecurity for SCADA systems and Industrial Control Systems and using machine learning for anomaly detection.

Dr Eirini Anthi is a lecturer in cybersecurity at the School of Computer Science & Informatics, Cardiff University. She teaches Operating Systems Security and Cybersecurity Operations. In addition, her research interests revolve around the security of the Internet of Things (IoT), SCADA, and Industrial Control Systems. More particularly, her research examines the security issues that come along with these devices/systems and focuses on developing intelligent and more robust cyber-attack detection mechanisms for such networks using machine learning and adversarial machine learning techniques. As part of her doctorate, she developed state-of-the-art tools to detect and defend against network-based cyber attacks in such infrastructures.

Omer F. Rana received the B.Eng. degree in information systems engineering from Imperial College of Science, Technology and Medicine, London, U.K., an M.Sc. in microelectronics systems design from the University of Southampton, U.K., and a Ph.D. in neural computing and parallel architectures from the Imperial College of Science, Technology and Medicine. He is a Professor of performance engineering with Cardiff University, Cardiff, U.K. His research interests include high performance distributed computing, data analytics/mining and scalable systems.

Neetesh Saxena is currently an Associate Professor (Senior Lecturer) with the School of Computer Science and Informatics at Cardiff University, UK with more than 16 years of teaching/research experience in academia. Before joining CU, he was an Assistant Professor with Bournemouth University, UK. Prior to this, he was a Post-Doctoral Researcher in the School of Electrical and Computer Engineering at the Georgia Institute of Technology, USA, and with the Department of Computer Science, Stony Brook University, USA and SUNY Korea. He was a DAAD Scholar at Bonn-Aachen International Center for Information Technology (B-IT), Rheinische-Friedrich-Wilhelms Universitt, Bonn, Germany and was also a TCS Research Scholar. His current research interests include cyber security and critical infrastructure security, including cyber-physical system security: smart grid, V2G and communication networks.

Pete Burnap is a Professor at Cardiff University and is seconded to Airbus Group to lead Cyber Security Analytics Research heading projects involving the application of Artificial Intelligence, Machine Learning and Statistical Modeling to Cyber Security problems (most recently malware analysis). Pete obtained his B.Sc. in Computer Science in 2002 and his Ph.D.: Advanced Access Control in support of Distributed Collaborative Working and Deperimeterization in 2010, both from Cardiff University. He has published more than 60 academic articles stemming from funded research projects worth over 8m and has advised the Home Affairs Biographical Sketch Select Committee, Home Office and Metropolitan Police on sociotechnical research outcomes associated with cyber risk and evolving cyber threats.