École Polytechnique Fédérale de Lausanne

NinjaCrane: a Cybersecurity Attack Demonstrator on a Polar Crane's Industrial Control System (ICS)

by Gaiëtan Renault

Master Thesis

Approved by the Examining Committee:

Prof. Dr. sc. ETH Mathias Payer
Thesis Advisor

Nicolas Sursin
External Expert

Nicolas Sursin
Thesis Supervisor

EPFL IC IINFCOM HEXHIVE
BC 160 (Bâtiment BC)
Station 14
CH-1015 Lausanne

August 11, 2023

At the end of the day, the goals are simple: safety and security.
— *Jodi Rell*

Acknowledgments

I would like to express my gratitude to my company supervisor, *Nicolas Sursin*, for all his guidance throughout my journey at *Électricité de France* (EDF). His dedication to my professional growth has been evident in the effort he has invested in my training and development. *Nicolas Sursin* has been a great supervisor, always available to answer my questions and provide feedback and ideas for my project.

I would show deep gratitude to my manager, *Jérôme-Alexandre*. Through all his anecdotes, his valuable advice and his enlightened vision on the industry domain, he allowed me to grasp and brush up on the complexity of the nuclear industry, on the safety priority, and on the importance to understand automation.

I would like to take a moment to express my acknowledgments to the cybersecurity expert *Ludovic* for his precious time and constructive criticism and who has always been around while I was eating a (cyber-)pizza.

Moreover, I would grateful all *Contrôle-Commande* team members; *Karim* for his kindness and availability to answer all my questions on the nuclear domain, *Nicolas* for his jokes and endlessly joy to help me, *Romain* for his help on trigrams and irradiation, *Océane* for her sociability and energy despite a lower availability, *Baptise*, a disruptive big bear full of good ideas, for his expertise on automation, handling system, and his D.I.Y. side, *Aude* a kind docking stations killer, *Arnaud* for his expertise on the qualification process, *Paul* for bringing some youth and energy to the team, *Evelina* for her spontaneity, *Jean-Paul* for his unbelievable stories and expertise on the industrial standards, *Laurent* for his investment despite his professional travels, *Thierry* for his teasing and outspoken nature, *Louis* for his sense of humor, and *Thibaud* for his amazing work on the polar crane. To all, thank you for your welcoming, for all the good moments we have shared, for your kindness, and for your help in the company.

Despite his very busy schedule, I would like to express my great respect to Prof. *Payer*, my academical supervisor, for providing me with invaluable and concise advice remotely that greatly enhanced my works.

Finally, I would like to thank my love, *Camille*, for her support, her joy, and her energy that she managed to transmit to me beyond the Swiss border. I address my last words to my family, who supported me with all their love every weekend when coming home, and my sister who made me laugh remotely so much with her adventures.

Lyon, August 11, 2023 Gaiëtan Renault

Abstract

Despite the sophisticated *Stuxnet* attack discovered in 2010, the cybersecurity awareness around the *Industrial and Control System* (ICS) remains a challenge [1]. The necessity to train ICS actors and to raise awareness on this issue increases [2] [3] [4]. Moreover, listing the cybersecurity best practices to follow regarding a security policy has a limited impact on their effectiveness compared to an awareness program [5] [6] [7]. In this context, this Master's Project builds a cyberattack demonstrator targeting a part of a nuclear power-plant ICS and more precisely a polar crane's *control system*. The described attack, called *NinjaCrane*, infects an engineering workstation through an USB cable or a maliciously tampered mouse and deploys a malware targeting the communication with a *Modicon M580*, a *Programmable Logic Controller* (PLC), by conducting a *Meet-In-The-Middle Attack* (MITM) exploiting the CVE-2022-45789 (Capture-replay attack) and the CVE-2021-22779 vulnerability (a.k.a. *ModiPwn*). This work illustrates and highlights the necessity to respect the ICS security policy and may well suit an awareness program.

Résumé

Malgré la sophistication de l'attaque *Stuxnet*, découverte en 2010, la prise de conscience sur l'importance de la cybersecurité des systèmes de supervision reste encore en marge [1] et la necessité de former les acteurs industriels sur cette problématique apparaît comme croissante [2] [3] [4]. De plus, lister les bonnes pratiques vis-à-vis d'une politique de sécurité peut avoir un impact limité sur leur réussite en comparaison à une démarche de sensibilisation [5] [6] [7]. Dans ce cadre, ce Projet de Master vise à construire un démonstrateur d'attaque visant une partie d'un système industriel de contrôle d'une centrale nucléaire et plus précisément le *Contrôle-Commande* (CC) du pont polaire. L'attaque présentée ici, appelée *NinjaCrane*, infecte une station d'ingénierie grâce à un câble USB infecté ou une souris modifiée et déploie un logiciel malveillant visant la communication avec le *Modicon M580*, un *Automate Prgorammable Industriel* (API), en conduisant une attaque de type *Homme Du Milieu* (HDM) exploitant les failles CVE-2022-45789 (une attaque par rejeu) et CVE-2021-22779 (appelée *ModiPwn*). Ce travail illustre et souligne la necessité du respect de la politique de sécurité et s'inscrit dans une démarche de sensibilisation.

List of Acronyms

Abbreviations	Meaning
APT	Advanced Persistent Threat
ARP	Address Resolution Protocol
BLE	Bluetooth Low Energy
CB	Reactor Containment Building or Reactor Building
CISA	Cybersecurity and Infrastructure Security Agency
DCS	Distributed Control System
DMZ	DeMilitarized Zone
DOE	Department of Energy
EDF	Électricité de France
FBI	Federal Bureau of Investigation
GRFICS	Graphical Realism Framework for Industrial Control Simulations
HID	Human Interface Device
HMI	Human Machine Interface
ICS	Industrial Control System
IIoT	Industrial Internet of Things
IS	Information System
IT	Information Technology
MITM	Meet-In-The-Middle
MSc	Master of Science
NPP	Nuclear Power Plant
NSA	National Security Agency
OPSWAT	Omni-Platform Security with Access Technologies
OT	Operational Technology
PLC	Programmable Logic Controller
RS232	Recommended Standard 232
RTU	Remote Terminal Unit
SCADA	Supervisory Control And Data Acquisition
TTL	Transistor-Transistor Logic (serial communication)
UAC	User Account Control
UMAS	Unified Messaging Application Services
USB	Universal Serial Bus
VM	Virtual Machine

List of Notation

Notation	Meaning
$x_{\text{(utf-16-le)}}$	UTF-16 little-endian encoded string
$x_{\text{(base64)}}$	Base64 encoded string
`SHA256(`x`)`	Secure Hash Algorithm 2 with a 256-bits digest of the input x
$\mathbb{N}$	Natural numbers set
$(x \bmod y)$	Signed remainder of the division of x by y
$\{0,...,x\}$	Set containing all the integers from 0 (included) to x (included)
$x \leftarrow x_0,...,x_n$	x is a string made of the ordered concatenation of the characters x_0 up to x_n
$s_1 + s_2$	Concatenation of the strings s_1 and s_2
`0xXX`	Hexadecimal representation of `XX` (e.g., `0xA5` is equal to 165)

Contents

Chapter 1

Introduction

The notorious *Stuxnet* attack [8], [9] the first *Advanced-Persistent Threat* (APT) that subverted an *Industrial and Control System* (ICS), spurred cybersecurity research on this domain. Since then, we observe increasing threats in the ICS domain as show the recent and major attacks, e.g., *Colonial Pipeline*, *CaddyWiper*, *Industroyer2* etc. Moreover, the energy sector is a *Critical Infrastructure* (CI) sector which makes it a target of particular interest [10]. Indeed, the *Kaspersky Lab* lists, as a half-year report, the most noticeable attacks in ICS and claims that in the energy sector in 2022 about 34.5% of the ICS computers (in *Kaspersky Security Network*) have blocked malicious objects [11]. However, cybersecurity in the energy sector still needs to be developed — in April 2022, the joint cybersecurity advisory "urge[d] critical infrastructure organizations, especially energy sector organizations, to implement detection and mitigation mechanisms and harden their ICS/SCADA devices" — DOE, CISA, NSA, and FBI [12]. In this context of enhancing cybersecurity in ICS, this work, as part of a Master's Project in industry at *Electricité de France* (EDF), builds a demonstrator attack called *NinjaCrane* on a part of an ICS representative of what can be found in a Nuclear Power Plant (NPP) station. *NinjaCrane* is a practical ICS cyberattack which raises awareness on the importance to respect the cybersecurity policies. The targeted ICS is a polar crane's control system, where the polar crane is a standalone equipment introducing and handling components within the reactor containment building.

Compared to the *Information Technology* (IT) cybersecurity [13], the *Operational Technology* (OT) cybersecurity is relatively new [14]. OT cybersecurity is a change of paradigm where priority is given to availability followed by integrity to achieve one goal — safety. The following typical characteristics and challenges can be noted in an OT environment; low powered hardware with small cybersecurity capabilities and low cybersecurity monitoring, presence of custom closed-source solutions due to the importance of patents in the industry, safety comes before cybersecurity, complexity to update the system as it can stop the industrial process and as any modification needs to be qualified, presence of 3rd-party solutions, and finally still a relatively low awareness on the cybersecurity policy despite the fact that the human is generally a key asset directly interacting with

the ICS.

To cite similar works that have been done to illustrate an industrial cyberattack, the *Graphical Realism Framework for Industrial Control Simulations* (GRFICS) models a destructive attack on the simplified *Tennessee Eastman Challenge Process* [15]. This attack is based on *ARP poisoning* and a MITM attack in a simplified network architecture that can be found in Figure A.1. *C. Ekisa* et al. built upon this GRFICS to make a *Virtualised ICS Open-source Research Testbed* [16] that emulates more closely an ICS and offers the possibility to practice several defense & attack techniques. Even though the virtualization approach offers a better flexibility it suffers from a lack of representativeness. As a consequence, virtualization reduces the possibility to model advanced threat and generally shows exploitation of only one vulnerability. As an example, virtualized environment can not model physical attack and always assumes that the attacker already infected the air-gap network. Moreover, the target audience of the demonstrator attack are operators and automation technicians, the ones that interact with the ICS. They may not be familiar with computer virtualization and may not see how containers interact with each other, how realist is the considered architecture, and how it shows a practical cyberattack on the ICS they handle on a daily basis. To overcome those limitations, demonstrator attack examples on real world testbeds of a physical process have been made like the *Check Point* chemical plant ICS testbed [17], the *OPSWAT* electric station ICS [18], or the *Bristol Cyber Security Group* testbed [19] [20] [21]. In the first two attack scenarios the adversary is however assumed to be already inside the production network which is a strong assumption. In both case, the adversary's goal is to make the industrial process stops. While in the Bristol testbed, the APT attack on the ICS starts from a maliciously modified manual PDF stored on the data aggregation server — the adversary is thus assumed to be in the SCADA — and the adversary's goal is to make the water industrial process enters into an unsafe state. In opposite, in this MSc thesis the adversary is assumed to be external (i.e., the production network and the control system are clean). The *NinjaCrane* attack targets a clean air-gap ICS containing a *Modicon M580* PLC controlling an industrial NPP-related demo environment made in LEGO™. The end goal of the *NinjaCrane* attack is to postpone the nuclear unit start by making it enter into a potentially unsafe state.

This MSc project is building upon an already made sketch of a polar crane in LEGO™ and its ICS architecture. This demo environment has originally been developed by EDF as part of an internal training program on automation and qualification. It has been designed by *B. Allard* according to the V model development cycle made by *T. Cassas*. The MSc project's goal is to design a demonstrator cyberattack on this sketch to increase awareness on the cybersecurity policy. The polar crane is an easy to grasp and visual physical process which makes it more interesting to tamper for a demonstrator. Moreover, having a physical sketch enable to create a full cyberattack with several stages; from intrusion into the air-gap to physical process full control. The core idea is to show that an attacker would exploit several weak links to end up having a full control over the physical process. The approach taken was to exploit already known vulnerabilities like the HID spoofing or the *ModiPwn* vulnerability which allow to take the control over the engineering workstation or the PLC respectively.

Specifically in the industry, the cybersecurity policies are more often by-passed by the operators and are less monitored for several reasons — cybersecurity policy may reduce ease-of-use and are generally IT-oriented, lack of cybersecurity awareness, consequences or success probability of an OT cyberattack are minimized, some OT materials are not secured by design and do not integrate by default defense mechanisms and monitoring and OT materials are often part of an air-gap. In overall, operators may be under-trained regarding OT cybersecurity and may minimize their role even though they are a key but weak asset [22]. For all those reasons, a pedagogical and real-world demonstrator attack is well suiting a cybersecurity awareness training.

This MSc project develops a hands-on demonstrator cyberattack on the polar crane's ICS to make the polar crane stop with a loaded charge above of the nuclear reactor vessel. To do so, an infected USB cable is plugged-into an engineering workstation to perform an HID spoofing attack and setup a MITM attack over the connection with the PLC. This attack, exploiting the lack of authentication of the communication protocol, allows the attacker to take control over this *Modbus*/*UMAS* connection and allows arbitrary read and write on the PLC's system bits. By taking control of the PLC, an attacker can take the control of the polar crane when the PLC is connected to the engineering workstation.

By presenting an attack demonstrator via an infected USB charging cable, this work highlights the importance of supervising and regulating the supply chain cybersecurity and the possible severe consequences of a lack of USB hygiene. Finally, targeting a specific physical process (e.g. a handling equipment) points out that an advanced OT cyberattack requires in-depth understanding of the physical process in order to make it enter into an unsafe state. Even if the polar crane is not directly interacting with the nuclear fuel, it is still a valuable target for the attackers as its compromise could lead to severe economical loss.

The demonstrator attack developed in this thesis is the first ICS demonstrator attack in the literature that targets a non-simulated specific system presents in a NPP. One novelty of this cyberattack also lies in its capacity to target an air-gap and to shape from an USB and IT-based attack to an OT attack on a PLC. Furthermore, the *NinjaCrane* attack by exploiting maliciously tampered mouse or USB cable with wireless capacity illustrates how diverse, stealth and flexible an USB-based attack can be. Finally, in this demonstrator, even if it simplifies a lot the process and its architecture, the cyberattack has been made to be as close to reality as possible by making the fewest possible assumptions.

Chapter 2

Background

This chapter introduces the background material related to the NPP and the cybersecurity of its associated ICS.

2.1 The Reactor Containment Building in a Nuclear Power Plant

As the demonstrator attack targets a NPP equipment, this section briefly summarizes the plant operation and associated cyber and safety risks with a focus on the role of the targeted equipment — the polar crane.

2.1.1 Nuclear Power Plant (NPP)

Nuclear Power Plants (NPP) are using nuclear fission in order to heat water into steam which spins a turbine combined with a generator to produce electricity. In France, in 2023, all of the 56 civilian nuclear power reactors in operation, distributed in 16 nuclear power stations, are Pressurized Water Reactors. PWR are the most common type of nuclear reactor and as such are the ones considered in this project. A simplified view of a *Pressurized Water Reactor* (PWR) can be found in Figure B.1. As an electrical production site, NPP are considered as CI. Moreover, the NPP criticity is increased because of the nuclear fuel handling. The worst-case scenario to avoid is the radioactive contamination. It can for example happen in case of uncontrolled nuclear meltdown in the reactor vessel.

Our cyberattack considers a particular situation of the NPP — its shutdown. The NPP shutdowns take place for maintenance, fuel reloading or ten-yearly inspection reasons.

2.1.2 Reactor Containment Building (CB)

The reactor *Containment Building* (CB), or reactor building, is a containment structure in which we find the primary circuit and a part of the secondary circuit. A cross section of the CB can be found in Figure C.1. The goal of this enclosure is to confine fission products. In return, large equipment cannot be easily delivered and must be delivered by the polar crane.

2.1.3 Rotary Overhead Crane

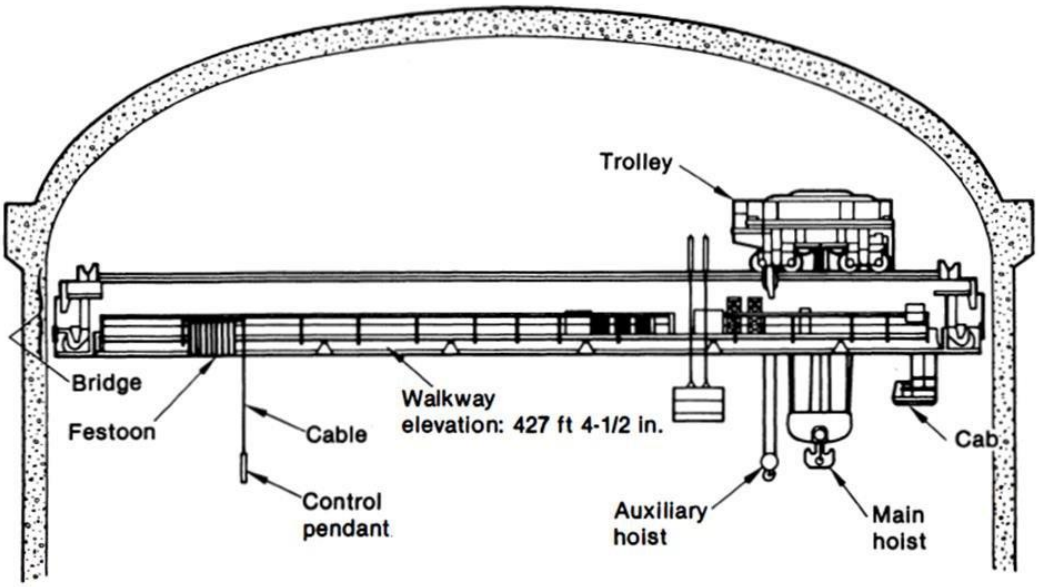


Figure 2.1: Rotary Overhead Crane [23], or Polar Crane. Three movements are possible in this system: polar crane rotation, trolley horizontal shift and charge lifting.

The attack demonstrator targets the polar crane's ICS. The operation of the polar crane is explained in this section. The rotary overhead crane (called thereafter polar crane) is a handling equipment which delivers and transports large components. It is for example used to lift and remove the reactor cover. The polar crane is only used during NPP shutdown, and its ICS is shutdown during the NPP operation. The polar crane equipment is standalone and is locally and manually operated by an operator.

Two OT cyberattack scenarios were identified as possible ways of creating maximal damage and significant safety event:

- Shutdown of the polar crane with a charge above the reactor. This would create significant

safety event with a possibility of the charge being drop above the reactor and breaking the first three barriers to radioactive release (i.e., fuel matrix, fuel cladding and boundary of the reactor coolant system) [24].

- Activate charge lifting of the polar crane when it is in an overload situation caused by friction. For example this situation could be caused by friction when lifting a concrete slab which is placed above the confinement pool. If a cyberattacker detects it and has full control over the PLC he could activate charge lifting to increase the overload which could either break the charge or make it fly.

2.2 Industrial Control System and Programmable Logic Controller

In this section are presented a cybersecurity introduction of the *Industrial Control System* (ICS) and its major component — the *Programmable Logic Controller* (PLC).

2.2.1 Industrial Control System (ICS)

Industrial Control System (ICS) is a network of interconnected systems that are designed to monitor and control industrial processes. Cybersecurity threats to ICS have far-reaching implications that can result in severe damage to property, loss of production, and even potential loss of life. Security of ICS thus requires a comprehensive approach that includes risk assessment, vulnerability identification, and implementation of appropriate protective measures.

ICS architecture are often modeled by the *Purdue Model* represented in Figure D.1. The attack surface is large and very diverse. It is large because the cyberattack could target any systems within the production network, the *Industrial Control System* (ICS), the *Supervisory Control And Data Acquisition* (SCADA), the corporate network etc. Attack surface is diverse because of the large number of different device types — actuators, sensors, *Industrial Internet of Things* (IIoT) and smart devices, *Remote Terminal Unit* (RTU), PLC, Engineering Workstation, *Human-Machine Interface* (HMI), servers, switches, routers etc.

2.2.2 Programmable Logic Controller (PLC)

Programmable Logic Controller (PLC), or similarly *Programmable Automation Controller (PAC)*, is an industrial microprocessor-based controller made of a CPU, ROM, RAM, OS and a firmware. The PLC stacks are represented in Figure 2.2. The studied PLC is the *Modicon M580 BMENOC0301* maintained by *Schneider Electric*. It supports for example the following communication protocols: Modbus TCP, Ethernet IP, HTTP, FTP etc. The M580 is programmed by an engineering workstation

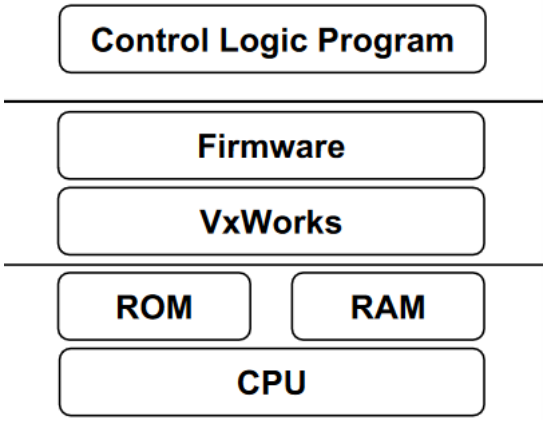


Figure 2.2: PLC Stacks of a PLC [25] designed following the IEC 61131 standard. The Control Logic Program is the program executed by the CPU; it is a STM extension file for Schneider Electric programs. Considered firmware in this study is the Modicon M580 SV2.7 firmware. Considered CPU in this study is the Modicon M580. VxWorks is the real-time operating system running on the CPU.

by using the *Unity Pro XL* software renamed as *EcoStruxure Machine Expert* in the later versions. To program the PLC the closed-source Modbus/UMAS protocol is being used.

2.3 About Cybersecurity in ICS

In the nuclear sector, most of the systems ensuring safety used to rely on analog devices. This technology adds a cybersecurity layer as analog devices are less complex and less connected. However, in the last decades, numerical devices with greater capabilities have penetrate the industrial market while the number of analog device manufacturers has dropped. This technology shift led to the success of several attacks [26]. On another hand, the nuclear industry specifically focus on a defense-in-depth approach to ensure operational security.

2.3.1 Previous Major ICS Attacks in the Energy Sector

Stuxnet

Stuxnet attack [27], [28], [29] targeted the Iran's nuclear facilities and involved the use of a sophisticated worm exploiting multiple zero-day vulnerabilities and a rootkit technique to hide from antivirus. This OT cyberattack was designed to cause significant damage to the centrifuges responsible for enriching uranium by carefully tampering their speed to 1,064 cycles per second

while keeping the attack stealth and persistent. The attackers used an attack cycle made of a combination of spear-phishing and network infiltration techniques to gain access to the targeted facility. Once inside, they deployed the *Stuxnet* worm, which was capable of infecting both Windows operating systems and Siemens PLC while remaining undetected.

Stuxnet relied on several vulnerabilities to carry out its mission successfully. Primary vulnerability, the .LNK W32/*Stuxnet* (CVE-2010-2568) vulnerability, took advantage of the use of USB drives to transfer data between systems. The associated exploit drops the trojan when the system automatically parses a shortcut file (.lnk) which results in the execution of a malicious Control Panel module malware. This USB-based attack does not require to reprogram the USB peripheral. *Stuxnet* approach allows replication by infecting USB thumb drives. *NinjaCrane* is using another type of USB-based attack based on HID injection, which requires the presence of a programmable micro-controller in the USB peripheral. This adds a difficulty layer to the attacker and a less stealth attack but it allows in counter-part to infect any OS not implementing HID spoofing protection measures.

Another vulnerability that the attackers exploited was the weak authentication protocols of the *Siemens* control systems. *NinjaCrane* similarly exploits another vulnerability — *ModiPwn* caused by a weak authentication of the *Schneider Electric* communication protocol.

Triton

The *Triton* attack is a sophisticated cyberattack that targeted the safety instrumentation systems of a Saudi Arabian petrochemical plant in 2017. The *Triton* attack started with the compromise of the plant's network through phishing. The *Triton* malware, once installed, was manipulating the safety controllers (the *Triconex* PLC from *Schneider Electric*) and the plant's safety systems that monitor and detect incidents. By doing so, the attackers could have overridden the safety protocols, potentially resulting in an explosion or a toxic gas leak.

First vulnerability exploited was the human factor — the plant's employees. The attackers used a spear-phishing email to trick an employee into downloading the *Triton* malware, which then spread throughout the entire network. This highlights the importance of employees training in preventing cyberattacks.

One key vulnerability that the attackers exploited was the plant's reliance on poorly-protected safety systems. The safety systems of the plant were not adequately secured against cyberattacks, and the use of legacy equipment, outdated firmware, and poorly configured network architecture made it easier for the attackers to breach the system and install the *Triton* malware. The attackers were also successful in infiltrating the system because they had prior knowledge of the plant's network topology, protocols, and industrial control systems, which helped them in crafting the malware to exploit the system's vulnerabilities.

Similarities can be noted with the *NinjaCrane* attack — the potentially catastrophic disaster of the cyberattack, human-factor role in the cyberattack, and prerequisite to reverse-engineer the PLC's communication protocol. This communication protocol is called *TriStation* for the *Triconex* PLC and Modbus/UMAS for the *M580 PLC*. Also, the malware is, as for the *NinjaCrane* attack, a python script compiled in an exe file using the *py2exe* extension. Even if *NinjaCrane* has persistence capabilities, it does not implement advanced hiding and anti-forensics techniques like *Triton* does.

2.3.2 From Operational Security to Cybersecurity in the Nuclear Industry

The defense-in-depth approach taken by the nuclear industry during the control system design integrates operational security. The operational security can for example result into material and software redundancy, system qualification, human-supervision, mechanical stops, safety levels, priority on actuation control system etc. Those various mitigation barriers could also act as cybersecurity barriers. For example, a polar crane can integrate a kinematics line monitoring system responding to motor overspeed or engine backfire. As another example, a polar crane can integrate a wired emergency stop with highest priority order to shutdown the polar crane. Those defense-in-depth barriers won't protect directly against the *NinjaCrane* attack but will prevent the attack to set a high polar crane's rotation speed or to deviate visually from normal operation as the automation operator will activate an emergency stop as soon as the the polar crane behaves differently than expected.

2.3.3 Security of the *Modicon M580*

Schneider Electric lists all released vulnerabilities and their state [30]. As a first observation, the number of discovered vulnerabilities on the *M580* is around ten per year. Moreover, the average time to patch a vulnerability is between two and four years. Unfortunately, patch deployment implies some code modification and thus requires a system re-qualification. It also requires industrial process stop. For those reasons a patch deployment can take more than a decade to be applied.

Among those vulnerabilities, the approach was to choose a communication protocol vulnerability as it allows to perform lateral movement from engineering workstation to PLC. Compromising directly the PLC would have require an access to the PLC. But the access to an ICS is physically restricted and PLC are usually placed into physically locked electrical cabinet. Moreover, in case of communication protocol vulnerability, remediation requires PLC firmware update and Unity Pro engineering workstation update. As date of 12 March 2023, Schneider Electric reports that a replay attack on the *Modicon M580* is still possible [31] and advises to harden the engineering workstation.

2.3.4 Overview on USB-related Threats

Number of USB-based attacks tends to increase over time and be more diverse [32]. *NinjaCrane* makes use of a USB-based attack that can be classified as programmable micro-controller hardware-based attack. Many such attack-capable devices are available on the market, e.g., *BadUSB*, *BadUSB 2*, *Rubber Ducky*, *USBHarpoon*, *OMG Cables*, *USB Ninja*, *Bash Bunny* etc.

NinjaCrane infects the ICS with the use of a *USB Ninja* cable type-C (or a maliciously modified mouse) that would be plugged to the engineering workstation. The USB Ninja cable and the malicious mouse are *Human Interface Devices* (HID) spoofing device injecting keyboard key-frames. For the USB Ninja cable, the attack can be triggered over Bluetooth and in its professional version the key-frames can even been sent over Bluetooth. Moreover, the *USB Ninja* cable is a working charging cable indistinguishable, at first glance, from a conventional charging cable. For the maliciously tampered mouse, any HID-capable device with extensive memory or embedded wireless connection can be placed in the mouse to execute the malware. Except from its weights, the mouse is indistinguishable from a conventional mouse.

Human Interface Device (HID) spoofing device [33] mimics legitimate HID to trick the computer to believe a keyboard is attached, the device then sends pre-configured keystrokes. To mimic a legitimate HID, the spoofing device operates an USB transport layer enumeration. It will then initiate data exchange under the format of input/output reports where the input report can contain keystrokes or mouse click to inject. As those keystrokes injection are impractical to deploy an advanced threat, the injected keystrokes generally perform an USB Drive-By attack in order to execute a downloaded malware.

Chapter 3

Threat Model

This chapter introduces the considered threat model for the *NinjaCrane* attack. This includes a description of the adversary model and its attack surface.

3.1 Adversary Model

The system under attack is the polar crane's ICS. The choice made by the adversary is justified as follow. Control and regulation functions in normal operation are generally made redundant with independent and diversified systems, and some components are non-electrical analog devices making it hard to compromise. However, during NPP shutdown less securely-proofed systems are used like the polar crane and can be seen as a weaker link.

The adversary type considered is an external adversary with medium capabilities such as hacktivists, hacking collectives or non-state cyber-terrorists. As such, adversary is considered to have no physical access to the NPP station, and would require unintentional insider threat to enter in the air-gap polar crane's ICS. Moreover, considered attacker is not highly skilled and would not exploit any 0-day vulnerability. However, the considered adversary has medium resources and has the capacity to chain known vulnerabilities and has an access to a M580 PLC, publicly accessible in the market, to reverse engineer it and test the cyberattack. As an hypothesis, some leaked information regarding the polar crane's ICS is assumed e.g., the supply chain leaked material references, unintentional data leak on social media from employee helped to identify automation operators interacting with the polar crane etc.

The campaign objective is industrial sabotage by system tampering up to destruction. More precisely, the adversary's goal is to slow down or to keep the energy production of the NPP stopped in order to inflict economical loss or corporate shaming. As a recall, a reactor unit with a 900 MW

capability generates around 3 M dollars per day. The cyberattack is performed during the periodic nuclear unit shutdowns and when the polar crane equipment is being used. Tampering or stopping the normal operations of the polar crane will delay the resumption of the energy production.

The campaign vehicle is an USB physical media such as a maliciously tampered USB cable or a maliciously tampered mouse. The infected USB physical media is brought to the NPP station via spear-phishing or supply chain compromise e.g., sending an USB Ninja cable or a malicious mouse to the automation operator as a fake gift from a subcontractor or a colleague.

The campaign weapons are HID spoofing and Modbus/UMAS exploit. The payload delivery is a keystrokes injection and a pre-compiled executable file.

The payload capabilities are keystrokes injection on the engineering workstation, data exfiltration of the information products, lateral movement from engineering workstation to PLC, Command and control by triggering on time the attack on the PLC and finally Denial-of-Service (DoS) of the PLC and polar crane to stop the use of both.

3.2 Attack Surface

This section describes the attack surface of the polar cane's ICS architecture. Figure E.1 represents a Data Flow Diagram of an ICS by listing the common assets, security controls and threat actors.

All components of the ICS have a physical access control, i.e., safety gantry, key system etc. Three scenarios for the external intrusion could be consider:

- Intrusion via the *Information System* (IS), e.g., an employee plugs his own infected laptop to the corporate network.
- Intrusion via the *Supervision Network* (SCADA), e.g., an operator on site plugs a found USB-capable device on the log servers or on an engineering workstation.
- Intrusion via the *Production Network*, e.g., an adversary maliciously tampers an electronic actuator before it is delivered on site.

By assuming that the supply chain of the polar crane is trusted, only the first two intrusions remain as the ICS is thus considered as clean after setup. The second scenario is the one kept in the following chapters for practicality, i.e., network architecture connecting the ICS and the IS is complex and is generally considered as an unidirectional network from ICS to IS.

Chapter 4

Testbed Overview

This chapter describes the testbed of the *NinjaCrane* attack. This testbed is used in the Evaluation Chapter 7 to test the attack.

4.1 The Polar Crane as a Physical Process

For illustration purpose the attack is performed on a sketch in LEGO™ of the polar crane. This sketch has been made by B. Allard and T. Cassas with the first goal being to introduce a practical control system and to create a training on software and equipment qualification.

4.1.1 Description

The sketch of the polar crane can be found in Figure F.1. This sketch made in LEGO™ is not to scale. The polar crane is composed of three BLE Hubs controlling the crane's rotation, trolley's shift, and charge lift-up. The cyberattack will target the crane's rotation in order to stop it.

4.1.2 Components

Here is the list of the sketch components (sensors and actuators):

- 3 × LEGO™ Powered Up Technic Hub (88012)
- 3 × LEGO™ Powered Up (88007) Color & Distance Sensor (e.g., computes the angular position of the polar crane to initialize it)

- 1 × LEGO™ Powered Up Technic XL Motor (88014), used for charge lift-up.
- 1 × LEGO™ Powered Up Technic Large Motor (88013), used for trolley shift.
- 4 × LEGO™ Powered Up (88011) Train Motor, used for polar crane rotation.

4.2 Electrical Cabinet

This section presents the physical Electrical Cabinet used in the testbed. The cabinet contains the PLC and the HMI and can be considered as representative as it has been qualified following the AFCEN RCC-E nuclear standard (i.e., the electrical cabinet used in this testbed could be placed in a NPP site).

4.2.1 Description

The PLC M580 is connected to the ESP32 via the MAX232 converter (RS232 to TTL serial respectively). The ESP32 is communicating over *Bluetooth Low Energy* (BLE) to the Bluetooth Hubs of the polar crane. This part is not representative of reality but the success of *NinjaCrane* cyberattack is independent from this choice.

4.2.2 Main Components

- 1 × HMI: Magelis HMIIG3U
- 1 × CPU: Modicon M580 (BMEXBP582040), firmware v2.70 (07/2018)
- 1 × Rack: Modicon X80, 12 slots, Ethernet backplane (BMEXBP1200)
- 1 × Fiber converter module: Modicon X80, single mode (BMXNRP0201)
- Diverses I/O modules used for buttons and lights — not itemized here
- 1 × ESP 32-WROOM-32
- 1 × Maxim Integrated Products MAX232 IC

4.3 Engineering Workstation

4.3.1 Description

The engineering workstation is connected to the PLC via an Ethernet connection. The protocol used to program and communicate is Modbus/UMAS over TCP/IP. This connection can be considered as representative of reality. However, the engineering workstation is assumed to be connected to the internet at some point for malware download (if the USB Ninja cable is being used) and data leakage during the cyberattack. This could for example happen if the automation operator connects the workstation to download a driver or update a system. Unity Pro is the software that modifies, compiles and uploads the program on the PLC. The program is stored in a .STU file.

4.3.2 Characteristics

The engineering workstation is a desktop computer with the following characteristics:

- Operating System: Windows 10 Professional 22H2 (build 19045.2604)
- Antivirus: Avast Antivirus Free (Version 23.3.8047.782)
- WiFi Adapter: WiFi Adapter RoHS NBS WFLD01
- Software:
  - Unity Pro XL V13.1 - 180823C (Schneider Electric Industries SA). Documentation of Unity Pro v13.1 software [34].
- Admin user account

4.4 Adversary's Tools

4.4.1 Laptop

- Operating System: Windows 11 Professional 22H2 (build 22621.1702)
- Software:
  - Python 3.11.1
  - nRF Connect 4.1.2
  - Wireshark 4.0.6

- Admin user account
- Bluetooth Sniffer Dongle : Nordic Semiconductor nRF 52840

4.4.2 USB Ninja Cable

USB Ninja cable from Lab401 can be configured with the *USBNinja github*. The USB Ninja cable is a regular USB cable that can be triggered via a wireless remote controller to execute a pre-compiled payload. The payload is uploaded via the Arduino IDE.

4.4.3 Malicious Mouse

The malicious mouse is made of a regular mouse in which were added a USB hub to multiplex the serial connection to add a microcontroller. The microcontroller then act as a HID-capable device and execute a pre-compiled payload.

The electric schematic can be found in Figure G.1. The design of the maliciously modified mouse offers flexibility as any microcontroller with micro USB type B connector can be inserted in the mouse. The *Adafruit Trinket M0* is the microcontroller used in this MSc thesis.

4.5 Network Architecture

The overall network architecture of the testbed is described in details in Figure F.2. The engineering workstation, connected in Ethernet to the PLC, uploads the program. The engineering workstation programs the HMI with the *Vijeo* software by temporarily connecting in serial to the HMI. The HMI and the PLC are connected in Ethernet allowing the PLC to send the variables state to the HMI and the HMI to send command orders to the PLC. The PLC program relays those orders to the ESP32 that interprets them and activates the corresponding BLE Hubs to control the polar crane.

Chapter 5

Design

This chapter discusses the design choices made for the *NinjaCrane* attack like the attack cycle, and some weaknesses of the Modbus/UMAS communication protocol.

5.1 Attack Cycle

In the following subsections are described the different steps taken by the attacker. The attack tree is described in Figure E.2. The entry point considered is a physical device connected to the air-gap network. For example, the compromise of the supply chain can induce a device to be brought into the the NPP site and be connected to the engineering workstation. This device will infect the engineering workstation via an HID attack and run a malware. The malware performs lateral movement to the PLC via a MITM attack to change the PLC's internal variables.

5.2 Entry Points

As the control panel cabinet is locked and devices in rotary overhead crane are considered as secure (see Chapter 3), the adversary could target only two systems i.e. the engineering workstation or the BLE communication. Taking back the initial network architecture, adversary's entry points are represented in Figure 5.1. For a better representativeness — there's no LEGO™ BLE in a NPP station — the choice has been made to target the engineering workstation as an entry point. Moreover, in the testbed, an attacker could easily connect with the BLE hub with a phone as there is no protection in the LEGO™ Wireless Protocol.

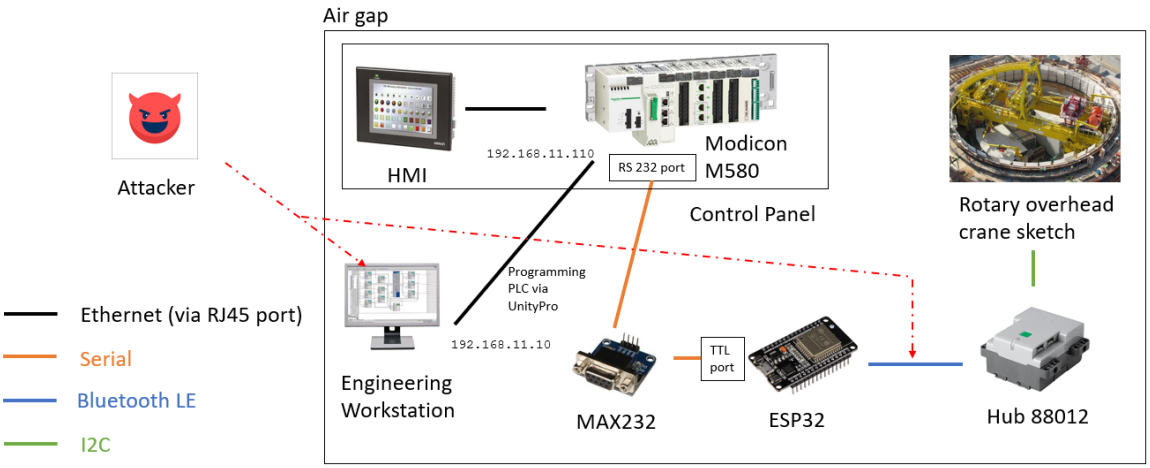


Figure 5.1: Adversary's Possible Entry Point

5.2.1 Malicious USB Device Delivery

The first step of the attack is to bring a compromised USB device to be connected to the engineering workstation. A spear-phishing attack can take the form of the effective USB drop attack — "Users Really Do Plug in USB Drives They Find" [35]. The supply chain compromise could exploit vulnerabilities in the Integrated Circuit (IC) Systems development life cycle (SDLC) that can lead to a compromised USB end-product [36] [37]. More details on the how a physical media would be brought and why it would be plugged in won't be discussed any further. The next Chapter will discuss two different malicious USB devices that can be used as a possible entry point to infect the workstation.

5.2.2 Engineering Workstation Infection

Engineering workstation are used by automation operators — who may not be trained enough regarding the security policy — to program the PLC. The engineering workstation could be infected by either a malicious mouse or a USB Ninja cable. Both could contain a pre-compiled payload that will be described in further details in the next Chapter. The goal of the infection is to take the control over the communication with the M580 in a persistent and stealth way. To this extent, a MITM attack could be setup to modify the PLC's variables state and thus take the control of the polar crane.

5.3 Communication Between the M580 and the Engineering Workstation

In this section is described the communication protocol between the PLC and the workstation i.e., the UMAS protocol.

5.3.1 Unity Pro and UMAS Protocol

The communication between the M580 PLC and the engineering workstation follows the UMAS/Modbus protocol over TCP/IP. The UMAS protocol is specific to Unity Pro — the software used to program the *Modicon* PLC — and is a proprietary communication protocol from *Schnieder Electric*. The overall packet format is ordered as follow:

1. **Ethernet**: `Destination MAC address, Source MAC address, Type`
2. **IP**: `Version`, ..., `Length Id`, `Checksum`, `Source IP address`, `Destination IP address`
3. **TCP**: `Source port, Destination port, Sequence number, Acknowledgment number, Flags`, `Window`, `Checksum`
4. **Modbus TCP**: `Transaction Id`, `Protocol Id`, `Length`, `Unit Id`
5. **Modbus**: `Function code` (e.g., `0x5A` for *Unity Pro*), `UMAS data`

Thanks to the work from CICLAB Laboratory of the University of Leon [38], the UMAS protocol has been almost completely reversed. According to this, UMAS packet takes the following request/response format:

- Modbus packet format from workstation to PLC (i.e., request): `0x5A`, `session`, `UMAS function code`, `data`
- Modbus packet format from PLC to workstation (e.g., response): `0x5A`, `session`, `status code` (i.e., `0xFE`), `data`

Below are some remarks regarding the UMAS protocol.

- The `session` token is used to authenticate the workstation's packets and is fixed until the end of the communication. This token is set to `0x00` when the authentication is disabled or for the first non-authenticated packets used to create this token. In the next part, it is assumed that authentication is enabled. To compute the `session` token the workstation must use the UMAS reservation mechanism. After this, workstation is synchronized to modify the PLC's program and variables.

- The `UMAS function code` is used by the workstation to specify the request function (e.g., `READ_MEMORY_BLOCK`, `READ_VARIABLES`, `KEEP_ALIVE`, ... [38]).
- The `status code` is used by the PLC to notify any error (i.e., `0xFD`) or a success (i.e., `0xFE`)

An example of UMAS packet over TCP/IP can be found in Figure 5.2.

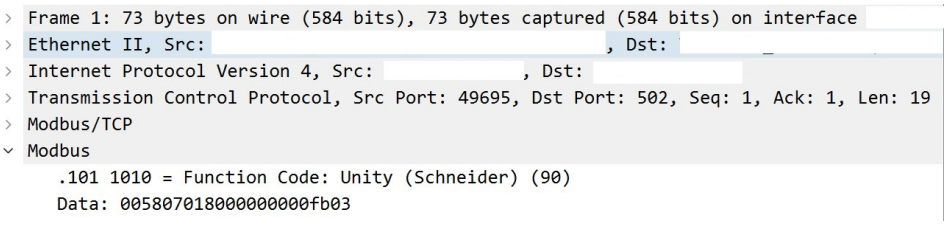


Figure 5.2: A Modbus/UMAS packet over TCP/IP. Regarding the Modbus packet; function code is 90_{10} = `0x5A`, the packet is not authenticated (session is `0x00`), it is a request sent by the workstation to check the PLC status (request `0x58`)

5.3.2 The Unity Pro Protection Passwords

Four different passwords can be set by the engineering workstation within Unity Pro:

- **Application Protection**: Password protecting the CPU and Unity Pro by preventing modification, upload and program opening. UTF-16-le encoding of this password will be denoted as $\texttt{app\_pwd}_{\text{(utf-16-le)}}$ hereafter.
- **Program Unit, Section and Subroutine Protection**: Password protecting the program sections and program units (from writing and reading). UTF-16-le encoding of this password will be denoted as $\texttt{prgrm\_pwd}_{\text{(utf-16-le)}}$ hereafter.
- **Data Storage Protection**: Password preventing malicious access to the data storage of the SD memory card. UTF-16-le encoding of this password will be denoted as $\texttt{data\_pwd}_{\text{(utf-16-le)}}$ hereafter. Default password is "datadownload".
- **Firmware Protection**: Password preventing malicious access to the module firmware via FTP. UTF-16-le encoding of this password will be denoted as $\texttt{firmware\_pwd}_{\text{(utf-16-le)}}$ hereafter. Default password is "fwdownload".

5.3.3 Communication Phases in Normal Operation

This subsection details the communication phases between the workstation and the PLC in normal operation. A first stage is to setup the passwords (see Password Setup Phase), send the project

information (see Project Setup Phase), and then upload the program (see Program Upload Phase). In a second stage, the engineering workstation will connect (again) to the PLC (see Connection Phase), get back from the PLC the project information running on it (see Information Gathering Phase), make a reservation (see Making a Reservation Phase), and finally send command to the PLC e.g., start PLC, modify variables, or upload a new program (see Send Command Phase).

Password Setup

When setting up new passwords, the workstation, via the `UPLOAD_BLOCK` function (`0x31`), sends to the PLC the base64 encoding of three bytes sequence; `salt2`, `hash1`, and `hash2` defined in Equation 5.1. This phase assumes that the engineering workstation already established a connection with the PLC.

$$\begin{aligned} \texttt{hash1} &:= \text{SHA256}(\texttt{salt1} + \texttt{data\_pwd}_{\text{(utf-16-le)}}) \\ \texttt{hash2} &:= \text{SHA256}(\texttt{salt2} + \texttt{firmware\_pwd}_{\text{(utf-16-le)}}) \end{aligned} \tag{5.1}$$

Where `salt1` and `salt2` are random 48-bits length bytes sequence generated by Unity Pro and sent in base64 to the PLC.

An example of a UMAS packet over TCP/IP sent when new passwords are set can be found in Figure 5.3.

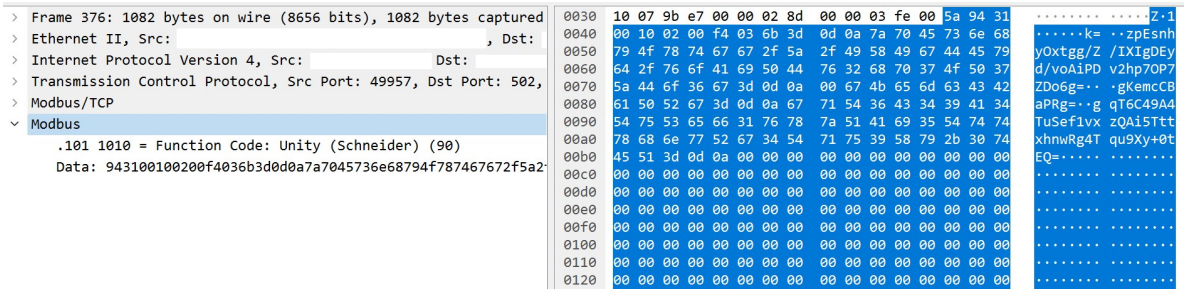


Figure 5.3: A Modbus/UMAS packet sent over TCP/IP to modify the passwords with the following values; $\texttt{hash1} := \text{SHA256}(\texttt{salt1} + \texttt{datadownload}_{\text{(utf-16-le)}}) = \texttt{zpEsnhyOxtgg/Z/IXIgDEyd/voAiPDv2hp7OP7ZDo6g=}_{\text{(base64)}}$. $\texttt{salt2} := \texttt{gKemcCBaPRg=}_{\text{(base64)}}$ and $\texttt{hash2} := \text{SHA256}(\texttt{salt2} + \texttt{fwdownload}_{\text{(utf-16-le)}}) = \texttt{gqT6C49A4TuSef1vxzQAi5TttxhnwRg4Tqu9Xy+0tEQ=}_{\text{(base64)}}$

Setup Phase

When uploading a new program onto the PLC, the workstation uses the `UPLOAD_BLOCK` function to write information regarding the project. Below is the list of the sent information according to an example of packet sent when uploading project information (see Figure 5.4.).

- Project name: `Projet`
- "Encrypted" program password: enc(`prgrm_pwd`) = `37713D7S`. The "encryption" is detailed in the next paragraph.
- `salt3` := `6Umv13jYhak=`$_{\text{(base64)}}$
- `hash3` : = SHA256(`salt3` + `app_pwd`$_{\text{(utf-16-le)}}$)
  = `XBhpWAvOQj/l67B7SVO0wh03M+7KO/sMstr2Teed/54=`
- Unity Pro software version: `V13.1`
- Computer name: `DESKTOP-8NOLHQU`
- Program file path. This the path to the .STU file that is being uploaded on the PLC: `E:\PROJET_2\P...OM_BA.STU`
- `salt1` := `5OiiBJuyhpk=`$_{\text{(base64)}}$. This salt is fixed (i.e., it stays fix over different password changes).
- `hash1` : = SHA256(`salt1` + `datadownload`$_{\text{(utf-16-le)}}$
  = `zpEsnhyOxtgg/Z/IXIgDEyd/voAiPDv2hp7OP7ZDo6g=`$_{\text{(base64)}}$
- `salt2` := `gKemcCBaPRg=`$_{\text{(base64)}}$
- `hash2` : = SHA256(`salt2` + `fwdownload`$_{\text{(utf-16-le)}}$)
  = `gqT6C49A4TuSef1vxzQAi5TttxhnwRg4Tqu9Xy+0tEQ=`$_{\text{(base64)}}$

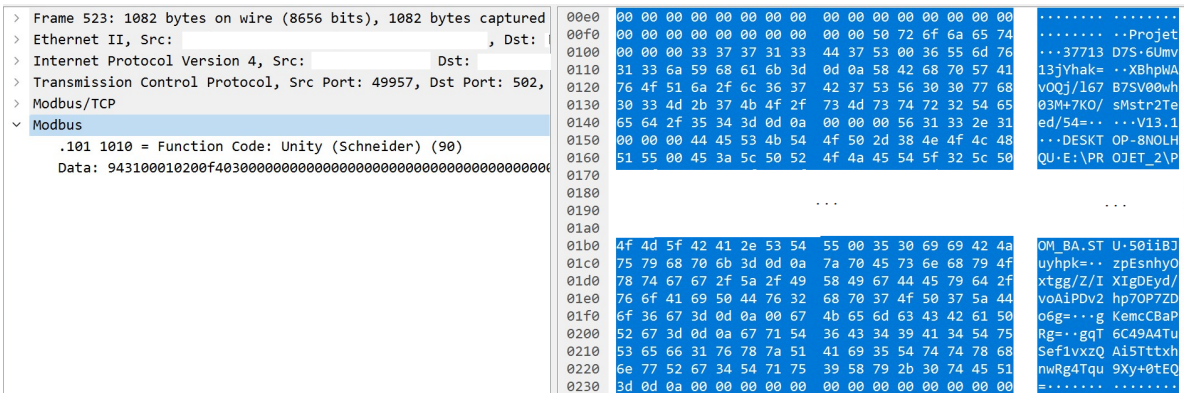


Figure 5.4: A Modbus/UMAS packet sent over TCP/IP to modify project information.

The "encryption" (as called by Schneider Electric) algorithm used to send Program Unit, Section and Subroutine Protection password is a custom-made hash algorithm. It has been reversed by Nicholas Miles [39] and a simplified version of it is described in Algorithm 1. This simplified version assumes that the password is only made of Basic Latin character (e.g., first Unicode Block).

For $x \in \mathbb{N}$, let f be defined as:

$$f : x \mapsto (x \bmod 2)((x \bmod 16 + 72)) + (x + 1 \bmod 2)((x \bmod 16 \bmod 11 \bmod 2)(x \bmod 16 + 48) + (x \bmod 16 \bmod 11 + 1 \bmod 2)(x \bmod 16 + 55)) \quad (5.2)$$

Algorithm 1: Password hash

Assumption: The plaintext password is encoded in utf-8 and is only made of Basic Latin characters

Input: $p = p_0, ..., p_{\text{len}(p)-1}$ (plaintext password), where p_i is a utf-8 character (i.e, a hex number) for $i \in \{0, ..., \text{len}(p) - 1\}$

Output: $e = e_0, ..., e_{\text{len}(p)-1}$ (password hash of same length as password)

$e_{-1} \leftarrow 0$

$S \leftarrow \sum_{i=0}^{\text{len}(p)-1} p_i$

for $i \leftarrow 0$ **to** $len(p) - 1$ **do**

  $e_i \leftarrow f(e_{i-1} * i + S + p_i * (i + 1))$ where f was defined previously in Equation 5.2

$e \leftarrow e_0, ..., e_{\text{len}(p)-1}$

return e

The work of M. Miles has been extended to break the hash algorithm. To find back a pre-image of e we need to solve the following equations system for $S, p_1, p_2, ..., p_{\text{len}(p)-1}$,

$$\begin{cases} e_0 = f(S) \\ e_1 = f(p_0 + p_1 + S) \\ e_2 = f(2 * (p_1 + p_2) + S) \\ ... \\ e_i = f(i * (p_{i-1} + p_i) + S) \\ ... \\ e_{\text{len}(p)-1} = f((\text{len}(p) - 1) * (p_{\text{len}(p)-1-1} + p_{\text{len}(p)}) + S) \end{cases} . \tag{5.3}$$

And then using the definition of S we obtain $p_0 = S - \sum_{i=1}^{\text{len}(p)-1} p_i$. The code can be found in the *NinjaCrane github* [40].

Moreover, by using SHA256 as such: SHA256(salt + password) and sending it in clear; the following vulnerability observations can be made regarding an active attacker:

1. The communication is flawed by design as the salt is also sent in clear. More precisely, the attacker can leave the sent salt unmodified but tampers the sent SHA256(salt + password) into SHA256(salt + tampered_password), where tampered_password is a password chosen by the attacker.
2. Making use of a hash of concatenation rather than a HMAC could lead to an extension attack. For example an adversary tampering the sent SHA256(salt + password) into SHA256(salt + password + adding_characters) will modify the correct password into password + adding_characters.
3. By modifying the salt the adversary could perform a collision attack. For example let's assume that the salt is equal to salt1 + x then we have that SHA256(salt + password) = SHA256(salt1 + x + password). A collision hash is created by removing x at the end of salt and by adding x to the beginning of password.

Program Upload Phase

Once the password setup and the information regarding the project have been sent to the PLC, the engineering workstation will upload the program using the `UPLOAD_BLOCK` request function.

Connection Phase

The reservation phase establishes the connection between the PLC and the engineering workstation. To do so, the engineering workstation sends a random nonce (denoted `client_nonce`) and the PLC replies by sending another random nonce (denoted `server_nonce`) as shown in Figure 5.5.

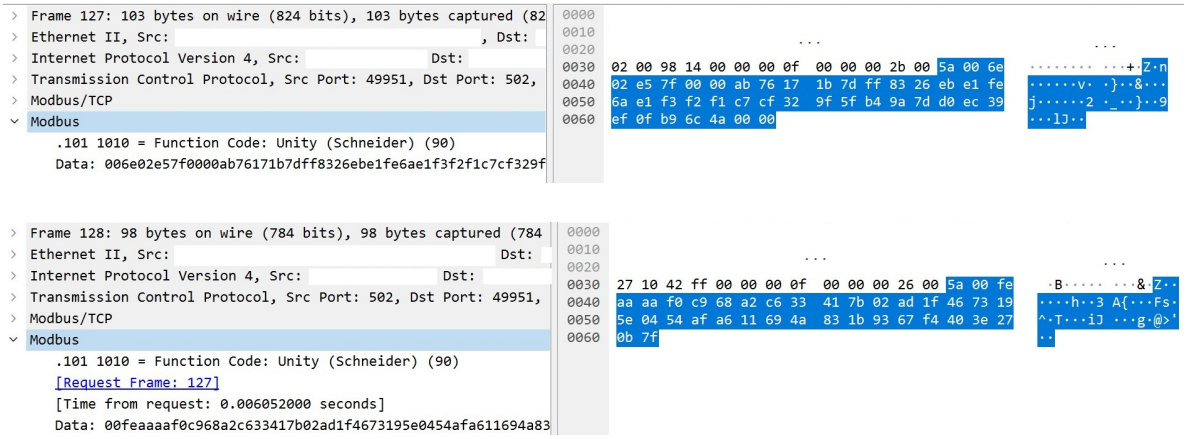


Figure 5.5: Top: Client (i.e., workstation) Nonce is `0xab76...6c4a`. Bottom: Server (i.e., PLC) Nonce response is `0xf0c9...0b7f`.

Information Gather Phase

In this phase the PLC sends back the packet it received in Section 5.3.3 Setup Phase.

Request Sending: Starting the PLC

Once the connection is made the engineering workstation can start & stop the PLC as shown in Figure 5.6, upload a new program, change variables in the PLC, or change the password.

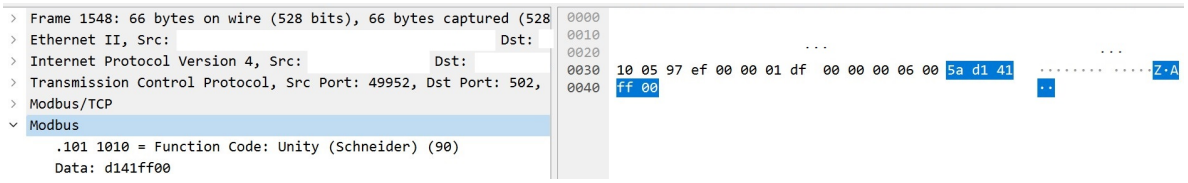


Figure 5.6: Packet sent by the workstation to start the PLC.

Diagram Synthesis

The synthesis diagram in Figure H.1 models the connection establishment between the PLC and the workstation. After this connection, the engineering workstation sends to the PLC the command to start it.

5.3.4 Exploiting the ModiPwn Vulnerability

The ModiPwn vulnerability (i.e., CVE-2021-22779 [41]) crafts an authenticated packet by simply spoofing the communication.

First, let `hw_id` denotes the hardware identifier. This identifier is fixed and specific to the PLC. The PLC sends this identifier when the engineering workstation requests for it as show in Figure 5.7.

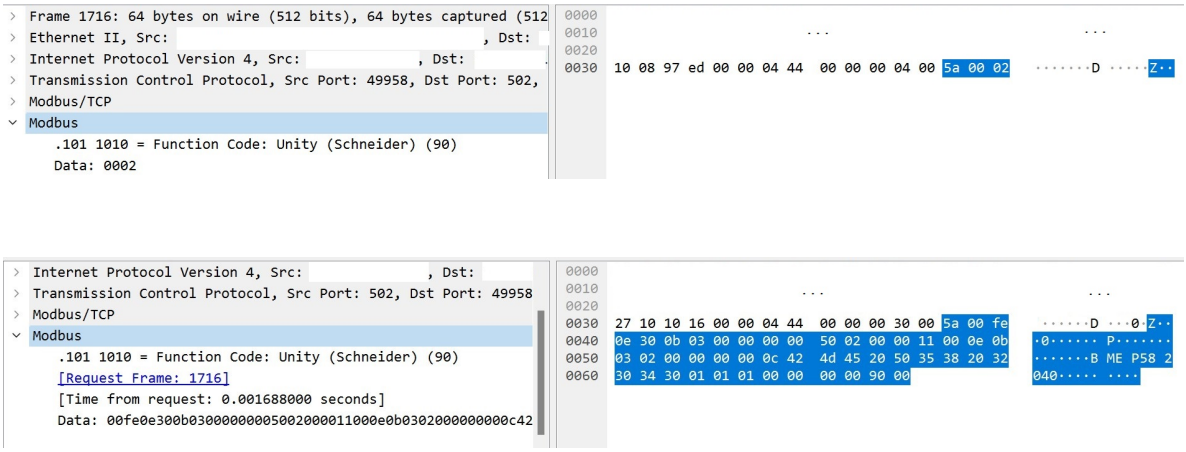


Figure 5.7: Top: Packet sent by the workstation to obtain the hardware identifier after connection establishment (i.e., request `0x02`). Bottom: Packet replied by the PLC with the hardware identifier (i.e., `0x0e0b0302`) and communication module name (i.e., `BMEP582040`).

A strong authenticated modbus command (e.g., start PLC) is composed as such :

`0x5A, session, UMAS function code, SHA256(SHA256(hw_id + client_nonce) + 0x5A + session + command + SHA256(hw_id + server_nonce)), command.`

As an example, the command to start the PLC is `command = 0x40FF00`.

The ModiPwn vulnerability exploits the fact that all values (`session`, `hw_id`, `client_nonce`, `server_nonce`) are known to an adversary only spoofing the communication. An adversary can thus craft any authenticated chosen-`command` packet.

As an important note, a patient adversarial spoofer observing any authenticated packet could perform a replay attack. Indeed, the packet authentication does not prevent from a replay attack as the packets do not depend on a timestamp or a tagging mechanism.

5.3.5 `MONITOR_PLC`: Arbitrary Read/Write on PLC's System Bits or Words

The UMAS communication protocol includes a `MONITOR_PLC` function (`0x50`) to read or write system bits or words. To do so, first, the workstation must send a packet to inform which variables to monitor and then must send a second packet to read or write one variable. The `MONITOR_PLC` packet takes the following format:

`0x5A`, `session`, `0x50 0x15 0x00 0x03 0x01`, `header`, `read/write`, `length`, `action` (read or write), `code`, `system bit`, `system bit to read/write + 4`, `unknown`, `system bit value to write` (if write), `0x00 0x00 0x00 0x01`

In short, the adversary needs to guess what is the system bit/word memory location. But, if the adversary has access to the program running on the PLC, it has access to this mapping and can thus read or write any wanted variable. For this reason the *NinjaCrane* attack includes program exfiltration, requiring the workstation to be connected to the internet. Furthermore, the *NinjaCrane* attack will then use this `MONITOR_PLC` function to change the variable associated to the speed of the polar crane or to the activation of the polar crane rotation.

Chapter 6

The NinjaCrane Attack

6.1 Entry Points

The implementation choice of the entry point infecting the workstation has been to create two equivalent entry points — the USB Ninja cable and the malicious mouse. Payloads stored on the USB Ninja cable can be triggered over BLE which makes it more interactive for a live demonstrator. The drawback of the USB Ninja cable is the storage capacity, too small to store the *NinjaCrane* malware. The *NinjaCrane* malware must be already downloaded or stored on the workstation (or on the device connected to this USB Ninja cable). To overcome this, a malicious mouse has been made. It offers a better flexibility as any board with USB-B connector can be placed and connected inside the mouse. This allows to store the malware directly inside the mouse.

6.1.1 USB Ninja

The BLE communication protocol used by the USB Ninja cable has been reverse-engineered by the Embedded Lab Vienna for IoT & Security [42]. From this work, a script controlling the USB Ninja cable from the BLE of a windows computer has been made with the help of the `bleak` python package.

Given the USB Ninja cable's Bluetooth MAC address, the script connects to its associated BLE GATT server. To trigger the payloads stored on the USB Ninja cable, two BLE packets are sent consecutively. The first one writes at BLE handle `0x36` the password. The second packet writes at BLE handle `0x36` either `A=L\r\n` or `B=L\r\n` to trigger the first or the second payload respectively. The script is used by the attacker's GUI; button `Deploy Payload` sends `A=L\r\n` and button `Trigger Attack A` sends `B=L\r\n` to the USB Ninja cable.

6.1.2 Malicious Mouse

The clean mouse is a Logitech M90 USB wired mouse. A DIY Tiny USB Hub [43], placed inside this mouse, splits the USB wired connection in order to connect both the mouse and a micro USB type-B male connector. A pre-programmed Adafruit Trinket M0 is plugged to this connector. The Trinket M0 is a HID-capable device that can emulate a keyboard or a mouse. Picture of this assembly can be found in Figure G.2. The Trinket M0 is programmed using `CircuitPython v8.2.0` by placing in its associated USB drive two different files (`code.py` and `boot.py`). When the Trinket is plugged-in the `boot.py` and the `code.py` scripts are respectively executed. The first script will deactivate Midi, REPL and USB drive (but not HID). The second script executes the HID payload and then activates the USB drive where the malware is stored.

6.1.3 Payloads of the HID-capable Device

The payload of the cable executes the following instructions: open Run command, open a hidden powershell terminal, write and execute a script that will detect internet connection, download a malware and execute it in background. Moreover, for a background persistence, a malware's shortcut is placed in the engineering workstation startup folder.

The payload of the mouse executes similar instructions: open Run command, open a hidden powershell terminal, and execute the malware in background. No internet connection is required as the malware is stored in the malicious mouse.

6.2 Attacker's GUI

No attacker's GUI are available to control the malicious mouse as the Trinket M0 has no wireless communication. But if the entry point is the USB Ninja cable, a python interface to control the attack can be used. This interface is shown in Figure I.1. Python interface has been written with *PySimpleGUI* package in order to better illustrate the steps taken by adversary and simplify the attack explanations. This GUI is made of one interface modeling the attack progress on the ICS architecture and two buttons. The first button `Deploy Payload` will trigger the first payload stored on the USB Ninja cable. The second button `Trigger Attack A` will trigger the second payload stored on the USB cable.

6.3 The *NinjaCrane* Malware

The *NinjaCrane* malware is a python script that is converted into an executable file named `malwar3.exe`.

6.3.1 MITM Script

The MITM script is a .pyw extension python file for better flexibility and stealthiness. This script first by-pass the UAC consent prompt to execute itself with admin rights. It then waits for a trigger to perform the MITM attack on the polar crane.

UAC Consent Prompt By-Passing

First, the script detects if it runs with administrative privilege. If not it will exploit the `fodhelper.exe` weakness (i.e., `fodhelper.exe` elevates itself to run in a higher integrity level) to execute again itself with admin privilege. Otherwise it continues with the attack.

To exploit the `fodhelper.exe` weakness, two register keys are created at `HKEY_CURRENT_USER\Software\Classes\ms-settings\shell\open\command`. The first key (`DelegateExecute`) is set to empty string and the second key (Nil) is set to the command to execute with higher integrity context (i.e., executes this python script again) as shown in Figure 6.1. No privilege are needed to modify and create those `HKEY_CURRENT_USER` register keys. Finally, the python script executes `C:\\Windows\\System32\\fodhelper.exe` to run itself again with higher privilege.

Name	Type	Data
(Default)	REG_SZ	C:\\Windows\\System32\\cmd.exe /c start /min "" C:\Users\gaietan\Desktop\powershell_script\malwar3.exe
DelegateExecute	REG_SZ	

Figure 6.1: Register keys created to By-Pass Windows Consent Prompt UAC

Waiting for Attack Trigger

Once the script executes with admin privilege it makes use of the `pydivert` package to listen and modify the packets sent by the workstation to the PLC. The `pydivert` package is itself based on Windows Packet Divert (named `WinDivert`), a capture-and-divert package for Windows.

The script captures only the UMAS packets (i.e. packets from/to the engineering workstation's 503 TCP port). The script extracts the key information e.g., `server_nonce`, `client_nonce`, `salts` &

hash, Unity Pro version, project name, project folder etc.

Data Exfiltration

As soon as the script extracts the project folder it will try to exfiltrate over internet the .STU program and a text file with project information. The text file with project information is stored in the \Desktop\powershell_script\extracted_info.txt. The script thus uploads from the workstation the two files on a transfer.sh server and sends an SMTP email with the corresponding urls with the use of the mailtrap solution. The attacker from outside, receiving this email can then download those two files as shown in Figure J.1.

This data exfiltration allows the adversary to perform aside an analysis of the .STU program running on the PLC and to better weaponize the malware. Indeed, as soon as the adversary has access to the program, it has access to the memory addresses of every variables, their use and their type. This allows the adversary to craft a valid authenticated packet to modify or read any of those variables.

Send Malicious Authenticated Packet

When the adversary triggers the attack (or when mouse is plugged-in), the USB Ninja (or malicious mouse respectively) will write to a text file. The script is continuously reading this text file. In case of file modification, the script will start to send malicious authenticated packets. For a better cyberattack illustration purpose the following malicious packets are sent:

- **Packet 1**: Sends a request to initiate the variables monitoring.
- **Packet 2**: Sets the rotation speed of the polar crane's rotation to 34 % of the maximum motor's speed.
- **Packet 3 & 4**: Activate the polar crane's rotation. First is a rising edge (i.e., sets the corresponding internal variable to 1) the other one is a falling edge (i.e., sets the corresponding internal variable to 0).
- **Packet 5**: Sets the rotation speed of the polar crane's rotation to 63 % of the maximum motor's speed.
- **Packet 6 & 7**: Activate the polar crane's rotation. Those packets are needed in order for the polar crane to apply the previously set rotation speed.
- **Packet 8 & 9**: Deactivate rotation. First is a rising edge, the other one is a falling edge.
- **Packet 10**: Stops the PLC.

To sum up, the above packets will make the polar crane rotate slowly first, then fast, then completely stop the polar crane and the PLC. This scenario could be improved to also show that the attacker can extract the value of the internal variables. To send the malicious packet the script completely modifies a sent `MONTOR_PLC` (i.e., `0x50`) packet with the corresponding Modbus data. `PyDivert` automatically recomputes the checksum. However the TCP sequence and acknowledgment numbers must be modified according to the modified packet length. This modification must apply to any fresh packet sent or received (i.e., the script synchronizes the ack and seq numbers in order to avoid duplicate ack/seq, TCP re-transmission, and connection loss).

6.3.2 Converting a Python Script to an Executable *Malwar3.exe*

As the engineering workstation has no python environment the script must be compiled. To compile the python script in one file, the `pyinstaller` package (or its Graphical Interface equivalent, `auto-py-to-exe 2.36.0`) has been used. Moreover the Ultimate Packer for eXecutables (UPX) reduces the compiled binary's size. The executed command to compile the python script is: `pyinstaller -noconfirm -onefile -windowed -upx-dir "path/to/upx_filr" "path/to/malware_script.pyw"`

Chapter 7

Evaluation

7.1 Raising Awareness with an Attack Demonstrator

Demonstration video of the polar crane's cyberattack can be found in the *NinjaCrane github* [40]. In parallel to this video a demonstration show has been created to better illustrate a real-world scenario and is detailed is the section.

7.1.1 Demonstration Setup

Two persons will act in the demonstration. The first plays the role of the polar crane's automation technician that programs the PLC with the use of his engineering workstation. The second one plays the role of the hacker. For simplicity, we assume that the hacker is in the same room as the technician. This is a violation with respect to the threat model (as a external adversary was assumed). The attacker can for example be a subcontractor. Finally the entry point used in this demonstrator is the USB Ninja cable (and not the malicious mouse).

The situation setup is the attacker talking with the technician and asking him to charge his phone (or any USB device) with the USB Ninja cable. The technician plugs the USB Ninja cable to charge his phone as shown in Figure 7.1.

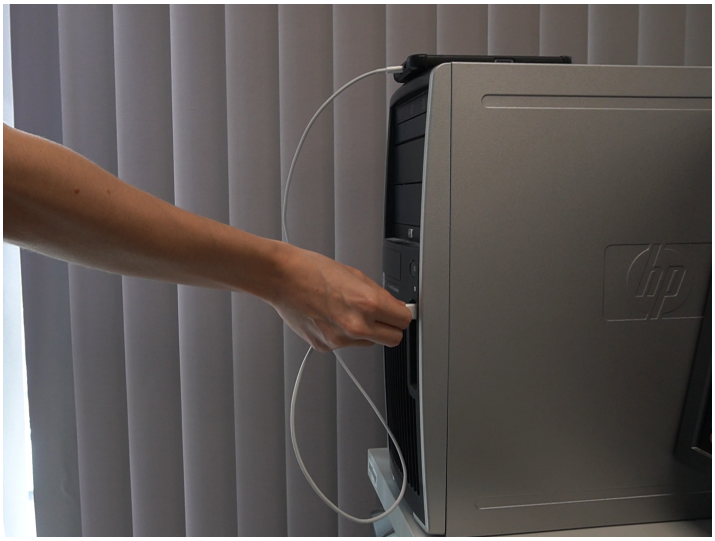


Figure 7.1: Plugging USB Ninja Cable.

7.1.2 Engineering Workstation Infection

Once the USB Ninja is plugged-in. The attacker will for example ask the technician to take a coffee break while leaving the session unlocked. Once done, the attacker triggers the USB Ninja cable with the adversary's GUI to run the payload. This will open the Execute service and open a powershell command in background to run the commands as shown in Figure 7.2.

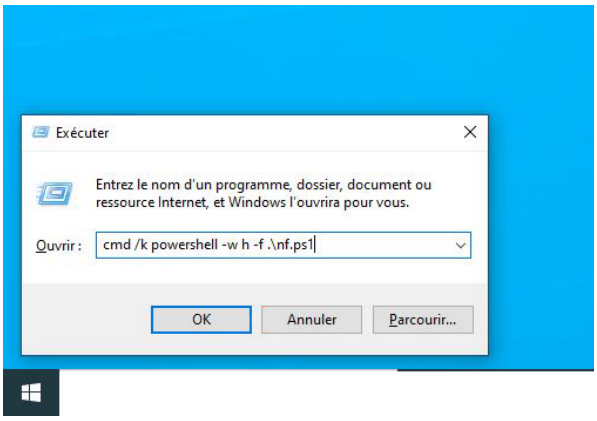


Figure 7.2: Opens the Execute service and opens a hidden powershell on the engineering workstation.

The commands will create four files; `VBs.vbs` and `script.ps1` are scripts to download the malware and run it in background at each session opening, `malwar3.exe` is the downloaded

NinjaCrane malware performing the MITM attack and finally `trigger.txt` is the text file containing the instruction to triggers the polar crane's attack. The four files are created in the `Desktop/powershell_script` folder as shown in Figure 7.3. The Figure K.1 is a flowchart describing the sequence to infect to engineering workstation. For the demonstration, a modified version of the commands can be used to have an offline scenario that assumes that the `malwar3.exe` is already located at `\Desktop\Malware\malwar3.exe`

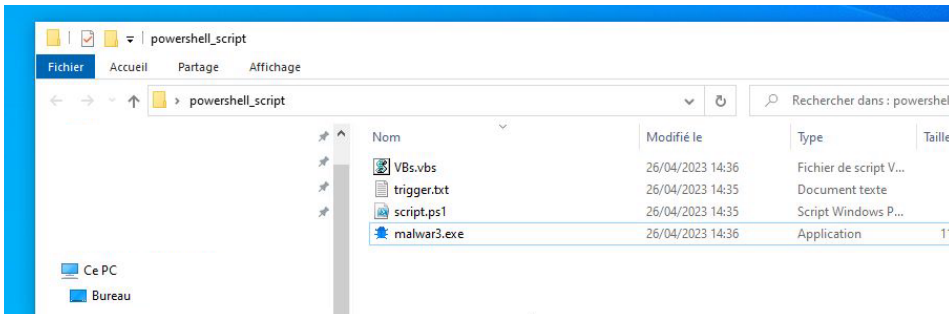


Figure 7.3: Created folders after the first USB Ninja payload execution.

7.1.3 Take the Polar Crane's Control

Once the malware is running on the engineering workstation, the attacker triggers once again the USB Ninja cable to effectively disturb the polar crane as described in section 6.3.1. The malware sends a packet to rotate the polar crane at 34% of maximum speed then at 63% of the maximum speed. This shows that the adversary could wear and tear the motor. Then it stops both the polar crane and the PLC. This shows that the adversary can stop the process at any time and for example stop the loaded polar crane above the reactor vessel.

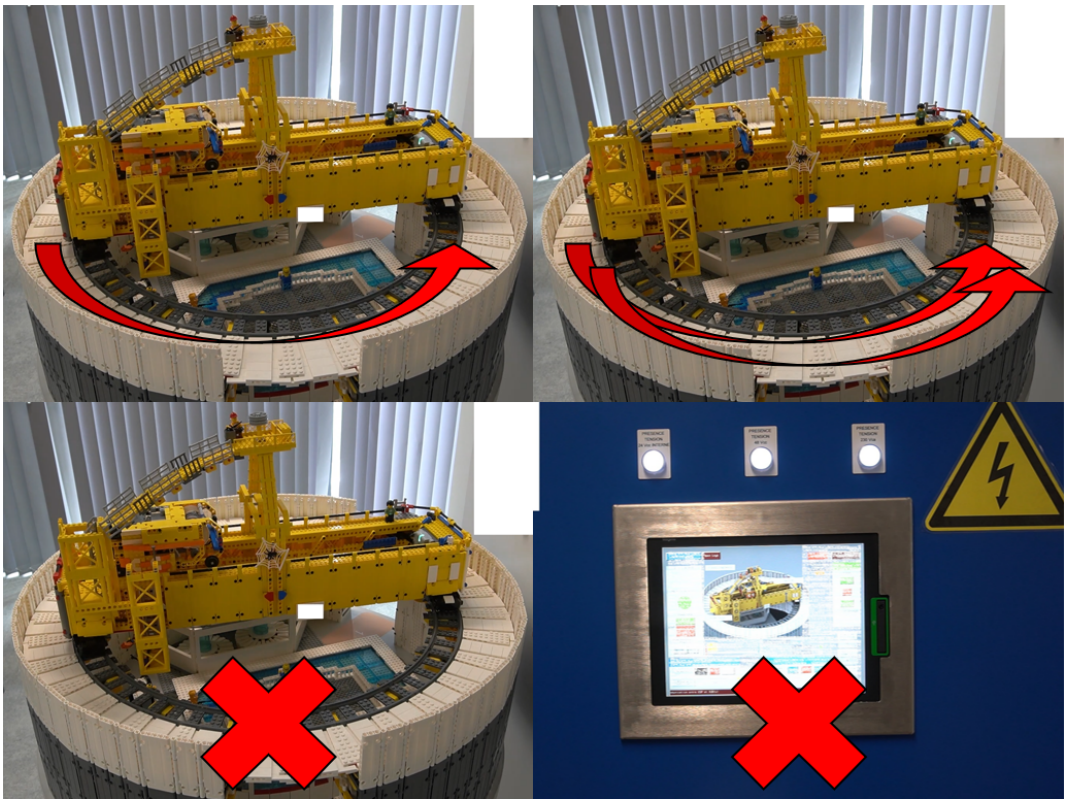


Figure 7.4: Attack Sequence order from left to right. Top-Left: Make the polar crane rotates at 43% of maximum speed. Top-Right: Make the polar crane rotates at higher speed (63% of maximum speed). Bottom-Left: Stop the polar crane. Bottom-Right: Stop the PLC.

7.2 Real World Feasibility of the Cyberattack

7.2.1 Engineering Workstation Infection

Several observations can be made regarding the success of the engineering workstation infection.

- Avast antivirus does not detect the malware executable as a threat. However, the avast antivirus fixed at a high protection level will notify the execution of the executable and ask the administrative user to block or not the execution. Moreover, the malware requires administrative privilege to execute. Thanks to the UAC bypassing method, it only requires that current logged-in user has administrative right and that the UAC level is not set to "Always Notify".
- There must be no protection regarding HID-capable device. Since July 2021, the Windows environment has a built-in solution to restrict USB devices called Microsoft Intune [44], [45].

The Intune service is a system policy that blocks any USB device not matching the hardware ID whitelist given by the user.

- HID-based attacks come with major drawbacks. First, an event must trigger the attack. It can be a time-based event (e.g., keyboard frames are sent 10 seconds after the connection) or attacker-based event (e.g., attacker triggers the attack over BLE, as for the USB Ninja). Moreover, if windows password account is unknown to the adversary, this trigger must be run during a session unlocked. And if windows password is known to the adversary, the workstation must be turned-on. Finally, as the payload is made of keyboard frames, the attack is not completely stealth during few seconds — the time to type all the key-frames.

7.2.2 Live Demonstration Feedback

This demonstration has been played during the *Journée Sûreté* at the *Direction Projet Nouveau Nucléaire* , during the *Plénière* of the *Département des Composants Electriques et Electromécaniques* and partially-played (i.e., only with a video support) at the *Cercle d'Experts Sûrté sur la Cybersécurté* at *EDF Saclay*. The demos were played with the goal to raise awareness on cybersecurity and to show how a malware could propagate from the IT to the OT systems in case of security policy breach. The demo shows a concrete social engineering scenario that leads to the industrial process failure — the polar crane stops as well as the connection between the workstation and the PLC. Anonymously collected feedback from the demo participants can be find in Appendix L. One downside regarding the reliability of the demo can be noted. Indeed, the connection between the PLC and the engineering workstation can sometimes drop due to time-out or the engineering workstation infection must sometimes be repeated in case of bluetooth connection lost with the USB Ninja cable or if a mouse click is made during the HID-keystrokes injection.

7.2.3 MITM Malware Attack

Similarly, some observations can be made regarding the success of the MITM attack.

- The MITM malware must catch the connection phase with the PLC to conduct later the attack phase. Moreover during the attack phase, engineering workstation and PLC must be connected. Unfortunately, due to connection time-out the connection between the workstation and the PLC may drop after half an hour. To overcome this, the timeout variable in Unity Pro can be increased.
- In the current state, the *NinjaCrane* attack is not resilient to program change that results in the change of variable memory location. Indeed, the attack crafts and sends pre-fixed packet to change the value of a specific memory location corresponding to a PLC's inner variable.

- The attack sequence played by *NinjaCrane* malware is fixed and does not consider the current state of the polar crane. During the demonstration, *NinjaCrane* malware assumes that the polar crane's and PLC are already in a specific state (i.e., PLC is in run and is in the mode to control the polar crane). However, this might not be the case and the *NinjaCrane* malware needs to be adjusted accordingly. For example the *NinjaCrane* malware could craft some packets to read some variable states or could be made reactive by adjusting the sent packet according to the information previously gathered by observing the communication.

7.3 Mitigation and Counter Measures

7.3.1 Engineering Workstation Hardening

Solutions for engineering workstation hardening are highly-dependent on the use-case, environment, and threat model. Below is a non-exhaustive list of solutions that would have blocked the *NinjaCrane* attack:

- Physically blocking the USB port (e.g., Lindy USB Port Blocker). It physically prevents the use of the USB port. However the blocker can be forced and the key to unlock the blocker is any-blocker compatible.
- Physically blocking the data transfer wires (e.g., USB Condom). The data blocker physically prevents any data transfer but can still be used for slow charging. Charge-only cables also exist. Figure 7.5 illustrates physically how the USB data blocker works.

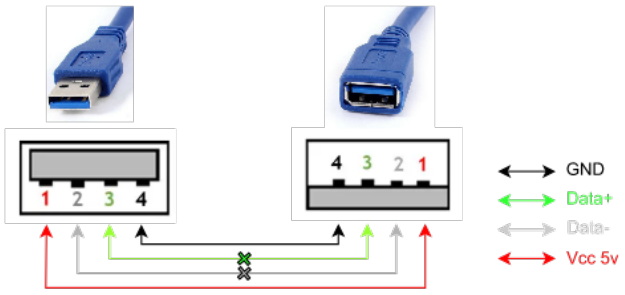


Figure 7.5: Data blocker. The data blocker cuts the Data+ and Data- transmission.

- De-activates USB port. This could be done via the BIOS, via Group Policy, Device Manager, 3rd party software, or uninstalling the USB driver.
- De-activates USB port based on whitelisting. This can be setup via Microsoft Intune, Group Policy or 3rd party software. USB ID device can however easily be spoofed.

- *USBFILTER* solution from Tian et al. [46] [47] (or similarly *Netfilter*) instruments the USB stack to filter USB packets at a fine granularity following user-defined rules. With correct rules, this solution prevents a mouse or a charging cable to act as a keyboard. Similarly the *GoodUSB* solution (currently only available for the Linux USB Stack) associate user's expectation of the device's functionality.

7.3.2 Cybersecurity of the Supply Chain

Cybersecurity over the Supply Chain is still in active development and mostly boils down to the cybersecurity principles — separation and least privilege, monitoring, life cycle development (e.g., V model), access controls, audits and regular security assessments, sensitive data encryption etc. Below is a non-exhaustive list of practical solutions that would have blocked the *NinjaCrane* attack:

- Blockchain technology (e.g., VeChain or ChainLink). Blockchain can help to ensure products authenticity, movements and origins via tamper-proof records available to the supply chain tenants.
- In-depth inspection. Inspection can take the form of randomized security assessments of the electronic devices. By opening the maliciously tampered mouse or the USB Ninja cable the alteration would have been detected. Furthermore, characteristics inspection would also be a working lead (e.g., mouse weight or USB Ninja cable heating).
- Strict security policy between the tenants of the supply chain. As an example, the USB Ninja cable could be offered to an employee as a gift.

Chapter 8

Related Work

In this chapter is described the closely related work, without any attempt to compile an exhaustive list. In the MSc thesis was developed an APT attack on a non-simulated NPP-related process testbed. Most of the literature is based either on process simulation or on system virtualization. First section describes some cyberattack demonstrators or testbeds of ICS based on virtualization or on a hybrid configuration (called Hardware-In-The-Loop) of software-virtualization and hardware. They offer better flexibility, lower cost, better portability but suffer from a lack of representativeness compared to this study. Second section focuses on related work that includes non-simulated industrial process.

8.1 Hardware-In-The-Loop-based or Virtualized Attack Demonstrator on ICS Based on Simulated Industrial Process

De Brito et al. [48] built a Modbus/TCP network testbed simulating the NPP process (with Asherah NPP Simulator) and including a SCADA system (ScadaBR), a rogue PLC (OpenPLC on Arduino), an inside adversary (Kali Linux system), and a historian. The proposed cyberattack assumes that the adversary already neutralized a control system and replaced it by a rogue PLC. The rogue PLC maliciously tampers a pump speed variable while the Kali Linux system conducts a MITM attack so that the SCADA does not notice this tampering. De Brito et al. also explored a network monitoring defense based on the "Time from request". This work proposes a low-cost, simplified, and hybrid network testbed to conduct cyberattack on an ICS of a NPP. As this Master thesis, a MITM attack on Modbus/TCP is conducted, however the adversary is assumed to be inside the network with a Kali Linux system and a rogue PLC and adversary directly targets the NPP process.

Puys et al. [49] proposed two Hardware-In-The-Loop cyber-ranges in order to build an ICS cybersecurity awareness and a student training program respectively. First one, WonderICS, includes

3D simulation of an industrial process (i.e., hazardous gases management, hydroelectric power plant, chemical process of Tennessee-Eastman), PLCs, protection relay, HMIs, Kali Linux Virtual Machine (VM), engineering workstation VM. Two attack scenarios were considered but the first one is the closest to the attack presented in this MSc project. It makes use of a Rubber Ducky USB key that performs cryptolocking, keyboard disabling, and PLC commands injection. The attack injects a malicious Modbus frame writing a fake value on the PLC's register. In comparison to the *NinjaCrane* attack, it offers less stealthiness - a malicious cable or mouse is more stealth than a Rubber Ducky - and no flexibility - as the charge delivery and exploitation are automatically and instantaneously launching when Rubber Ducky is plugged-in. The second HIL cyber-range is called G-ICS and it offers a student training program. The students can then practice malicious packet injection, firewall rule editing (to protect against a known exploit), industrial proprietary protocols reverse-engineering and protocol fuzzing to find a protocol vulnerability.

Much research have been made to construct an ICS testbed for cybersecurity training such as the KYPO4INDUSTRY [50], the Micro-CI Testbed from Hurst et al. [51], or the EPS-ICS Testbed from Gao et al.[52]. None of them includes an APT scenario, however they all propose a flexible, pedagogical, portable and low-cost solution.

8.2 Cybersecurity Testbed on ICS Based on Non-simulated Industrial Process

Several universities have developed more realistic industrial processes such as Technische Hochschule Augsburg with LICSTER [53]. LICSTER is a simple open-source and low-cost ICS testbed where several single step attacks at any ICS level can be conducted such as sniffing, DoS or MITM. It benefits from its simplicity while still proposing a haptic understanding of the consequences of the cyberattack on the ICS. The testbed includes open-source Remote IO, PLC, SCADA and HMI.

The New Orleans University SCADA testbed from Ahmed et al. [54] includes three (simulated and non-simulated) industrial processes. It is made of PLC, Relay, switch, historian and HMI. Even though no cyberattack has been made it offers diversity, with the presence of three different vendors PLC to reverse-engineer the proprietary communication protocols.

A practical cyberattack demonstrator has also been made by *Bristol Cyber Security Group* [19] [20] [21], where A. Rashid et al. created a full attack scenario starting from cloud IoT infrastructure and causing at the end water treatment process enter into an unsafe state. They exploited *Tomcat* vulnerabilities, malicious PDF and a zero-day *SCADAPack* vulnerability. This testbed illustrates how complex an attack in several steps can be made. The attacker is still assumed to be inside the SCADA network but A. Rashid et al. suggest that a malicious USB device could be the attack vector. This ICS training testbed including an APT scenario is believed to be the most advanced one and realist in the literature. In comparison, the *NinjaCrane* attack can be seen as

another APT scenario that reinforces the need to monitor the supply chain and the need to integrate cryptographic authentication standard in the industrial proprietary protocol.

This need to have a better fidelity in the industry field pushed companies to build their own ICS testbed like the OPSWAT (CIP Lab) electric station ICS [18] or the *Check Point* chemical plant ICS testbed [17]. *OPSWAT* (i.e. *CIP Lab*) built an electric station ICS to promote their *OTfuse* defense solution [18]. The second testbed, the *Check Point 1200R* security gateway is connected to the ICS network and prevents part of the fuzzing attacks, value-tampering attacks or exploitation of known vulnerabilities. This work is focusing on the ICS defense and not on the attack exploration nor the awareness part unlike presented work. As a side observation, the *NinjaCrane* attack exploiting the *ModiPwn* vulnerability would not have been detected by their security gateway.

Chapter 9

Conclusion

9.1 Future Work

The major drawbacks of HID-based attacks, like the malicious mouse or cable, is the need to perform the attack while the workstation is on and session is unlocked — if windows account password is unknown to the attacker. This generally only happens when the workstation is in use. Moreover, as the attack emulates keystrokes, it is never completely stealth and a window flashes on the computer. The moment to trigger the HID payload plays a key role to conduct a stealth attack. It is generally triggered using a timer, over WiFi, or over BLE as in this study. In order to perform the HID-based attack while the session is locked — and windows account password is unknown — a maliciously modified keyboard spoofing the password (i.e., hardware key-logger) and then performing the HID-based attack can be considered. This can be done by modifying the electrical schematic from Figure G.1 by replacing the mouse by a keyboard and adding a spliced GPIO-to-USB converter that transmits the sent keystrokes from keyboard to the Trinket M0. Alternatively, other USB attacks can be considered with stealth auto-run capability (e.g., LNK stuxnet/fanny USB flash drive exploit, AutoRun exploits, Buffer overflow based attack etc.) and apparition of a fake lock-screen to spoof the windows password (see *NinjaCrane github* [40]) for privilege escalation.

The malware performing the MITM attack is a heavy executable (around 11 Mo) due to the fact that it is a python converted script and can thus not be stored on a microcontroller Read Only Memory (ROM). To overcome this, the malware could be re-written in low level language with the help of `WinDivert` [55].

Moreover, a third point to be improve is the need to run the malware with admin privilege. The malware can bypass UAC consent prompt but it requires that the current logged-in user has admin privilege. To relax this assumption, windows privilege escalation technique must be used like the `winPEAS` solution [56] working on previous Windows versions. As an observation, the engineering

workstation is rarely connected to the internet and the non updated workstation assumption might be added in the threat model.

The polar crane testbed offers an ideal environment to study the UMAS protocol and to explore fuzzing techniques on the PLC like boofuzz or mutiny fuzz [25]. Furthermore the *NinjaCrane* attack is a first stage to explore the defense mechanisms to protect for example against USB attacks. The *Microsoft Intune* [44] solution is not a silver bullet as it raises problems about maintaining the HID whitelist and it does not protect against all USB-based attacks. Other software-based solution could be integrated as the integrity system from Griscioli et al. [57] making use of cryptographic primitives (but not preventing HID-based attack), the *GoodUSB* or the *USBFILTER* solution from Tian et al. [46] [47] trying to protect the USB stack or to add a USB firewall respectively.

9.2 Conclusion

This study developed a cybersecurity attack demonstrator called *NinjaCrane* on the polar crane's ICS. The *NinjaCrane* attack is the first cyberattack demonstrator in the literature targeting a specific nuclear equipment of a NPP — the polar crane. The polar crane is a handling equipment situated in the CB and used during nuclear unit shutdown to carry and move heavy loads. By targeting a non-simulated PLC controlling the polar crane, the *NinjaCrane* offers a framework for an awareness program on OT cybersecurity with high realism level.

The *NinjaCrane* attack first infects the engineering workstation in the air-gap polar crane's ICS via the connection of a malicious USB device — a malicious mouse or a malicious USB Ninja cable — on this network. It then executes a persistent malware performing data exfiltration and lateral movement. Regarding data exfiltration, the malware running on the engineering workstation will send an email if it is connected to the internet with the control logic program and some PLC and workstation information. Regarding lateral movement, the malware performs a MITM attack to write arbitrary system bits in the PLC and take the control of the polar crane process. By taking control over the polar crane's ICS it is possible to create a significant safety event — a load stuck above the reactor pressure vessel.

The *NinjaCrane* attack highlights the importance to monitor the USB-capable devices brought by the supply chain in the information system and to respect the security policy. Moreover, it reveals the authentication weaknesses of the cryptographic scheme used by the UMAS protocol. This protocol used by Schneider Electric to program PLCs offers no encryption, no integrity and weak authentication.

Availability

All the material used for this Master thesis is available at `https://github.com/grennault/NinjaCrane` and is released under GNU GPLv3 and GNU FDLv1.3.

Bibliography

[1] Animesh Pattanayak and Matt Kirkland. "Current Cyber Security Challenges in ICS". In: *2018 IEEE International Conference on Industrial Internet (ICII)*. 2018, pp. 202–207. DOI: 10.1109/ICII.2018.00013.

[2] Keyong Wang, Xiaoyue Guo, and Dequan Yang. "Research on the Effectiveness of Cyber Security Awareness in ICS Risk Assessment Frameworks". In: *Electronics* 11.10 (2022). ISSN: 2079-9292. DOI: 10.3390/electronics11101659. URL: https://www.mdpi.com/2079-9292/11/10/1659.

[3] Lai-Wan Wong, Voon-Hsien Lee, Garry Wei-Han Tan, Keng-Boon Ooi, and Amrik Sohal. "The role of cybersecurity and policy awareness in shifting employee compliance attitudes: Building supply chain capabilities". In: *International Journal of Information Management* 66 (2022), p. 102520. ISSN: 0268-4012. DOI: https://doi.org/10.1016/j.ijinfomgt.2022.102520. URL: https://www.sciencedirect.com/science/article/pii/S0268401222000548.

[4] CISA - Cybersecurity and Infrastructure Security Agency. "ICS Training Available Through CISA". In: URL: https://www.cisa.gov/ics-training-available-through-cisa (visited on 06/21/2023).

[5] Hyeun-Suk Rhee, Cheongtag Kim, and Young U. Ryu. "Self-efficacy in information security: Its influence on end users' information security practice behavior". In: *Computers & Security* 28.8 (2009), pp. 816–826. ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2009.05.008. URL: https://www.sciencedirect.com/science/article/pii/S016740480900056X.

[6] Elena Sitnikova, Ernest Foo, and Rayford Vaughn. "The power of hands-on exercises in SCADA cyber security education". In: *Information Assurance and Security Education and Training: 8th IFIP WG 11.8 World Conference on Information Security Education, WISE 8, 7 and 6, Revised Selected Papers [IFIP Advances in Information and Communication Technology, Volume 406]*. Ed. by R C Dodge and L Futcher. Germany: Springer, 2013, pp. 83–94. DOI: 10.1007/978-3-642-39377-8_9. URL: https://eprints.qut.edu.au/66352/.

[7] Giddeon Angafor, I. Yevseyeva, and Ying He. “Bridging the Cyber Security Skills Gap: Using Tabletop Exercises to Solve the CSSG Crisis”. In: Oct. 2020. ISBN: 978-3-030-61813-1. DOI: 10.1007/978-3-030-61814-8_10.

[8] Ralph Langner. *To Kill a Centrifuge. A Technical Analysis of What Stuxnet's Creators Tried to Achieve*. Nov. 2013. URL: https://www.langner.com/wp-content/uploads/2017/03/to-kill-a-centrifuge.pdf (visited on 05/31/2023).

[9] Nicolas Falliere, Liam O Murchu, and Eric Chien. *W32.Stuxnet DossierVersion 1.4*. Feb. 2011. URL: https://docs.broadcom.com/doc/security-response-w32-stuxnet-dossier-11-en (visited on 05/31/2023).

[10] CISA - Cybersecurity and Infrastructure Security Agency. “Critical Infrastructure Sectors”. In: URL: https://www.cisa.gov/topics/critical-infrastructure-security-and-resilience/critical-infrastructure-sectors (visited on 06/21/2023).

[11] Kaspersky ICS CERT. *Reports*. URL: https://ics-cert.kaspersky.com/publications/reports/ (visited on 05/31/2023).

[12] DOE, CISA, NSA, and FBI. *Joint cybersecurity advisory - APT Cyber Tools Targeting ICS/SCADA Devices*. 2022. URL: https://www.cisa.gov/sites/default/files/publications/AA22-103A_APT_Cyber_Tools_Targeting_ICS_SCADA_Devices.pdf (visited on 05/31/2023).

[13] NIST - Information Technology Laboratory Computer Security Resource Center. “Information Technology - Cybersecurity”. In: URL: https://www.nist.gov/cybersecurity (visited on 06/21/2023).

[14] NIST - Information Technology Laboratory Computer Security Resource Center. “Operational Technology Security - Publications”. In: June 4, 2021. URL: https://csrc.nist.gov/Projects/operational-technology-security/publications (visited on 06/21/2023).

[15] David Formby, Milad Rad, and Raheem Beyah. “Lowering the Barriers to Industrial Control System Security with GRFICS”. In: *2018 USENIX Workshop on Advances in Security Education (ASE 18)*. URL: https://www.usenix.org/conference/ase18/presentation/formby, https://github.com/Fortiphyd/GRFICSv2. Baltimore, MD: USENIX Association, Aug. 2018.

[16] Conrad Ekisa, Diarmuid Ó Briain, and Yvonne Kavanagh. “VICSORT - A Virtualised ICS Open-source Research Testbed”. In: *2022 Cyber Research Conference - Ireland (Cyber-RCI)*. 2022, pp. 1–8. DOI: 10.1109/Cyber-RCI55324.2022.10032670.

[17] Check Point Software Technologies Ltd. “Ensure the Safety and Integrity of your Operational Technology (OT) Environment”. In: URL: https://www.youtube.com/watch?v=-JS_J_nSoBA, https://pages.checkpoint.com/ics-demo.html.

[18] OPSWAT (CIP Lab). "OPSWAT Launches World's First Interactive Mobile Lab for Critical Infrastructure Organizations". In: URL: https://www.youtube.com/watch?v=00O6Dw3xL3s, https://www.opswat.com/blog/opswat-launches-worlds-first-interactive-mobile-lab-for-critical-infrastructure-organizations.

[19] Joseph Gardiner, Barnaby Craggs, Benjamin Green, and Awais Rashid. "Oops I Did It Again: Further Adventures in the Land of ICS Security Testbeds". In: *Proceedings of the ACM Workshop on Cyber-Physical Systems Security & Privacy*. CPS-SPC'19. London, United Kingdom: Association for Computing Machinery, 2019, pp. 75–86. ISBN: 9781450368315. DOI: 10.1145/3338499.3357355. URL: https://doi.org/10.1145/3338499.3357355.

[20] B. Craggs, A. Rashid, C. Hankin, R. Antrobus, O. Şerban, and N. Thapen. "A reference architecture for IIoT and industrial control systems testbeds". In: *Living in the Internet of Things (IoT 2019)*. May 2019, pp. 1–8. DOI: 10.1049/cp.2019.0169.

[21] Awais Rashid, Joseph Gardiner, Benjamin Green, and Barnaby Craggs. "Everything Is Awesome! or Is It? Cyber Security Risks in Critical Infrastructure". In: *Critical Information Infrastructures Security*. Ed. by Simin Nadjm-Tehrani. Cham: Springer International Publishing, 2020, pp. 3–17. ISBN: 978-3-030-37670-3.

[22] Tashfiq Rahman, Rohani Rohan, Debajyoti Pal, and Prasert Kanthamanon. "Human Factors in Cybersecurity: A Scoping Review". In: July 2021. DOI: 10.1145/3468784.3468789.

[23] ONR Research - Office of Nuclear Regulatory Research. "TMI-2 Knowledge Management Library - (P11) Reactor Building Polar Crane". In: URL: https://tmi2kml.inl.gov/Documents/0a-Photos/(P11)%20Reactor%20Building%20Polar%20Crane.jpg (visited on 06/23/2023).

[24] *Defence in Depth in Nuclear Safety*. INSAG Series 10. Vienna: INTERNATIONAL ATOMIC ENERGY AGENCY, 1996. ISBN: 92-0-102596-3. URL: https://www.iaea.org/publications/4716/defence-in-depth-in-nuclear-safety.

[25] Gao Jian. "Debacle of The Maginot Line: Going Deeper into Schneider Modicon PAC Security". In: NSFOCUS, GEWU Lab - Hitcon 2021. URL: https://hitcon.org/2021/agenda/b128a44d-c492-410f-b04c-045548ce0590/Debacle%20of%20The%20Maginot%20Line%EF%BC%9AGoing%20Deeper%20into%20Schneider%20Modicon%20PAC%20Security.pdf.

[26] Zakarya Drias, Ahmed Serhrouchni, and Olivier Vogel. "Analysis of cyber security for industrial control systems". In: *2015 International Conference on Cyber Security of Smart Cities, Industrial Control System and Communications (SSIC)*. 2015, pp. 1–8. DOI: 10.1109/SSIC.2015.7245330.

[27] Internet Archive - Unknown. "Stuxnet code". In: Sept. 12, 2015. URL: https://archive.org/details/Stuxnet.

[28] David Kushner. "The real story of stuxnet". In: *IEEE Spectrum* 50.3 (2013), pp. 48–53. DOI: 10.1109/MSPEC.2013.6471059. URL: https://ieeexplore.ieee.org/document/6471059.

[29] Eric Chien Nicolas Falliere Liam O Murchu. “W32.Stuxnet Dossier - Version 1.3 - Symantec Security Response”. In: 2010. URL: https://docs.broadcom.com/doc/security-response-w32-stuxnet-dossier-11-en.

[30] Schneider Electric. “Cybersecurity Support Portal - Security Notifications”. In: URL: https://www.se.com/ww/en/work/support/cybersecurity/security-notifications.jsp (visited on 06/23/2023).

[31] Schneider Electric. “Schneider Electric Security Notification - 10-Jan-23 (14-Mar-23) Document Reference Number – SEVD-2023-010-06 Page 1 of 9 EcoStruxure Control Expert, EcoStruxure Process Expert and Modicon M340, M580 and M580 CPU Safety”. In: URL: https://download.schneider-electric.com/files?p_Doc_Ref=SEVD-2023-010-06&p_enDocType=Security+and+Safety+Notice&p_File_Name=SEVD-2023-010-06_Modicon_Controllers_Security_Notification.pdf (visited on 06/26/2023).

[32] Nir Nissim, Ran Yahalom, and Yuval Elovici. “USB-based attacks”. In: *Computers & Security* 70 (2017), pp. 675–688. ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2017.08.002. URL: https://www.sciencedirect.com/science/article/pii/S0167404817301578.

[33] Mathew Nıcho and Ibrahim Sabry. “Threat and Vulnerability Modelling of Malicious Human Interface Devices”. In: vol. 21. isresoffice@gmail.com: ISRES Publishing, 2022, pp. 241–247. DOI: 10.55549/epstem.1225679.

[34] Schneider Electric. “Unity Pro 13.1 EN - Complete Documentation”. In: Mar. 11, 2019. URL: https://www.se.com/us/en/download/document/UnityPro_EN/.

[35] Matthew Tischer, Zakir Durumeric, Sam Foster, Sunny Duan, Alec Mori, Elie Bursztein, and Michael Bailey. “Users Really Do Plug in USB Drives They Find”. In: *2016 IEEE Symposium on Security and Privacy (SP)*. 2016, pp. 306–319. DOI: 10.1109/SP.2016.26.

[36] Andrew M. Tun. “Cybersecurity-Centric Analysis of Complex Integrated Circuit’s System Development Life Cycle Supply Chain”. AAI29252384. PhD thesis. 2022.

[37] Steven A. Melnyk, Tobias Schoenherr, Cheri Speier-Pero, Chris Peters, Jeff F. Chang, and Derek Friday. “New challenges in supply chain management: cybersecurity across the supply chain”. In: *International Journal of Production Research* 60.1 (2022), pp. 162–183. DOI: 10.1080/00207543.2021.1984606. eprint: https://doi.org/10.1080/00207543.2021.1984606. URL: https://doi.org/10.1080/00207543.2021.1984606.

[38] CICLAB Laboratory of the University of Leon - Luis Martin. “The Unity (UMAS) protocol”. In: URL: https://lirasenlared.blogspot.com/2017/08/the-unity-umas-protocol-part-i.html (visited on 03/12/2023).

[39] Nicholas Miles. “Examining Crypto and Bypassing Authentication in Schneider Electric PLCs (M340/M580)”. In: July 13, 2023. URL: https://medium.com/tenable-techblog/examining-crypto-and-bypassing-authentication-in-schneider-electric-plcs-m340-m580-f37cf9f3ff34.

[40] Gaiëtan Renault. “NinjaCrane: a Cybersecurity Attack Demonstrator on the Polar Crane’s ICS”. In: Aug. 11, 2023. URL: https://github.com/grennault/NinjaCrane.

[41] NIST: Natinoal Vulnerability Database. “CVE-2021-22779 Detail”. In: URL: https://nvd.nist.gov/vuln/detail/CVE-2021-22779.

[42] University of Applied Sciences Campus Vienna: Embedded Lab Vienna for IoT & Security (ELVIS). “Exploiting the USB Ninja BLE Connection”. In: URL: https://wiki.elvis.science/index.php?title=Exploiting_the_USB_Ninja_BLE_Connection.

[43] RETROCUTION. “Easy DIY Tiny USB Hub For Raspberry Pi Projects”. In: Jan. 15, 2020. URL: https://www.retrocution.com/2020/01/15/easy-diy-tiny-usb-hub-for-raspberry-pi-projects/.

[44] Microsoft. “Restrict USB devices and allow specific USB devices using Administrative Templates in Microsoft Intune”. In: URL: https://learn.microsoft.com/en-us/mem/intune/configuration/administrative-templates-restrict-usb (visited on 02/22/2023).

[45] Microsoft - Barak Manor. “Introducing the ability to apply layered Group Policy”. In: URL: https://techcommunity.microsoft.com/t5/windows-it-pro-blog/introducing-the-ability-to-apply-layered-group-policy/ba-p/2608462 (visited on 07/04/2021).

[46] Dave Jing Tian, Adam Bates, and Kevin Butler. “Defending Against Malicious USB Firmware with GoodUSB”. In: *Proceedings of the 31st Annual Computer Security Applications Conference*. ACSAC ’15. Los Angeles, CA, USA: Association for Computing Machinery, 2015, pp. 261–270. ISBN: 9781450336826. DOI: 10.1145/2818000.2818040. URL: https://doi.org/10.1145/2818000.2818040.

[47] Dave Jing Tian, Nolen Scaife, Adam Bates, Kevin Butler, and Patrick Traynor. “Making {USB} Great Again with {USBFILTER}”. In: *25th USENIX Security Symposium (USENIX Security 16)*. 2016, pp. 415–430.

[48] Israel Barbosa de Brito and Rafael T. de Sousa. “Development of an Open-Source Testbed Based on the Modbus Protocol for Cybersecurity Analysis of Nuclear Power Plants”. In: *Applied Sciences* 12.15 (2022). ISSN: 2076-3417. DOI: 10.3390/app12157942. URL: https://www.mdpi.com/2076-3417/12/15/7942.

[49] Maxime Puys, Pierre-Henri Thevenon, and Stéphane Mocanu. “Hardware-In-The-Loop Labs for SCADA Cybersecurity Awareness and Training”. In: *Proceedings of the 16th International Conference on Availability, Reliability and Security*. ARES 21. Vienna, Austria: Association for Computing Machinery, 2021. ISBN: 9781450390514. DOI: 10.1145/3465481.3469185. URL: https://doi.org/10.1145/3465481.3469185.

[50] Pavel Čeleda, Jan Vykopal, Valdemar Švábenský, and Karel Slavíček. "KYPO4INDUSTRY: A Testbed for Teaching Cybersecurity of Industrial Control Systems". In: *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*. SIGCSE '20. Portland, OR, USA: Association for Computing Machinery, 2020, pp. 1026–1032. ISBN: 9781450367936. DOI: 10.1145/3328778.3366908. URL: https://doi.org/10.1145/3328778.3366908.

[51] W Hurst, N Shone, A El Rhalibi, A Happe, B Kotze, and B Duncan. "Advancing the Micro-CI Testbed for IoT Cyber-Security Research and Education". In: *CLOUD COMPUTING 2017 : The Eighth International Conference on Cloud Computing, GRIDs, and Virtualization*. IARIA, Feb. 2017, pp. 129–134. URL: http://researchonline.ljmu.ac.uk/id/eprint/5763/.

[52] Haihui Gao, Yong Peng, Kebin Jia, Zhonghua Dai, and Ting Wang. "The Design of ICS Testbed Based on Emulation, Physical, and Simulation (EPS-ICS Testbed)". In: *2013 Ninth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*. 2013, pp. 420–423. DOI: 10.1109/IIH-MSP.2013.111.

[53] Felix Sauer, Matthias Niedermaier, Susanne Kießling, and Dominik Merli. "LICSTER - A Low-cost ICS Security Testbed for Education and Research". In: *CoRR* abs/1910.00303 (2019). arXiv: 1910.00303. URL: http://arxiv.org/abs/1910.00303.

[54] Irfan Ahmed, Vassil Roussev, William Johnson, Saranyan Senthivel, and Sneha Sudhakaran. "A SCADA System Testbed for Cybersecurity and Forensic Research and Pedagogy". In: *Proceedings of the 2nd Annual Industrial Control System Security Workshop*. ICSS '16. Los Angeles, CA, USA: Association for Computing Machinery, 2016, pp. 1–9. ISBN: 9781450347884. DOI: 10.1145/3018981.3018984. URL: https://doi.org/10.1145/3018981.3018984.

[55] basil. "WinDivert". In: 2020. URL: https://reqrypt.org/windivert.html.

[56] CarlosPolop. "winPEAS". In: URL: https://github.com/carlospolop/PEASS-ng/tree/master/winPEAS (visited on 07/04/2023).

[57] Federico Griscioli and Maurizio Pizzonia. "Securing promiscuous use of untrusted USB thumb drives in Industrial Control Systems". In: *2016 14th Annual Conference on Privacy, Security and Trust (PST)*. 2016, pp. 477–484. DOI: 10.1109/PST.2016.7907002.

[58] U.S. Nuclear Regulatory Commission. *Pressurized Water Reactors*. URL: https://www.nrc.gov/reactors/power/pwrs.html (visited on 06/16/2023).

[59] ONR Research - Office of Nuclear Regulatory Research. "TMI-2 Knowledge Management Library - (P11) Reactor Building, Cross-Sectional View Looking South (GEND-INF-069)". In: URL: https://tmi2kml.inl.gov/Documents/0a-Photos/(P11)%20Reactor%20Building,%20Cross-Sectional%20View%20Looking%20South%20(GEND-INF-069).jpg (visited on 06/23/2023).

[60] JEGO Systems. *System Integration*. URL: https://www.jegosystems.com/system-integration (visited on 06/02/2023).

Appendices

Appendix A

Graphical Realism Framework for Industrial Control Simulation

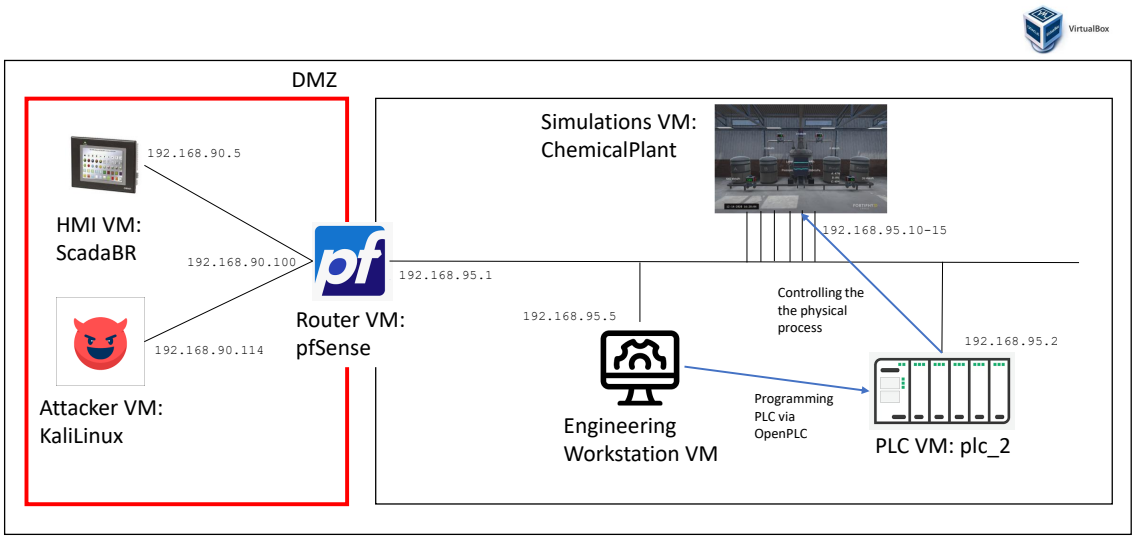


Figure A.1: Network architecture of the GRFICSv2. The GRFICSv2 makes use of six VMs (e.g, attacker, HMI, firewall, Engineering Workstation, PLC, and 3D process simulation) to model a cyberattack leading to an explosion of a chemical process.

Appendix B

Pressurized Water Reactor (PWR)

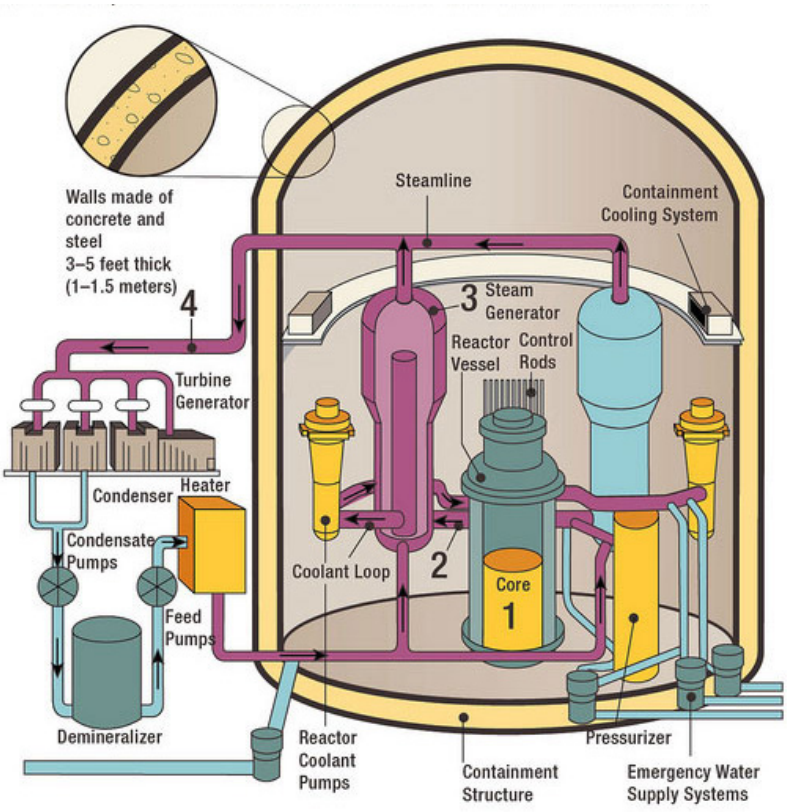


Figure B.1: Pressurized Water Reactor [58]. 1. The core, situated in the reactor vessel, produces heat from nuclear reaction. 2. Pressurized water in the primary circuit carries the heat to the steam generator. 3. Inside the steam generator, heat from the primary circuit vaporizes the water in a secondary circuit, producing steam. The steam then rotates the rotor of the turbine generator, producing electromagnetic field variation which induces alternating current in the stator.

Appendix C

Reactor Containment Building (CB)

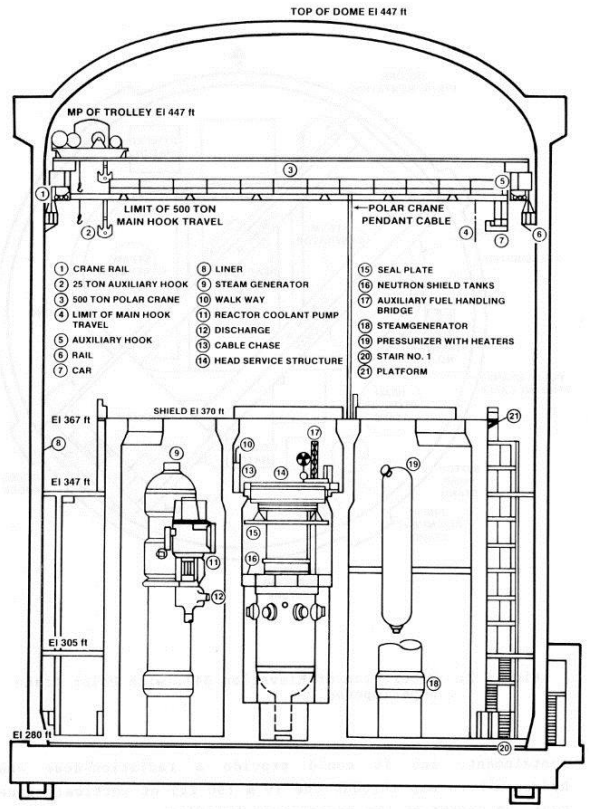


Figure 1. Cross Section of TMI-2 Reactor Containment Building (Looking North to South).

5

Figure C.1: Reactor Containment Building (CB) [59]. The Three Mile Island NPP CB is a 140 meters high and 80 cm thickness building that resists to internal accident situations (e.g., break of the primary circuit) and external aggressions (e.g., human origin or natural hazard).

Appendix D

Purdue Model

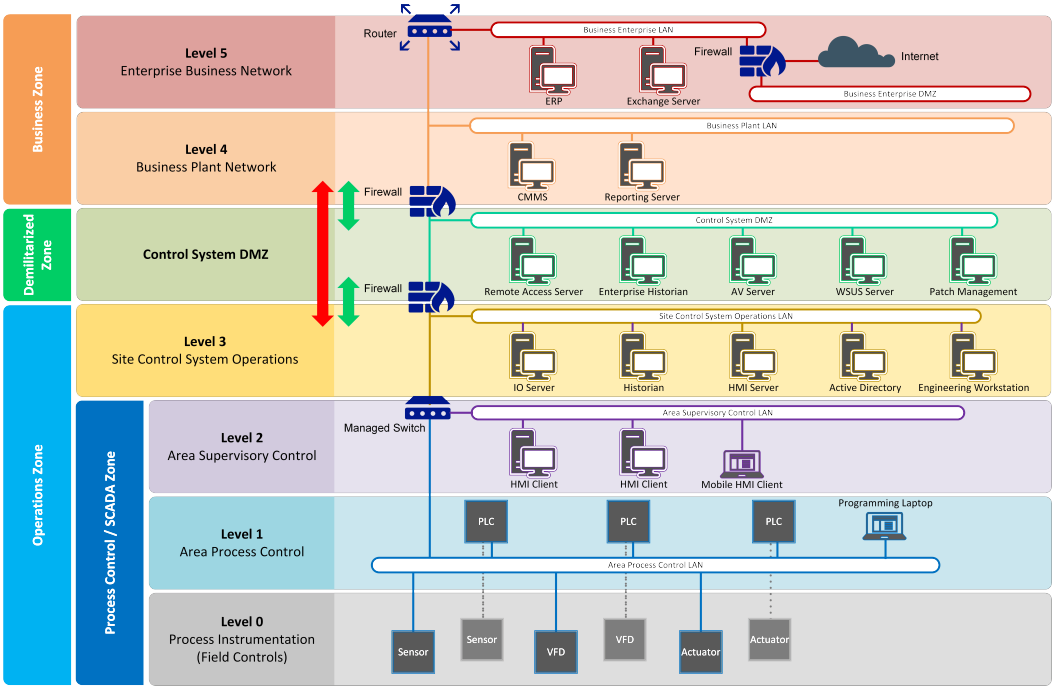
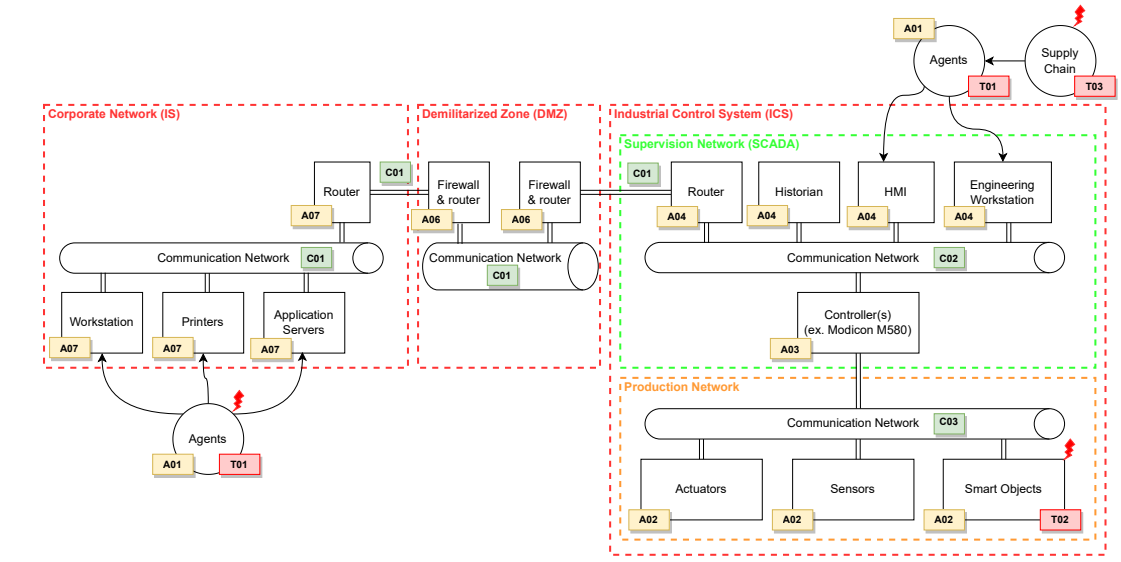


Figure D.1: The Purdue Model: a Simplified Architecture of an ICS [60]. The different levels an ICS architecture from Top to Bottom. 1. Level 5, in red, is the Corporate Network. This delocalized area is for example made of servers, or employees computer. 2. Level 4, in orange, is the Plant Network. It is mainly used for process reporting. 3. Level 3, in yellow, is the site control operations. It is separated with Level 4 by a DMZ that globally acts as a one-way firewall. The Level 3 is an area made for example of historians and process maintenance servers. The Level 2, in purple, is the SCADA. It is made of HMIs and allows technicians to visualize and supervise the process. The Level 1, in cyan, is the area process control it is mainly made of DCS (e.g., PLC, RTU or IIoT devices). The Level 0 is the process instrumentation made of actuators and sensors. As a side note, a polar crane's ICS is generally smaller and is an air-gap not connected to the Level 3 to 5.

Appendix E

Threat Model



Assets

ID	Description
A01	Agents
A02	Level 0 - Process devices
A03	Level 1 - Direct control devices
A04	Level 2 - Plant supervisory devices
A05	Level 3 - Site operation devices
A06	DMZ
A08	Level 5 - Entreprise network devices

Security Controls

ID	Description
C01	SSL/TLS (confidentiality, integrity and authentication)
	FTPS,IPPS, ...
C02	Modbus over TCP with authentication & integrity
C03	Radio Frequence (RF) - no security ? (P25 maybe ?)
	Bluetooth Low Energy (BLE) - security (no security to BLE 5.0) No authentication
	Modbus over TCP with authentication & integrity

Threat Actors

ID	Description
T01	Compromised Agents
T02	Compromised Smart Objects
T03	Compromised Supply Chain

Figure E.1: Data flow diagram. From an ICS architecture, this data flow diagram represents the three entry points identified – the corporate network, the supervision network, or the production network. The *NinjaCrane* attack infects first the supervision network via a malicious HID USB to then tamper the production network.

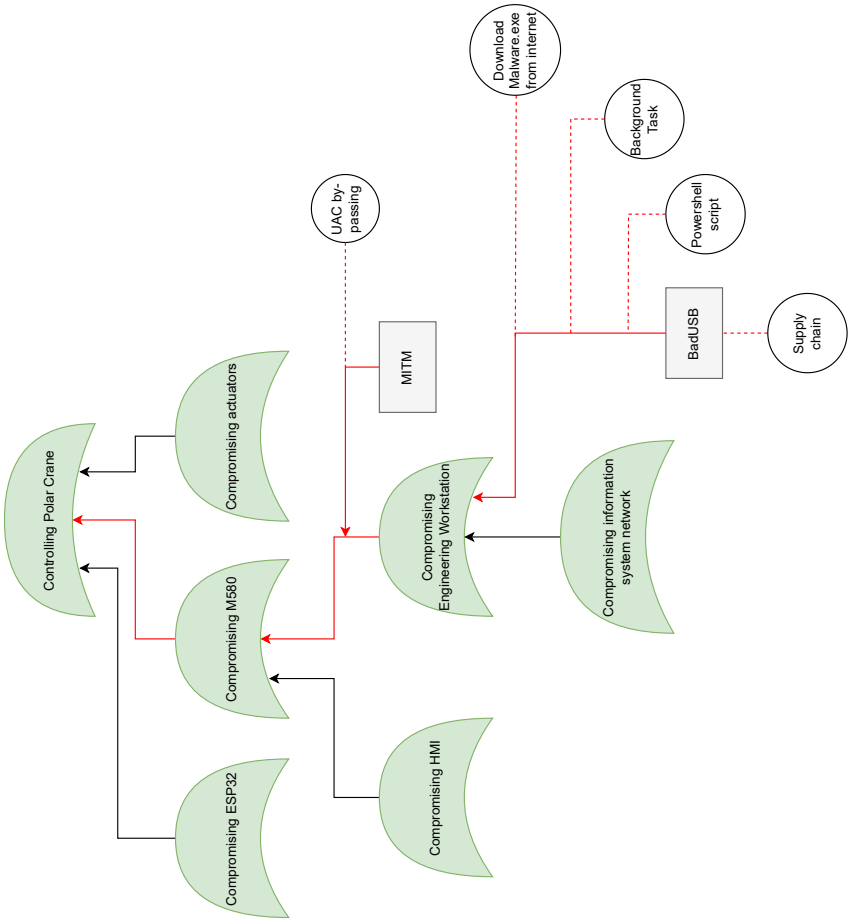


Figure E.2: Attack tree. The attack tree models the path taken to take control over the polar crane. The first step is the supply chain compromising that brings a HID-capable device to be connected to the engineering workstation. When connected, the USB device runs a powershell script to download and execute in background a malware. This malware by-pass the windows UAC and performs a MITM attack to modify PLC's internal variables and take control over the polar crane.

Appendix F

Polar Crane's Testbed

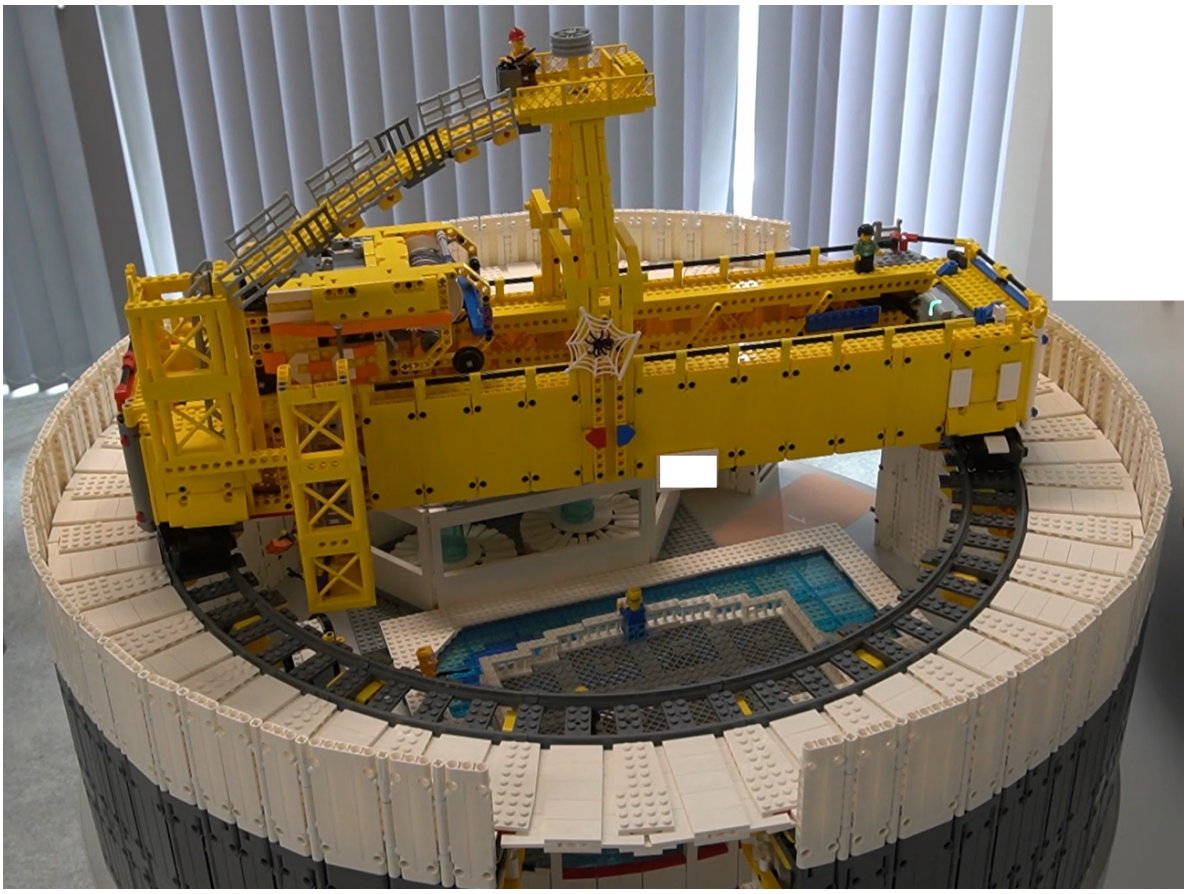


Figure F.1: Polar Crane's Sketch. Sketch in LEGO™of the polar crane which models three different movements; crane rotation, trolley rectilinear displacement, and charge lift. Those movements are controlled via an HMI connected to the *Modicon M580* PLC. The PLC interprets and transmits the commands to three LEGO™BLE Hubs via an ESP32 microcontroller establishing the Bluetooth connection.

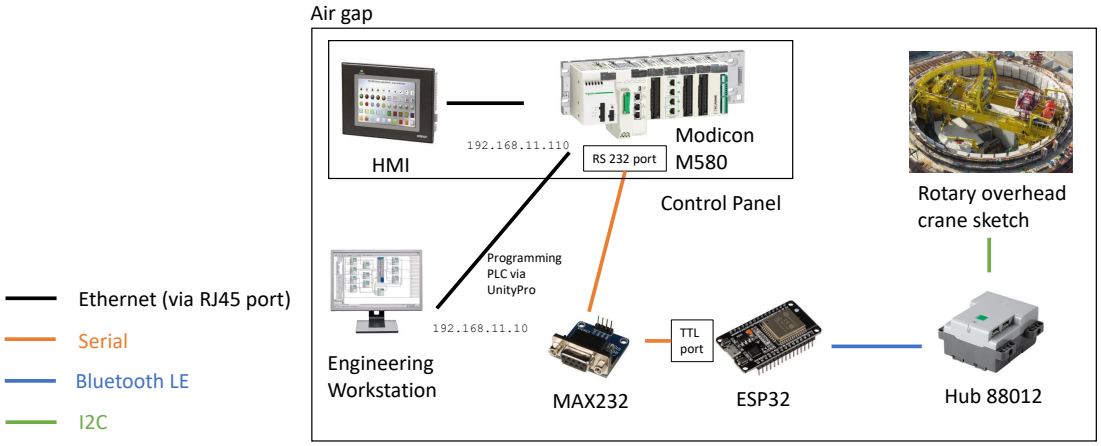


Figure F.2: Network Architecture of the Testbed. The Engineering Workstation programs the M580 PLC via Unity Pro. The M580 stores and executes the polar crane's program. The HMI displays the state (i.e., position, speed etc.) of the polar crane and allows a technician operator to control the polar crane. Due to its rotation, the polar crane must be controlled over the air. In this testbed the choice has been made to control the polar crane with BLE. The ESP32 is sending over BLE the commands it received from the PLC.

Appendix G

Malicious Mouse

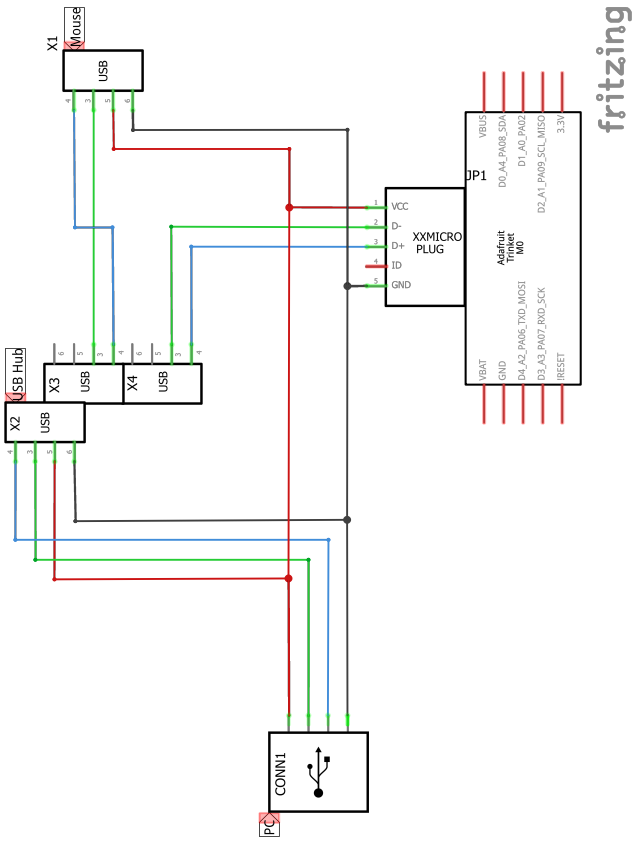


Figure G.1: Electric wired schematic of the malicious mouse. The mouse connection originates from the `CONN1` (e.g., PC USB connection) and proceeds to an USB hub that splits the connection into two pathways – one leading to the mouse itself and the other one to an HID-capable microcontroller.

Figure G.2: Picture of the maliciously modified mouse. A Logitech mouse opened and modified in order to include an HID-capable microcontroller. On the photograph can be seen the Adafruit Trinket M0 connected in micro-USB to the hub.

Appendix H

Modbus/UMAS Communication

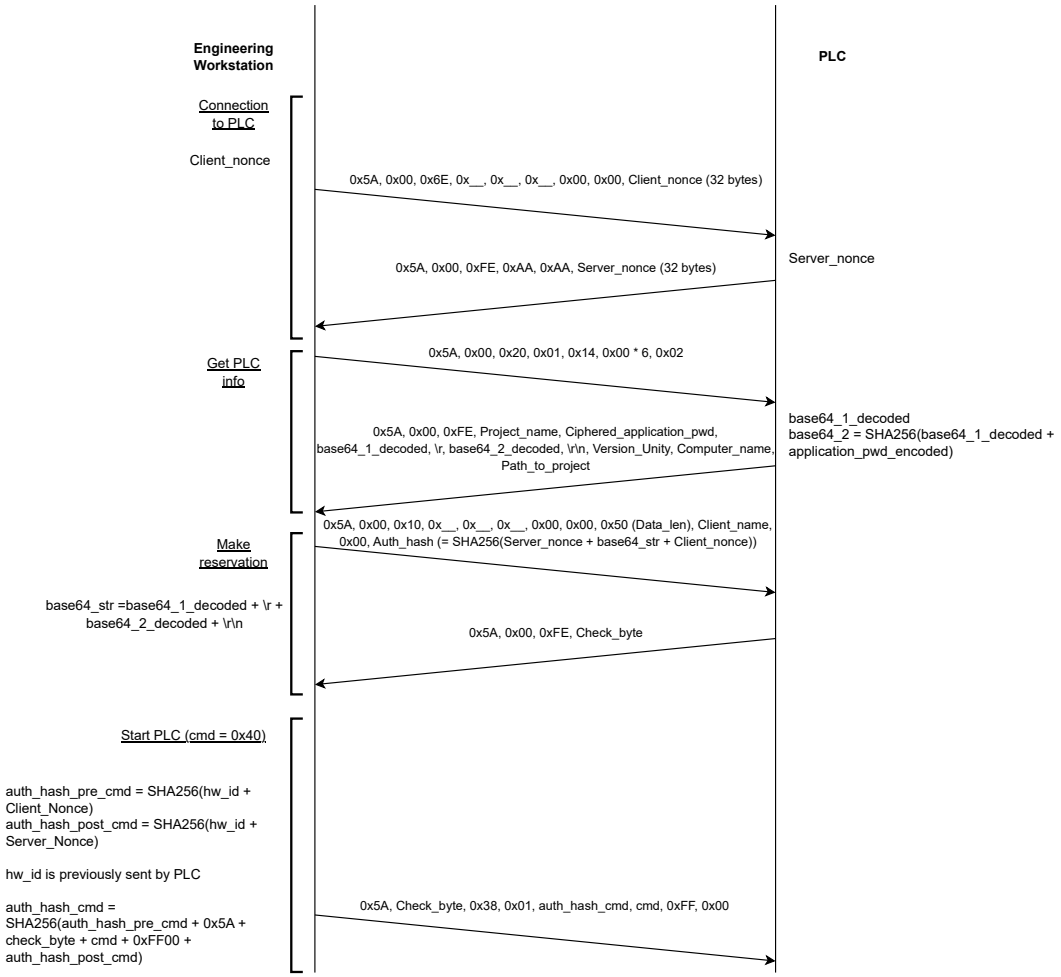


Figure H.1: Communication between the M580 and the engineering workstation

Appendix I

Adversary's Graphical User Interface (GUI)

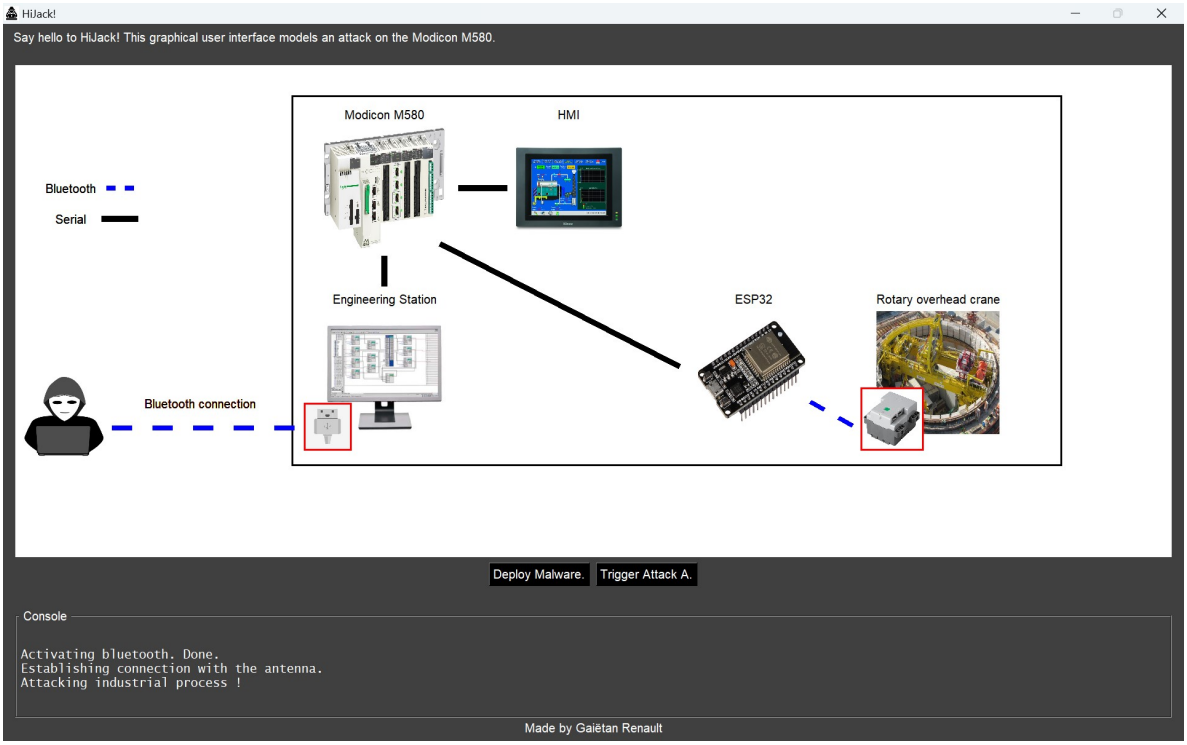


Figure I.1: Adversary's Graphical User Interface (GUI). This interface, running on a laptop, is made with *PySimpleGUI* and allows to take control of the USB Ninja cable over BLE by triggering two different payloads. The first one deploys the malware on the PC, the second one starts the *NinjaCrane* attack.

Appendix J

Data Exfiltration

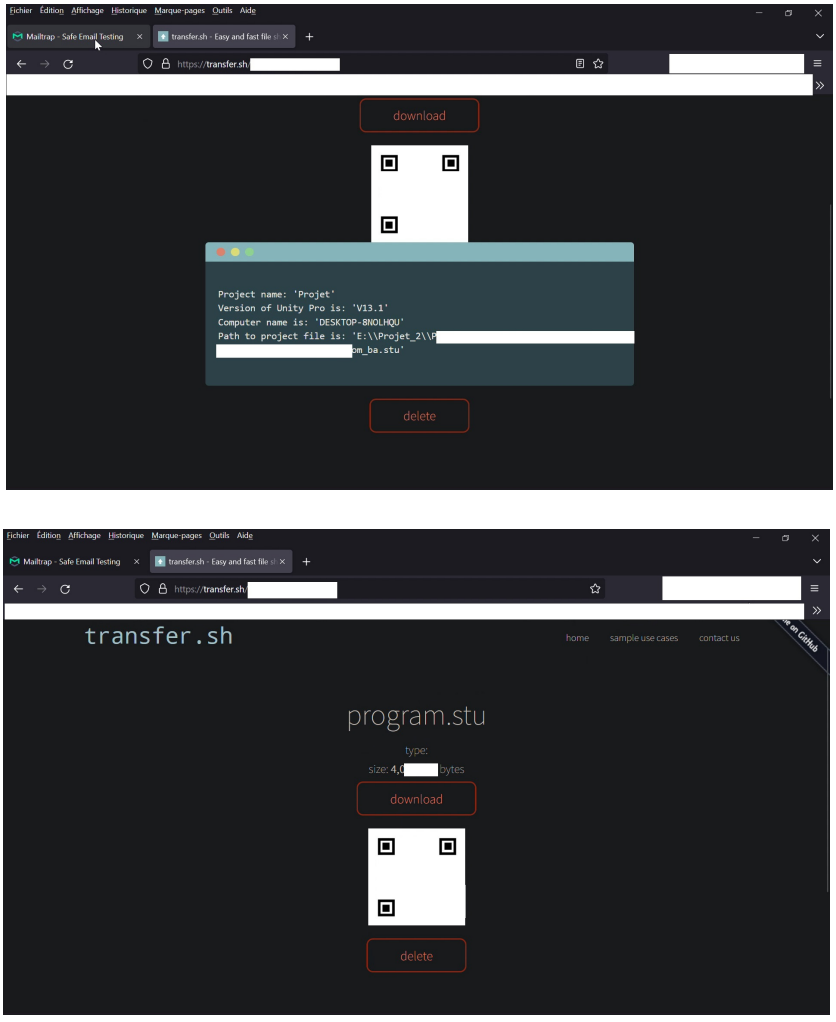


Figure J.1: Data exfiltration over internet. Top: Project information. Bottom: .STU program.

Appendix K

HID-device's Flowchart Payload

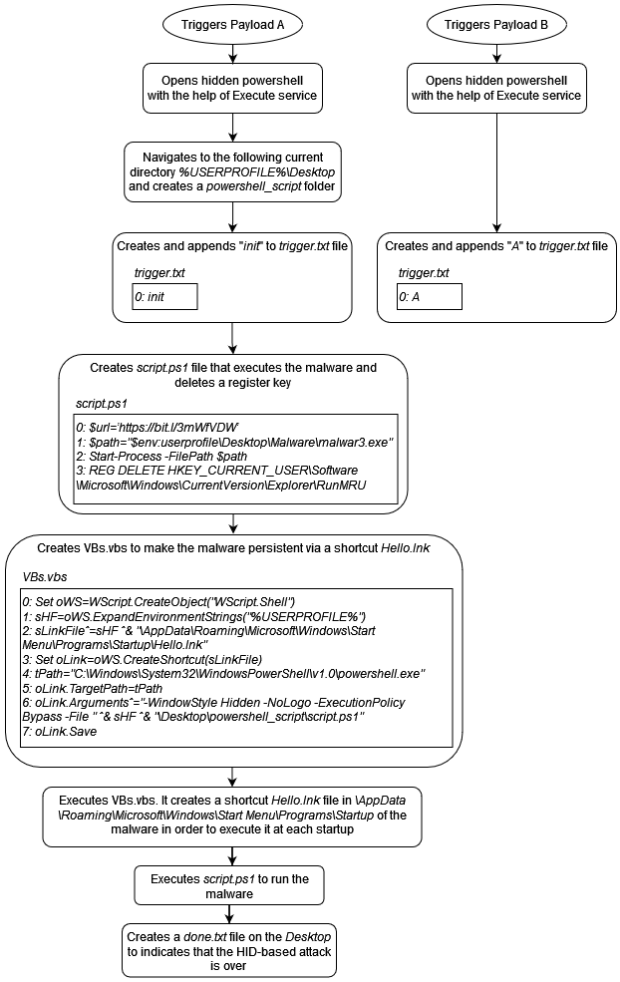


Figure K.1: HID-device's flowchart payload.

Appendix L

Live Demonstration Feedback from the Participants

- « The cyberattack of the polar crane was extremely well realised because it was very assiduous and very practical: we directly observe how a human-origin mistake (USB cable plug-in) could have consequences on a NPP site. »
- « Cybersecurity workshop was interesting and playful. »
- « Cybersecurity workshop was very interesting. Congrats to the job done for preparing it, it was immersive and great. »
- « Cybersecurity, very good in form and content, it's an eye-opener on the importance of ensuring safety. »
- « Cybersecurity: Discovering new dangers for nuclear facilities. Complexity of protecting against this type of risk, given the development of new technologies (AI in particular). »
- « Cyber-crime: Excellent! A return to childhood via the lego sketch, which can be moved and even controlled by the PLC. The workshop on cyber-security really made us realize that evil can lurk anywhere... very instructive! »
- « Polar crane hacking: very good, very entertaining, with visual elements. The trainee presented extremely well (very good popularization for non-experts), [...]. I'd perhaps recommend a longer workshop (to allow more questions) and with fewer people, to allow people to move around the polar crane demo and the engineering workstation. »