Physics-Aware Targeted Attacks Against Maritime Industrial Control Systems

Giacomo Longo[a,1], Francesco Lupia[b,*], Andrea Pugliese[b] and Enrico Russo[a]

[a] University of Genoa, Italy
[b] University of Calabria, Italy

ARTICLE INFO

Keywords:
Physics-Awareness
Targeted Attacks
Process Mining
Maritime Industry
Industrial Control Systems

ABSTRACT

The advancement of the maritime industry towards technologically integrated and automated systems has significantly increased the complexity of onboard Industrial Control Systems (ICS), raising concerns about cybersecurity risks. In this paper, we examine typical onboard ICS configurations through an adversarial lens. We introduce a threat model that leverages domain-specific peculiarities, e.g., maritime protocols, and targets vulnerability vectors to execute software attacks against the infrastructures of shipboard ICS. This includes a case study on a critical subsystem of ship machinery: the steering gear system. We have developed a novel attack methodology intended for use by targeted malware. A comprehensive experimental assessment confirms the feasibility of attacks devised according to our methodology.

1 Introduction

In recent years, the maritime industry has accelerated its drive towards technological integration, increased automation, and the implementation of new and improved systems. This transition has also occurred onboard ships, which have developed into increasingly large and complex Industrial Control Systems (ICS). Aimed at assisting and enhancing efficiency for maritime personnel, this evolution has promised to elevate safety and security during their operations. However, these advancements have also raised concerns regarding cybersecurity risks [17, 61], challenging such improvements.

Despite the sector having little to no incentive to disclose incidents to the public, analysis within the scientific community has offered an overview of maritime cybersecurity incidents during the decade 2010-2020 [53]. The analysis confirms a substantial and rapid increase in the attacks related to onboard or off-ship systems with consequences concerning security (e.g., disruption of operations) and safety (e.g., return to port). Moreover, most of these attacks exploited common and generic vectors (e.g., IT vulnerabilities, USB keys, or social engineering) and prominently utilized ransomware as a primary technique.

Nevertheless, the experience with ICS breaches has demonstrated that the most severe consequences have arisen from targeted attacks and their capability to exploit the specificity of the running context. Stuxnet [18] and Triton [15] are famous examples illustrating this phenomenon. Stuxnet specifically targeted Supervisory Control and Data Acquisition (SCADA) systems used in Iran's nuclear program, demonstrating a highly targeted approach to disrupt a particular industrial process. Similarly, Triton was designed to manipulate Safety Instrumented Systems in critical infrastructure, showcasing the extreme danger posed by targeting these systems responsible for ensuring plant safety. Drawing insights from incidents such as the above, we argue that it is crucial to study how onboard systems are vulnerable to similar attacks.

To this aim, the paper focuses on the typical onboard ICS configurations from an adversarial standpoint. The main contributions of the paper can be summarized as follows.

- We propose a threat model that resembles the targeted malware described above, exploiting vectors such as maritime protocols, shipboard ICSs, and international regulatory constraints to execute and maximize the potential impact of an attack.
- We present a realistic case study involving a critical subsystem within the shipboard ICSs. The subsystem governs the directional control and accurately reproduces the peculiarities of the system that are exploitable by attackers.
- We introduce a novel attack methodology embedded in a standalone targeted malware. The methodology includes three main phases. In the *reconnaissance* phase, the malware analyzes the specific characteristics of available registers in order to discover and characterize those of interest for the construction of effective attacks. Then, the malware collects data from onboard communication systems. In the *weaponization* phase, the malware applies process discovery techniques to the data and metadata produced during reconnaissance in order to dissect the behavior of the

This work was partially funded by the NextGenerationEU project "Security and Rights in CyberSpace" (SERICS) and supported by research funding from Fincantieri E-phors S.p.A.

*Corresponding author
giacomo.longo@dibris.unige.it (G. Longo); francesco.lupia@unical.it (F. Lupia); andrea.pugliese@unical.it (A. Pugliese); enrico.russo@unige.it (E. Russo)
ORCID(s): 0000-0003-0025-7191 (G. Longo); 0000-0003-0775-6890 (F. Lupia); 0000-0003-4385-958X (A. Pugliese); 0000-0002-1077-2771 (E. Russo)

[1] G. Longo's work was carried out while he was enrolled in the Italian National Doctorate on Artificial Intelligence run by the Sapienza University of Rome in collaboration with the University of Genoa.

various control systems and build models of the processes that may have an effect on the direction of the ship — a critical requirement of such models is that of accurately reflecting the expected system operations. Following this, the malware generates attacks in the form of sequences of payloads. In the final *delivery* phase, when specific triggering conditions are met, the malware injects the payloads into the automation network.

- We perform extensive experiments to assess the feasibility of the attacks based on our methodology. The results confirm that our malware can generate attacks that have an immediate impact on the shipboard ICSs. Such attacks have hugely disruptive consequences on the ship, similar to the case of the grounding of *Ever Given*, which resulted in heavy financial price and loss of revenues for Egypt and consumers worldwide [23].

This paper also aims to highlight the importance of enhancing and refining existing countermeasures.

Structure of the paper. The remainder of the paper is organized as follows. Section 2 provides preliminary notions about various systems and protocols used onboard modern vessels and process mining techniques. In Section 3, we describe the assumptions we make about attackers, and in Section 4, we provide details on the case study targeted in the paper. In Section 5, we develop our proposed methodology to build and execute attacks. In Section 6, we report on our experimental assessment. Finally, we discuss possible remediations in Section 7, review related work in Section 8, and draw our conclusions in Section 9.

2 Preliminaries

2.1. Integrated systems in onboard maritime operations

On modern vessels, onboard systems seamlessly integrate to ensure efficient coordination and control of various ship functions. The pivotal components are the Integrated Bridge System (IBS) [62] and the Integrated Platform Management System (IPMS) [74]. IBS focuses on navigation, communication, and control of onboard systems from the bridge, whereas IPMS manages and monitors the ship's automation.

Figure 1 depicts a typical architecture used to integrate onboard equipment.

It comprises multifunction control consoles, navigational sensors, and Remote Terminal Units (RTU). The consoles are workstations that offer bridge operators access to essential navigation functions within the IBS, such as RADAR, Electronic Chart Display and Information System (ECDIS), and autopilot. They also serve as the Human Machine Interface for the IPMS in various control rooms, such as the Engine Control Room. Navigational sensors gather and transmit data about the ship's location, orientation, velocity, and the surrounding environmental conditions. RTUs (see Section 2.3) provide process-level data acquisition and control. They interface with the actuators and sensors of the ship machinery. In particular, they interact with the subsystems that generate thrust for propelling (Propulsion) and control pumps and compressors (Auxiliaries), electrical generators and switchboards (Electrical), and the ship's direction (Steering).

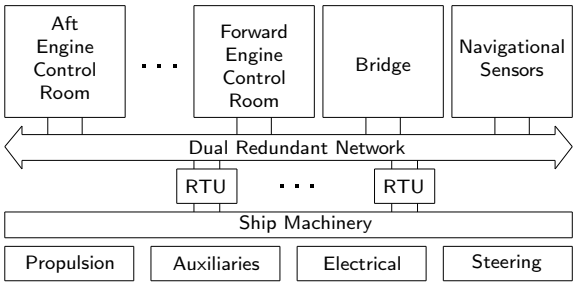

Figure 1: Typical architecture for integrated onboard systems.

The overall architecture configuration follows an integration pattern where any connected endpoint can use a dual redundant network to receive and add messages. Navigational sensors transmit actual data. RTUs send messages with setpoints from ship machinery and receive commands for actuators. Consoles consume, process, and visualize exchanged data. They offer operators extensive information and facilitate seamless data fusion, fostering comprehensive situational awareness and decision support. This architecture also simplifies the redundancy of control rooms, e.g., a forward and aft engine control room, and the duplication of control stations, such as placing an engineering station in the bridge to empower officers in monitoring vital functions of machinery subsystems.

Integration is ensured even among equipment from different manufacturers, facilitated by an open architecture that leverages standard protocols. In particular, navigational sensors and maritime equipment use the NMEA 0183 [35] standard (see Section 2.5). RTUs adhere to protocols used in Operational Technology (OT) installations. There is no single standard, but onboard installations favor two widely adopted protocols, namely Modbus and OPC [49]. In this paper, we focus on Modbus (see Section 2.4) since it is still the protocol that most vendors choose to implement [3].

In terms of cybersecurity, the protocols used lack inherent security features, e.g., encryption or authentication. These weaknesses imply attackers could eavesdrop on transmitted data and inject false information. In addition, they could exploit the integrated configuration to perform lateral movements between the different systems.

Recently, with support from the International Maritime Organization's high-level recommendations on cyber risk [40], the design or refit of ships is starting to consider this aspect. Rather than full integration, the current preference is to separate systems to create zones grouped by function [62], such as navigation and automation, with least privilege [66]. However, to keep some control consoles operational, these restrictions *must be kept relaxed*. For example, the control of the steering subsystem relies on the real-time

monitoring of the ship's behavior [5], e.g., roll angle, pitch angle, and heave, and needs to access both navigation and automation to correlate data from navigational sensors and ship machinery.

2.2. Steering gear system

In the maritime context, the *steering gear system* refers to the equipment used to steer the vessel. It typically includes the rudder, mechanical linkages, actuators, and an associated Steering Gear Control System (SGCS).

The International Convention for the Safety of Life at Sea (SOLAS) and its amendments [36, 37, 72, 73] define the mandatory functional and operational requirements for steering gear systems found onboard ships undertaking international voyages. An example of the level of detail covered by these regulations is the explicit requirement within the ship's design of a rudder angle indicator on the navigation bridge, giving the helmsman or officer on watch real-time information about the angle of the rudder. As a result of these detailed regulations, the implementations often rely on a select few time-tested solutions with well-established specifics.

In this paper, we focus on an electro-hydraulic type steering, which is the most common among modern large commercial vessels [30]. Specifically, the units tasked with supplying the essential force to steer the vessel and regulate its direction, namely Power Units (PU), comprise an electric motor, its corresponding electrical components, and an interconnected pump.

SOLAS requirements that are crucial to our research focus are:

- A minimum of two independent power units. These units are under constant control by the SGCS, which regulates the rudder angle through fluid circuit valves and power unit adjustments.
- Unit independence. This critical feature ensures the isolation of any potential failure within the piping system or individual power units. This is essential to maintaining or quickly restoring control capabilities.
- Relief valves to prevent excessive pressure build-ups that could lead to system failure or damage.

We refer the reader to Section 4 for details on the inner workings of an electro-hydraulic SGCS.

2.3. Programmable logic controllers and operational cycles

Programmable Logic Controllers are digital systems engineered for the industrial environment, as defined by the IEC 61131 standard [70]. At the core of a PLC lies a CPU tasked with executing the user-defined logic program while managing the system's operational cycle and communicating with peripheral devices.

A PLC's operational cycle is a continuous sequence initiated by the CPU. This sequence involves scanning and interpreting input data, processing the data according to the user program, and updating the outputs to control the industrial process. The user program is stored in a programmable memory, while the cycle's data is held temporarily during the execution. Each PLC thus operates under a stringent three-phase operational cycle, known as the *scan cycle*: (1) read inputs, (2) process, (3) write outputs.

In most cases, a remote control solution, like onboard consoles (see Section 2.1), does not directly interface with the PLCs of the controlled system. Instead, it relies on a single dedicated remote terminal unit (RTU) to communicate and interact with the functionalities of field devices. The RTU acts as both a communications intermediary and a hiding place for the complexity of the underlying system. From an operational perspective, it can be considered a specialized application of a PLC that supports the functionality described above.

2.4. Modbus

The Modbus protocol was designed to deal with several mediums, such as satellite, telephone, and radio devices, from different manufacturers without focusing on a single brand architecture. Modbus communications are of two types: (i) query/response, i.e., between a coordinator and a worker, or (ii) broadcast, i.e., from a coordinator to all the workers. Modbus maps the temporary memory of a PLC program to four different types of registers: discrete output coils, discrete input contacts, analog input registers, and analog output holding registers. Coils and discrete inputs are 1-bit registers. Input and holding registers are 16-bit registers. The latter can be combined to create general memory registers of different sizes, such as 32- or 64-bit registers. The commands that manipulate these registers are called *function codes*. They allow read/write access on the coil and holding registers and read-only access on the discrete and input registers. A Modbus transaction comprises a single query, response, or broadcast frame. A Modbus frame message contains the address of the intended receiver, the command the receiver must execute, and the data needed to execute the command. Modbus TCP embeds a Modbus frame into a TCP frame. TCP/IP coordinators and workers listen and receive Modbus data via port 502.

2.5. NMEA

The NMEA 0183 standard [35] defines an electrical and data exchange format between maritime electronics. Most onboard sensors and equipment communicate via NMEA [38]. In particular, any device can discover, listen, and communicate with multicast flows via the standard IGMP protocol [9]. This configuration, known as Multiple talkers and multiple listeners–Ethernet interconnection, is often referred to as *Lightweight Ethernet (LWE)*.

Each message (or *sentence*) comprises a start character followed by comma-delimited fields and a simple XOR checksum terminated by a two-byte delimiter. Of particular interest is the *talker sentence* format in which each message contains a two-letter talker identifier, a three-letter sentence type, and a variable number of fields. For instance, the following example sentence with talker identifier

```
$SGRSA,13.74,A,,V,*40
```

represents a message emitted by the steering gear (SG), with a sentence type related to the rudder sensor angle (RSA), and indicating a valid (A) rudder angle measurement of starboard 13.74°, no secondary rudder (empty field followed by V) and having a checksum of $40_{16} = 64_{10}$.

2.6. Process mining

Process mining is a powerful approach to analyze complex processes [1]. It consists of looking at a set of events recorded in an event log related to the execution of the activities of an underlying process. The primary goal is to shed light on the actual behavior of the process at hand.

In this paper, we focus on *process discovery*, a fundamental task in process mining aiming at constructing a graph of causal dependencies over various activities within a process. Basically, this task is devoted to automatically deriving a causal dependency graph from a collection of *traces* (called event log), which records the sequence of activities performed during the operational cycle of a process. The goal is to build a dependency graph that can be used to design a full process model. This model can then help in supporting workflow management systems or to shed light on the actual process behavior.

However, dependency-graph discovery is a challenging problem due to log incompleteness. To this end, the opportunity to use background knowledge, which domain experts have access to, has been proposed as a solution to enhance the accuracy of these models [25]. In the context of ICS, process discovery algorithms have been successfully applied to generate process models that accurately reflect the expected behavior of the system, learning from the device logs generated by ICS devices like PLCs [57]. These devices are often characterized by multiple activities operating concurrently, each executing a series of steps that are critical for the physical process. The order of events in an ICS process is crucial – any deviation from the expected sequence can disrupt the process. In such scenarios, even valid ICS events, if executed in an incorrect sequence, have the potential to disrupt the industrial operation.

As further discussed in this paper, discovering the normal behavior of a system can provide valuable information for conducting both offensive and defensive strategies.

3 Adversary Model and Assumptions

This work focuses on an advanced threat actor that operates within the maritime sector. In particular, we consider adversaries that the Maritime Cyber-Risk Assessment (MACRA) categorizes as $tier_3$ attackers [69]. They have the expertise and resources for crafting targeted malware [55] and employing diverse techniques for its deployment. These techniques leverage maintenance operations [10, 47], compromising the supply chain [54], social engineering, or the vulnerabilities present in onboard workstations [48, 68].

We assume that the targeted vessel complies with established maritime industry standards and regulations. Specifically, it adopts the NMEA 0183 standard through an LWE configuration, as detailed in Section 2.5. In addition, it operates with a single rudder electro-hydraulic steering gear system designed to adhere to SOLAS (see Section 2.2). We also assume attackers have deployed the malware on an onboard system that both receives NMEA data and has access to the RTU of the steering subsystem using Modbus.

The goal is to manipulate the ship's direction as desired by gaining control of the steering system. This aims to create disorder or interfere with navigation, resulting in significant economic or reputational harm while compromising safety.

We determine that the malware must work under the following requirements to fulfill the objective.

- It works as a standalone and can carry out malicious activities without contacting a command and control infrastructure.
- It maintains a covert operational approach. This condition requires the malware to orchestrate the attack by observing system behavior rather than making trial-and-error changes directly.
- It is capable of running on onboard systems, including legacy ones, without requiring significant resources. Excessive resource usage can increase the likelihood of detection.
- Upon activation, it is able to bring the ship to the desired state in a limited time, minimizing the crew's capacity to respond.

4 Case Study

In this paper, we consider an ocean-going vessel underway. The vessel is equipped with a SOLAS-compliant electro-hydraulic SGCS featuring a dual-pump solution to ensure the required redundancy of the PUs. Figure 2 depicts a schema of the subsystem.

The rudder is operated by two linked dual-acting hydraulic cylinders (*lRam* and *rRam*) counteracted by the self-centering force of the rudder. Both PUs (*lPump* and *rPump*) are fixed displacement pumps rotating respectively at *lPumpSpeed* and *rPumpSpeed* revolutions per minute. They feed a central mixing valve called *mixValve*. This valve allows them to supply each ram individually with a single PU or to work together by combining their efforts. Valves at the rams (*lValve* and *rValve*) enable choosing a preferred actuation direction or neutralizing hydraulic force by redirecting fluid to the return line. This return line is also connected to spring-operated over-pressure relief valves (*lReliefvalve* and *rReliefValve*). Within the circuit, we consider an ISO VG 100 [42] compressible work fluid contained within two tanks (*lTank* and *rTank*) at a standard [41] ambient pressure (101.325kPa) and temperature (15°C). For brevity, The refill of tanks and drainage piping are not depicted in the picture. Control-wise, several holding registers accessible via an

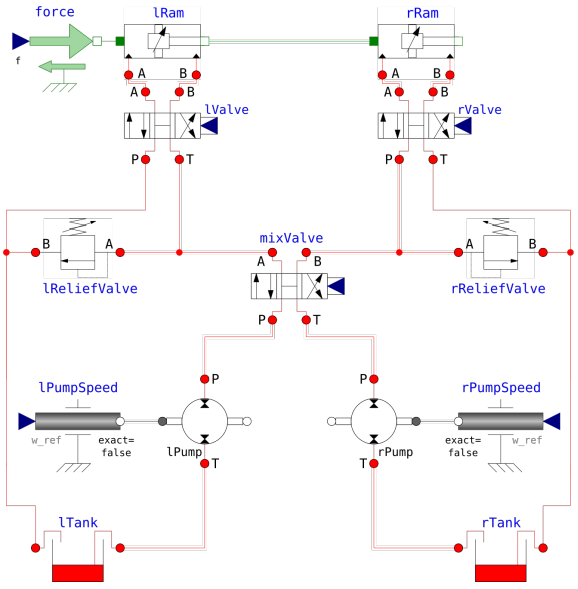

Figure 2: SGCS Schema.

Table 1
RTU Registers.

Register	Description	Unit	Domain
QW100	Desired rudder angle	deg	$[-35, 35]$
IW101	Current rudder angle	deg	$[-35, 35]$
QW102	Operating Mode	–	$\{0, 1, 2\}$
QW103-105	{Mix,L,R} Valve command	–	$\{-1, 0, 1\}$
QW106-107	{L,R} Pump governor	%	$[0, 1]$
IW108-109	{L,R} Pump speed	rpm	$[0, 3000]$
IW110-111	{L,R} Circuit pressure	bar	$[0, 115]$
IW112-113	{L,R} Pump flow rate	m^3/s	$[0, 2.5]$
IW114-115	{L,R} Pump suction pressure	bar	$[0, 115]$
IW116-117	{L,R} Pump downstream pressure	bar	$[0, 115]$
IW118-119	{L,R} relief pressure	bar	$[0, 115]$
IW120-121	{L,R} relief flow rate	m^3/s	$[0, 2.5]$
IW122-123	{L,R} relief opening	%	$[0, 1]$
IW124-124	{L,R} return flow rate	m^3/s	$[0, 2.5]$
IW125-126	{L,R} tank level	%	$[0, 1]$
IW127-130	{L,R} ram {A,B} pressure	bar	$[0, 115]$
IW131-138	{L,R} valve {P,T,A,B} pressure	bar	$[0, 115]$
IW139-142	Mix valve {P,T,A,B} pressure	bar	$[0, 115]$

RTU and its interconnected PLCs enable monitoring and controlling of the SGCS. Overall, the automation system operates with 312 registers that hold actuator setpoints and read sensor inputs.

Table 1 summarizes holding registers and the essential inputs associated with the rudder steering. For each entry, the table reports the address, a brief description, the unit of measurement, and the accepted values. Specifically, addresses starting with *IW* denote input registers, while those beginning with *QW* designate holding registers.

In short, the writable *desired rudder angle* register accepts inputs aligned with the commands from the helm. The *current rudder angle* register holds the current angle of the rudder. The *operating mode* sets which PUs are to be used by the system: left (0), right (1), or both (2). The writable *valve command* registers control the flow across their four ports. In position 0, all ports are equalized together. Instead, the other positions split the bottom flows by sending it either straight through (-1) or to the opposite side (1) of the valve. The *pump governors* set the target rotational speed of the PUs.

Table 2
Rudder behavior associated with valves and PUs states.

Valve position			Active PUs	Action
Mix	L	R		
-1	-1	0	Left	To starboard
-1	1	0	Left	To port
-1	0	0	Left	To center
-1	0	1	Right	To starboard
-1	0	-1	Right	To port
-1	0	0	Right	To center
0	-1	1	Both	To starboard
0	1	-1	Both	To port
0	0	0	Both	To center

Afterward, a set of registers gathers sensor data from rams, pumps, valves, and tanks across the left and right circuits. They measure data of interest for system monitoring, such as the current rotational speed of the PUs, hydraulic pressure, flow rate, and tank levels.

The automation system runs a cyclic process that reads the desired rudder angle, calculates the deviation from the current rudder angle, and corrects the deviation by adjusting the rudder angle. This repositioning process involves maneuvering the rudder to the port, center, or starboard positions by governing the valves through the associated registers.

Table 2 illustrates the potential rudder actions based on the configurations of the left, right, and mix valves and according to the number of active PUs. It is worth noting that the valves can assume configurations beyond those outlined in the table, but activating these configurations could lead to significant system anomalies and failures.

5 Methodology

In this section, we delve into the inner workings of our methodology. Figure 3 illustrates its workflow, which the stealth malware can exploit to conduct the attack. We break it down into three main phases, inspired by the cyber kill chain [32]. During the *reconnaissance* phase, the malware begins by discovering distinctive aspects of the onboard automation system responsible for monitoring and controlling the SGCS, e.g., registers it uses and data formats. These data are required to create logs that are suitable to be ingested by the model discovery algorithm. During the *weaponization* phase, the malware employs a technique to reverse engineer the operational procedure for controlling the rudder utilized by the automation system. Subsequently, the malware can craft an attack that alters the ship's direction. Finally, in the

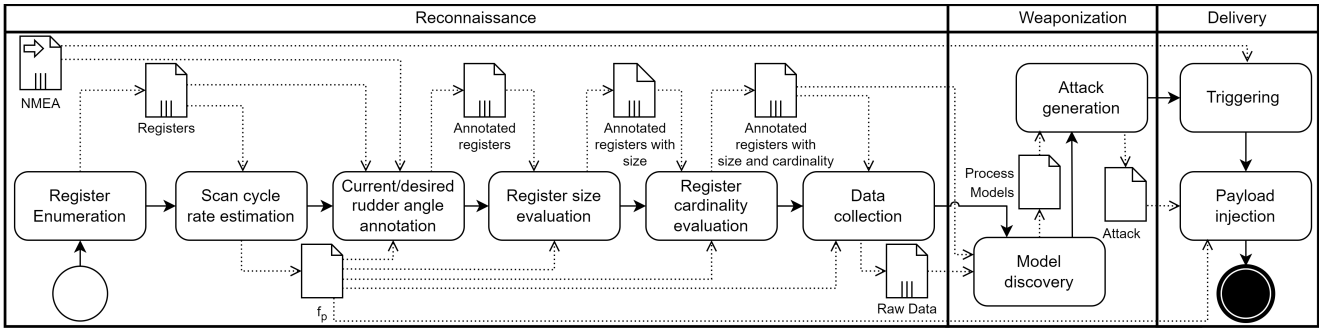

Figure 3: Workflow of the stealth malware.

delivery phase, the malware identifies the appropriate time to inject the weaponized packets into the automation network and execute the attack.

The remainder of this section details each of the tasks included in the above phases.

5.1. Register enumeration

This task includes the identification of the specific registers utilized by the SGCS automation system within the available ones. As Modbus lacks a built-in discovery mechanism, attackers must implement a method for this purpose, by often resorting to enumeration techniques, i.e., scanning the complete address range of the Modbus protocol and monitoring successful read attempts. Common scanning tools such as NMAP [63] leverage this approach.

This task ends with generating a list of registers with their respective types (see Section 2.4).

5.2. Scan cycle rate estimation

Each PLC operates according to a scan cycle (see Section 2.3) that runs at a fixed rate f_p. This scan cycle rate is unknown to attackers. Determination of this rate is required during both the weaponization and delivery phases. Attackers can ensure a consistent and accurate sampling interval by aligning their reads after each write during weaponization. In the delivery phase, taking control involves hijacking legitimate values right after they are written.

The estimation of f_p can be performed by sampling one of the entries from the register list at a greater rate f_s. Specifically, the recurrence relation used for updating the current scan cycle rate estimate $\hat{f}_p$ is

$$\left[\frac{1}{\hat{f}_p}\right]_t = \begin{cases} \alpha\frac{n_t}{f_s} + (1-\alpha)\left[\frac{1}{\hat{f}_p}\right]_{t-1} & \text{if } x_t \neq x_{t-1}, \\ \left[\frac{1}{\hat{f}_p}\right]_{t-1} & \textit{otherwise.} \end{cases}$$

This update is performed every time the register value x changes, performing an exponential averaging operation (with smoothing factor α) on the number of samples n_t since the last update. Estimation completes once $\hat{f}_p$ receives an update smaller than a threshold, e.g., within 5% of its previous value.

Figure 4 shows the convergence of $\hat{f}_p$ for different values of f_s over 5000 tests in our case study. Notably, increasing

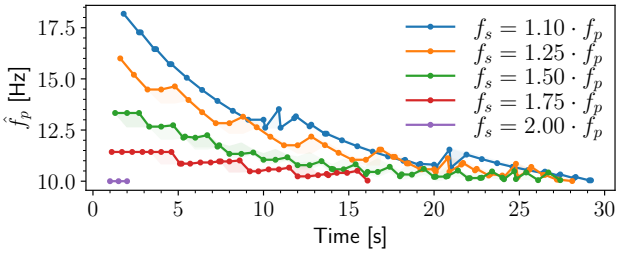

Figure 4: Relation between f_s and convergence of $\hat{f}_p$ ($f_p = 10$).

f_s correlates with shorter convergence times for $\hat{f}_p$. For attackers, selecting f_s involves balancing the time needed to discover $\hat{f}_p$ against the volume of rogue traffic generated when sampling a register. The task ends in delivering the estimated $\hat{f}_p$.

5.3. Current/desired rudder angle annotation

As described in Section 4, the automation system controlling the SGCS provides an input register keeping the current rudder angle and a holding register used to designate the desired rudder angle. In this task, we identify their corresponding entries in the registers list.

In particular, this task relies on listening to the broadcasted NMEA traffic and capturing the two sentences that transport the matching values: (*i*) RSA (Rudder Sensor Angle) indicating the current angle and (*ii*) ROR (Rudder Order Status) representing the desired angle. These sentences feed different equipment in the navigation network, such as the autopilot or the conning display, i.e., the station that oversees and manages the vessel's navigation and maneuvering.

The identification occurs by correlating RSA and ROR sentences with the values returned by input and holding registry entries in the registers list, respectively. In detail, attackers synchronize time-stamped RSA and ROR values with the data gathered from the SGCS registers. Then, they compute the Pearson correlation coefficient for each potential match. Finally, the register with the highest correlation coefficient is assumed to hold the sought quantity.

We tested the feasibility of the approach on data from our case study. Analysis of the time-synchronized data points shows that the highest correlation with RSA sentences is found in the IW101 register, suggesting that IW101 is a linear transformation of the rudder angle. Similarly, ROR sentences have the highest correlation with the QW100 register. Both studies demonstrate a high level of correlation ($R^2 > 0.999$) in accurately identifying the correct registers.

5.4. Register size evaluation

As stated in Section 2.4, multiple analog input and output holding registers can collectively store 32- or 64-bit data. Therefore, registers may vary in size, extending beyond the default 16-bit.

Algorithm 1 outlines our heuristic for evaluating register sizes, featuring an *Evaluate* procedure applied to analog input and output holding entries in the registers list. This

Algorithm 1 Register size evaluation

```
1:  procedure GatherHistory(regs)
2:      h ← ∅
3:      while |h| ≤ k_hist do
4:          v_{i=1...|regs|} ← ReadAll(regs)
5:          Append(h,v)
6:          Sleep(1/f̂_p + δ)
7:      end while
8:      return h
9:  end procedure
10: procedure IsIEEE754(h,bits)
11:     maybe_754 ← ∅
12:     for i in 1 ... |h| do
13:         v ← h_i
14:         if IsZero(v) then
15:             Append(maybe_754,i)
16:         else if IsNormal(v) and k_ϵ ≤ x ≤ k_big then
17:             Append(maybe_754,i)
18:         end if
19:     end for
20:     return maybe_754 ≥ k_754|h|
21: end procedure
22: procedure Evaluate(r)
23:     h_f32 ← GatherHistory({r, r + 1})
24:     h_f64 ← GatherHistory({r, r + 1, r + 2, r + 3})
25:     if IsIEEE754(h_f32,32) then return f32
26:     else if IsIEEE754(h_f64,64) then return f64
27:     else
28:         return integer
29:     end if
30: end procedure
```

procedure determines whether the value stored in register r corresponds to a float 32 by referencing the next register or a float 64 by considering the subsequent three registers for its representation.

It starts by calling the *GatherHistory* procedure that returns a list of historical values h associated with two (Line 23) or four registers (Line 24). This history is formed by gathering k_{hist} values (Line 3), achieved through periodic reads contents of the registers (Line 4-5) at short intervals of δ in relation to the scan cycle rate f_p (Line 6).

The procedure *IsIEEE754* tries to decode each entry of the above histories as a floating-point number first considering 32-bit (Line 23) and then 64-bit (Line 24) aggregations of registry values. It checks how many of these aggregations are either zeros (Line 14) or values in $[k_\epsilon, k_{big}]$ (Line 16). The definitions for *IsZero* and *IsNormal* align with the specifications outlined in the IEEE 754 standard [33]. Then, it identifies a register as a floating-point number when over k_{754} percent of these values (Line 20) meet either of these criteria. Finally, if all other classification attempts fail, it defaults to the single-register data, i.e., an integer (Line 28).

This task ends by updating the analog input and output holding entries with their size in the annotated register lists.

5.5. Register cardinality evaluation

This task is intended to assess the cardinality of the registers, to identify their diversity and the type of data they can hold within the SGCS system.

Attackers initiate this task by querying values from analog inputs and outputs holding entries in the register list. The task then involves capturing their values or states over multiple cycles. This data collection enables the creation of a comprehensive dataset that records the variations and patterns exhibited by the registers over time. From this dataset, attackers can apply analysis and statistical methods, such as frequency counting or variance examination across cycles, to derive an estimation of the registers' cardinality. The task ends by updating the registers list with their respective estimated cardinality.

To evaluate the accuracy of cardinality estimation in our case study, we gathered 1000 consecutive segments, each with a duration of at least two hours. For each segment and register, we determined its actual cardinality and measured the time required to accurately estimate its value. Our method involves using bit vectors, with each bit indicating the presence of a value in the set. As we process the data, we mark the bit corresponding to each element. This approach has a time complexity of $O(n)$, where n is the set size, and space complexity of $O(m/8)$, with m being the range of possible values (e.g., 65536 for a Modbus register). This method is ideal for our scenario, as it deals with values within a finite range and prioritizes space efficiency.

Figure 5 illustrates the time taken to accurately estimate the cardinality of different registers. The process quickly converges to the correct value in cases where registers are almost homogeneous. It takes about 500 seconds to estimate the cardinality of half of the identified registers. A rough estimate of 2 hours is sufficient for all registers to accurately determine their cardinality.

5.6. Data collection

This task completes the reconnaissance phase by collecting data from the SGCS system automation for the model discovery task. It works similarly to the *GatherHistory* procedure of Algorithm 1. The main difference is that attackers

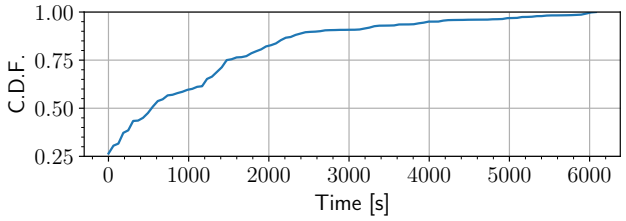

Figure 5: Cumulative probability distribution of time required for correct cardinality estimation.

collect the final sensor and actuator values, considering the size specified in the registers list. The output is raw data from readings across multiple operational cycles. Each entry comprises the timestamp of the reading, the register ID, and its current value. For values using multiple registers, the ID of the first register is taken as the reference for the entry.

5.7. Model discovery and attack generation

We now describe our method for supporting a malicious reverse engineering activity, which aims to derive a model of the SGCS behavior. This is then used to build attacks.

Preprocessing

As a first step, the analysis of ship dynamics requires transforming collected raw data into an event log suitable for process mining techniques.

Inspired by the work done in [57], we start by introducing the concept of a device status record, which represents an individual entry that is generated by a SCADA device.

Definition 1 (Device Status Record)**.** *Let T, VN, and RA denote sets of timestamps, variable names, and attribute names, respectively. In addition, let VV_v denote the set of possible values of the variable named $v \in VN$. A* device status record *r is a tuple of attribute name/value pairs. The value of attribute $a \in RA$ for device status record r is denoted by $a(r)$. Every device status record r has at least the following attributes:*

- *$time(r) \in T$ is the timestamp of the record;*
- *$vName(r) \in VN$ is the variable name in the record;*
- *$vVal(r) \in VV_{vName}$ is the value in the record.*

We denote the set of all possible device status records as R.

Definition 2 (Device Log)**.** *A device log $L \subseteq R$ is a set of device status records.*

A device log represents the sequential recording of system states and activities derived from the various registers' readings. Table 3 illustrates a portion of a device log, including the values recorded for different variables during two consecutive scan cycles.

A first filtering step eliminates the records referring to $vName$s that are not of interest for the process models we will build as a basis for attacks. Specifically, we eliminate the records referring to input coils and input registers (see Section 5.1) except for the current rudder angle (Section 5.3). In addition, we remove the records referring to the desired rudder angle (Section 5.3).

Table 3
Example portion of a raw device log.

time	vName	vVal
00:00:00	QW100	32768
00:00:00	IW101	32768
00:00:00	QW102	0
00:00:00	QW103	0
00:00:00	QW104	32768
00:00:00	QW105	32768
00:00:00	QW106	32768
00:00:00	QW107	32768
00:00:01	QW100	32768
00:00:01	IW101	32768
00:00:01	QW102	0
00:00:01	QW103	0
00:00:01	QW104	32768
00:00:01	QW105	32768
00:00:01	QW106	32768
00:00:01	QW107	32768

We also eliminate records that represent "non-events", i.e., records where the register's value remains unchanged with respect to previous cycles. This may happen often in SCADA systems, as such systems frequently generate logs with periodic status updates, which in turn include records that are not indicative of significant operational events. Thus, for each scan cycle and each $vName(r)$, we assess whether $vVal(r)$ has changed. Otherwise, the corresponding record is classified as a non-event and filtered out.

Next, we derive *activity names* from the raw data.

Definition 3 (Activity Name Construction)**.** *Given a device status record with a variable name $v \in VN$ and its associated value $val \in VV_v$, the activity name is obtained as follows:*

- *If v is a numerical variable (i.e., $|VV_v| > 3$), the activity name indicates the change relative to the last recorded value for v. Specifically, we denote an increase in value by "v_Increasing" and a decrease by "v_Decreasing".*
- *If v is a boolean or ternary variable ($|VV_v| \leq 3$), the activity name captures the specific transition from the previous recorded value. Specifically, we use "v_(previousvalue)_to_(currentvalue)" as activity name.*

If a previous value is not in the log, the activity name for v is not specified. The set of all possible activity names is denoted as A.

Observe that $|VV_v|$ corresponds to the cardinality of the registers (see Section 5.5). In our example, after this step,

Table 4
Example portion of a device log with activity names.

time	vName	vVal	activity
00:00:00	QW100	32768	None
00:00:00	IW101	32768	None
00:00:00	QW102	0	None
00:00:00	QW104	32768	None
00:00:00	QW105	32768	None
00:00:00	QW106	32768	None
00:00:00	QW107	32768	None
00:00:08	QW100	16103	QW100_Decreasing
00:00:08	QW104	65536	QW104_32768_to_65536
00:00:08	QW106	65536	QW106_Increasing
00:00:09	IW101	32458	IW101_Decreasing
00:00:10	IW101	31832	IW101_Decreasing
00:00:11	IW101	31102	IW101_Decreasing
00:00:12	IW101	30227	IW101_Decreasing
00:00:13	IW101	29325	IW101_Decreasing
00:00:14	QW100	31550	QW100_Increasing
00:00:14	IW101	28477	IW101_Decreasing

we get the log shown in Table 4—we then remove records without an activity name.

We now introduce *case identifiers*. Intuitively, a case identifier associates a subset of status records to a distinct operational cycle. This association (i) is based on the assumption that the log follows a cycling logging paradigm and (ii) makes use of specific status records that correspond to the end of a logging cycle — specifically, those records with a distinct variable name $v_{sc} \in VN$. In our case study, v_{sc} is IW101 (see Section 5.3).

Definition 4 (Case Identifier)**.** *Given a device log L, a* case identifier *is a function $cid : L \to \mathbb{N}$. Consider the set $\{r_1, r_2, \ldots, r_k\} \subseteq L$ of device status records such that $\forall i \in [1,k]$, $vName(r_i) = v_{sc}$. Then:*

- $\forall r \in L$ *s.t.* $time(r) \leq time(r_1)$, $cid(r) = 0$.
- $\forall i \in [1,k]$, $\forall r \in L$ *s.t.* $time(r_i) < time(r) \leq time(r_{i+1})$, $cid(r) = i$.

After this step, we get the log shown in Table 5. We then remove "singleton cases" (i.e., cases with only one event), as they are irrelevant from a process mining perspective.

Finally, we remove attributes $vName$ and $vVal$ and leverage domain knowledge (which is assumed to be accessible to informed attackers) to partition the log into two sub-logs: one containing the cases that end with v_{sc}*_decreasing* and another with the cases that end with v_{sc}*_increasing*. Log partitioning is a common practice in process mining, especially when dealing with unstructured processes. It also helps reduce the complexity of the models discovered later from the logs. Tables 6 and 7 show the initial portions of the final event sub-logs.

Extraction and generation

These steps aim to understand the intrinsic process structure behind the SGCS, leveraging the process model extracted from the event log to automate the generation of attacks.

Several key factors influence the choice of a process discovery algorithm. The first factor is the algorithm's performance in terms of computation time, an aspect that directly affects its practical applicability. Another critical factor is memory usage, especially in our target scenario, where resources are limited. Finally, yet equally important, the process model's representation should be comprehensible and interpretable within the given context.

Along these lines, the *Heuristics Miner* algorithm [75] emerges as a suitable choice for our analysis. This algorithm, which is very popular in the process mining community, begins by constructing a dependency graph and applies heuristics to assign a causal score to each edge of the graph. The causal score provides a formal measure that quantifies the dependency relationship between two activities. This score essentially distinguishes genuine sequential dependencies from instances of interleaved concurrency. Another notable strength of Heuristics Miner lies in its resilience to minor data variations, emphasizing the identification of predominant dependencies.

Applying this algorithm to the event log, we are able to successfully model the operational dynamics of the SGCS. The resulting models reflect the system's standard behavior patterns and are crucial to our subsequent automatic attack generation.

Figure 6 provides a visual representation of possible models extracted from the final event sub-logs of our example scenario. Note that this visualization validates our algorithmic choice and lays the basis for developing targeted and automated attack strategies. Indeed, by mirroring the operations observed along the most frequent path in the process model —from the starting to the terminating activity— we are equipped with a way to automate the crafting of an attack. The attack consists of a sequence of specific Modbus packets appropriately designed to modify the values associated with the targeted registers according to operational patterns identified in the process model. The process models visualized in Figure 6 can be used to control the steering gear exactly like the original PLC program.

As an example attack scenario, consider a situation where the SGCS is operating under normal conditions. Upon reaching the triggering conditions, the malware activates and starts to deliver Modbus packets to the key registers of the SGCS. An example attack based on the IW101_Increasing model of Figure 6 is shown in Table 8. The attack starts by injecting the Modbus registers associated with the valve actuators, specifically QW104 and QW105. The malware injects valid but malicious Modbus packets having the function code "write single discrete output coil" (0x05) (Lines 1–2 in Table 8) so that these registers are set to atypical operational states (0.0 and 65536, respectively). Subsequently, the malware injects "read discrete output coil" packets (function code 0x01) in order to retrieve the current state of the governors associated with registers QW106 and QW107 (Lines 3–4). These two actions (h_1, h_2) are "hidden activities" in the sense that they do not appear in the process model (because they are not registered in the log),

Table 5
Example portion of a device log with activity names and case identifiers.

time	vName	vVal	activity	caseid
00:00:08	QW104	65536	QW104_32768_to_65536	0
00:00:08	QW106	65536	QW106_Increasing	0
00:00:09	IW101	32458	IW101_Decreasing	0
00:00:10	IW101	31832	IW101_Decreasing	1
00:00:11	IW101	31102	IW101_Decreasing	2
00:00:12	IW101	30227	IW101_Decreasing	3
00:00:13	IW101	29325	IW101_Decreasing	4
00:00:14	IW101	28477	IW101_Decreasing	5
00:00:14	QW104	0	QW104_65536_to_0	6
00:00:14	QW106	35607	QW106_Decreasing	6

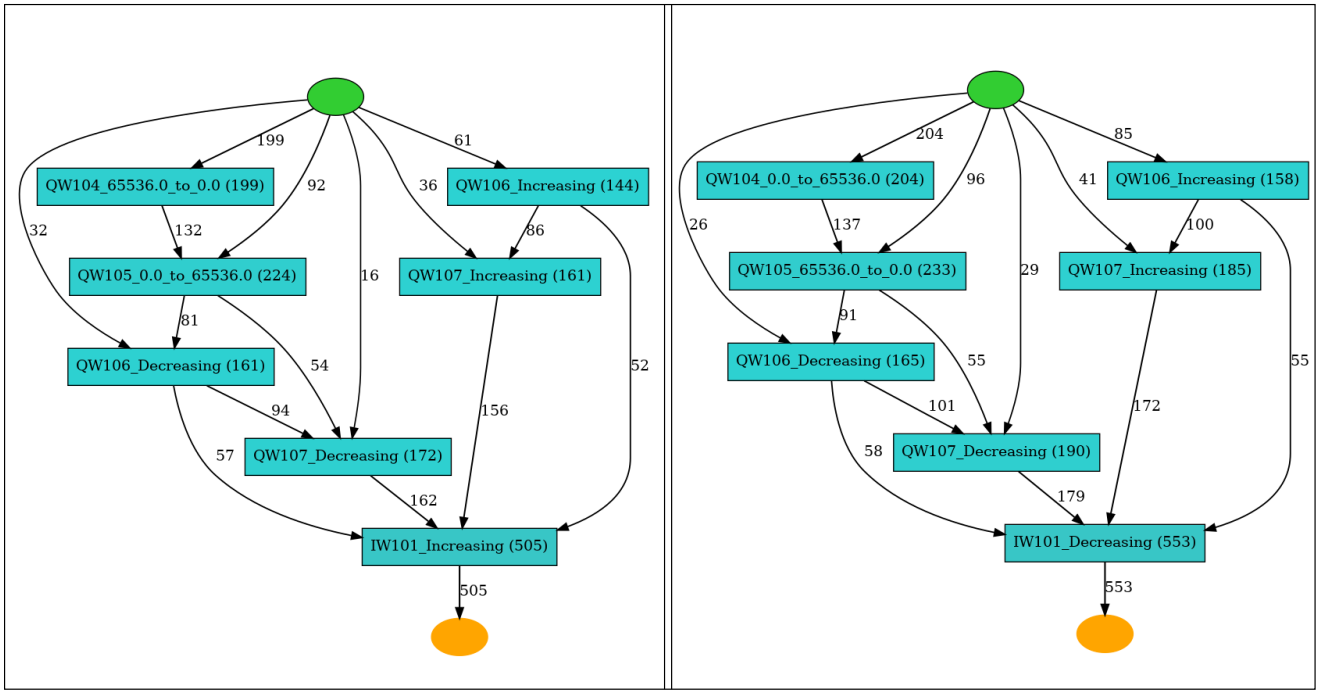

Figure 6: Example process models for the final event sub-logs.

yet they play a role in the enactments.[1] In fact, h_1 and h_2 activate the events on Lines 5 and 6, respectively. After the measurements readings of QW106 and QW107, the attack proceeds by sending "write single discrete output coil" packets to manipulate the operation of the pumps as much as needed to deliver the desired effect (e.g., a significant alteration in the ship's direction).

The execution of the sequence in the attack leads to a disruption in the normal operation of the SGCS.

5.8. Triggering and payload injection

This task leverages the capability to receive NMEA traffic and analyze sentences to track the ship telemetry, nearby vessels, tracked targets, and weather conditions. These data can help determine the most suitable time to launch an attack and maximize the attackers' chances of achieving their objectives. For example, attackers can determine if the GPS position is on a dangerous route, if the ship is navigating through congested areas, or if it is encountering potentially limited visibility due to nighttime or weather conditions.

When it is time to execute, the malware processes the incoming attack file. It extracts the payload from each line (see Table 8) and sends it to the SGCS RTU via the Modbus protocol. This transmission utilizes the computed f_p, injecting payloads immediately after the scan cycle executes the write outputs operation.

6 Experimental Evaluation

6.1. Experimental setup

All of the data used in this paper was gathered in a simulation setting, partially inspired by [44]. The vessel's

[1] In process discovery applications, hidden activities are frequently used to enrich models [25].

Table 6
Example portion of the final event sub-log (with IW101_decreasing).

time	activity	caseid
00:00:08	QW104_32768.0_to_65536.0	9
00:00:08	QW106_Increasing	9
00:00:09	IW101_Decreasing	9
00:00:53	QW106_Increasing	54
00:00:53	QW107_Increasing	54
00:00:54	IW101_Decreasing	54
00:01:54	QW104_0.0_to_65536.0	115
00:01:54	QW105_65536.0_to_0.0	115
00:01:54	QW106_Decreasing	115
00:01:54	QW107_Decreasing	115
00:01:55	IW101_Decreasing	115
00:02:06	QW106_Increasing	127
00:02:06	QW107_Increasing	127
00:02:07	IW101_Decreasing	127

Table 7
Example portion of the final event sub-log (with IW101_increasing).

time	activity	caseid
00:00:14	QW104_65536.0_to_0.0	15
00:00:14	QW106_Decreasing	15
00:00:15	IW101_Increasing	15
00:00:31	QW102_0.0_to_2.0	32
00:00:31	QW103_0.0_to_32768.0	32
00:00:31	QW105_32768.0_to_65536.0	32
00:00:31	QW107_Increasing	32
00:00:32	IW101_Increasing	32
00:00:42	QW104_0.0_to_65536.0	43
00:00:42	QW105_65536.0_to_0.0	43
00:00:43	IW101_Increasing	43

movements were simulated according to [52] and the SGS model presented in Section 4. These simulations were conducted using OpenModelica [21] (with a tolerance of 10^{-9}), yielding to a balanced system containing 2095 differential equations and variables. In addition, two autopilots control the boat telegraph and desired rudder angle, allowing the simulated vessel to navigate according to pre-planned routes. Figure 7 depicts the simulation setup. The quantities and setpoints in the physics simulation are then exposed to a simulated ship network either as NMEA sources or via the Modbus RTU.

In our experimental evaluation, we drew inspiration from the *Ever Given* container ship grounding incident to simulate a realistic scenario. This grounding, a notable event that disrupted global shipping in 2021, saw the ship becoming lodged in the Suez Canal.

To assess the effectiveness of the reconnaissance phase, we simulated 1000 ship transits through the Suez Canal under conditions similar to those held during the incident. These simulations were equally divided between the canal's northbound and southbound routes, as officially designated

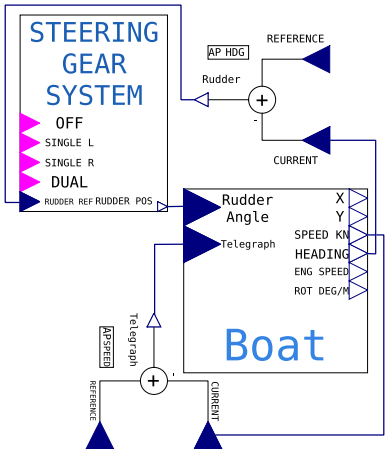

Figure 7: Simulation setup.

by the Suez Canal Authority [67]. Figure 8 displays the ship's positional data over time during these simulated transits.

For each of these transits, we follow the workflow of Figure 3 until the *Triggering* task. As such, each experiment simulates a distinct reconnaissance performed on a distinct data set. This data collection process emulates the one of malware programmed not to trigger during its first canal crossing. Then, we reinitialize the vessel status at a random point associated with the return voyage and execute the *Payload injection* task — finally, we record the effects of the malicious actions on the simulation. The simulation ends once the ship veers off the canal center line by more than 500m, a distance higher than the maximum canal width, indicating an inevitable collision.

Experiments were performed on a Fedora 39 virtual machine equipped with 128GB RAM, running on a Proxmox hypervisor with dual Intel Xeon E5-2699v4 CPUs. This hardware configuration, required to run the compute-intensive simulation process satisfactorily, is not representative of the onboard environment targeted by the malware. To address this, we simulate a less powerful workstation by executing the malware in a controlled environment. This is achieved through a restrictive *cgroup*, which limits the malware to using no more than 2 CPU cores with a maximum of 5% utilization per core and restricts the available RAM to 4GB.

6.2. Results

The experimental assessment results confirm that our methodology consistently allows us to identify the appropriate setpoints associated with a given rudder actuation. This is clearly shown in Figure 9, which captures the ship's trajectories during the simulated attacks. In the figure, two zoomed-in sections highlight the results of attacks associated with two sequences of events (see Figure 6). The leftmost zoom shows an attack associated with *current rudder angle is increasing*, and the second zoom depicts an attack aimed at reproducing the *current rudder angle is decreasing* condition. Regardless of the specific trajectory,

Table 8
Example attack.

#	Event	Payload	Description
1	QW104_65536_to_0.0	07d40000	Inject *L Valve command register* with "write single discrete output coil" packet to set 0.0 value
2	QW105_0.0_to_65536	07d5ffff	Inject *R Valve command register* with "write single discrete output coil" packet to set 65536 value
3	h_1	07d60001	Sends a "read discrete output coil" packet to *L Pump Governor*
4	h_2	07d70001	Sends a "read discrete output coil" packet to *R Pump Governor*
5	QW106_Decreasing	07d65d7a	Sends a "write single discrete output coil" packet to *L Pump Governor*
6	QW107_Decreasing	07d77de8	Sends a "write single discrete output coil" packet to *R Pump Governor*

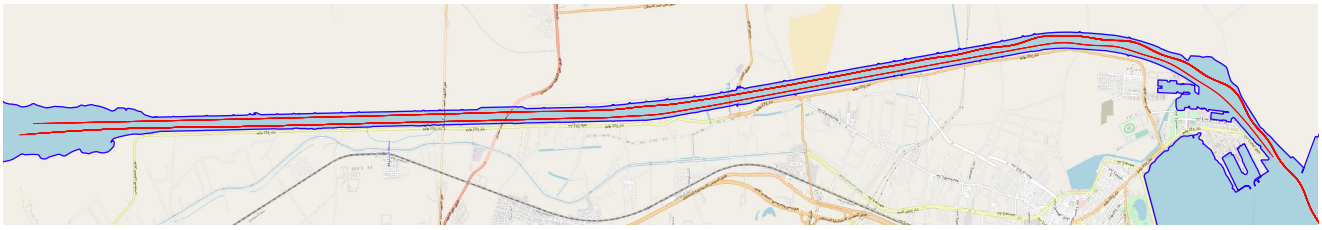

Figure 8: Traces of the 1000 voyages along the Suez Canal.

each scenario results in the ship's inevitable collision with the canal border walls within approximately two minutes. Red markers indicate such impact points on the map.

Further insights are provided in Figure 10, which details the ship's angular speed as the attack unfolds. This visualization reveals that the attack's effect is analogous to either a hard to starboard or a hard to port maneuver. The correct identification of the action sequence allows the quick movement of the rudder blade to its maximum angle, steering the ship into the attackers' desired collision course.

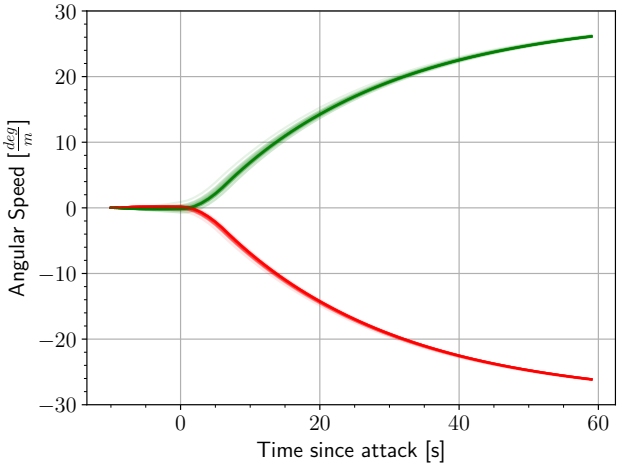

Figure 10: Resulting angular speed. Values referring to attacks associated with increasing (resp., decreasing) rudder angle are shown in green (resp., red).

Focusing instead on the attack effects on the SGS itself, Figure 11 illustrates the peak hydraulic fluid pressure observed within the four ram chambers. Upon initiation of the attack, the chamber is quickly energized. The plot captures the increase in water resistance associated with the rudder's change in angle of attack. This resistance peaks when the rudder reaches its maximum extension and hits the end-of-travel damper approximately 16 seconds post-attack. Subsequently, the system rebounds from the pressure drop triggered by this abrupt movement.

Table 9 presents the average measured costs (runtime, memory usage, and storage usage) for the Reconnaissance and Weaponization phases. Runtimes were measured using the system's monotonic clock, while memory and storage were monitored using Python's *tracemalloc* and *os* modules, respectively. The data collection task, lasting the entire canal voyage, is the most time-intensive activity, spanning several hours. This task also requires approximately 200 MiB of storage to record a full voyage, which is relatively modest in terms of space requirements. All other tasks are completed in less than 10 minutes and require no storage space and less

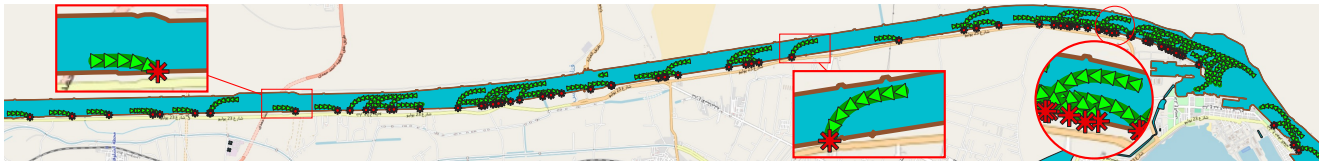

Figure 9: Resulting traces of trajectories and impacts.

Table 9
Average costs.

Phase	Reconnaissance					Weaponization
Task	Enumeration	Rate	Rudder annotation	Size/cardinality evaluation	Data collection	Model discovery
Runtime [s]	68.52	11.38	42.32	3,400.21	6,035.39	70.72
Memory [MiB]	17.30	0.30	6.14	44.02	0.84	29.11
Storage [MiB]	–	–	–	–	180.15	–

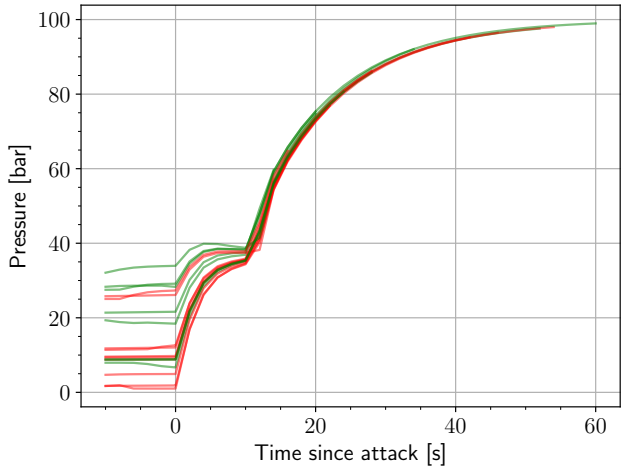

Figure 11: Resulting pressure on ram chambers affected by the attack. Values referring to attacks associated with increasing (resp., decreasing) rudder angle are shown in green (resp., red).

than 100 MiB of memory. This confirms that our proposed approach can be effectively executed on devices with limited resources, such as those in a ship's computer systems.

7 Remediations

The class of attacks similar to the one designed in this paper exploits a set of inherent configurations and weaknesses of onboard systems, making it impractical to recommend a single and harmonized remediation. Instead, it is necessary to consider a series of actions guided by the principle of reducing risks to a level deemed as "Low As Reasonably Practicable" (ALARP).

On the prevention side, preparedness and proactive assessment are strategic to anticipate and address potential cyber attack risks. To this aim, the International Maritime Organization has released specific guidelines [39] to safeguard shipping from current and emerging cyber threats and vulnerabilities. Several other organizations with an interest in the maritime sector have also published best practice documents [8, 16, 34, 58] to address such risks and reduce the likelihood of compromise.

These documents emphasize the significance of regularly conducting vulnerability assessments and implementing thorough defense-in-depth practices. The first activity aims to reduce the possibility of exploiting known vulnerabilities to penetrate the onboard infrastructure. The second isolates the different subsystems into minimal-privilege zones by deploying various security layers, such as network segmentation and firewalling. It prevents attackers from executing further later movements within the infrastructure, restricting their access to critical systems like SGCS, even after an initial breach.

Considerable focus is on enhancing maritime personnel's awareness and training regarding cyber risks. The human operators must be capable of distinguishing such risks from common anomalies and must understand their potential impact. In particular, providing operators with efficient contingency plans seamlessly integrated into their operating procedures is crucial to ensure the safety and reliability of the involved systems.

Another effective mitigation strategy involves monitoring vulnerable systems by collecting, correlating, and analyzing data from sources like navigational sensors, automation systems, logs, and network traffic. Early detection and response can limit the impact or even prevent significant damage from an attack [19, 20].

Several solutions showcase the utilization of process mining techniques for monitoring ICS event logs and detecting anomalies. A prominent approach encompasses the conformance checking activities, i.e., comparing logged events against a model of the expected behavior. The primary objective is to identify system data utilization and control-flow patterns discrepancies [4, 57]. Process-oriented methods monitor deviations against constraints or expected values of a process variable [11, 13, 56, 59]. For example, an alert might be triggered when a valve pressure falls outside the expected range of values observed over time. For known attack patterns, an alternative approach consists of creating the corresponding process model and possible new variants to identify attack traces in real time [14, 29]. Finally, physics-aware ICS honeypots are an effective countermeasure for detecting the presence of attackers early on and discovering their strategies [46, 50].

8 Related Work

This paper outlines an attack that is specifically designed for the maritime domain. Its objective is to disrupt or interfere with navigation by deriving a model of the automation process of the industrial system that controls the steering

gear and interfering with it. Related work includes studies that present and analyze cyber-attacks against shipboard systems with the above objective and against broader OT systems using similar techniques to the one we introduced.

Numerous attacks in the maritime context exploit the lack of authentication and confidentiality of the NMEA protocol [48, 71] to hijack naval sensor data. The Bridge Attack Tool [31] serves as a cybersecurity assessment tool for integrated bridge systems and provides a taxonomy and implementation of attacks. Notably, it demonstrates that the effects on nautical equipment can result in inaccurate estimations, potentially leading navigators to make incorrect and harmful decisions. Similar issues in network protocols used by RADAR systems [45, 76, 43] can be exploited to introduce deceptive vessels on the RADAR display. Again, this false information can cause operators to make incorrect decisions, especially during collision avoidance maneuvers. Our malware also leverages the weaknesses of NMEA, but the attack directly compromises the steering system without relying on human error. For this reason, our approach significantly amplifies the likelihood of success and the inherent danger.

Regarding attacks targeting OT systems, existing research has mostly explored the exploitation of operating systems weaknesses, firmware vulnerabilities, flaws in industrial protocols, and methods to bypass traditional security mechanisms [2, 7, 60]. While these works effectively uncover general vulnerabilities that could be exploited on the PLC regardless of its managed processes, they typically assume that attackers have the capability to upload new control logic onto the target system. Furthermore, these attacks often result in generalized impacts, such as denial of service.

In order to orchestrate finely targeted attacks that can have sustainable impacts on target systems beyond denial of service, understanding operational processes and their significance is a crucial aspect [24, 27, 28, 65]. This approach requires a deep comprehension of the system's operational behavior, achieved by dynamically analyzing observed operational data and event logs. Although the potential of such approaches has already been demonstrated [22], extensive research on practical physics-aware attacks still needs to be done. A limitation of current studies is that they suppose a specific degree of visibility and authority within industrial systems and the ability to list control logic by monitoring PLC memory, which limits viability in real-world attack scenarios. Recent advancements [12] include methodologies for deriving approximate models of controlled physical processes. Our work shares some similarities but differs in applying process mining techniques to extract actionable process knowledge. Moreover, we expand on prior findings by advancing the attacks based on standalone malware with automated reconnaissance, payload generation, and attack initiation. To the best of our knowledge, our application of this approach is the first instance within the maritime context.

9 Conclusions

In this paper, we have looked at cybersecurity issues of Industrial Control Systems used onboard ships, whose latest advancements result in increased security risks. After defining a threat model and a specific case study, we have proposed a methodology that, when adopted by targeted malware, can build attacks that have disruptive consequences on the ships. We have demonstrated our approach's effectiveness (and efficiency) through extensive experiments in a real-world scenario. Although this work sheds light on the potential for attackers to exploit specialized malware designed to disrupt the physical operations of ships, there is a significant need for further research to develop robust defense strategies against such threats. To address this, in future work, we will explore the integration of process mining-based detection techniques of physics-aware attacks with game theory frameworks [26, 64]. Furthermore, to model the behavior of malware more accurately, we plan to study more sophisticated and richer representations, such as declarative process models [6]. Indeed, this approach combined with *explainable AI* and effective application of the Shapley value [51], could allow to create models that are not only more accurate but also human-understandable, thereby improving their practical application in our defensive strategies against specialized malware.

References

[1] van der Aalst, W.M.P., Adriansyah, A., de Medeiros, A.K.A., Arcieri, F., Baier, T., Blickle, T., Bose, R.P.J.C., van den Brand, P., Brandtjen, R., Buijs, J.C.A.M., Burattin, A., Carmona, J., Castellanos, M., Claes, J., Cook, J.E., Costantini, N., Curbera, F., Damiani, E., de Leoni, M., Delias, P., van Dongen, B.F., Dumas, M., Dustdar, S., Fahland, D., Ferreira, D.R., Gaaloul, W., van Geffen, F., Goel, S., Günther, C.W., Guzzo, A., Harmon, P., ter Hofstede, A.H.M., Hoogland, J., Ingvaldsen, J.E., Kato, K., Kuhn, R., Kumar, A., Rosa, M.L., Maggi, F.M., Malerba, D., Mans, R.S., Manuel, A., McCreesh, M., Mello, P., Mendling, J., Montali, M., Nezhad, H.R.M., zur Muehlen, M., Munoz-Gama, J., Pontieri, L., Ribeiro, J., Rozinat, A., Pérez, H.S., Pérez, R.S., Sepúlveda, M., Sinur, J., Soffer, P., Song, M., Sperduti, A., Stilo, G., Stoel, C., Swenson, K.D., Talamo, M., Tan, W., Turner, C., Vanthienen, J., Varvaressos, G., Verbeek, E., Verdonk, M., Vigo, R., Wang, J., Weber, B., Weidlich, M., Weijters, T., Wen, L., Westergaard, M., Wynn, M.T., 2011. Process Mining Manifesto, in: Business Process Management Workshops - BPM 2011 International Workshops.

[2] Abbasi, A., Hashemi, M., Zambon, E., Etalle, S., 2016. Stealth Low-Level Manipulation of Programmable Logic Controllers I/O by Pin Control Exploitation, in: International Conference on Critical Information Infrastructures Security, CRITIS, Revised Selected Papers, Springer. pp. 1–12.

[3] Adhane, G.W., Kim, D., 2017. Distributed control system for ship engines using dual fieldbus. Comput. Stand. Interfaces 50, 83–91.

[4] Alizadeh, M., Lu, X., Fahland, D., Zannone, N., van der Aalst, W.M.P., 2018. Linking data and process perspectives for conformance analysis. Comput. Secur. 73, 172–193.

[5] Ariffin, A., Laurens, J., Mansor, S., 2016. Real-time evaluation of second generation intact stability criteria. Proceedings of the RINA, Royal Institution of Naval Architects— Smart Ship Technology .

[6] Bernardi, M.L., Cimitile, M., Maggi, F.M., 2023. Data-aware process discovery for malware detection: an empirical study. Mach. Learn. 112, 1171–1199.

[7] Biham, E., Bitan, S., Carmel, A., Dankner, A., Malin, U., Wool, A., 2019. Rogue7: Rogue Engineering-Station attacks on S7 Simatic PLCs. Black Hat USA .

[8] BIMCO, 2021. The Guidelines on Cyber Security Onboard Ships. URL: https://www.bimco.org/about-us-and-our-members/publications/the-guidelines-on-cyber-security-onboard-ships.

[9] Cain, B., Deering, D.S.E., Fenner, B., Kouvelas, I., Thyagarajan, A., 2002. Internet Group Management Protocol, Version 3. RFC 3376.

[10] Cantelli-Forti, A., Colajanni, M., Russo, S., 2023. Penetrating the Silence: Data Exfiltration in Maritime and Underwater Scenarios, in: IEEE Conference on Local Computer Networks, LCN.

[11] Caselli, M., Zambon, E., Amann, J., Sommer, R., Kargl, F., 2016. Specification Mining for Intrusion Detection in Networked Control Systems, in: USENIX Security Symposium.

[12] Ceccato, M., Driouich, Y., Lanotte, R., Lucchese, M., Merro, M., 2022. Towards Reverse Engineering of Industrial Physical Processes, in: Computer Security. ESORICS International Workshops - Cyber-ICPS, SECPRE, SPOSE, CPS4CIP, CDT&SECOMANE, EIS, and SecAssure.

[13] Colbert, E., Sullivan, D., Hutchinson, S., Renard, K., Smith, S., 2016. A process-oriented intrusion detection method for industrial control systems, in: International Conference on Cyber Warfare and Security, ICCWS.

[14] Coltellese, S., Maggi, F.M., Marrella, A., Massarelli, L., Querzoni, L., 2019. Triage of IoT Attacks Through Process Mining, in: On the Move to Meaningful Internet Systems: OTM Conferences - Confederated International Conferences: CoopIS, ODBASE, C&TC.

[15] Di Pinto, A., Dragoni, Y., Carcano, A., 2018. TRITON: The first ICS cyber attack on safety instrument systems. Proc. Black Hat USA 2018, 1–26.

[16] DNV, 2021. DNV cyber secure class notation. URL: https://www.dnv.com/services/cyber-secure-class-notation-124600.

[17] Erstad, E., Hopcraft, R., Palbar, J.D., Tam, K., 2023. CERP: A Maritime Cyber Risk Decision Making Tool. International Journal on Marine Navigation and Safety of Sea Transportation 17, 269–279.

[18] Falliere, N., Murchu, L.O., Chien, E., 2011. W32.Stuxnet dossier. White paper, Symantec corp., security response 5, 29.

[19] Fortino, G., Greco, C., Guzzo, A., Ianni, M., 2022. Neural Network based Temporal Point Processes for Attack Detection in Industrial Control Systems, in: IEEE International Conference on Cyber Security and Resilience (CSR).

[20] Fortino, G., Greco, C., Guzzo, A., Ianni, M., 2023. Identification and prediction of attacks to industrial control systems using temporal point processes. J. Ambient Intell. Humaniz. Comput. 14, 4771–4783.

[21] Fritzson, P., Pop, A., Abdelhak, K., Ashgar, A., Bachmann, B., Braun, W., Bouskela, D., Braun, R., Buffoni, L., Casella, F., Castro, R., Franke, R., Fritzson, D., Gebremedhin, M., Heuermann, A., Lie, B., Mengist, A., Mikelsons, L., Moudgalya, K., Ochel, L., Palanisamy, A., Ruge, V., Schamai, W., Sjölund, M., Thiele, B., Tinnerholm, J., Östlund, P., 2020. The OpenModelica Integrated Environment for Modeling, Simulation, and Model-Based Development. Modeling, Identification and Control: A Norwegian Research Bulletin 41, 241–295.

[22] Garcia, L., Brasser, F., Cintuglu, M.H., Sadeghi, A., Mohammed, O.A., Zonouz, S.A., 2017. Hey, My Malware Knows Physics! Attacking PLCs with Physical Model Aware Rootkit, in: Annual Network and Distributed System Security Symposium, NDSS.

[23] Gerson, A., 2023. Stranding of the Mega-Ship Ever Given in the Suez Canal: Causes, Consequences, and Lessons to Be Learned. The Suez Canal: Past Lessons and Future Challenges , 231–252.

[24] Giraldo, J., Urbina, D.I., Cárdenas, A.A., Valente, J., Faisal, M.A., Ruths, J., Tippenhauer, N.O., Sandberg, H., Candell, R., 2018. A Survey of Physics-Based Attack Detection in Cyber-Physical Systems. ACM Comput. Surv. 51, 76:1–76:36.

[25] Greco, G., Guzzo, A., Lupia, F., Pontieri, L., 2015. Process Discovery under Precedence Constraints. ACM Trans. Knowl. Discov. Data 9, 32:1–32:39.

[26] Greco, G., Lupia, F., Scarcello, F., 2020. Coalitional games induced by matching problems: Complexity and islands of tractability for the Shapley value. Artif. Intell. 278.

[27] Green, B., Derbyshire, R., Krotofil, M., Knowles, W., Prince, D., Suri, N., 2021. PCaaD: Towards automated determination and exploitation of industrial systems. Comput. Secur. 110, 102424.

[28] Green, B., Krotofil, M., Abbasi, A., 2017. On the Significance of Process Comprehension for Conducting Targeted ICS Attacks, in: Workshop on Cyber-Physical Systems Security and PrivaCy.

[29] Guzzo, A., Ianni, M., Pugliese, A., Saccà, D., 2020. Modeling and efficiently detecting security-critical sequences of actions. Future Gener. Comput. Syst. 113, 196–206.

[30] Göksu, B., Yüksel, O., Şakar, C., 2023. Risk assessment of the Ship steering gear failures using fuzzy-Bayesian networks. Ocean Engineering 274, 114064.

[31] Hemminghaus, C., Bauer, J., Padilla, E., 2021. BRAT: A BRidge Attack Tool for Cyber Security Assessments of Maritime Systems. International Journal on Marine Navigation and Safety of Sea Transportation 15, 35–44.

[32] Hutchins, E.M., Cloppert, M.J., Amin, R.M., 2011. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. Leading Issues in Information Warfare & Security Research 1.

[33] IEEE, 2019. Standard for Floating-Point Arithmetic.

[34] International Association of Classification Societies, 2022. Recommendation on Cyber Resilience. Rec 166 Corr.2.

[35] International Electrotechnical Commission, 2016. 61162-1 Maritime Navigation and Radiocommunication Equipment and Systems—Digital Interfaces—Part 1: Single Talker and Multiple Listeners.

[36] International Maritime Organization, 1975. Recommendation Concerning Regulations for Machinery and Electrical Installations in Passenger and Cardo Ships. Resolution A.325(IX).

[37] International Maritime Organization, 1979. Improved Steering Gear Standards for Passenger and Cargo Ships. Resolution A.415(XI).

[38] International Maritime Organization, 2007. Adoption of the revised performance standards for integrated navigation systems (INS). Resolution MSC.252(83).

[39] International Maritime Organization, 2021. Guidelines on maritime cyber risk management. MSC-FAL.1/Circ.3/Rev.1.

[40] International Maritime Organization, 2022. Guidelines on Maritime Cyber Risk Management.

[41] International Organization for Standardization, 1975. ISO 2533 - Standard Atmosphere.

[42] International Organization for Standardization, 1992. ISO 3448 - Industrial liquid lubricants – viscosity classification.

[43] Longo, G., Merlo, A., Armando, A., Russo, E., 2023a. Electronic attacks as a cyber false flag against maritime radars systems, in: 2023 IEEE 48th Conference on Local Computer Networks (LCN), pp. 1–6. doi:10.1109/LCN58197.2023.10223370.

[44] Longo, G., Orlich, A., Musante, S., Merlo, A., Russo, E., 2023b. MaCySTe: A virtual testbed for maritime cybersecurity. SoftwareX 23, 101426.

[45] Longo, G., Russo, E., Armando, A., Merlo, A., 2023c. Attacking (and Defending) the Maritime Radar System. IEEE Trans. Inf. Forensics Secur. 18, 3575–3589.

[46] Lucchese, M., Lupia, F., Merro, M., Paci, F., Zannone, N., Furfaro, A., 2023. HoneyICS: A High-interaction Physics-aware Honeynet for Industrial Control Systems, in: International Conference on Availability, Reliability and Security, ARES.

[47] Lund, M.S., Gulland, J.E., Hareide, O.S., Josok, O., Weum, K.O.C., 2018a. Integrity of Integrated Navigation Systems, in: IEEE Conference on Communications and Network Security, CNS.

[48] Lund, M.S., Hareide, O.S., Josok, O., 2018b. An Attack on an Integrated Navigation System. Necesse 3, 149–163.

[49] Luo, X., 2020. Research on Communication Technology of Ship Integrated Monitoring System Based on OPC, in: International Conference on Intelligent Transportation, Big Data & Smart City, ICITBS.

[50] Lupia, F., Lucchese, M., Merro, M., Zannone, N., 2023. ICS Honeypot Interactions: A Latitudinal Study, in: IEEE International Conference on Big Data.

[51] Lupia, F., Mendicelli, A., Ribichini, A., Scarcello, F., Schaerf, M., 2018. Computing the Shapley value in allocation problems: approximations and bounds, with an application to the Italian VQR research assessment program. J. Exp. Theor. Artif. Intell. 30, 505–524.

[52] Martelli, M., Viviani, M., Altosole, M., Figari, M., Vignolo, S., 2014. Numerical modelling of propulsion, control and ship motions in 6 degrees of freedom. Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment 228, 373–397.

[53] Meland, P.H., Bernsmed, K., Wille, E., Rooseth, O.J., Nesheim, D.A., 2021. A Retrospective Analysis of Maritime Cyber Security Incidents. International Journal on Marine Navigation and Safety of Sea Transportation 15, 519–530.

[54] Mitre Corporation, a. MITRE ATT&CK - Supply Chain Compromise. https://attack.mitre.org/techniques/T1195/. Accessed May 10, 2024.

[55] Mitre Corporation, b. MITRE CAPEC: Targeted Malware. https://capec.mitre.org/data/definitions/542.html. Accessed May 10, 2024.

[56] Molinaro, C., Moscato, V., Picariello, A., Pugliese, A., Rullo, A., Subrahmanian, V.S., 2014. PADUA: Parallel Architecture to Detect Unexplained Activities. ACM Trans. Internet Techn. 14, 3:1–3:28.

[57] Myers, D., Suriadi, S., Radke, K., Foo, E., 2018. Anomaly detection for industrial control systems using process mining. Comput. Secur. 78, 103–125.

[58] NATO Cooperative Cyber Defence Centre of Excellence, 2022. Cybersecurity Considerations in Autonomous Ships.

[59] Nivethan, J., Papa, M., 2016. A SCADA Intrusion Detection Framework that Incorporates Process Semantics, in: Annual Cyber and Information Security Research Conference, CISRC.

[60] Nochvay, A., 2019. Security research: CODESYS Runtime, a PLC control framework. Kaspersky ICS CERT , 56–61.

[61] Oruc, A., Kavallieratos, G., Gkioulos, V., Katsikas, S., 2024. Cyber Risk Assessment for SHips (CRASH). International Journal on Marine Navigation and Safety of Sea Transportation 18, 115–124.

[62] Perera, L.P., Soares, C.G., 2015. Collision risk detection and quantification in ship navigation with integrated bridge systems. Ocean Engineering 109, 344–354.

[63] Rudakov, A., . Script modbus-discover. https://nmap.org/nsedoc/scripts/modbus-discover.html. Accessed May 10, 2024.

[64] Saraeian, S., Shirazi, B., 2020. Process mining-based anomaly detection of additive manufacturing process activities using a game theory modeling approach. Computers & Industrial Engineering 146, 106584.

[65] Sarkar, E., Benkraouda, H., Maniatakos, M., 2020. I came, I saw, I hacked: Automated Generation of Process-independent Attacks for Industrial Control Systems, in: ACM Asia Conference on Computer and Communications Security.

[66] Sicard, F., Hotellier, E., Francq, J., 2022. An Industrial Control System Physical Testbed for Naval Defense Cybersecurity Research, in: IEEE European Symposium on Security and Privacy, EuroS&P.

[67] Suez Canal Authority, 2020. Rules of Navigation.

[68] Svilicic, B., Brcic, D., Zuskin, S., Kalebic, D., 2019. Raising Awareness on Cyber Security of ECDIS. TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation 13, 231–236.

[69] Tam, K., Jones, K., 2019. MaCRA: a model-based framework for maritime cyber-risk assessment. WMU Journal of Maritime Affairs 18, 129–163.

[70] Tiegelkamp, M., John, K.H., 2010. IEC 61131-3: Programming industrial automation systems. volume 166. Springer.

[71] Tran, K., Keene, S., Fretheim, E., Tsikerdekis, M., 2021. Marine Network Protocols and Security Risks. Journal of Cybersecurity and Privacy 1, 239–251.

[72] United Nations, 1974. International Convention for the Safety of Life at Sea.

[73] United Nations, 1978. Protocol relating to the International Convention for the Safety of Life at Sea, 1974.

[74] Warżała, R., 2020. Modern Integrated Platform Management System Laboratory for Polish Naval Academy: Design and Implementation. Scientific Journal of Polish Naval Academy 220-221, 59–71.

[75] Weijters, A.J.M.M., Ribeiro, J.T.S., 2011. Flexible Heuristics Miner (FHM), in: IEEE Symposium on Computational Intelligence and Data Mining, CIDM.

[76] Wolsing, K., Saillard, A., Bauer, J., Wagner, E., van Sloun, C., Fink, I.B., Schmidt, M., Wehrle, K., Henze, M., 2022. Network Attacks Against Marine Radar Systems: A Taxonomy, Simulation Environment, and Dataset, in: IEEE Conference on Local Computer Networks, LCN.