University of Padova

Department of Mathematics "Tullio Levi-Civita"

Ph.D. Schoool: Brain, Mind, and Computer Science

Cybersecurity of Modern Cyber-Physical Systems

Supervisor
Prof. Mauro Conti
University of Padova

Co-supervisor
Prof. Luciano Gamberini
University of Padova

Ph.D. Candidate
Federico Turrin

Academic Year
2022-2023

Dedicated to all those who have accompanied and supported me in this intense, but unforgettable, journey.

Abstract

Cyber-Physical Systems (CPSs) refer to those systems characterized by the interconnection of information technology and the physical process domains. These systems are nowadays employed in a wide range of applications, such as health monitoring, industrial control systems, and transportation. The recent digitalization and smartification of the processes required to integrate the Internet connection into CPSs, enabling functions like remote connection and cloud computing, but at the same opening new dangerous vulnerabilities surfaces. Indeed, recent events in history have shown many cyber-attacks and vulnerabilities discovered on CPSs. For this reason, there is still a need to contribute to securing such systems, both from the design and implementation points of view. In this thesis, we analyze the cybersecurity of modern CPSs, identifying and highlighting the current vulnerabilities, the research gaps in terms of security, and the threats affecting them. Then, we propose novel security mechanisms to prevent potential cyber-attacks. This thesis is composed of three parts as follows.

In the first part of the thesis, we will focus on the security of Industrial Control Systems (ICSs). These systems are used to control and monitor critical infrastructures and industrial processes. As a first step, we gather all the knowledge in this field from the literature, and we provide a systematic analysis of the testing platform and the detection systems solutions operating on them. To motivate the necessity of improving the security of current industrial systems, we performed a measurement study highlighting the dramatic exposure of the communication protocols and services of more than 50 industrial endpoints. Then, we developed and deployed an innovative ICS honeypot. While measuring the honeypot exposure, we noted that industrial systems are still highly targeted and interacted with by malicious actors over the internet on specific vulnerable industrial services.

In the second part of the thesis, we will look at the security of vehicular systems. Like ICSs, modern vehicles present vulnerabilities due to the adoption of legacy components, enabling the possibility of malicious exploits. To this end, we will focus on the internal communication bus of cars, we examine its vulnerabilities, the current solutions in the literature, and their limitations, and propose an innovative cryptographic key distribution system among the network nodes. We will then focus on the emerging electric vehicle paradigm. We identified two possible cyber-attacks on this ecosystem. The first is based on a relay attack vulnerability, which implies charging illegitimate vehicle recharging fees. Instead, the second one consists of a privacy leakage from the current absorbed during the vehicle's recharging process.

In the third part of the thesis, we leverage the knowledge of our studies to investigate the security of CPS cross-domain applications. In particular, we first present a survey on Power Side-Channel (PSC) exploits in the literature, focusing on existing attacks and countermeasures. Indeed, PSCs have been proven effective in reversing and profiling the functioning of

many embedded devices (e.g., smart cards, vehicles, and laptops). Then, we develop a novel framework to fingerprint Universal Serial Bus devices from their power consumption. This funding can be used, for instance, to securely authenticate a personal device and avoid malware delivery injection in critical applications (e.g., Stuxnet). Finally, we present the first security analysis of the emerging Hyperloop transportation technology. Hyperloop merges the concepts of ICS since it consists of a critical, distributed, and sensing infrastructure, and the concept of vehicle, due to the pod communication management. As a result, Hyperloop inherits all the vulnerabilities and risks of the two systems.

Contents

1 Introduction

Cyber-Physical Systems (CPSs) are characterized by the deep intertwining between the digital and physical worlds. These systems generally implement a control loop that perceives information from the physical processes through sensors and compute the control actions necessary to adjust the underlying process through actuators [1]. CPS systems are nowadays pervading our society with many applications [2]. Some examples of CPS are Industrial Control Systems (ICSs), vehicles, and medical devices. These applications are often considered critical since damaging them may cause terrible consequences to the surrounding people and environment and the business of the organizations involved. For this reason, monitoring and maintaining the security of CPSs is of central importance for both companies and government institutions.

Most modern CPSs are composed of long-term and legacy devices designed in the late 1900s. At that time, these systems were supposed to operate in an air-gapped environment without Internet access and remote connection capabilities. However, the increasing digitalization of the devices required the adaptation of CPSs to smart processes and, therefore, the integration of the connection of cyber-physical devices with the Internet. This increased their vulnerability surfaces and the possibility of cyber-attacks. In the remainder of the thesis, we always refer to an *attack* as a *cyber-attack*, unless noted otherwise. Generally, the software and hardware integrated into CPS components are basic since it only aims to run programs to complete a control process. For this reason and also due to the real-time availability constraints, and the production costs, CPS rarely implement security measures in their software, communications, and hardware. The security of CPSs is therefore at risk due to the complexity necessary to keep

the components updated with modern security measures. In fact, the update of the devices is considered critical, both in terms of data availability and compatibility with the entire system.

Despite the continuous effort of the research community, government, and companies, the trend of attacks in CPS is continuously growing. Many historical events have highlighted the exposure of CPSs to attacks. One of the most famous attacks in the industrial sector is the Stuxnet [3] malware. Stuxnet was developed to specifically target the turbines of an Iranian nuclear plant, causing serious damage to its functioning. However, this malware has been succeeded by many others over the years, such as TRITON [4] and BlackEnergy [5]. More recently, other attacks have been documented, such as the colonial pipelines ransomware [6] in 2021 and the Florida water system malfuction [7] in 2022. Both of them halted normal infrastructure operations and caused serious damage to the companies involved in terms of business and reputation. However, other applications of CPSs have also been shown to be dangerously vulnerable to attacks. For instance, in 2015, Miller et al. [8] demonstrated how it was possible to take remote control of a machine, commanding its functions.

For these reasons, the research in the security of CPSs still needs contributions from a different perspective.

Contribution

This thesis explores and proposes cutting-edge directions for the security of CPSs. Despite many research in recent years highlighted vulnerabilities of these systems and proposed solutions, malicious actors continue to exploit such systems, putting the security of users, businesses, and the environment at risk (e.g., Stuxnet [3], car remote exploit [8]). Therefore, there is necessary to analyze these systems' security and propose innovative mechanisms to prevent the compromission of these critical assets and provide secure design guidelines for the current and the next generation of CPSs.

Current researches focus on a single CPS application domain, e.g., vehicle network only, without considering common element with other CPSs which has been highlighted as vulnerable (e.g., side-channel leakages and protocols vulnerabilities). Furthermore, most proposed solutions do not meet companies' business constraints, are not transferable to similar applications, or are not backward compatible, limiting the overall solution's applicability [9].

To solve these issues and enforce the security of CPSs, this project aims to study different CPSs applications to find common design patterns and propose common innovation solutions by considering the companies' requirements and real-world design constraints. In particular,

this research project investigates the following Research Questions (RQs):

- *RQ1*: What is the state-of-the-art security of the modern CPSs and what are the most common threats affecting them?

- *RQ2*: What are the limitations of the common security solutions applied on CPSs, and how can we overcome these limits and improve their protection effectiveness?

- *RQ3*: Is it possible to transfer the knowledge of specific domains of CPSs to others CPSs applications and identify similar security patterns and threats?

To answer the previously stated RQs, the project includes the following phases:

1. **Background acquisition**. Analysis and study of the current implementation of modern CPSs security.

2. **Vulnerability Identification**. Based on the background analysis, we identify potential threats and attacks that may pose a dangerous threat impacting the user, the company, and environmental safety. The analysis focuses on the following aspects:

    - *Privacy exposure*. This analysis investigates if the current system implementation may expose sensitive information of the end user.
    - *Infrastructure vulnerabilities*. This analysis investigates vulnerabilities in the current CPS implementations (e.g., communication protocols and hardware vulnerabilities).

3. **Solution Proposal**. Once identified the security concerns in the previous analysis, we propose countermeasures and guidelines to support the development of future secure CPSs.

4. **Cross-domain security analysis**. The heterogeneous and distributed nature of the CPS imposes security challenges from different angles and cross-domain applications. In this phase, the knowledge acquired from the previous analysis will be leveraged to study cutting-edge directions to protect cross-domain applications of CPSs.

Thesis Organization and Papers References

The thesis is organized as follows. Chapter 2 focuses on the security of modern ICSs, surveying the related literature and presenting two measurements we performed on their exposure. These findings are based on the results of the following papers:

- M. Conti, D. Donadel, and F. Turrin, "A survey on industrial control system testbed and datasets for security research," *IEEE Communications Surveys & Tutorials (COMST)*, pp. 2248–2294, 2021.
- G. Barbieri, M. Conti, N. O. Tippenhauer, and F. Turrin, "Assessing the use of insecure ICS protocols via IXP network traffic analysis," in *2021 International Conference on Computer Communications and Networks (ICCCN)*, 2021, pp. 1–9.
- M. Conti, F. Trolese, and F. Turrin, "Icspot: A high-interaction honeypot for industrial control systems," in *2022 International Symposium on Networks, Computers and Communications (ISNCC)*. IEEE, 2022, pp. 1–4.

Then, Chapter 3 analyzes the security of vehicular systems, focusing on their internal communication network and the Electric Vehicle (EV) environment. These findings are based on the results of the following papers:

- S. Soderi, R. Colelli, F. Turrin, F. Pascucci, and M. Conti, "Senecan: Secure key distribution over can through watermarking and jamming," *IEEE Transactions on Dependable and Secure Computing (TDSC)*, 2022.
- M. Conti, D. Donadel, R. Poovendran, and F. Turrin, "Evexchange: A relay attack on electric vehicle charging system," in *European Symposium on Research in Computer Security (ESORICS)*. Springer, 2022.
- A. Brighente, M. Conti, D. Donadel, and F. Turrin, "Evscout2.0: Electric vehicle profiling through charging profile," *ACM Transactions Cyber-Physical Systems (TCPS)*, 2022.

Chapter 4 presents two case studies of cross-domain CPS security where we applied the knowledge acquired from the studies in the previous chapters: Universal Serial Bus (USB) Power Side-Channel (PSC) analysis and Hyperloop. Finally, Chapter 5 concludes the thesis by summarizing the findings achieved and identifying future work directions. These findings are based on the results of the following papers:

- H. Liu, R. Spolaor, F. Turrin, R. Bonafede, and M. Conti, "USB powered devices: A survey of side-channel threats and countermeasures," *High-Confidence Computing*, p. 100007, 2021.

- R. Spolaor, H. Liu, F. Turrin, M. Conti, and X. Cheng, "Plug and power: Fingerprinting USB powered peripherals via power side-channel," in Proceedings of the IEEE International Conference on Computer Communications (INFOCOM). IEEE, 2023.

- A. Brighente, M. Conti, D. Donadel, and F. Turrin, "Hyperloop: A cybersecurity perspective," 2022. *Currently under review*.

Other papers published during the Ph.D.

Aside from the paper previously mentioned, I also produced the following papers during my Ph.D.

- G. Bernieri, M. Conti, M. Sovilla, and F. Turrin, "ALISI: a lightweight identification system based on Iroha," in *Proceedings of the 35th Annual ACM Symposium on Applied Computing (SAC '20)*, 271–273, 2020.

- M. Conti, and F. Turrin, "Cyber Forensics for CPS", *Encyclopedia of Cryptography, Security and Privacy*, pp. 1-3, 2019.

- S. Soderi, A. Brighente, F. Turrin, and M. Conti, "VLC Physical Layer Security through RIS-aided Jamming Receiver for 6G Wireless Networks," in *Proceedings of the 2022 19th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, 2022.

- F. Turrin, A. Erba, N. O Tippenhauer, and M. Conti, "A statistical analysis framework for ICS process datasets", in *Proceedings of the 2020 Joint Workshop on CPS&IoT Security and Privacy (CPSIoTSec)*, pp. 25-30, 2020.

- L. Attanasio, M. Conti, D. Donadel, and F. Turrin, "MiniV2G: An Electric Vehicle Charging Emulator,", in *Proceedings of the 7th ACM on Cyber-Physical System Security Workshop (CPSS)*, pp. 65-73, 2021.

2

Industrial Control System Security

In the first part of the thesis, we will focus on one of the most studied CPS applications, the ICS security. These systems gained special attention from the security community due to the increasing attack spectrum targeting and cyber incidents targeting them, from Stuxnet [3] to the most recent colonial pipeline attack [6] and Florida water system [7] attacks. These events highlighted the insecurity of such infrastructures, posing severe problems to the population, the surrounding environment, and the company business.

This chapter gathers the results obtained from different papers. Section 2.1 is dedicated to introducing the ICS environment based on the finding of our survey [1]. In particular, will be introduced the ICS infrastructure, the different devices composing the network and the most used protocols, the worldwide testing platform for security research (i.e., datasets and testbeds), and the Intrusion Detection System (IDS) applied on them. Then, in Section 2.2 to motivate the need for security research in this field, we present our measurement study on ICS infrastructure protocol usage [10], showing the dramatic usage of insecure protocols over the internet (i.e., no encryption and authentication mechanisms). Lastly, in Section 2.3 we present a novel Honeypot conceived for ICS that we designed and deployed [11], and we show the data collected from the interaction with external sources.

2.1 An ICS Overview

Critical infrastructures and the emerging Industry 4.0 are increasingly using more advanced technologies such as computers, and electrical and mechanical devices to monitor physical processes. The networks resulting from smart computing integration for the processes monitoring are called ICSs or, sometimes, Supervisory Control And Data Acquisition (SCADA) systems.

ICSs are composed of two macro areas. The Operational Technology (OT) network includes hardware and software used to monitor and manage industrial equipment, assets, processes, and events, e.g., Programmable Logic Controllers (PLCs), sensors, and actuators. Instead, the traditional Information Technology (IT) network includes workstations, databases, and other classical machines used to manipulate information. IT and OT networks were originally disconnected. However, due to the so-called IT/OT Convergence [12], the two networks have been interconnected to facilitate the digitization of processes, opening new vulnerability surfaces.

For CPSs, which also contains ICSs, the classical CIA triad (Confidentiality, Integrity, Availability) is considered reversed, in order of importance, as Availability, Integrity, and Confidentiality [13, 14]. In this context, reliability becomes the most critical request since, differently from IT systems where the main concerns are about the confidentiality of the data, for an ICS instead, the availability is fundamental since it can guarantee human safety and fault tolerance [15]. For instance, in a nuclear plant environment, data availability (e.g., the temperature of the core) is more important than its confidentiality [16].

Since these systems control physical and sometimes dangerous processes, security is a fundamental need. However, in recent years several viruses targeting ICSs were identified. One of the first attacks targeting SCADAs systems dates back to 1982 [17] when a trojan targeting the Trans-Siberian pipeline causes a massive explosion. In successive years, many incidents exposed the security weaknesses of ICSs. Stuxnet [3, 18] is probably the most famous malware discovered in this field. Stuxnet was a worm discovered in 2010 targeting PLCs used in gas pipelines and power plants. It was able to cause the self-destruction of 984 centrifuges in a uranium-enrichment plant in Iran. In 2014, the third version of a known trojan family, BlackEnergy [5], was developed to target ICSs. In the following years, this trojan was spread mainly inside a Microsoft Word document that, once open, request to activate macros that hide the virus. Victims of these attacks are media and energy companies, mining industries, railways, and airports in Ukraine. On December 23, 2015, an attack employing BlackEnergy3 caused a three-hour disconnection of 30 substations in the Kyiv Power Distribution company, leading

to several hours of blackouts in the area. More recently, in 2017, another important malware, TRITON [4], was identified after an unscheduled shutdown of a Saudi Arabian petrochemical processing plant. TRITON reprograms some special PLCs used for safety purposes, causing them to enter a failed state.

According to a report of Kaspersky Lab [19], in the second half of 2016 the 39.2% of the industrial machines secured by Kaspersky's products have been attacked, a clear sign that threats to ICS are a growing problem nowadays. The vulnerabilities affecting these systems are also reported in recent studies on real ICS traffic over the Internet [10, 20], showing a dramatic lack of security features on the communication.

A successful attack on ICS implies a huge economic impact on the organization. These consequences include operational shutdowns, damage to the equipment, business waste, intellectual property fraud, and significant health and safety risks. Nozomi Network reports that known shutdown events of an ICS [21] due to an attack cost from 225K$ up to 600M$. An increasing attack trend against ICS is Ransomware, which aims to obtain economic rescue [22]. According to Coveware [23], in Q4 of 2019, the average ransom payment increased by 104% to 84,116$, up from 41,198$ in Q3 of 2019. One of the most recent Ransomware is EKANS, which was discovered targeting 64 ICS [24].

To prevent such catastrophic events, it is fundamental to implement novel security-by-design approaches, and where it is impossible to apply them, prevention or mitigation techniques must be integrated. However, developing a new security-by-design concept, it is required a complete testing infrastructure. Generally, researchers rely on scaled-down versions of a real ICS, created ad-hoc to reproduce real-world systems but in a controlled environment, called *testbed*. Testbeds could be based on physical devices to provide reliable data at the cost of being more expensive or virtual if the application does not require exact measures. However, the development of a new testbed is not straightforward, instead, it is challenging from different points of view, ranging from implementation costs, sharing capability, and fidelity (Section 2.1.4).

To develop prevention and mitigation techniques, nowadays, researchers involve machine learning techniques that exploit big amounts of data to train classification algorithms to detect misbehavior or potential attacks. The straightforward approach to collecting data is to record and provide to researchers with data from real ICSs. However, since these systems are generally critical and fundamental for society, this strategy can be challenging in many aspects. For instance, it is difficult, if not impossible, to deploy attacks in a real environment because they can damage the physical process or some devices. Moreover, privacy is a problem: pri-

vate companies could be reluctant to share system data from their ICSs. In fact, disclosing this data can cause intellectual property theft or reveal the infrastructure's vulnerabilities, attracting malicious attackers' attention. From a testbed, it is possible to generate data and share them with other researchers to compare and improve different detection algorithms' results. These captures are called datasets and can be composed of physical measures (i.e., data from OT sensors) and/or network traffic (i.e., data from network communications). Datasets are an excellent testing solution due to their simplicity and availability. However, they are also challenging from many points of view, for instance, in the generation process, and lack modularity (Section 2.1.5).

In this section, we present a comprehensive survey specifically targeting the security research platform in the ICS field. This work aims to collect all the information related to the testbed and dataset to support future research and studies on this sector. Furthermore, for each dataset identified, we report the best score achieved by an IDS in terms of F1-Score (F1), Accuracy (Acc), and Precision (Pr), which are the most common metrics. We have accurately analyzed all the testbeds, datasets, and IDSs to provide the readers with an exhaustive overview of the current ICS state of the art. The study aims to assist interested readers: (i) to discover the different testbeds and datasets which can be used for security research in ICS with a description of the design key points, (ii) to have a clear baseline when developing an IDS on a particular dataset, and (iii) to understand the challenges and the good practices to keep in mind when designing an ICS testbed or dataset.

2.1.1 Related ICS Surveys

Literature includes different surveys comparing testbeds and datasets created for applications in the ICS field. However, to the best of our knowledge, no detailed analysis gathers and describes both the ICS datasets and testbeds, but also the main IDSs implemented on them. For every dataset, we also report the algorithm with the best performances and the most interesting and innovative detection approaches. We believe that this could be useful for future research in this field and set a baseline to compare the different detection results.

In 2015, Holm et al. [25] proposed a complete revision of several papers related to ICS testbeds. The authors then focused on the objective and the component's implementation of 30 different testbeds. Furthermore, the authors provided an analysis of each testbed's main requirements (i.e., fidelity, repeatability, measurement accuracy, and safe executions). The paper's main scope was to provide an overview of the actual state of such systems' development

without detailing each single testbed composition. In fact, except for a table that indicates each testbed's location, all the others show only aggregated information. Moreover, since the paper is not recent, some of the presented testbeds are quite old and not widely used nowadays, while others that were only designed have never been made (e.g., [26]).

McLaughlin et al. [27], in 2016, presented a complete survey on the state-of-the-art in ICS security. The paper briefly introduces the ICS operation's key principles and the history of attacks targeting ICSs. The authors then addressed the vulnerability assessment process, outlining the cybersecurity assessment strategy advised for ICS and providing a list of steps to study the security and vulnerabilities of an industrial system. In the end, the authors focused on new attacks and mitigation techniques. Moreover, the paper briefly presents a small list of some testbeds which can be used for security research in this field. Differently, our work is less focused on providing a complete landscape on ICS security, while it offers a more in-depth analysis and review of all the most used testbeds and datasets for ICS security research.

In 2017, Cintuglu et al. [28] presented a comprehensive survey focused on smart grid testbeds, providing a systematic study with a particular focus on their domains, research goals, test platforms, and communications infrastructures. There are some intersections between smart grid and ICS fields, such as some used components and protocols. Nevertheless, some different concepts require a separate ICS analysis, like the specific applications and sensors used, combined with the complexity of smart grids. To classify smart grid testbeds, the authors provide different possible taxonomy. Some of them can be applied to the entire ICS field (e.g., platform type). Instead, some others are specific to Smart Grid (e.g., National Institute of Standards and Technology (NIST) grid domain). The employed testbed classification in [28] is mainly based on the research area, which motivates the development of each system. In this work, instead, we classify testbeds mainly based on the platform type, providing the reader an overview of the most suited ICS testbeds and datasets for his research.

Recently in 2019, Geng et al. [29] presented a survey on ICS testbeds based on the same four requirements of [25]. Besides analyzing different ICS datasets, the authors also present the different techniques that can be employed to build a testbed, including application scenarios, the main challenges, and future development directions. However, the authors' only introduced an analysis of each testbed's structure without going into details or providing comparison tables.

In the same year, Choi et al. [30] gathered and analyzed datasets for ICS security research, providing different comparison tables to understand the most suitable dataset depending on the case study. The authors based the comparison on the attack vector strategy. The paper

includes 11 commonly used datasets. Some existing datasets not widely used or without attack data are intentionally not considered. However, even if not suited for anomaly detection tasks, the latter could be useful to study the ICS environment's behavior. Also, one of the presented datasets (i.e., DEFCON23) is no longer available. Gosh and Sampalli [31] in 2019 presented a survey that classifies the security threats and the existing security schemes in industrial systems. Particular interest was given to the threats and the corresponding security measures related to Quantum Computing. However, differently from our work, the survey [31] does not focus on the testbeds and datasets available in the literature.

In 2020, in [32] the authors present an exhaustive survey with guidelines and good practices to help the building of an ICS testbed, highlighting the main challenges and the results of a focus group involving security experts to identify relevant design factors and guidelines. In the same years, Green et al. published another survey in this direction [33] with interesting guidelines for each ICS layer and a set of characteristics to consider when outlining testbed objectives, architecture, and evaluation process. While these works are interesting and give a comprehensive insight into the process of designing and evaluating a testbed, they do not consider datasets and IDSs, their requirements, and relationships.

Another interesting survey presented in 2020 by [34] discusses the transition of ICS standalone systems to cloud-based environments. The authors presented in detail the benefits, the main security challenges, and different case studies related to cloud-based ICSs. However, differently from our work, the authors mainly focused on reviewing the current ICS cloud transition state. Instead, we are interested in providing an overview of the testbed and datasets designed for security research. Pliatsios et al. [35] in 2020 present an exhaustive survey on SCADAs systems with a particular focus on the threats and vulnerabilities rising from the insecure design of the industrial protocols. The authors presented in detail the security issues of the protocols in terms of CIA Triad and how an attacker can exploit such vulnerabilities. Recently, Alcaraz and Lopez present a survey on how the new IT devices are being adopted to improve automation and control processes, allowing the convergence of IT with OT [36] However, few spaces on the survey were dedicated to the testbed, while the dataset and the most recent Intrusion Detection techniques were not discussed.

Differently from previous works, our survey aims to collect all the platforms (i.e., testbeds and datasets) useful for ICS security research. We base the existing literature to provide a detailed analysis of the current research issues, challenges, and future directions characterizing this field. We also report the best performance of the IDS on every dataset, which can be helpful for future IDS research baseline.

2.1.2 Industrial Control Systems

In this section, we offer a background on ICSs, useful when start approaching this field and to understand the remainder of this work. Firstly, in Section 2.1.2, we focus on the architecture of such systems compared to the classical systems architecture. Then we present a summary of the widely used ICS components in Section 2.1.2.

ICS Architecture

ICSs are composed of the interconnection of different computers, and electrical and mechanical devices used to manage physical processes. These systems are usually very complex and include heterogeneous hardware and software components such as sensors, actuators, physical systems and processes being controlled or monitored, computational nodes, communication protocols, SCADA systems, and controllers [37]. Control can be fully automated or may include a human in the loop that interacts via a Human Machine Interface (HMI). ICSs are widespread in modern industries (e.g., gas pipelines, water treatments) and critical infrastructures (e.g., power plants and railways).

Unlike classical IT systems, ICSs are composed of standard network traffic over Transmission Control Protocol (TCP)/IP stack and data from physical processes and low-level components. This interconnected and intertwined nature can open a wide space for new generation attacks exploiting new vulnerability surfaces. Several protocols are used in ICS, based on the specific purpose of each system. Industrial protocols are specifically designed to deal with real-time constraints and legacy devices in an air-gap environment. Many protocols do not implement any encryption or authentication mechanism due to these constraints, opening several vulnerabilities surfaces. Moreover, sometimes, the industrial protocols are customized from the company opening, again, many documentation and vulnerability issues.

The reference architecture of the ICS is the Purdue Model [29, 38]. As depicted in Figure 2.1, the Purdue module divides an ICS network into logical segments with similar functions or similar requirements:

1. **Enterprise Zone**, or IT network, includes the traditional IT devices and systems such as the logistic business systems and the enterprise network.

2. **Demilitarized Zone (DMZ)**, controls the exchange of data between the Control Zone and the Enterprise Zone, managing the connection between the IT and the OT networks in a secure way;

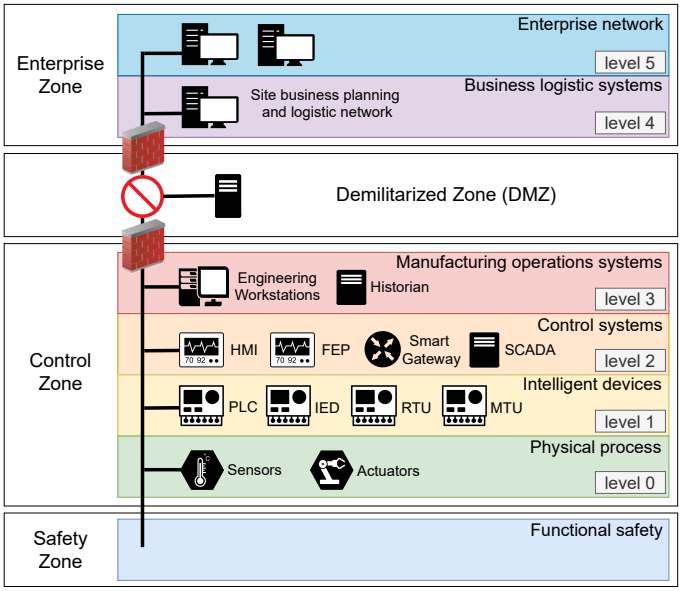


Figure 2.1: ICS Purdue Model architecture and corresponding level of the different industrial devices.

3. **Control Zone**, sometimes also referred to as OT network, includes systems and equipment for monitoring, controlling, and maintaining the automated operation of the logistic and physical processes. It is divided into four sub-levels:

    - *Level 0* includes sensors and actuators that act directly on the physical process;
    - *Level 1* includes intelligent devices such as PLC, Intelligent Electronic Device (IED), and Remote Terminal Unit (RTU);
    - *Level 2* includes control systems such as HMI, alarms, and control room workstations;
    - *Level 3* includes manufacturing operation systems that are often responsible for managing control plant operations to produce the desired end product;

    Level 2 and Level 3 devices can communicate with the Enterprise Zone through the DMZ.

4. **Safety Zone** includes devices and systems for managing ICS security by monitoring for anomalies and avoiding dangerous failures;

The role of the DMZ is to filter the internal communication of the network. In fact, according to the Purdue model, all the traffic Exchanged between OT and IT networks must pass

through the DMZ. However, this is rarely respected in the real-world, mainly due to the implementation difficulty or, more generally, the companies' insufficient attention to the industry-building phase's security aspects. This condition exposes the critical part of the system (i.e., OT network) to potential attacks.

Compared with the classical IT environment, ICSs need a different risk-handling strategy. The reliability is fundamental, and outages are not tolerated due to the critical nature of the processes monitored, unlike IT, where occasional failures are acceptable. The risk impact is also different: in the IT environment, the principal risk is the compromising of privacy and confidentiality (e.g., loss or unauthorized alteration of data). Instead, in the OT environment, a data compromise can cause a loss of production, equipment, and, in the worst case, a loss of lives or environmental damage.

Another difference with respecting traditional IT systems relies on information handling performance: in an IT environment, the throughput must be high enough, while delays and jitter are accepted. On the other hand, in the industrial field, communication is defined with regular polling time. Generally, this polling time is in second or millisecond orders, but delays are serious concerns. Finally, in IT systems, recovery can be made by rebooting. In contrast, in the OT system, fault tolerance is essential since a reboot would imply shutting down the entire industry and can lead to enormous economic losses [26].

For all these reasons, and considering that nowadays most of the ICS are connected with the Enterprise zone, it is essential to protect them using new and precise technologies.

ICS Components

Industrial Control Systems are composed of a wide range of heterogeneous devices and components with a specific role in the system. In this section, we briefly introduce the most common devices in the ICS fields. These devices are generally installed or simulated in the testbed to replicate the ICS environment. We reported in Figure 2.1 the level of the Purdue Model on which each device is installed.

PLC. PLC is a microprocessor-controlled electronic device that reads input signals from sensors, executes programmed instructions using these inputs and orders from supervisory controllers, and creates output signals that may change switch settings or move actuators. PLC is generally the boundary between the OT network and the physical process. It is often rugged to operate in critical environmental conditions such as very high or low temperatures, vibration, or in the presence of big electromagnetic fields. As with most ICS components, PLCs are

designed to last more than 10-15 years in continuous operations. The Real-Time Operations System installed in each PLC makes it suited for critical operations. The time to read all inputs, execute logic, and write outputs is generally very short.

RTU. An RTU is a microprocessor-controlled electronic device. Like PLC, it is designed for harsh environments and is generally located far from the control center, for instance, in voltage switch-gear. There are two types of RTUs: station and field RTUs. Field RTU receives input signals from field devices and sensors and then executes programmed logic with these inputs. It gathers data by polling the field devices/sensors at a predefined interval. It is an interface between field devices/sensors and the station RTU, which receives data from field RTUs and orders from supervisory controllers.

IED. An IED is a device containing one or more processors that can receive or send data from an external source. Examples of IEDs are electronic multi-function meters, digital relays, and controllers. Thanks to the higher complexity compared to a PLC and RTU, IED can perform more operations. An IED can be used for protection functions like detecting faults at a substation or for control functions such as local and remote control of switching objects and provide a visual display and operator controls on the device front panel. Other functions can be related to monitoring (for instance, a circuit breaker condition), metering (e.g., tracking three-phase currents), and communications with supervisory components.

Engineering Workstation. The Engineering Workstation is generally a desktop computer or server running a standard operating system hosting various software for controllers and applications. Engineers use this platform to manage the controllers.

HMI. The HMI is software installed on desktop computers, tablets, smartphones, or dedicated flat panel screens that permit operators to check and monitor the automation processes. As illustrated in Figure 2.2, the HMI shows the state of a plant operator, such as process values, alarms, and data trends. An HMI can monitor multiple process networks and several devices. An operator can use the HMI to send manual commands to controllers, for instance, to change some values in the production chain. Generally, the HMI shows a diagram or plant process model with status information to facilitate such a job.

Data Historian. A Data Historian is a software application used to collect real-time data from processes and aggregate them into a database for analysis. Data Historian mainly collects the same information shown in an HMI. The database and the hardware, generally a desktop workstation or a server, is designed for a very fast ingest of data without dropping data and uses industrial interface protocols.

Front End Processor (FEP). The FEP is a dedicated communications processor used to poll

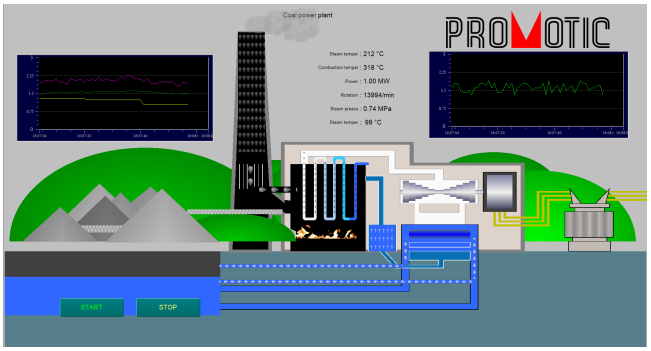


Figure 2.2: An example of HMI interface generated with Promotic Open-Source Tool [39].

status information from multiple devices to give operators the possibility to monitor the system's overall status.

Communications Gateways. A Communications Gateway is essential to make communications possible between devices from different manufacturers that use different protocols. Gateways can translate packets from a sending system to the receiver protocol.

Master Terminal Unit (MTU). MTUs manage the communication with the RTU or the PLC, gathers data from the PLCs, and process them. The communication between the MTU and the PLCs is bidirectional, but only the MTU can initiate the communication. Therefore, the MTU uses a master-slave communication where the MTU is the Master and PLCs are the slaves. Messages from the MTU to the PLCs can be triggered by an operator or be automatically triggered. These messages can either read memory parts representing current values or either write values in the memory and modify the configuration.

SCADA. SCADAs devices are placed on the higher level of the ICS hierarchy and are used to monitor and control centralized data acquired from different field sites. Furthermore, they manage the communication between the various devices and represent the remote connection point for the remote operators with the OT network. Over the year, SCADAs systems protocols moved from proprietary standards towards open international standards, resulting in attackers knowing precisely the protocols. That is why there is a gain of interest in reinforcing industrial control systems security.

ICS Field Devices. Field devices include all the components that are in direct contact with the physical process. The controllers can use them to get information regarding the physical process (e.g., the measure of temperature or pressure using sensors). Instead, actuators can interact with a physical process following commands from a controller (e.g., control motors, pumps, valves, turbines, and agitators). Communication with the controllers is generally performed via

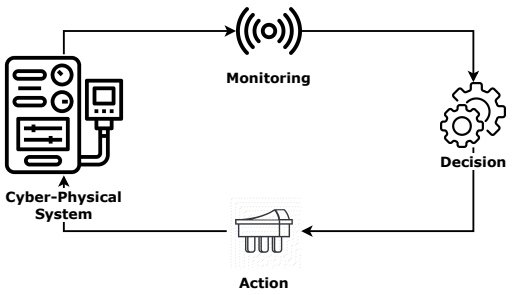


Figure 2.3: The CPS close loop.

I/O modules. Field Devices are implemented in the so-called CPS closed-loop to perform the three CPS main functions: monitoring using sensors, making decisions using PLCs, and applying actions using actuators. These three functions operate within a feedback loop covering, as shown in Figure 2.3.

Industrial Protocols Security

With the growth of the ICSs, several new protocols have been developed to support the specific requirements of the OT environment, like fault tolerance and reliability. The majority of these protocols were designed to operate in an air-gapped environment. Therefore originally, less importance was given to the security aspects with respect to the real-time constraint. Some of them have no security features at all (e.g., Authentication, Encryption). However, after the IT and OT convergence, they have still been used in practice [10].

This section reports the main industrial protocols focusing on the security properties initially and currently implemented. Standards like PowerLink Ethernet, EtherCAT, Constrained Application Protocol (CoAP), Message Queuing Telemetry Transport (MQTT), ZigBee PRO, WirelessHART, or ISA100.11.a are not used in the datasets and testbeds identified in this study. Therefore we decided not to report them. However, some of these protocols are widely used in different fields, for instance, in Industrial Internet of Things (IIoT) scenarios.

Table 2.1 summarized the main information related to the protocols presented in this section. It includes:

- Name of the **Manufacturer**;
- Standard **Ports** according to Internet Assigned Numbers Authority (IANA) [40];
- Information related to the **Original** protocol:

		Original					Enhancement				
Manufacturer	**Ports**	**Name**	**Year**	**E**	**I**	**A**	**Version**	**Year**	**E**	**I**	**A**
Schneider Electric	502/802	Modbus	1979				Modbus/TCP Security	2008	✓	✓	✓
GE Harris	20000	DNP3	1990			✓	DNP3-SA	2020	✓	✓	✓
Siemens	102	S7Comm	1994				S7CommPlus	2014	✓		✓
PROFINET Int.	-	PROFINET	2003				PROFINET Security Classes	2019	✓	✓	✓
ODVA	44818	EtherNet/IP	2000				CIP Security	2015	✓	✓	✓
OPC Foundation	4841	OPC	1996				OPC-UA	2006	✓	✓	✓
IEC	2404	IEC 104	2000				IEC 62351	2007	✓	✓	✓
IEC	102	IEC 61850	2003				IEC 62351	2007	✓	✓	✓

Table 2.1: Summary of the main protocol's characteristics and them security measures implemented in the enhancement extensions. In particular, the table shows **E**: Encryption; **I**: Integrity; and **A**: Authentication. The Enhancement part refers to the version proposed by the manufacturer.

  - **Name** of the protocol;
  - **Year** of release;
  - A tick if **Encryption**, **Integrity**, or **Authentication** are available;

- Information related to the **Enhancement** version with security measures offered by the manufacturer:
  - Name of the new **Version** of the protocol;
  - **Year** of release;
  - A tick if **Encryption**, **Integrity**, or **Authentication** are available.

Table 2.2: Protocol used in the presented testbeds and datasets. ● indicates that the protocol is supported/available. In some work, it is not indicated the version of Modbus (TCP, RTU, or ASCII) is adopted. In such cases, a ● is used to indicate a general version of the protocol.

Name	**Modbus**	**S7Comm**	**EtherNet/IP**	**DNP3**	**Logs**	**Phy.**	**Others**
4SICS	TCP	●	●	●			
Aghamolki et al.	●			●			IEEE-C37.118
Ahmed et al.	●		●				Profinet
Alves et al.	TCP						
BATADAL						●	
Blazek et al.							IEC61850
BU-Testbed			●				
CockpitCI	TCP						
CyberCity Dataset	TCP		●				NetBIOS
CyberCity Testbed	TCP		●				NetBIOS
D1: Power System					●	●	
D2: Gas Pipeline	TCP						
D3b: Water Storage Tank	●						

Table 2.2 – continued from previous page

Name	Modbus	S7Comm	EtherNet/IP	DNP3	Logs	Phy.	Others
D4: New Gas Pipeline	•						
D5: Energy M. S. D.					•		
Davis et al.	TCP						Custom TCP
DVCP	•						Profinet
Electra Modbus	•						
Electra S7Comm		•					
EPIC (IPSC)	TCP						
EPIC (iTrust)	TCP						IEC61850
EPIC Dataset	TCP					•	
EPS-ICS							Unknown
Farooqui et al.							Unknown
Gas Pipeline testbed	TCP						
Genge et al.	•			•			Profinet
Giani et al.	•			•			
Gillen et al.			•				CIP
GRFICS	•						
HAI Dataset						•	
HAI Testbed							Fieldbus
Hui Nuclear		•					Profinet, Custom TCP
HVAC_Traces		•					DCE/RPC, NetBIOS
HYDRA	•						
Jarmakiewicz et al.							IEC61850, IEC104
Jin et al.				•			
Kaouk et al.	tcp						
Kim et al.							Variable
Koganti et al.	•						
Koutsandria et al.	•						
KYPO4INDUSTRY	•			•			
Lancaster's testbed							Converted to IP
Lee et al.				•			IEC61850
LegoSCADA	•			•			
Lemay Covert	•						
Lemay SCADA	•						
LICSTER	TCP T						
Maynard SCADA							IEC104, OPC
Microgrid							Unknown
MiniCPS	TCP		•				
Mississipi Ethernet							Ethernet
Mississipi Serial	RTU, ASCII			•			
Modbus SCADA #1	TCP, RTU						
MSICST	TCP	•	•				
NIST	TCP		•				DeviceNet, OPC
PNNL							Not specified
PowerCyber				•			IEC61850
Queiroz et al.	TCP						
QUT_DNP3				•	•		
QUT_S7 (Myers)		•				•	
QUT_S7Comm		•					

Table 2.2 – continued from previous page

Name	Modbus	S7Comm	EtherNet/IP	DNP3	Logs	Phy.	Others
Reavers & Morris	TCP, RTU						
RICS-el							IEC104
S4x15 ICS	TCP						BACnet
Sayegh et al.							FINS
SCADA-SST	•						
SCADASim	TCP			•			
SCADAVT	TCP						Custom TCP
SGTB							Unknown
Singhet et al.				•			IEC104
SNL Testbed	TCP			•			IEC104 (partially)
SWaT Dataset			•			•	CIP
SWaT			•				CIP
T-GPP	TCP			•			
TASSCS	TCP						
Teixeira et. al	•						
TETRIS							
Turbo-Gas Power Plant	TCP		•				
VPST			•				
VTET	•	•					OPC
WADI Dataset						•	
WADI	TCP						
Wang et al.	TCP			•			
WUSTL-IIOT-2018	•						
Yang et al.							IEC104
Zhang et al.							Unknown

Industrial Protocols

Modbus. Modbus is a serial communication protocol initially published by Modicon (now Schneider Electric) in 1979 for use with its PLCs. Today Modbus [41] is one of the most used and famous protocols in the ICSs. Over the years, various versions of Modbus have been released. The first version was thought for serial communications, allowing to establish of asynchronous serial communications on RS-232 and RS-485 interfaces. Modbus is also adapted to transmission means other than copper, such as optical fiber and radio links. A typical communication via Modbus consists essentially of three stages: the formulation of a request from one device to another, the execution of the actions necessary to satisfy the request, and the resulting information's return to the initial device.

Security. Modbus was designed to be used in environments isolated from the Internet regarding the application layer's security. Therefore it does not include any security mechanism on this layer. These deficiencies are magnified by the fact that Modbus is a protocol designed for legacy programming control elements like RTUs or PLCs making the injection of malicious

code into these elements easier. Modbus Security [42] offers a Modbus/TCP version enhancement focused on using port 802. This new version enables Transport Layer Security (TLS) to provide confidentiality, integrity, and authentication using x.509v3 certificates. Moreover, it specified certificate-based authorization using role information transferred via certificate expansions. Researchers have also proposed different modifications to introduce confidentiality [43] or authentication [44] via covert-channel on Modbus.

DNP3. DNP3 is an open, intelligent, robust, and efficient SCADAs protocol organized into four layers: physical, data link, pseudo-transport, and application. In serial implementations, commands are issued broadcast. DNP3 contains significant features that make it more robust, efficient, and interoperable than older protocols such as Modbus, at the cost of higher complexity. The protocol's primary goal is to maximize system availability by putting less care into confidentiality and data integrity factors. DNP3 organizes data into data types such as binary inputs/outputs, analog inputs/outputs, counters, time and date, and file transfer objects.

Security. As previously mentioned, DNP3 is a protocol designed to maximize system availability by putting less care into confidentiality and data integrity factors. At the application level, some efforts have been made to provide a safe authentication standard in DNP3. While in the beginning, pre-shared keys were used to authenticate, according to the standard Institute of Electrical and Electronics Engineers (IEEE) 1815-2010 (deprecated), the latest versions implement Public Key Infrastructure (PKI) with remote key changes (standard IEEE 1815-2012). Recently, in 2020, GE Harris presents DNP3 version 6, introducing DNP3-SA [45], a separate protocol layer that supports Message Authentication Code to provide secure communication sessions, including authentication and integrity. Moreover, this version supports encryption to offer data confidentiality by using the AES-256 algorithm. Some other solutions have been proposed in the literature to implement cryptography protections, such as end-to-end encryption [46] and Virtual Private Network (VPN) for IP networks [47].

S7Comm. Introduced in 1995, S7comm (S7 Communication) [48] is a Siemens proprietary protocol that runs between standard PLCs of the Siemens S7-200/300/400 family and new generation PLCs like S7-1200/1500. It is a proprietary and closed standard without significant literature related to it. Siemens has a proprietary HMI software for the SIMATIC products and an Ethernet driver that provides connectivity to devices via the Siemens TCP/IP Ethernet protocol. In addition to this driver, there are also 3rd-party communication suites for interfacing and exchanging data with Siemens S7 PLCs.

Security. S7Comm is a closed protocol, so there is no related documentation. However, as various works underline, the base version of S7Comm does not include security features, and it is

vulnerable to replay attacks [49]. However, in 2010 Stuxnet exploited the security vulnerabilities of S7Comm to compromise a Nuclear Plant in Iran. As a result of this incident, Siemens has developed a new version of the protocol, called S7CommPlus, with replay-attack protection. It has been proven that this version is also vulnerable to reverse debugging attacks [49].

PROFINET. Developed by PROFIBUS & PROFINET International (PI), PROFINET [50] is an open standard for Industrial Ethernet standardized in International Electrotechnical Commission (IEC) 61158 and IEC 61784. Introduced in 2003, it is an evolution of the PROFIBUS standard, whose lines can be integrated into the PROFINET system via an IO-Proxy. This protocol follows the provider-consumer model for data exchange in a cascading real-time concept.

Security. PROFINET is a protocol operating in the application, link, and physical layers. The link layer in this protocol uses Fieldbus Data Link (FDL) to manage access to the medium. FDL operates with a hybrid access method that combines master-slave technology with the passing of a token, indicating who can initiate communication and occupy the bus. These measures ensure that devices do not communicate simultaneously. However, FDL constitutes any safety mechanism and may be susceptible to attacks involving traffic injection or Denial of Service (DoS). In 2019, PI introduced three Security Classes to offer a way to select security measures based on consumer needs [51]. Class 1 improves robustness through a digital signing of General Station Description files using a PKI infrastructure, an extended Simple Network Management Protocol (SNMP) configuration, and a DCP in read-only mode. Class 2 expands the previous class by offering integrity and authenticity via cryptographic functions and confidentiality only of the configuration data. Instead, Class 3 offers all the previous characteristics and the confidentiality of all the data.

ODVA's networks. Founded in 1995, ODVA [52] is a global association whose members comprise the world's leading automation companies with the mission of developing advanced open and interoperable communication technologies for industrial automation. The primary interest is developing the Common Industrial Protocol (CIP), supporting the various network adoptions such as DeviceNet, CompoNet, ControlNet, and the widely used EtherNet/IP. CIP encompasses a comprehensive suite of messages and services to collect industrial automation applications such as control, safety, energy, synchronization, motion, information, and network management. EtherNet/IP is an adaption of CIP to the Ethernet TCP/IP stack, while DeviceNet provides a way to use CIP over the CAN technology. ControlNet uses CIP over a Concurrent Time Division Multiple Access data link layer, and CompoNet implements CIP on a Time Division Multiple Access data link layer.

Security. Recently, in 2015, ODVA introduced the CIP Security framework [53] to provide

security measures to CIP protocol. Since different systems might need different security levels, CIP Security provides different security specifications profiles to help users configure interoperable devices. On EtherNet/IP, it enables TLS and DTLS to secure the TCP and User Datagram Protocol (UDP) transport layer protocols. TLS and DTLS provide authentication of the endpoints using X.509 certificates or pre-shared keys, message integrity and authentication employing TLS message authentication code, and optional message encryption.

Open Platform Communications (OPC). The classic OPC [54], developed in 1996, was designed to provide a communication protocol for personal computer-based software applications and automation hardware. In 2006 a new version, Open Platform Communications Unified Architecture (OPC-UA), was released as an operational framework for communications in process control systems. The general layout of the communication is simple: the hardware devices (e.g., PLC, Controller) act as data sources, and the software applications (e.g., SCADAs, HMI) play the role of data consumers, whereas the OPC interface acts as connectivity middleware, enabling the data flow. Using the OPC, the client applications access and manage the field information without knowing the physical nature of data sources. With OPC-UA improvements, the protocol is widely used in critical and industrial fields such as energy automation, virtualized environment, and building automation.

Security. OPC-UA implements a security model and five security classes, bringing greater security to the architecture at the cost of slightly higher complexity [55]. It is also possible to implement only a fraction of the security measures by using one of the five security classes provided. The security model allows for generating a secure channel that provides encryption, signatures, and certificates at the communication layer. Furthermore, a session in the application layer is used to manage user authentication and user authorization.

IEC 60870-5-104 (IEC104). Released in 2000, IEC104 protocol [56] is an extension of the IEC 101 protocol with the changes in transport, network, link, and physical layer services to suit the complete network access. There are two different methods of transporting messages. The first provides bit-serial communications over low-bandwidth communications channels. In the second, introduced with IEC104, the protocol's lower levels have been completely replaced by the TCP/IP transport and network protocols.

Security. IEC104, has been proven to be vulnerable to different types of attacks, such as man-in-the-middle and replay attacks [57]. A more recent and secure standard of the IEC family is IEC 62351. This version implements end-to-end encryption to prevent attacks such as replay, man-in-the-middle, and packet injection. However, due to the higher complexity, industries rarely upgrade IEC104 to IEC 62351.

IEC 61850. Like IEC104, IEC 61850 [58] was originally designed to enable communications inside substations automation systems. In recent versions, an extension of IEC 61850 allows substation-to-substation communication and provides tools for translation with other protocols such as IEC 60870-5, DNP3, and Modbus. The protocol is devised using an object-oriented design suited for communication between devices of different vendors.

Security. IEC 62351 standard provides various security measures, offering guidelines and developing a secure operation framework. Since time-critical application encryption is not suitable due to the 3ms delivery overhead, the standard recommends using digital signature generated by SHA256 and RSA public key algorithms. Furthermore, the employment of IEC 61850 in heterogeneous networks exposes the system to protocol mapping vulnerabilities. It is possible to prevent these vulnerabilities by developing ad-hoc security by design architectures [59].

Other protocols. In addition to previously presented protocols, some datasets contain few packets related to other generic protocols used in a wide range of applications. Published in 1995, BACnet is a data communication protocol for building automation and control networks supported by some HVAC components but not widely used [60]. Distributed Computing Environment/Remote Procedure Calls (DCE/RPC) is a remote procedure call system that allows programmers to write distributed software as if it were on the same computer. One of the datasets presented in this study includes DCE/RPC, together with NetBIOS, a networking protocol allowing applications on separate computers to communicate over a Local Area Network (LAN). Other generic packets are visible in some datasets like Address Resolution Protocol (ARP) and Domain Name System (DNS) requests but are generally not related to the industrial field.

Industrial Protocols Employment

In Table 2.2, we provide the complete list of the datasets and testbeds analyzed in this work, together with the protocol used in the specific platform. In detail, the table associates to each testbed the protocols supported and to each dataset the protocols available. Moreover, it indicates if data logs and physical measures are provided in the datasets. As previously described, there are several different protocols employed in the ICS field. In Figure 2.4, we reported the percentage of usage of each protocol in the testbeds and datasets investigated in this survey. Modbus, and its different versions (i.e., TCP, RTU, ASCII), are the most used protocols, while EtherNet/IP, DNP3, and S7Comm follow with a lower but significant employments.

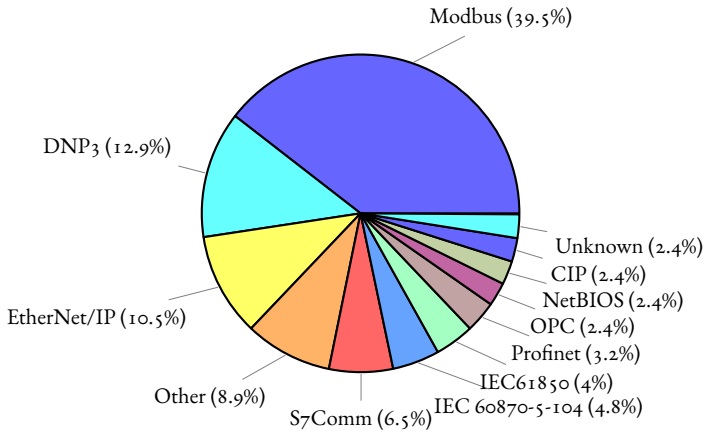
Since the testbed and dataset employed should represent an approximation of real-world

scenarios, it is interesting to compare if the distribution of the protocols implemented in the different datasets is similar to the protocol's distribution in the real industrial system. Verifying this claim is a challenging task due to the various privacy concerns of companies in disclosing information. Various works tried to deal with this problem by measuring the industrial traffic present on the Internet. Although with limitations, the traffic measurement can represent a reasonable estimate of the most popular industrial protocols currently used. By leveraging Censys search engine, Xu et al. [61] scanned the Internet for about two years (from 2015 to 2017), examining for industrial devices exposed. In particular, they focused on five protocols: Modbus, S7Comm, DNP3, BACnet, and Tridium Fox. Results show a significant prevalence of Modbus and Tridium Fox devices (with more than 20K devices found), a middle spread of BACnet (about 11K devices) and S7Comm (about 4.5K devices), and a lower number of devices using DNP3 (less than 1K). Furthermore, the authors noticed an increasing number of Modbus and S7Comm devices during the two years of recording, while the number of DNP3 devices decreased. A similar study, presented by Barbieri et al. [10], leveraged Shodan and an Internet Exchange Point (IXP) in Italy to measure ICS host exposure. They discover many devices using Modbus, MQTT, and Niagara Fox. Furthermore, the authors also identified EtherNet/IP, S7Comm, and BACNet devices but with significantly lower samples.

In addition to measurement works, we can also rely on market analysis. According to an HMS report [62], the overall market share of Industrial Ethernet protocols increased in 2020. In particular, EtherNet/IP and Profinet obtained first place with 17% of the market share, while in third place there is EtherCat with a share of 7%. On the other side, Fieldbus protocols such as Profibus and DeviceNet showed a decrease of 5% in the market share with respect to the previous year. Interestingly the Modbus protocol (TCP and RTU variants), despite being heavily employed in testing, results in a 10% of market share (5% RTU, 5% TCP).

These findings show that Modbus/TCP is the most employed protocol in testbeds, datasets, and Internet measurements. Nevertheless, it obtained a low market share in the last year (i.e., 2020), outranked by EtherNet/IP, which is also employed in a significant part of the testbeds and datasets presented in this survey, and Profinet, which instead is used in only the 3.2% of the testing system analyzed in this work. Therefore, Profinet can be an interesting protocol to introduce in future testbeds and datasets to follow the market trend. Other protocols that have an increasing market share are BACnet, TridiumFox, and NiagaraFox, which are not present in the testing platforms, except for one dataset that contains BACnet packets. Finally, another protocol that could be interesting to include in testing platforms is EtherCAT, which has a 7% of market share. However, no testing system currently supported it.

Figure 2.4: Percentage distribution of different protocols in the datasets and testbeds analyzed.



2.1.3 ICS Adversary Models

In this section, we offer an overview of the various attacks and defense mechanisms in ICSs. In particular, in Section 2.1.3 we present an overview of the typical attacks which can target ICSs and are implemented in the different testbeds and datasets. Instead, in Section 2.1.3 we proposed a brief overview of the techniques employed to detect and mitigate attacks in this field.

Typical Attacks

ICSs are extremely complex systems, which connect IT components with sensors, actuators, and other OT devices. Such an interconnected scenario with a wide variety of various components may hide attack surfaces caused, for instance, by device-specific vulnerabilities or misconfigurations. Recently, [36]

Having a clear idea of the different attack typologies is essential in building and testing defenses. Based on that, testbeds should be capable of simulating verisimilar attacks, while datasets should include not only normal operation data but also attack data. Reproducing attacks is a challenging task because the simulation should precisely emulate a realistic abnormal operating condition. However, it is impossible to replicate every type of attack due to the devices' potential damage. Some attacks could also shift the testbed's operating behavior in a dangerous state and seriously damage the machines. Furthermore, the limited class of attacks implemented could raise a generalization problem of the detection strategy, not transferable to novel and unknown attacks.

In a CPS scenario, by definition, there are two possible attack vector surfaces on the system. *Network-based* attacks, targeting the networking part of the network such as packets, protocols or routing policies, and *Physical-based* attacks, aimed at corrupting the physical process of the devices. Sometimes, these two attack categories' goals may also converge or combine to reach a specific target.

Network Attacks. The most common attack models include the Control Zone network access by the attacker to compromise an ICS. An attacker can obtain the network control through a phishing attack to the site operators [4] or by exploiting the security lack of the legacy devices connected to the Internet [10]. There are different actions that malicious actors can perform, but it is possible to categorize the main ones into five different classes [63, 64] of network attack. These attacks are also implemented in the testbeds to generate abnormal operating conditions.

- **Reconnaissance Attack** aims to identify potential victims within a network. Usually, this class of attack is used to plan other moves, such as identifying other vulnerable devices. These attacks can be passive (e.g., port mirror) or active (e.g., nmap). Reconnaissance Attack can be performed to understand the topology of the ICS with the consequently vulnerable devices or to identify the physical process involved.

- **Man-in-the-Middles (MiTMs) Attack** allows an attacker to sit in the middle of communicating parties. The attacker is then able to read or modify the communications, inject commands, or drop packets. As introduced in Section 2.1.2, most industrial protocols suffer from insecurity by design and, therefore, an attacker can perform malicious protocol exploitation. In [35] the authors reported a detailed taxonomy of attacks against Modbus and DNP3 protocols. The final aims of this type of attack can vary from the control of some devices to the disruption of the ICS's normal state to damage the system's owner or the system itself. MiTM attacks on ICS can intercept wired communication, which requires the installation of network tap or wireless by using antennas.

- **Injection Attack** aims at supplying untrusted and malicious inputs to a system. Typically, in an ICS, an attacker can inject data such as false measures from sensors or actuators (Data Injection Attack) or command (Command Injection Attack). Often a compromised node launch this type of attack, but, in some cases, the injected data can originate from other sources (e.g., a new entry point for the network). This type of attack also includes the injection of Malware (e.g., Worm [3], Ransomware [24]) or Backdoors in the devices, which allow the attacker to drive the system to an unsafe state.

- **Replay Attack** is based on retransmitting a valid message previously seen in the network. This attack is difficult to be detected, and it can lead to malfunctions in the system. For example, in a nuclear plant context, the attacker could retransmit a message with a low temperature of the reactor instead of rising, inhibiting the activation of safety measures.
- **DoS Attack** makes devices unavailable by overloading the system resources to disrupt the communication between machines. Usually, a common technique is packet flooding and, if packets are generated from many different sources, it is called Distributed Denial of Service (DDoS). This attack can stop some devices, making them unavailable and leading to unpredicted behaviors in the ICS. As previously explained, industrial devices are generally legacy and have low computational power, therefore even a low amount of packets can stop their normal functioning.

Physical Process Attacks. This class of attacks aims to alter the system's physical process and complex relations to manage it. Cyber-Physical Systems enable such attack surfaces due to the field device (i.e., sensors and actuators), sometimes in remote places. To achieve these attacks, the attacker could have previously obtained access to the system with one or more of the network attacks previously described. Generally, physical process attacks represent the final attack chain goal, which starts with the network as an entry point.

- **Stealth Attack** generates small perturbations in the system process to create long-term damages (e.g., loss in production terms or the devices' degradation). The stealth attack can use a static perturbation by introducing a constant error in the physical measure (e.g., increasing or decreasing the production) or dynamic by rapidly oscillating between upper and lower measurement bounds (e.g., causing turbulence in the flows). This class of attacks is generally difficult to detect since it maintains the process within its limits. Generally, if the stealth attacks compose a sequence, then the corresponding threat is generally indicated as Advanced Persistent Threat (APT), as in the case of Stuxnet or BlackEnergy [36].
- **Internal Logic Modification:** as introduced in Section 2.1.2, the control logic of the PLC is generally programmed in ladder logic, structured text (IEC 61131-3) or functional blocks. An unauthorized modification of the internal logic aims at modifying the physical condition monitored by the device. According to [65] there can be two types of modification: logic modification and function modification. Logic modification aims to modify the internal boolean logic of the device to bypass the control condition, while

function modification attempts to change the internal parameters updates. Although this attack requires high access to the target device, it can cause dangerous consequences to the system by driving it to unstable conditions or damaging the equipment.

- **Device Manumission** is achieved by physically tampering with the field device to compromise the data recorded. This attack aims to induce wrong measurements in the system by exploiting the distributed and, therefore, less monitored nature of these systems.
- **Direct Damage Attacks** aims to disrupt and damage the entire process or physical equipment by introducing significant process variations that bring the system into an unsafe state. This attack may also severely affect the population or the environment around the site.

ICS Defence Techniques

Enforcing ICS security by implementing security-by-design architectures is possible [66, 2]. For instance, it is possible to use DMZ as specified in the Purdue Model (Figure 2.1), enforcing network separation and segregation. Furthermore, boundary protections and firewalls with ICS-specific rules help protect an ICS from external attacks. The NIST proposed a complete guide explaining how to set up a secure network to protect an ICS [13]. Furthermore, security designers can rely on public databases, such as MITRE ATT&CK [67], ICS-CERT [68], and NIST's national vulnerability database [69], that collect the most common attacks, threats, and vulnerabilities. Another good practice is to implement a secure version of the industrial protocols. However, security-by-design can be challenging to consider in ICSs due to implementation constraints. Sometimes, it could also happen that companies consider the security aspects after the construction phase, which can raise problems in integrating measures into the infrastructure. Detection mechanisms can solve this limitation and be integrated into the system after construction, for instance, in central nodes or with network tap.

One of the most widely adopted techniques to secure an ICS is represented by IDSs. IDSs are algorithms designed to detect attacks by passively or actively monitoring the system. If the IDS is passive, it will only raise passive alerts in case of an anomaly. Instead, if the IDS is active, it will also take active response action in case of an anomaly (e.g., shutting down part of the system). IDSs represent a cost-effective solution since they can be installed without changing the system topology or substituting all network devices. In the following, we briefly report the two main categories of IDS, which employ two different approaches to detect attacks or domain drifts.

Knowledge-based intrusion detection (also called *misuse-based*) focuses on looking for runtime features that match a specific pattern of misbehavior. This method aims to exploit the stationary of ICSs, which, unlike IT systems, are characterized by control loop operation regulated by a constant polling time communication. The most famous misuse-based solutions include Snort [70] and Suricata [71], which has also been extended to include industrial protocol rules. The knowledge-based approach requires low computational power and has a low false-positive ratio since the system reacts only to known threats. However, it has the disadvantage of offering no protection against zero-day vulnerabilities. For this reason, the research community is focusing on developing a dynamic mechanism that can identify domain shifts without the need for signatures [72].

The current research trend focuses on the *anomaly-based* intrusion detection, which looks for runtime features that differ from normal behavior. The normal behavior pattern can be defined using unsupervised approaches training the model with live data or semi-supervised utilizing a set of truth data. This approach is called *behavior specification-based* intrusion detection. It represents a suitable ICS solution since it aims to dynamically learn the regular behavior model of network traffic and physical models. Again, ICS systems are generally characterized by constant time communication, thus helping the definition of a more robust model.

This last method is promising thanks to modern machine learning and deep learning techniques that can be used to automatize the anomaly detection classification process. A common requirement of these algorithms is the need for a considerable quantity of data: generally, the more data you provide to the training phase, the more precise your detection will be.

IDS are also classified according to the data source. *Network-based* IDS uses network adapters to collect and analyze packets in real-time. On the contrary, *host-based* IDS monitors the documents, processes, and other information specific to a particular device to identify. The disadvantage is that monitoring regard only one node in the network, while with the former approach, the network is under control. On the other hand, *host-based* can also detect threats coming from sources other than the network (e.g., USB sticks) [73].

The basic idea behind IDS is to exploit the massive amount of data collected from the sensors and predict an ICS's operations. Many features can be extracted from network traffic (e.g., timings of packets, bytes transmitted) and the physical process (e.g., sensor measurements, actuator statuses). This information can be employed to detect abnormal behaviors and attacks in an ICS.

A novel detection design concept that exploits the correlation of multiple ICS points was proposed by Bernieri et al. [74]. In this work, the authors proposed a distributed detection

approach to consider the different information characterizing ICSs to identify more complex vulnerabilities. Similarly, Ghaeini and Tippenhauer [75] proposed a hierarchical IDS which can combine data from different levels to detect attacks having a distributed effect on the system. Another novel approach has been proposed by Caselli et al. [76], who designed a sequence-aware IDS based on monitoring sequences of events instead of single ones.

Other common defense techniques adopted to protect ICSs include Honeypots and Intrusion Preventions Systems (IPSs). Honeypots mimic a real system to deceive an attacker and collect information about the attackers' typical actions [11, 77]. Instead, IPSs refer to systems aimed at detecting anomalies and attacks, but differently from IDSs, IPS are by design reactive, and they take actions to reduce the malicious impact [78]. Other solutions instead propose real-time attack traceability systems through consensus []

2.1.4 ICS Testbeds

In this section, we present a comprehensive analysis of the various ICS testbeds available in the literature. Firstly, in Section 2.1.4 we introduce the classification method we employ in this work. Instead, in Section 2.1.4 and Section 2.1.4 we recall, respectively, the requirements for an effective testbed and the main challenges in developing an ICS testbed. Then, we propose a detailed description of a set of interesting testbeds we choose, dividing them into the three categories that we design. In particular, Section 2.1.4 contains physical testbeds, Section 2.1.4 presents virtualized testbeds, and Section 2.1.4 describes hybrid testbeds which are a conjunction point of the other two categories.

Testbeds Classification

There are different possible classifications of a testbed, based on its sector, construction methodology, or the process involved. In this survey, and particularly in this section, we consider the functional elements involved in the testbed, classifying them as *Physical*, *Virtual*, or *Hybrid* testbed. The different testbed categories are illustrated in Figure 2.5, even if sometimes the difference between them can be minimal. For instance, many of the virtual testbeds presented can be interconnected with physical devices or wholly virtualized. Instead, Hybrid systems were designed with some real components and, without them, they could not work correctly.

Physical testbeds use real hardware and software to configure both the network and physical layers. They are a suitable approach when researchers need a solution to collect realistic measurement variation and latencies. Furthermore, it is possible to exploit the vulnerabilities

of a specific device. On the other hand, physical testbeds are expensive both in construction and maintenance. They generally have a long building time, and they may not provide a safe execution of dangerous physical processes (e.g., the nuclear sector).

On the contrary, virtual testbeds leverage software simulations and emulations with single or multiple programs to reproduce the entire network and all the different components. A virtual testbed represents a low-cost solution, but it is not easy to simulate high-fidelity physical processes due to the virtualized environment. Despite this lack of precision, dangerous and risky processes (e.g., the nuclear sector) can be, in this way, simulated in a laboratory. Matlab, Modelica, Ptolemy, and PowerWorld are software used in the process simulation phase. Other tools are used to model control center communication networks (e.g., DETER, Emulab, CORE, ns3) and other devices used in the system such as PLCs (e.g., STEP7, RSEmulate, Modbus Rsim, Soft-PLC). Despite not generating data with perfect fidelity, these approaches are easy to update and upgrade, which gives them good flexibility and extendibility.

A widely diffused approach is developing testbeds composed of both physical devices and software simulations. This approach represents a good trade-off between physical and virtual solutions and is called a hybrid testbed. The main difference between the complete physical testbeds is that part of the components is simulated using specialized software. This solution can reduce the system's fidelity, but on the other hand, it permits containing the cost and development time. However, as stated before, the separations between Virtual and Hybrid testbed is not always well defined. Sometimes virtual testbeds can be modified to work as a hybrid testbeds by supporting physical devices. For example, VTET [79] can be deployed using physical PLCs to replace the simulated ones. In this work, we consider as Hybrid a virtualized testbed composed of at least one real industrial device (e.g., PLC, IED, actuator, sensor).

In Figure 2.6, we reported the geographic distribution of the Physical and Hybrid around the world. We think that this representation could help the reader see the current research trend in this sector of the world. In particular, Figure 2.6a provides a high-level view of where the testbeds are placed in the world, while Figures 2.6e, 2.6d, and 2.6c show close-up of the countries with more than one testbed. In these figures, the marker size represents the estimated cost of the testbed. Simultaneously, the color indicates the Citations of the associated reference according to Google Scholar at the writing time. Furthermore, we developed a website with an interactive map to collect and also provide information about future ICS testbeds and datasets[1]. Our goal is to continue to update this collection in the future. Moreover, in Table 2.3 we reported a brief comparison between the testbed presented in this survey, highlighting their main

[1] https://spritz.math.unipd.it/projects/ics_survey/

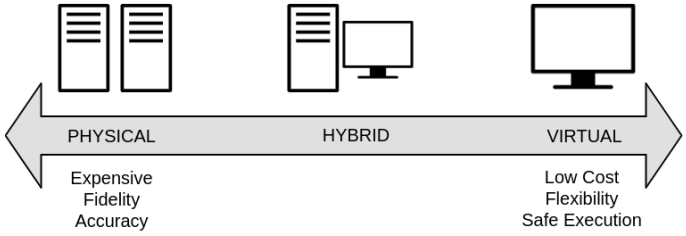


Figure 2.5: A summary of differences between testbed types.

information and features. In particular, for every testbed, we reported the following information.

- **Name** of the testbed (or of the authors if a name is not provided);
- **Institution** in which the testbed has been developed;
- **Country** The country on which is based the testbed or the institution of the first author;
- **Sector** indicates the field of the represented process;
- **Category** of the testbed. It can be *Physical*, *Virtual*, or *Hybrid*;
- **Physical Process** indicates how is implemented at the physical level. It can be *Simulated* with software or *Real* if consists of a physical implementation;
- **License** of the testbed. It can be:
  - *Open-source* if the source code is freely available;
  - *Open description* if despite the source code is not provided, the description is sufficiently detailed to allow a reader to develop a similar copy;
  - *Education* if it is maintained by a university and open to collaborators;
  - *Collaborations* if it is maintained by an institution that can accept collaborations;
  - *Not available* if it is owed by a private company and so not accessible or not available online.
- **Scope** indicates the applications of the testbed. It can be:
  - *Security* if the main scope is related to cybersecurity research;

  - *Forensic* if the target scope is to provide a way to perform forensics research;
  - *Pedagogy* if the main scope is to provide education to students;
  - *General* if a precise scope is not specified.

- **Cost** estimated for the testbed implementation. When available, the cost estimation is based on our analysis of the documentation of the testbed. Furthermore, we consider the testbed's size, and the number and type of devices employed (e.g., a single professional PLC cost at least 300$).

  It can be:

  - *Low* for a cost estimated < 500$. These testbeds are generally composed of a single PLC or different low-cost devices (e.g., Raspberry or Arduino).
  - *Medium* for a cost estimated between 500$ and 10*K*$. These testbeds are composed of a discrete number of devices and hardware not extremely expensive, but at the same time, they have a reasonable degree of complexity.
  - *High* for a cost estimated > 10*K*$. This category is conceived for testbeds with a high degree of complexity and a large number of devices employed.

- **Reference** includes a reference to a description of the testbed.

- **Resource**, if available, indicates a resource for the download.

This information was not always available or easy to retrieve; therefore, the degree of detail may vary according to the specific dataset.

Testbeds Requirements

When researchers need to work with a real-world ICS environment, the proper solution is to build a testbed for conducting rigorous, transparent, and replicable testing of new technologies. The different testbeds vary in dimension, complexity, or sector. According to [29], an effective testbed needs to satisfy four main requirements: i) *Fidelity*, ii) *Repeatability*, iii) *Measurement Accuracy*, and iv) *Safe execution*. Sometimes, it could be challenging to satisfy all these requirements together; therefore, it is important to determine an optimal trade-off based on the research needs during the design phase.

A testbed should be developed to achieve good *fidelity* by accurately replicate the devices and processes from a real-world ICS. This is an expensive and space-consuming task, making

it difficult for other researchers without much funding to replicate the same environment to verify the results. In these cases, mathematical models can be employed to virtualize physical processes in a cheap but less accurate way.

Repeatability is an essential property for a testbed: it allows other researchers to reproduce the findings and compare other solutions on the same system. This property can be easily achievable for completely simulated testbeds, while it can be extremely challenging for ICSs that employ physical components or processes.

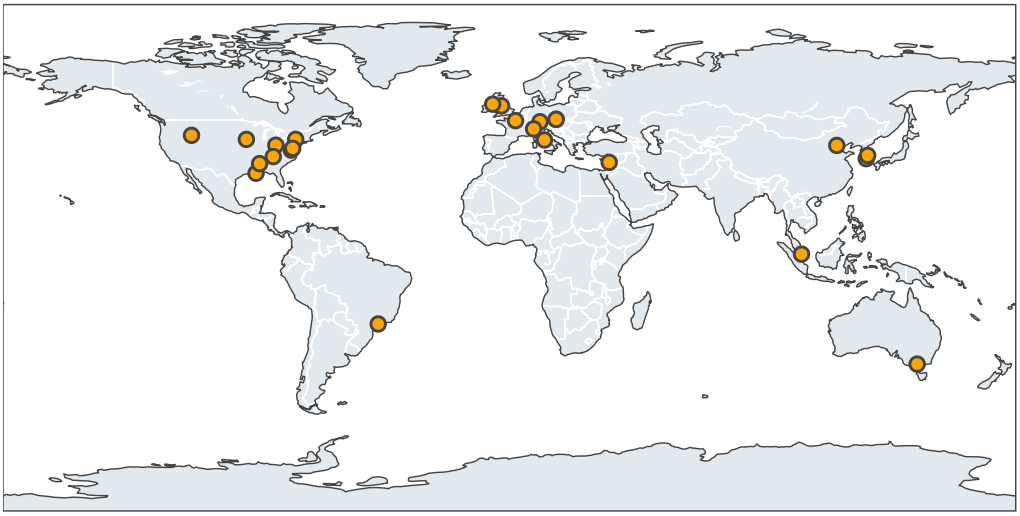
A testbed should monitor a physical process and take *accurate measurements* without interfering with it. Sensors must be placed smartly, and if different points of measures are available, they must be carefully synchronized to provide accurate and reliable data.

Often ICSs are used to manage critical physical processes (e.g., chemical reactions, nuclear plants). If under attack, these kinds of *processes can be dangerous* and can cause physical damage to the system itself. Since researchers need to study countermeasures' effect and effectiveness to attacks, testbeds must be provided with safe execute risky processes. This design challenge can be mitigated by employing simulations at the cost of a loss of accuracy. In other cases, processes are instead less critical. However, they can have an expensive or time-consuming recovery after an attack (e.g., after an attack completely empties a container into a water treatment system, it will take time to refill it again). In these scenarios, a virtual approach can be an excellent alternative to the physical replication [79].

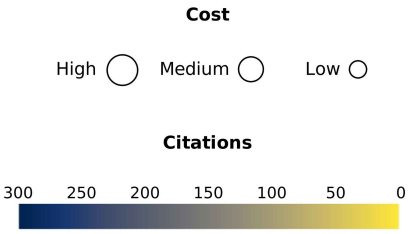
Challenges in Developing a Testbed

The development of an industrial testbed is challenging from several points of view. Different works analyze the challenges in developing a well-designed ICS testbed [80, 81]. Based on the existing literature, in the following, we present the main problems related to the development of such a testbed.

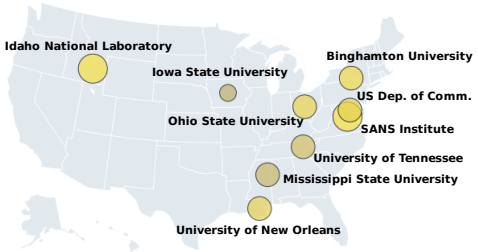
- **Design Guidelines**: When a research group decides to venture into building a testbed, it is fundamental to have a clear idea of the architecture. A clear and defined architecture can be useful in the development phases and to project further expansions. Moreover, it can guide other groups in building their own testbeds and therefore enabling the experiment repeatability. However, it is difficult to identify clear guidelines that help design a testbed from the engineering perspective.



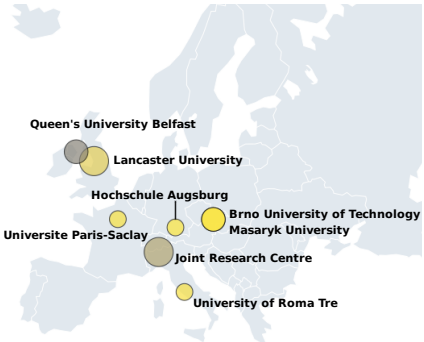
(a) Physical and hybrid with physical process testbeds distribution around the World.



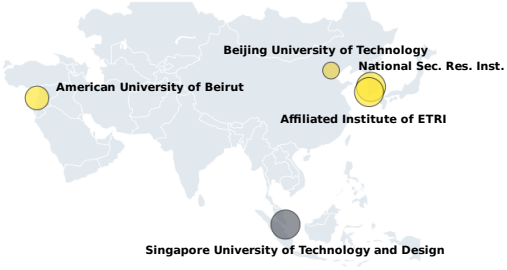
(b) Legend



(c) North America



(d) Europe



(e) Asia

Figure 2.6: Physical and hybrid with physical process testbeds distribution on the continents with more than one testbed: North America, Europe, and Asia. If there is more than one dataset in a place (e.g., SUTD), we aggregated the information.

Table 2.3: Summary of testbeds presented in the literature. We denote **Category** as H: Hybrid, P: Physical, and V: Virtual. To specify the ICS process **Physics** we use S: Simulated, R: Real, M: Mixed, and No: if there is not a physical process. To denote the **License**, we use OD: Open Description, E: Education, OS: Open-Source, C: Collaboration, and NA: Not Available. To denote the **Scope**, we use S: Security, G: General, P: Pedagogy, and F: Forensic. To denote the **Cost** we use L: Low, M: Medium, and H: High. The entries in Yellow indicate that the testbed is hybrid, the entries in blue mean that the testbed is physical, while entries in green correspond to virtual testbeds.

Name or authors	Category	Institution	Country	Sector	Physics	License	Scope	Cost	Reference	Resource
Aghamolki et al.	H	USF	Florida, US	Power Grid	S	OD	S	M	[82]	-
Alves et al.	H	UAH	Alabama, US	Gas Pipeline	S	OD	S	L	[83]	-
CockpitCI	H	University of Coimbra	Portugal	Power Grid	S	OD	S	M	[84]	-
CyberCity	H	SANS Institute	-	City	M	E	S, P	H	[85]	-
EPIC (IPSC)	H	JRC Ispra	Italy	General CPS	S	OS	S	L	[86]	[87]
EPS-ICS	H	BIT	China	Generic ICS	R	NA	G	L	[88]	-
Gillen et al.	H	ORNL	Tennessee, US	Cooling System	S	OD	S	M	[89]	-
Hui Nuclear	H	Queen's University	UK	Nuclear Plant	S	OD	S	M	[90]	-
HYDRA	H	University of Roma Tre	Italy	Water Distribution	R	OS	S	L	[91]	[92]
Jarmakiewicz et al.	H	MUT	Poland	Power Grid	S	OD	G	M	[93]	-
Kaouk et al.	H	University of Grenoble	France	Generic ICS	S	OD	S	L	[94]	-
Kim et al.	H	NSRI	South Korea	6 Different ICS	R	E	S, P	H	[95]	-
Koutsandria et al.	H	Sapienza University	Italy	Power Grid	S	OD	S, F	M	[96]	-
KYPO4INDUSTRY	H	Masaryk University	Czech Republic	Linear Motor	R	OD	P	M	[97]	-
LegoSCADA	H	Universite Paris-Saclay	France	Vehicular	R	OS	S	L	[98]	[99]
Microgrid	H	OSU	Ohio, US	Power Grid	M	OD	G, P	M	[100]	-
MSICST	H	-	China	4 Different ICS	S	OD	S	H	[101]	-
NIST	H	USDOC	US	4 Different ICS	M	C	S	M	[102]	-
PNNL	H	PNNL	Washington, US	Generic CPS	S	OD	S	L	[103]	-
Queiroz et al.	H	RMIT University	Australia	Water Distribution	R	OD	S	L	[104]	-
SNL Testbed	H	SNL	New Mexico, US	Generic ICS	No	OD	S	L	[105]	-
VPST	H	University of Illinois	Illinois, US	Power Grid	S	OD	S, G	L	[106]	-
Ahmed et al.	P	UNO	Louisiana, US	3 Different ICS	R	OD	S, F, P	M	[107]	-
Blazek et al.	P	BUT	Czech Republic	Power Grid	R	OD	S	M	[108]	-
BU-Testbed	P	Binghamton University	New York, US	Power Plant	R	OD	S	M	[109]	-
EPIX (iTrust)	P	SUTD	Singapore	Electric Power	R	E	S	H	[110]	[111]
HAI Testbed	P	ETRI	South Korea	Power Plant	R	OD	S	H	[112]	-
Lancaster's	P	Lancaster University	UK	Generic ICS	R	C	G, P	H	[113]	-
LICSTER	P	HS-Augsburg	Germany	Generic ICS	R	OS	S, P	L	[114]	[115]
Mississipi Ethernet	P	MSU	Mississippi, US	2 different ICS	R	E	S, P	M	[116]	-
Mississipi Serial	P	MSU	Mississippi, US	Industrial op.s	R	E	S, P	M	[116]	-
PowerCyber	P	Iowa State University	Iowa, US	Power Grid	R	OD	S, P	L	[117]	-
Sayegh et al.	P	AUB	Lebanon	Generic ICS	No	OD	S	M	[118]	-
SGTB	P	INL	Idaho, US	Power Grid	R	NA	S	H	[119]	-
SWAT	P	SUTD	Singapore	Water Treatment	R	C	S	H	[120]	[121]
Teixeira et. al	P	IFET	Brazil	Water Distribution	R	OD	S	M	[122]	-
T-GPP	P	JRC Ispra	Italy	Power Plant	R	OD	S	H	[123]	-
WADI	P	SUTD	Singapore	Water Distribution	R	C	S, P	H	[124]	[125]
Yang et al.	P	QUB	Irland	Power Grid	R	OD	S	M	[126]	-
Zhang et al.	P	University of Tennessee	Tennessee, US	Nuclear Plant	R	OD	S	M	[127]	-
Davis et al.	V	University of Illinois	Illinois, US	Power Grid	S	OD	S	L	[128]	-
DVCP	V	TUHH	Germany	Chemical Process	S	OS	S, F	L	[129]	[130]
Farooqui et al.	V	NUST	Pakistan	Generic CPS	S	OD	S	L	[131]	-
Gas Pipeline	V	MSU	Mississippi, US	Gas Pipeline	S	NA	S	L	[132]	-
Genge et al.	V	JRC Ispra	Italy	Generic ICS	S	OD	S	L	[133]	-
Giani et al.	V	UC Berkeley	-	Generic ICS	S	OD	S	L	[134]	-
GRFICS	V	Georgia Tech	Georgia, US	Chemical Process	S	OS	S, P	L	[135]	[136]
Jin et al.	V	UIUC	Illinois, US	Generic ICS	S	OD	S	L	[137]	-
Koganti et al.	V	University of Idaho	Idaho, US	Power Grid	S	OD	S	L	[138]	-
Lee et al.	V	Ajou University	Korea	Power Plant	S	OD	S	L	[139]	-
Maynard SCADA	V	Queen's University	UK	Generic ICS	S	OS	S	L	[140]	[141]
MiniCPS	V	SUTD	Singapore	Generic CPS	S	OS	S	L	[142]	[143]
Reavers & Morris	V	Georgia Tech	Georgia, US	Generic ICS	S	OD	S	L	[144]	-
RICS-el	V	FOI	Sweden	Power Grid	S	OD	S	L	[145]	-

Table 2.3 – continued from previous page

Name or authors	Category	Institution	Country	Sector	Physics	License	Scope	Cost	Reference	Resource
SCADA-SST	V	KFUPM	Saudi Arabia	2 different ICS	S	OS	S	L	[146]	[147]
SCADASim	V	RMIT University	Australia	Generic ICS	S	OS	S	L	[148]	[149]
SCADAVT	V	RMIT University	Australia	Water Distribution	S	OD	S	L	[150]	-
Singhet et al.	V	C-DAC	India	Power Grid	S	OD	S	L	[151]	-
TASSCS	V	University of Arizona	Arizona, US	Power Grid	S	OD	S	L	[152]	-
VTET	V	SKL-MEAC	China	Chemical Process	S	OD	S	L	[79]	-
Wang et al.	V	Tsinghua University	China	Generic ICS	S	OD	S	L	[153]	-

- **Real Word Representation**: An industrial system must represent a real-world industrial scenario, including all the physical processes related to the environment. Furthermore, the Industrial testbed must include the most common industrial devices installed in the real world ICSs and supporting the most used protocols. Also, it is crucial to consider different versions of devices, knowing their different security features [103, 154]. The testbed should also include the different vulnerabilities that could, however, lead to a bias in the attack strategy vector.

- **Replication in Safety**: The physical processes controlled by ICSs are wide different, ranging from manufacturing processes to critical nuclear plants. The most delicate processes cannot always be replicated in a scaled-down version inside a laboratory. Furthermore, during attacks targeting the process's stability, even the less critical operation can express important safety issues [154].

- **Complexity**: Industrial systems devices can be hard to configure and maintain due to their specificity and because they are designed to perform a precise and unique task. It is also challenging to find IT experts who have the needed knowledge to manage and maintain a complex ICS containing several OT devices. The maintenance requirements must be considered from the early design stages since the increasing complexity can become even more expensive and difficult to manage [154].

- **Cost**: To build physical industrial testbeds, research groups have to deal with building and maintenance costs. Expenses are one of the main reasons why there are not many testbeds available for research, and the ones that exist are generally not easily accessible by everyone. To overcome this problem, virtualized and emulated solutions are relatively diffuse in the field, even if they cannot provide the same fidelity and replication accuracy.

- **Lack of Documentation**: Another challenge in ICS research is the lack of documentation of the existing systems. Companies do not share internal information related to their system's architecture, the devices implemented, or the devices' software version. This is primarily due to the companies' privacy concerns, protection of intellectual proprieties, and security reasons. In fact, if a company discloses the presence of legacy devices with well-known vulnerabilities, it can attract several malicious actors' attention. This absence of documentation made the implementation of effective real-word testbeds difficult. Furthermore, the lack of documentation can be problematic for the in-laboratory testbed. If poor documentation is provided, new researchers who start to work on a testbed might spend much time understanding the system's behavior and components and have a concrete idea. To provide exhaustive documentation, it is essential to write it step-by-step during the testbed building process, avoiding writing it after the testbed is entirely built, which can be difficult and not cost-effective [113].

- **Reproducibility**: Due to the complexity of an ICS, it could be challenging to reproduce the experimental conditions of another research to replicate the results or test other solutions. The differences between the original conditions and the reproduced one can be minimal but, in some cases, can be sufficient to lead to different results. To facilitate the deployment, experiment-management systems can help researchers with the setup and the management of a testbed (e.g., [155]) by using a template or code generation. Moreover, scripts for auto-configuration of an emulated testbed can be offered by developers (e.g., [142]) to simplify the sharing process. However, suppose the testbed is composed of physical processes and components. In that case, it could be difficult to perfectly replicate them since many external variables can influence the system behavior (e.g., the temperature, the pressure) [156].

- **Scalability**: If expanding a simulated or emulated testbed is generally straightforward, doing it with a physical testbed can be challenging. Real devices are expensive, and researchers are not always able to afford them. Alternatives to expand physical processes are Hardware-In-the-Loop (HIL), i.e., mathematical representations of physical processes inserted in the chain. HILs offer great scalability of the system even if generally they are not advisable due to the lack of accurate mathematical models. A cheap way to add new devices is to employ software simulations. Software simulations are cost-effective solutions with the drawback of less precise and reliable physical representation. To provide system scalability and intelligent reconfiguration of all the physical devices implemented,

virtualization and Virtual Local Area Networks (VLANs) can be an excellent solution to be implemented in ICS without any substantial disadvantages [154].

- **Data Collection**: A not trivial aspect of building a testbed is the data access and recording. It is generally a manual process, but it is vital to develop strategies to automate the collection precisely, providing reliability and synchronization between the different data collection points, for example, by introducing a central historian server.

Physical Testbeds

Ahmed et al. [107] presented a physical testbed built at the University of New Orleans, which models three industrial processes on a small scale but by employing real-world equipment such as transformers and PLCs. A small gas pipeline that transports compressed air was built using a pipe fed with an air compressor. A valve regulates the other end of the pipe. Instead, the second system is a power transmission and distribution that carries electricity from power generation sources to individual consumers. This system is composed of a power station and four substations. Finally, the third system developed is a wastewater treatment system composed of sedimentation, aeration, and clarification processes. Each system is controlled by one PLC connected through a switch to a historian and an HMI. This last device makes it possible to visualize and control the systems. The industrial protocols employed are Modbus, EtherNet/IP, and PROFINET.

Electrical Power and Intelligent Control (EPIC) [110, 111] is a high-cost 72kVA electric power testbed that mimics a real-world power system in small scale smart-grid, and it is available for rent. The testbed is shown in Figure 2.7b and it is composed of four stages, namely: Generation, Transmission, Micro-grid, and Smart Home. Each stage is controlled by PLCs connected to a master PLC using switches and then to a SCADAs gateway. The physical process is entrusted to two motor-driven generators, photovoltaic panels, and a battery. Communications occur using the IEC 61850 standard protocol for the electrical substation and automation system that runs over TCP/IP stack. The authors also present false data injection attacks, malware attacks, power supply interruption attacks, and physical damage attacks, together with possible mitigation techniques. The testbed resides at the Singapore University of Technology and Design (SUTD), and it is used to supply power to two other testbeds inside the same institution (i.e., Secure Water Treatment System (SWaT) [120], and Water DIstribution System (WADI) [124]) to create also the possibility for research related to a cascade-connected ICSs. The authors also shared a related dataset, which will be analyzed in Section 2.1.5.

HAI Tesbted (HIL-based Augmented ICS) [112, 157] is an extensive and expensive interconnection of three independent real ICSs coordinated by a real-time HIL developed at The Affiliated Institute of ETRI, Republic of Korea. Emerson's boiler control system, GE's turbine control system, and FESTO's water treatment control system are built-in small-sized by employing components used in industrial environments. The HIL is used to simulate the power plant to combine the three control systems and form an integrated power generation system. In [112] the authors present various physical attacks targeting the pump and the pressure of the boiler system. An expansion of the testbed [157] was built to make it possible to launch also network attacks using tools like Nessus or Acunetix.

BU-Testbed [109] is a physical reproduction of two power generation systems developed at Binghamton University. The first one is composed of an Alternate Current (AC) motor directly coupled to a permanent magnet Direct Current (DC) motor, generating up to 400V. The other one instead contains an AC motor used to drive a 12-volt DC blower motor used to generate electricity. The communication uses the EtherNet/IP protocol. Furthermore, the authors explain some cyber-physical attacks which are practicable on the testbed. These attacks regard different categories: 1) attacks on networks (i.e., MiTM, DNS poisoning); 2) network congestions and delay (i.e., DoS); 3) attacks on controllers, sensors, and drivers (i.e., malicious software injection and firmware modification); and 4) attacks on HMI and programmable stations (malware injection).

Lancaster's testbed by Green et al. [113] at the Lancaster University is a big physical scaled-version of a generic industrial ICS (the testbed does not explicitly the physical processes involved). It is composed of six Manufacturing Zones, a DMZ, and an Enterprise Zone. Each core zone is split at the network level using VLANs. The legacy serial-based communications have been upgraded to IP to reduce the complexity and allow communications with a vast number of ICS devices. The connections are almost all physical, apart from two manufacturing zones connected using 3G, 4G, and satellite communications. The authors are continuously improving the testbed to make it more usable and more complete. Students and researchers of the university use the testbed, but the authors also plan to make it more available for external researchers.

Morris et al. [116] at the Mississippi State University built seven different small physical testbeds for security research and pedagogy purposes. Five of them have communications based on Modbus/ASCII, Modbus/RTU, and DNP3 (henceforth called **Mississipi Serial**) and represent respectively: 1) a gas pipeline used to move petroleum products to market; 2) a storage tank used in the petrochemical industry; 3) a raised water tower used to provide pressure in the

water distribution system; 4) a factory conveyor belt control system, and 5) an industrial blower used to force air through an exhaust system. These five systems are controlled by the same HMI but on different screens. It enables the control of all the systems from the same point and simulates a more extensive system by making them operate simultaneously. The remaining two testbeds are connected through an Ethernet network (and then are called **Mississipi Ethernet**) and include: 1) a steel rolling operation; and 2) a smart grid transmission system. The authors also use the testbeds to generate datasets that are freely available online [158].

Smart Grid Test Bed (SGTB) [159, 119] deployed by Idaho National Laboratory is the world's first full-scale replication of a smart grid, and it is part of the United States National SCADA Test Bed Program. Portions of the power loop can be isolated and reconfigured for independent, specialized testing. As planned in 2017, the authors obtain more funds to expand SBTB with a SCADAs testbed to be installed in the command and control shelter to allow operators to observe, manage, and manipulate test line configurations and record testbed operating parameters. However, to the best of our knowledge, the authors never release updates about the project. This testbed is not an ordinary scaled-down version of real systems. Instead, SGTB is a full-size plant. Even if it represents an impressive and valuable work, unfortunately, students and researchers have limited access to such a facility [117].

SWaT [120, 160] is a six-stage water treatment plant developed by the SUTD represented in Figure 2.7a. One PLC (plus one for backup) controls each stage, and the overall testbed leverages a distributed control approach. Furthermore, through a HMI, an operator can manually control all the system components. In the paper, the authors implemented various attacks to manipulate plant operations. The different attacks leverage different assumptions on the attack model. The testbed is accessible only for collaborations or by renting it. Recently, a python-based software simulation of the testbed was developed and released with open-source code [161, 162]. Also, datasets based on different data collection are openly available upon request. These datasets contain both network and physical packets in normal behavior and with the system under attacks [163]. We present the dataset in Section 2.1.5.

Teixeira et al. [122] implemented an ICS testbed to model a simple water storage tank's control system. The storage tank is equipped with two-level sensors to control the water level. When it reaches the maximum level, the upper sensor sends a signal to the PLC, which turns off the water pump used to fill the tank. At the same time, another pump is activated to draw water from the tank. When the water reaches the lower sensors, a signal is sent to the PLC, which will reverse the two pumps' state to fill up the tank again. The SCADAs system gets data from the PLC using the Modbus protocol and displays them to the system operator through

(a) SWaT Testbed by iTrust of Singapore.

(b) EPIC Testbed by iTrust in Singapore.

Figure 2.7: Two of the Singapore Testbeds.

the HMI interface. To complete the study, the authors tested some attacks such as scanning, device identification, and not authorized read of actuators. By recording SCADAs network traffic for 25 hours, a dataset has been released [164] and will be presented in Section 2.1.5. In 2019, minor improvements of the testbed had been presented [165], such as embedding a turbidity sensor and a turbidity alarm to add analog input to the system.

Turbo-Gas Power Plant (T-GPP) testbed [123] is an experimental platform presented by Fovino et al. at the Joint Research Centre of Ispra (Italy) to perform security research on a SCADAs system. It is a physical testbed that replicates a power plant's dynamics process and its control systems providing additional mechanisms for running and analyzing the system. The testbed is composed of seven different functional elements: 1) Field Network, used to link PLCs with the SCADAs servers, actuators, and sensors; 2) Process Network, which interconnects the different physical subsystems; 3) Intranet, the internal private network connecting PCs and server of the company; 4) Demilitarized Zone, used to separate IT area from OT components; 5) External Network, such as the Internet; 6) Observer Network, a network of meshed sensors to gather a massive quantity of raw data useful for the analysis; and 7) Horizontal Services Network, used for the management of the laboratory. The paper profoundly analyzes such systems' vulnerabilities, highlighting those related to the protocols implemented (i.e., Modbus/TCP and DNP3), and describes various attacks deployed on the testbed: DoS, worm, and malware infection on the process network, phishing attack, and local DNS poisoning. Finally, the authors propose different countermeasures to the attacks.

WADI [124, 166] is a scaled version of a water distribution testbed build by the SUTD to perform security researches. It consists of five stages controlled by three PLC and two RTU, which

can supply 10 US gallons/min of water. The communication happens using Modbus/TCP protocol at Layer 0, while at Level 1 network between PLCs uses TCP over Ethernet instead of RTUs that exploit High-Speed Packet Access using GPRS modem to generate a precise real-world scenario. The authors also implemented different attacks against the testbed by manipulating data from sensors to cut off the consumer tank's water supply. Furthermore, WADI is available to organizations for joint research programs and usage, but a dataset generated upon request is available. We will analyze the dataset in Section 2.1.5.

In 2014, **Yang et al.** [126] proposed a physical SCADAs power grid testbed specifically designed to test their detection approach. At the control network level, the testbed is composed of an HMI, a database to log events and data, a host used to perform the attacks, and different networking components (e.g., protocol gateway, switch, firewall, router). Instead, the physical network is composed of various IED simulated, connected to a real photovoltaic system. The IDS proposed by the authors was installed between the HMI and the Protocol Gateway. It monitors all the incoming connections to the substation and the LAN network through port mirroring.

Zhang et al. [127] presented a security research on a physical process ICS testbed which simulates a two-loop nuclear power system. The primary loop includes a 9kW heater representing the reactor core, controlled by the SCADAs master through an open-loop controller. It also contains a variable speed coolant pump, upper and lower delay tanks, and other instrumentation such as a flow meter and temperature detectors. The secondary loop is composed of valves, a magnetic flow meter, and two temperature detectors. In the same paper, the authors proposed some attacks to the testbed (e.g., MiTM, DoS).

Virtual Testbeds

Davis et al. [128] present a power grid simulated testbed based on a client-server paradigm. The network is emulated using RINSE [167] which allows clients to launch different commands to simulate attacks (e.g., DoS attacks), defense techniques (e.g., filtering), diagnostic tools, device controls, and simulator data. The authors present various attacks, such as DDoS and network overload, comparing the results with and without security measures. To the best of the authors' knowledge, the testbed is not available online.

In [129] the authors present **Damn Vulnerable Chemical Process (DVCP)**, an open-source framework developed for cyber-physical security experimentation based on two models of chemical processes. In particular, the framework includes **DVCP-Tennessee Eastman (TE)** and

DVCP-VAC, two simulated ICS testbed based respectively on Tennesse-Estman [168] and Vacuum-assisted closure (VAC) [169] chemical processes simulated with Matlab. However, for this implementation, the authors did not share any code or further implementation information.

Genge et al. [133] proposed a framework based on Emulab [170] for the emulation of the components and to Simulink [171] for the physical processes simulation. The architecture comprises three layers: the cyber layer containing the regular emulated ICT components used in SCADAs systems, the physical layer providing the simulation of physical processes, and the link-layer to connect the cyber and physical layers through the use of a shared memory region. There are many supported protocols such as Modbus, Profinet, and DNP3. The testbed, implemented in C#, is not available online to the best of the authors' knowledge.

Giani et al. [134] developed a virtual SCADAs testbed for security-related researches purposes. However, this work represents a preliminary study presenting the testbed at a high-level, but without a practical implementation description. At the center of the architecture, there is the SCADAs master station containing the SCADAs server and the HMI. The authors depict various possible attacks (e.g., DoS, integrity attacks, phishing attacks) and suggestions about security mechanisms. To the best of our knowledge, the testbed is not publicly available online.

GRFICS [135] is a graphical and open-source [136] ICS simulation tool based on the Tennessee Eastman process (Figure 2.8). Currently, the testbed is designed for educational purposes and allows only the use of pre-defined functions. The testbed allows running many predefined attacks such as MiTM, Command Injection, False Data Injection, Reprogramming of PLCs (i.e., Stuxnet), Loading Malicious Binary Payload (i.e., TRITON), and Common IT attacks (i.e., password cracking, buffer overflow). Once the attacks are launched, the interface allows monitoring the testbed attacks' consequences, log the process information, and how much cost is wasted through the purge. Finally, the testbed allows the installation of the Snort detector [70] and to customize it with new rules. The communications on the testbed are based on Modbus protocols.

Maynard et al. [140] proposed **Maynard SCADA**, an open-source, scalable framework for deploying a replication of a SCADAs network. The testbed is composed of a collection of scripts used to build and configure virtual machines that, by default, are emulated using Oracle VirtualBox [172]. Furthermore, the paper [140] shows a common metering application using seven virtualized nodes with detailed instructions to replicate it. The instructions also include an accurate description system's requirements and a comparison between some other testbeds in the same document. The framework is entirely open-source, and it is accessible on

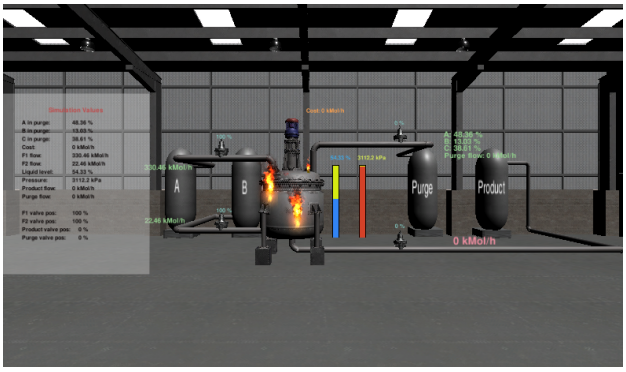


Figure 2.8: Example of GRFICS simulator rendering.

GitHub [141], where are also available some datasets.

MiniCPS [142] by Antonioli and Tippenhauer is a toolkit used to create an extensible and reproducible research environment for network communications, control systems, and physical layer interactions in CPS. MiniCPS is an extension of Mininet [173], a widespread network simulator built around the Software-Defined Networking paradigm that exploits lightweight system virtualization using Linux containers. The authors present an attack scenario of a MiTM attack on a replicated model of the SWaT testbed [120], also providing different countermeasures based on a custom SDN controller. The testbed and its documentation are open-source and available on Github [143].

Morris et al. [132] presented a virtual gas pipeline system (called **Gas Pipeline testbed**) that is a simulation of a testbed previously built. The testbed consists of four components running in different virtual machines: a virtual physical process, a Python-based PLC simulation, a network simulation, and an HMI. The various components communicate through Modbus/TCP over a virtual network and may be connected to real devices. The virtual system allows modeling a pump, a valve, a pipeline, a fluid, and a fluid flow. The models are based on a previous physical testbed [116], allowing to compare measures from the two testbeds. The virtual testbed mimics the physical device's behavior but with some difference in pressure change frequency. Also, the startup process is similar but not identical. The authors present a command injection attack to the virtual testbed, but the resulting behavior is not compared with the physical testbed. To the best of our knowledge, this virtual simulator is not publicly available online.

Reavers & Morris [144] develop an open and complete platform for creating virtual testbeds. The resulting system is highly scalable, and it is possible to install plenty of different virtual devices. The testbed's main components are process simulators, data loggers, and configuration files used to configure virtual devices and connections among them. All the simulations are im-

plemented with Python without adopting off-the-shelf network simulation tools. The process simulator includes four components: 1) a simulator module, 2) a communication interface, 3) an update queue, and 4) configuration files. The simulator communicates directly with the virtual test devices via a "backchannel" to transmit measurements and inputs. Starting from this work, Thornton and Morris in 2015 [174], deployed a similar platform that permits the usage of Simulink [171] instead of Python to simulate the physical processes.

RICS-el testbed [145] is a virtual testbed representing a power system built on top of the Cyber Range And Training Environment (CRATE) infrastructure at the Swedish Defence Research Agency (FOI) [175]. All the hosts of the testbed are run on virtual machines using VirtualBox [172]. Ongoing work is focusing on adding realistic traffic to each segment by emulating users and different scenarios.

SCADASim [148] is a simulator for SCADAs systems created on top of OMNET++ [176]. The testbed is developed to satisfy specific requirements: 1) it allows plug-n-play to create simulations to allow system experts to set up the software; 2) allows connectivity to multiple external hardware or software that can be used to expand the simulator; and 3) supports multiple industry-standard protocols such as Modbus/TCP, DNP3, and the integration of proprietary protocols. For the evaluation, the authors present two simulations: a smart meter and a wind power plant. A DoS attack and a spoofing attack are also deployed on the systems and analyzed in the paper. SCADASim is an open-source project available on Github [149].

SCADAVT [150] is a framework to build a virtual SCADAs model-based testbed designed for research in security field purposes. Two attack scenarios are also presented: a DoS and manipulation of command messages. The framework, which supports real devices' connection, is described in detail, but the source code is not publicly available.

TASSCS (Testbed for Analyzing Security of SCADAs Control Systems) [152] was developed by the University of Arizona mainly to test a novel technique to protect SCADAs systems from attacks, which the authors called Autonomic Software Protection System (ASPS). The authors present various attacks, including spoofing, MiTM, DoS, and data injection. Two of these attacks scenario are also implemented: a DoS attack and a compromised HMI scenario to force a complete network blackout. The protection system under testing was able to detect the two launched attacks.

Virtual Tennessee-Eastman Testbed (VTET) [79] is a simple virtual testbed that simulates a chemical ICS with Matlab. The architecture is based on four components: a physical PLC, a PC used for network communication, and two other PCs simulating the physical process and a PLC. Unfortunately, the testbed is not available online to our knowledge, but the description

Figure 2.9: A figure representing CyberCity testbed.

is quite complete on the paper.

Hybrid Testbeds

CyberCity Testbed [85, 177] is a physical representation of an entire city (Figure 2.9) developed by the SANS Institute to test security measures on the ICS field. It includes a bank simulation, a hospital, a power plant, a train station, a water town, and many other available infrastructures. Nowadays, the testbed is mainly used to teach cybersecurity on ICS as part of the SANS Institute courses and federal agencies to perform security research.

Experimentation Platform for Internet Contingencies (EPIC) by Siaterlis et al. [86] is an innovative hybrid testbed to simulate CPSs based on Emulab [170]. The testbed architecture comprises two control servers, a pool of physical resources used as experimental nodes (e.g., PCs, routers), and a set of switches employed to interconnect the nodes. Physical processes are simulated using Simulink Coder [178] and managed by a software simulation unit. The software part is open-source and freely available online [87] with complete documentation.

EPS-ICS [88] is a framework to implement a hybrid testbed, principally developed by the Technical Assessment Research Lab (CNITSEC) in Beijing, China. The testbed implements a multi-level design approach where Level 3, the corporate network, and Level 2, the supervisory control LAN, are emulated. Instead, Level 1 devices, including Distributed Control Systems (DCS) controllers, PLCs, and RTUs, are real physical devices. Finally, a mathematical model is used to simulate the physical process at Level 0, and it is implemented with Simulink [171]. This approach allows replicating the interactions between the ICS components.

Gillen et al. [89] presented a hybrid replication of the cooling system for Oak Ridge Na-

tional Laboratory's 200-petaflop Summit supercomputer, currently declared the fastest open-science computer in the world [179, 180]. Summit consists of over 4600 nodes and has a peak power draw of 13MW. The cooling system cycles through over 4000 gallons of water each minute. The authors collected 30 days of data from the real Summit cooling system historian to correctly emulate sensors and actuators. Then they used emulation scripts to generate data from the devices. To validate the testbed, the authors compared its behavior with the real Summit supercomputer cooling system. Considering the alerts, logs, and historian data, all the data are replicated accurately.

Hui et al. [90] introduce **Hui Nuclear**, a hybrid testbed modeling a nuclear reactor built at the Center for Secure Information Technologies at the Queen's Univerity of Belfast. The testbed's scope is to generate a realistic network interaction and a simple way to collect network data to be used in the CPS security field. The network architecture is exhaustive and contains all the Purdue model areas, together with firewalls, IDS, and logging services.

HYDRA [91] is a low-cost and open-source physical emulator for critical infrastructures developed at the Università Roma Tre in Italy. It can be used for investigating fault diagnosis, cybersecurity strategies, and testing control algorithms. The testbed is designed to emulate a simple water distribution system's behavior. The authors also present an attack scenario of a data modification attack. The code and all the testbed technical details are open-source and available on Github [92].

Kim et al. [95] proposed a platform to perform cybersecurity exercises for national critical infrastructure protection. The testbed was designed to replicate a realistic ICS environment that matches the characteristics of the Cyber Conflict Excercise (CCE). The paper [95] describes an implementation of the proposed platform, which includes six different critical infrastructures: a power grid, a nuclear plant, a water purification plant, railroad control, airport control, and traffic light control. The system contains two PLCs of different vendors that control some typical actuators (e.g., mechanical relay, magnetic switch, motor).

In [96], **Koutsandria et al.** presented a hybrid testbed for testing a real-time Network IDS. To simulate the ICS environment, the authors employ a combination of simulated and real devices. The testbed is based on Matlab Simulink to simulate the physical and control networks. In particular, the authors model the physical system with Simulink by simulating IED and field devices controlled by a PLC via Modbus in a master/slave communication model. To validate this architecture and its capabilities, the authors also present three attacks (i.e., two network communication alterations and a physical behavior violation) scenario showing the effectiveness of the detection rules.

KYPO4INDUSTRY [97] is a training facility for students based on open-source hardware and software, built at Masaryk University in the Czech Republic. This testbed consists of a laboratory room designed to help computer science students to learn cybersecurity in a simulated industrial environment. Finally, the paper introduces the university's course syllabus that employs the facility, showing the arguments addressed on each of the 13 weeks of the course.

LegoSCADA [98, 99] is a cost-effective hybrid testbed developed at the Universite Paris-Saclay in France. The testbed's conceptual architecture is based on three block elements: the controller, the system, and the sensors. To test the architecture, the authors have developed a test scenario based on Lego Mindstorms EV3 brick [181] which emulates a PLC on a car, a Raspberry Pi [182] to emulate an RTU connected to the vehicle, and a personal computer as a controller. MiTM attacks are deployed on the developed testbed, in particular replay attacks and injection attacks. Moreover, a watermark authentication technique has been tested to stop the attacks with interesting results.

LICSTER [114, 115] is an open-source and open-hardware testbed and was presented at the Hochschule Augsburg in Germany. Its main target is to give students and researchers an affordable system to perform security research with an expense of about 500 euros. Modbus/TCP is the protocol used to enable communication between components. Different attack scenarios on LICSTER are presented and tested. The authors cover widely used threats to levels 0, 1, and 2, such as passive/active sniffing, Dos, MiTM, and manipulation over the network. For each attack, an evaluation is presented containing useful information (e.g., impact, skill level, detection difficulty). Scripts and instructions on the implementation are available on the Github repository [115].

Microgrid [100] is a flexible and adaptable testbed developed by The Ohio State University, composed of a hybrid setup of physical hardware and real-time simulations. The testbed contains Power Hardware-In-the-Loop (PHIL) able to emulate power hardware not installed in the testbed, along with a real-time SCADAs system with an OPNET [183] based real-time System-In-the-Loop communication network simulation system. The testbed is designed to study topics related to smart grids and provide hands-on experience to students.

MSICST (Multiple-Scenario Industrial Control System Testbed) [101] is a hybrid representation of four different ICS scenarios: a thermal power plant, a rail transit, a smart grid, and intelligent manufacturing. Physical processes are always simulated while the control systems are built using commercial hardware and software. Furthermore, in some scenarios, a combination of software simulation and actual physical equipment is used to build a more realistic scenario. MSICST also contains an attacker model and a monitoring network. Some vulnerability

discovery experiments have been done on MSICST, ranging from discovering vulnerabilities on a specific type of PLC to some attacks to known vulnerabilities of S7Comm and Modbus provided by the lack of encryption and identity authentication. Some security measures are presented as well, like a whitelist-based host protection software and a new IDS solution that combines traditional IT system IDS with behavior-based ICS-specific IDS.

NIST developed a cybersecurity testbed for ICS presented in detail in [102]. The testbed is designed to emulate three real-world industrial systems without replicating the entire plant or assembling a complete system. The testbed is available upon request to academia, government, and industry to analyze new technologies. Based on the research on these testbeds, NIST published a long and complete guide to ICS security in 2015 [13].

PNNL [103] by Edgar et al. at Pacific Northwest National Laboratory is a remotely configurable and community-accessible hybrid testbed to support research on cyber-physical equipment. This testbed combines physical, simulated, and virtual components giving considerable implementation flexibility. Furthermore, users can control different areas of the architecture, including the environment (used to simulate the physical process), devices (e.g., PLCs, RTUs), network communication (representing the backbone communication), simulation, and device integration. The testbed is accessible following the indications provided in the PNNL website [184] and using Arion [185] as modeling software.

SNL Testbed [105] is a complex hybrid testbed built by Sandia National Labs in Albuquerque, USA. It contains simulated components (represented using a model in OPNET [183]), emulated nodes (i.e., using real software running on an emulated machine), and physical (i.e., real software running in real hardware) devices. The testbed is presented as a case study used to model a complex scenario, containing: the corporate network (connected to the Internet), a DMZ, a control system network (containing HMI, the SCADAs Server, Engineering Workstation, and Front End Processor), and the field layer (containing sensors, RTUs, and IEDs). The protocols implemented are Modbus/TCP, DNP3, and IEC 60870. Finally, the authors present a security assessment of the testbed considering different threats and attacks such as reconnaissance, resistance to standard penetration tools (e.g., Metasploit [186]), and MiTM.

VPST (Virtual Power System Testbed) [106] of the University of Illinois is designed to be integrated with other testbeds across the country to explore SCADAs protocols and equipment's performance and security. Thanks to its easy integration with real devices and testbeds, VPST has the advantage of having actual HIL and a faithful communication system. The architecture is divided into three main subsystems: the first handles electrical simulation using PowerWorld [187], the second simulates the communication systems using RINSE [167], and

the third includes all the actual devices. The paper presents some example use cases: attack robustness analysis, incremental deployment analysis, and Human-in-the-loop event analysis. However, thanks to its flexibility, the testbed is suited for many different types of research.

2.1.5 ICS Datasets

In this section, we provide a description of the ICS dataset available in the literature, highlighting the key design point and the most interesting and performant IDS applied to them. In Section 2.1.5 we outline the classification method that we use in the following sections, while in Section 2.1.5 we introduce the main requirements and challenges in developing a dataset. In Section 2.1.5 we briefly recall the common evaluation metric for IDS. Then, in Section 2.1.5 we present the datasets offering only physical-level data, while in Section 2.1.5 we describe network-level datasets. Finally, in Section 2.1.5 we highlight datasets containing both the information.

Datasets Classification

Datasets are a collection of data recorded from a testbed or synthetically produces, which can be used to train and test an IDS. Unlike datasets concerning IT systems, which are composed only of network traffic, to characterize an ICS, a dataset must contain both network traffic, representing the communications between the various devices, and the physical processes' measurements.

Datasets are generally shared as csv, arff, or pcap format files, depending on the typology of data collected. An interesting solution introduced by Morris et al. [188] consists of providing also some datasets containing only a subset of the data. They can be used, for instance, to quickly look at the data without downloading huge files or training a preliminary algorithm during the early stages of development.

There are many ways to categorize datasets. For example, Choi et al. in [30] groups datasets based on attack path. In this survey, we decided to divide datasets based on the typology of the collected data. The capturing can contain data at *physical level* i.e., field data such as measures from sensors, actuator, and other physical level devices, or *network level* data, i.e., packet or flow sent in the channel under control. However, datasets can contain both the typology of data, and so they are considered both *physical and network level*. Sometimes, it is possible to find other types of data, like device logs, to better understand the ICS's behavior. To perform our study and provide reliable statistics, we downloaded every dataset and analyzed it reporting the main interesting properties.

Table 2.4 summarizes the main features and statistics of the presented datasets. We reported the following features.

- **Name** of the dataset (or of the authors if a name is not provided);
- **Sector** indicates the field of the source ICS;
- **Data** type provided. Can be
  - *Logs* if logging information of the system during the process are available;
  - *Network* if network traffic data are provided;
  - *Physical* if measurements of sensors and actuator states are available;
- **Time** provide an approximation of the duration of the recording;
- **Entries** indicates an approximate number of entries contained in the dataset. In the case of datasets containing different versions, the most used or the most recent is considered;
- **Reference** includes a reference to a description of the dataset;
- **Resource** indicates a webpage in which the dataset is downloadable or information about how to retrieve it are available;
- **Attacks** specified the categories of attacks contained in the dataset, if any. Can be Reconnaissance, Replay, MiTM, DoS, Injection, or Others which contains less used categories. More information about the attacks are presented in Section 2.1.3.
- **%** indicates the percentage of data under attack on the total entries, if any;
- **Format** indicates the format of the files containing the capture. Can be:
  - `pcap` is a widely used format containing network packets;
  - `csv` is an extension for files containing Comma Separated Values;
  - `log` contains textual logging of events;
  - `xslx` is a format for spreadsheet files;
  - `arff` is a format used to save data for databases in a textual format. It is generally used with Weka [189];

  - `inp` contains data of emulations. Generally, it is used with Epanet [190].

- **IDS** contains a reference to the best IDS available in literature applied on the testbed at the best of authors knowledge;

- **F1**, **Acc**, and **Pr** represent the evaluation metrics of the IDS specified, according to Section 2.1.5.

The detection algorithms selected are implemented on the whole dataset and not on a fraction of it. Furthermore, the selection does not take into consideration the rank of the publication venue of the paper. For some datasets and IDSs was not possible to obtain all the information since the related paper does not provide exhaustive information. Thus, the degree of depth of analysis may not be the same for all work.

Datasets Challenges & Requirements

There are several challenges in generating a valuable dataset. Therefore it is fundamental to create it by following a suitable methodology and keeping in mind the design requirements. Gomez et al. [63] described a framework useful to generate reliable anomaly ICS datasets to be employed in anomaly detection tasks. Firstly, it is important to select a priori, one or more attacks that will be implemented. To do so, researchers must know the main protocols used in the field of interest, discover the related threat, and design attacks according to the related vulnerabilities. Then, attacks can be deployed, carefully choosing the nodes affected, each attack's duration, and its starting time. Finally, it is possible to capture network packets and/or data from sensors and actuators: it is essential to define the data capture duration, and the sampling frequency and smartly choose the collecting point. Generally, the latter should be a central node of the system. The last step is the final dataset generation. To generate the dataset to release, it is important to carefully choose the features useful to describe the system under consideration. The behavior of the system can be represented at packet-level, flow-level, or physical-level data.

The **deployment of attacks** in datasets is probably the most challenging phase. In fact, if not accurately performed, the attacks generated can lead to an inaccurate system representation or bias in the detection methodology. There are principally two ways to generate attacks. The first and most accurate one is to attack the testbed in real-time, recording the corresponding network traffic or the ICS's physical state. Another strategy is to insert synthetic malicious

data, a posterior, in a dataset with regular operation. However, this strategy could lead to inaccuracies and may not accurately represent the real system behavior response. In fact, if we want to inject packets on a dataset with normal operations, we must consider all the complex cascade relations of the systems. By breaking these relations, we would leave a trace that an IDS can exploit to detect anomalies, creating detection bias. Since this property is not present in real systems, the IDS will miss most of the attacks in physical environments, reducing the detection generalization in other systems. It is one of the main problems of Lemay et al. dataset [191], which uses tools such as Metasploit [186] to inject malicious traffic.

Another critical concern causing the lack of available datasets from real environments (i.e., ICS of companies) is related to the collected **data's privacy**. In fact, companies may be reluctant to share their internal configurations, intellectual property, or proprietary protocols. Moreover, giving the public access to industrial site data may allow malicious users to identify vulnerabilities and exploit them to attack the company. As a result, many datasets are generally generated from scale-down testbeds and a few real ICS environments.

Since many intrusion detection techniques are supervised, a complete dataset must provide **labels** indicating normal or abnormal data. Furthermore, labels are essential as ground truth for evaluating detection performances during the test phase. However, the labeling process is not always straightforward. For example, some attacks can move the system in abnormal behavior after a long time the malicious packets have been sent. In this scenario, the data labeled as malicious should start when the actual attack starts or when the system's behavior starts to be compromised? An analysis of this problem can be found in [192] and [191]. In both cases, the solution could raise a problem in the ground truth. Therefore, there is no right or wrong answer to this question. It depends on the context and the attack type, but it must be specified in the dataset's documentation to allow researchers to act accordingly.

Evaluation Metrics

In this section, we briefly recall the metrics used to evaluate the performances of the detection algorithms. According to the literature, the most common metrics are Acc and F1. They are defined as follows.

- **Acc**: represents the fraction of correct predictions of the model under consideration. In the binary classification case, the accuracy is defined in terms of positives and negatives

samples classified as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{2.1}$$

where TP = True Positives, TN = True Negatives, FP = False Positives, and FN = False Negatives.

- **F1**: is a metric used to evaluate a classification, defined as the harmonic mean between *precision* and *recall* as follows:

$$F1 - score = 2 \cdot \frac{precision \cdot recall}{precision + recall}, \tag{2.2}$$

where the *true negative rate*, or precision is:

$$precision = \frac{TP}{TP + FP}, \tag{2.3}$$

while the *positive and negative predictive values*, or Recall (Re), is:

$$recall = \frac{TP}{TP + FN}. \tag{2.4}$$

Physical Level

BATADAL (BATtle of the Attack Detection ALgorithms) [193, 194, 195] was a design challenge aimed at the creation of an attack detection algorithm. Every participant was provided with three datasets containing observations of a simulated C-Town network [196], a real-world, medium-sized water distribution system operated through PLCs and SCADAs systems, which allows modeling the hydraulic response of a water distribution network under attack. The dataset is free and available in csv format. There is also available an inp file that can be used with EPANET2 to simulate the system. A new dataset version is also available at [197], contains sensor readings without concealment, and is discussed in [198].

Table 2.4: Summary of datasets presented in the literature. The **Data** type indicated as L: Logs, N: Network; P: Physical. **Times** of recording are estimations and measure units are h: hours, d: days, m: months. **Entries** numbers are estimations, too. We denote the **Attacks** launched during the recording as RC: Reconnaissance; RP: Replay; M: MiTM; I: Injection; D: DoS; O: Others. The **%** column indicates the percentage of data under attack with respect to the whole dataset. File **Formats** are indicated as P: pcap; C: csv; L: log; A: arff; I: inp; and X: xlsx. *: the version of WADI dataset considered is the one dated November 2019; the one of SWaT dataset is instead A1 dated 2015, the most used one as the best of the authors' knowledge.

Name	Sector	Data	Time	Entries	Ref.	Res.	Attacks	%	Formats	IDS	F1	Acc.	Prec.
D5: Energy M.S.D.	Energy Manag.	L	30d	6M	-	[158]	-	-	C	-	-	-	-
QUT_DNP3	Power Grid	N, L	40d	31M	[64]	[199]	RC, RP, M, I, O	~0.01	P, L	-	-	-	-
QUT_S7Comm	Mining Refinery	N, L	17.5h	2M	[200]	[201]	M	~10	C, L, P	-	-	-	-
4SICS	Generic ICS	N	46h	3M	-	[202]	unk	unk	P	[203]	~1	~1	-
CyberCity Dataset	City	N	16d	170K	[85]	[204]	I, M, D, RC, O	16.58	P	-	-	-	-
D2: Gas Pipeline	Gas Pipeline	N	-	400K	[205]	[158]	I	0.97	C	[205]	0.75	-	0.75
D3b: Water S. T.	Water Storage	N	-	230K	[188]	[158]	RC, I, D	27	A	[206]	0.981	0.981	0.981
D4: New Gas P.	Gas Pipeline	N	-	270K	[132]	[158]	M, I, O	21.86	A	[206]	0.988	0.988	0.988
Electra Modbus	Power System	N	>12h	16M	[63]	[207]	RC, I, RP	5.2	C	[63]	0.987	-	0.988
Electra S7Comm	Power System	N	>12h	387M	[63]	[207]	RC, I, RP	1.42	C	[63]	0.996	-	1.000
HVAC_Traces	HVAC	N	7d	40M	[208]	[209]	-	-	P	-	-	-	-
Lemay Covert	Breakers	N	6.55h	1.6M	[191]	[210]	Covert Channel	100	P, C	-	-	-	-
Lemay SCADA	Breakers	N	~6h	900K	[191]	[210]	RC, I, O	3.29	P, C	[211]	1.000	1.000	-
Modbus SCADA #1	Liquid Pump	N	~24d	41M	[212]	[213]	M, D	4.81	P	[214]	0.775	0.812	0.964
S4x15 ICS	Generic ICS	N	<1h	310K	-	[215]	unk	unk	P	[203]	~1	~1	-
WUSTL-IIOT-2018	Water Control	N	25h	7M	[122]	[164]	O	6.07	C	[122]	1.000	1.000	1.000
D1: Power System	Power System	P, L	-	78K	-	[158]	I, O	71.02	C, A	[216]	0.955	0.950	0.980
EPIC Dataset	Generic ICS	P, N	4h	5K	[110]	[217]	-	-	P, C	-	-		
QUT_S7 (Myers)	Generic ICS	P, N	8.5h	15M	[218]	[219]	I, M	<0.001	P, L, X	[218]	0.744	-	0.727
SWaT Dataset*	Water Treatment	P, N	11d	950K	[163]	[217]	I, O	5.76	P, C, X	[220]	0.889	-	0.919
BATADAL	Water Distribution	P	22m	13K	[193]	[195]	RP, M, O	1.69	C, I	[221]	0.970	0.989	0.987
HAI Dataset	Power Plant	P	10d	1M	[157]	[222]	RP, M	1.83	C	[223]	0.780	-	0.950
WADI Dataset*	Water Distribution	P	16d	950K	[124]	[217]	M	1.04	C	[220]	0.804	-	0.908

The challenges' participants developed different algorithms for intrusion detection ranging from Random Forest (RF) to Recurrent Neural Network. Housh and Ohar [221] achieved the best result by proposing a model-based fault detection approach that employs a simulator to generate benign data and then compare them to the available SCADAs readings to detect anomalous behaviors. This approach is composed of three main phases: 1) available SCADAs data are used in a Mixed-Integer Linear Program to estimate the water demand in each node; 2) EPANET simulator is used to generate reference values, which are used to produce simulation errors when compared to actual readings; and 3) a multi-level classification approach is implemented to classify the obtained simulation errors into events and normal conditions. The result shows a Pr of 0.987, and Acc of 0.989, and an F1 of 0.970. In [224] Kravchik and Shabtai present a detection approach base on under-complete Autoencoder in the frequency domain, which could reach an F1 of 0.937, which is high, considering the simplicity and non-specificity of the used algorithm. The presented paper was applied to the first version of BATADAL. Finally, we must consider that BATADAL is synthetically generated. Therefore this dataset does not suffer from significant noisy problems, making anomaly detection easier.

Morris et al. presented different ICS datasets, which are available online [158]. Each dataset's name is labeled with a number from 1 to 5 and the involved industrial sector.

Dataset 1: Power System Datasets are a collection of three datasets provided by Morris et al. [158, 225] containing the same data but with various labels. One dataset has binary labels (i.e., Normal and Attack). The second dataset has three-class labels (i.e., Attack, Natural, and No Events), which identity as "Natural" events single line-to-ground (SLG) faults and line maintenance and as "No Events" the normal operations. Finally, the third dataset includes 41 different labels containing more information about the attacks and various events. In particular, one label is reserved for "No Events" (i.e., Normal Operation), eighth labels contain different classes of the "intensity" of the "Natural" samples previously mentioned. The remaining 32 labels identify different attacks such as Data Injection, Command Injection, and Relay Setting Change. All the details about the labeling process are available in the readme file at [158]. Physical measures and logs from the control panel, relays, and Snort captures are collected from a physical testbed containing two power generators, four IEDs that can switch four breakers, all connected through switches and routers. Data are provided as a csv file (for the first two datasets) and ARFF format (for the third dataset).

Different interesting IDSs are implemented on these datasets [225, 216, 226, 227]. In [216] different machine learning-based anomaly detection algorithms are tested against the three datasets. The most performant method was JRipper algorithm [228] together with ADA Boost

(ADA) [229] to improve the performance. The algorithms were trained on voltage and current measurements of the four synchrophasors (29 features each). This information was combined with information on frequencies, impedances, and status flags of relays for a total of 128 features. Results show an F1, Re, and precision almost always greater than 0.8, with a peak of F1 of 0.955 in the three-class dataset. Also, Acc was always greater than 0.85. Even if the authors did not include any numerical results based on common metrics, it is worth mention another approach presented in [225]. The authors presented a specification-based intrusion detection framework, which is tested in the discussed dataset. They implemented a Bayesian network to model different threat scenarios. The authors' purpose was to build a network with a unique path for each threat scenario. In other words, each scenario must be described as a sequence of system states, actions, and events that uniquely identify it. For each threat identified, the system collected related measurable variables and events. Then, each scenario is divided into actions that cause the system state transition. Finally, the Bayesian network is built on an independent path of states, computed for each threat. An IDS was implemented starting from the Bayesian network obtained, which reads states and logs to track the system states. The obtained IDS can classify ten different scenarios containing both faults and attacks by monitoring the state transitions, with different precision based on the relay location.

Dataset 5: Energy Management System Data [158] is a large anonymized log collected by an Energy Management System in a utility in the United States of America.

The dataset's csv contains the timestamp and ID of each event, the SCADAs category (i.e., information of the type of event), each device type, the event message, the priority code, the name of the substation, and the area of responsibility (i.e., the controlling authority). Data are collected in a period of 30 days. Since the dataset contains only normal operations, no attacks are provided. For this reason, to the best of the authors' knowledge, there are not IDS implemented on this dataset.

HAI Dataset (HIL-based Augmented ICS) [230, 157, 222] is a collection of physical data from three physical control systems (a GE's turbine, Emerson's boiler, and a FESTO's water treatment systems). Data were sampled every second in 59 points representing the variables measured or controlled by the control system. Basing on the GitHub repository [222] (which currently differs from [230]), the data collected contains seven days of normal system behavior, a day with 20 different attack scenarios on each control loop, and two days with 14 attacks on multiple control loops, for a total of 10 days of capturing. Totally, there are around 1 million samples, 1.83% of which are labeled as under MiTM attacks, in particular relay and modification attacks. However, all the attacks are deeply explained in [230]. Data are provided in a csv

format with a document that accurately depicts the testbed architecture and the dataset's data.

Due to the novelty of the dataset, released in 2020, there is a lack of IDS implemented on this dataset. However, in [223] the authors present an anomaly detection strategy based on clustered deep one-class classification. It is an unsupervised approach that combines clustering algorithms with Deep Learning (DL) models. In particular, K-means were applied for clustering on the training set. Then, different types of neural networks (e.g., Deep Neural Network (DNN), Convolution Neural Network (CNN), Recursive Neural Network (RNN)) were implemented to predict the clusters and return softmax values classified with the iForest algorithm. Currently, on the HAI Dataset, the higher precision is achieved using DNN as cluster predictor (0.95) while the overall higher scores are obtained with CNN as cluster predictor (F1: 0.78; Pr: 0.78). To complete the research, the same algorithms are tested on another popular dataset, SWaT [163], showing the best results with the same algorithms (i.e., CNN and DNN).

WADI [124, 125] is a dataset with data collected from WADI, a water distribution testbed, created as an extension of the SWaT testbed [120]. The system comprises three subsystems: a primary grid, a secondary grid, and a return water grid. It is also able to simulate water consumption following time-varying demand patterns. The dataset collects 16 days of continuous operation: 14 under regular operation and two days within an attack scenario (a total of 15 attacks). The adversary aimed to cut off the water supply to the consumer tanks. In the attacker model, the adversary has remote access to the SCADAs system. The data recorded represent the state of all the 123 sensors and actuators connected using Modbus/TCP protocol. The dataset is free upon request [217], and it is provided as csv files.

There are many IDSs designed and tested on WADI Dataset in literature. MAD-GAN [231] is an unsupervised multivariate anomaly detection method based on Generative Adversarial Networks (GANs). This method uses a generative model to create a fake time series and a discriminator to distinguish between normal and abnormal data. A peculiarity of this work is that, instead of considering each data stream independently, the framework considers the entire variable set concurrently to capture the latent interactions among variables. To do so, the authors implement a sliding-window approach to divide the multivariate time series into sub-sequences. On WADI, MAD-GAN obtains a precision of 0.53 and an F1 of 0.62. Better results were achieved by Kravchik and Shabtai [224] which obtain a Pr of 0.83 and an F1 of 0.75. They employ an Autoencoder with sequences of length 7 in the time domain. With respect to SWaT [163] and BATADAL [193], the authors also mention that it was impossible to apply the AutoEncoder on the frequency domain because most of the features do not have a clear dominant frequency. However, the best results on WADI were obtained by DAICS [220], a

deep learning solution for anomaly detection in ICSs. The authors propose a 2-branch feature extraction framework. The wide branch, containing only one fully connected layer, is used to memorize the normal state of sensors and actuators. Instead, the deep branch comprises two fully connected layers between two convolution layers and provides the generalization degree required to handle events not covered in the training set. Moreover, DAICS introduces the *few-time-steps algorithm* which can be used to efficiently reconfigure DAICS in a production environment when operators encounter false alarms. DAICS can achieve a Pr of 0.919 and an F1 of 0.804 on WADI.

Network Level

CyberCity Dataset [85, 177, 204] is a dataset collected by the SANS Institute from their own ICS CyberCity testbed. CyberCity testbed is a complete simulation of an entire city containing a bank, a hospital, a power plant, and many other generally available components in a small town. There is also a tabletop scale model of the city, which shows an electric train's behavior, a water tower, and a miniature traffic light. A pcap file is freely downloadable online [204] containing over 170k network packets recorded as a dataset for the Holiday Hack cybersecurity challenge in 2013. The data are unlabeled, but in [85] the authors estimate that about 16% of the data is under attack. Various attacks are included, such as scanning, information disclosure, command injection, MiTM, and DoS. The ICS components use Modbus/TCP, EtherNet/IP, and NetBIOS as communication protocols. For each attack presented, some preventative measures are proposed and evaluated. Some examples are awareness training, system patching, IDS, or anti-virus, but it is remarked that neither one is 100% effective. It is worth noting that, at the best of the authors' knowledge, there is no precise and official documentation of the dataset provided by the SANS Institute.

Dataset 2: Gas Pipeline Datasets [158, 205] contains a collection of labeled Modbus/RTU telemetry streams from a gas pipeline system in Mississipi State University's Critical Infrastructure Protection Center [116]. Each stream is composed of some selected features, including, for instance, an identification bit to discriminate between command and responses, states of components, length of data, and physical measurements. The authors include different command injection and data injection attacks, alongside some data in normal behavior. The dataset contains about 397k samples, divided into csv files with a name indicating the particular attack. The dataset also includes a feature to identify the samples that are effectively part of an attack, with information about the attacker's action in the particular moment. The total percentage

of samples with abnormal behavior is 0.97%. Unfortunately, the dataset does not include each sample's timestamps, making it impossible to analyze timing information.

The dataset was used to test different machine learning algorithms as a discriminator of malicious RTU transactions to detect the deployed attacks [205]. Features are derived by analyzing each raw packet individually to extract the protocol command values. K-Nearest Neighbors and RF are the two algorithms that provided better results across all the attacks, with a Re/Pr of 0.75 or higher for five of the seven attacks. More in detail, the most problematic attacks were burst values (i.e., sending multiple successive pressure values, faster than the data display rate, to the operator interface) and setpoint value injection (i.e., the attacker sends false pressure values equal to the setpoint). Yüksel et al. [232] formally describe a user-understandable framework with effective anomaly detection techniques for ICSs. The test implemented using Modbus/RTU employs the Dataset 2: Gas Pipeline by dividing the attacks into scanning, illegal values, timing, and illicit command. The features extracted were only related to the ICS protocol fields of each packet, such as the type of command, location, or address the host is accessing. To improve results, the authors also performed a feature selection by excluding features with low importance (e.g., incremental fields). The results are highly variable depending on the trade-off between the detection rate and the false positive rate. However, by fine-tuning the algorithm, it was possible to achieve a detection rate of 0.9991 and a false positive rate of 0.001.

Dataset 3: Gas Pipeline and Water Storage Tank by Morris et al. [188, 158] are two different datasets from physical testbeds containing both physical data field and network traffic. The first comprises data deriving from a gas pipeline, while the second contains data from a water storage tank. Both the datasets come from testbeds at the Mississipi State University's Critical Infrastructure Protection Center [116] and are shared as ARFF files. A bump-in-the-wire approach was used to capture data logs and inject attacks in Modbus communication in both cases. The implemented attacks are reconnaissance, response and command injection, and DoS. They cover around 27% of the total data. The authors also provide two short datasets created using 10% of the complete datasets, suited for rapid tests during the preliminary IDS development phases. As explained in [158, 132], the gas pipeline dataset contains unintended patterns that cause some algorithms to identify attacks and non-attacks in unrealistic ways easily. Therefore, we do not report this work in the corresponding dataset tables. Instead, we consider the second version of this dataset, called Dataset 4: New Gas Pipeline.

Dataset 4: New Gas Pipeline [132, 158] is a new version of the Dataset 3: Gas Pipeline dataset. This version was proposed to fix dataset problems causing machine learning algorithms

models that do not match real system behavior and lead to overly optimistic classification accuracy. In this version, the authors implement 35 attacks and precisely document them in the paper and the dataset. The dataset includes different labels for each attack, which cover 21.86% of the capture. Like the previous version, the protocol used is Modbus and data are available as an ARFF dataset containing both physical data and information about the network packets.

D3 and D4 datasets are widely used in the study of IDS for ICS. Feng et al. [233] presented a multi-level anomaly detector using package signatures and Long Short-Term Memory (LSTM) networks. The detection architecture provided is composed of two-level. First, a packet-level anomaly detector based on a Bloom Filter is applied; second, the first-level not-anomaly data are used as input to a stacked LSTM neural network model time-series level anomaly detection. The anomaly detector was tested on Dataset 4: New Gas Pipeline using two LSTM layers of 256 nodes, each achieving a Pr of 0.94, Acc of 0.92, and an F1 of 0.85. The most problematic attack to be detected was the injection of malicious state commands for which a Gaussian Mixture Model performed better. Demertizis et al. [206] proposed the Spiking One-Class Anomaly Detection Framework (SOCCADF), which employs the advanced evolving Spiking Neural Netowork (eSNN). eSNN is a modular connectionist-based system that evolves its structure and functionality in a continuous, self-organized, online, adaptive, and interactive way using incoming information. The framework is supervised and was tested on both the Dataset 3: Water Storage Tank (Pr 0.981; Acc 0.981; F1 0.981) and the Dataset 4: New Gas Pipeline (Pr 0.988; Acc 0.988; F1 0.988). The same authors adopted eSNN on GRYPHON [234], which simplifies the validation mechanisms to work in a semi-supervised way, getting as input only data in standard behavior (i.e., labeled as normal packets). This approach was able to get a Pr of 0.980, an Acc of 0.980, and an F1 of 0.980 on the Dataset 3: Water Storage Tank, while a Pr of 0.975, an Acc of 0.977, and an F1 of 0.970 on the Dataset 4: New Gas Pipeline. Another interesting work is the metaheuristic approach by Mansouri et al. [235]. In this work, the authors provide an anomaly detector based on neural networks with a pre-processing step able to act with a different algorithm based on the packet's delay to have as little impact as possible on the real-time communications. When computational speed is required, computationally efficient Evolutionary System [236] optimization is used. Instead, a more accurate but computationally expensive Grey Wolf optimizer [237] is used if with higher latency scenarios. A neural network is then used to detect malicious data with an accuracy up to 98% on the Dataset 4 New Gas Pipeline.

Electra [63] dataset was obtained from a real scenario of an electric traction station used in the railway industry. Electra is composed of 5 PLCs, a SCADAs system, a switch, and a fire-

wall. All the communications between the components implement Modbus and S7comm over TCP/IP with a master-slave model. There are two different datasets, one for each communication protocol. The implemented and labeled attacks are false data injection, replay attack, and reconnaissance attacks in both cases. The attacks were deployed with a new device attached to the network with a MiTM configuration. In both **Electra Modbus** and **Electra S7comm** datasets, the capture lasts about 12 hours in which 94% and 98% of the data are in normal condition, respectively. The data amount is enormous, containing 387M entries for S7Comm (36.8GB) and 16M for Modbus (1.5GB). The two datasets are freely available on the web [207] in csv format.

Together with the datasets' presentation, Gómez et al. provided an implementation of the main algorithms used for anomaly detection. The authors implemented both supervised and semi-supervised algorithms (i.e., One-class Support Vector Machine (SVM), Isolation Forest, SVM, RF, and Neural Network). Features were obtained by packet inspection and by considering only the control protocol fields such as MAC and IP addresses, timestamps, errors, and the application data. On Electra Modbus, a simple supervised RF with 200 estimators was sufficient to achieve a Pr of 0.988 and an F1 of 0.987, while a single layer supervised Neural Network with 128 neurons was able to reach a Pr of 0.9999 and an F1 of 0.996 on the S7Comm version. On the other hand, the semi-supervised OCSVM performed properly on both the dataset, reaching 0.996 of Pr in Electra S7Comm. In the successive year, the same authors proposed SafeMan [238], a framework to manage both cybersecurity and safety in the manufacturing industry. It is composed of a set of applications and services used to monitor and analyze the industrial process in real-time. SafeMan is based on Edge Computing (EC) to achieve low latency and fast deployment of applications and services. Furthermore, EC allows performing the necessary computing tasks close to the manufacturing activity or the network edge. The framework contains several components to assist the deployment, and the risk assessment, together with the cyber threats detection application proposed in [63]. A different and innovative approach was introduced by Li et al. [239] who design an anomaly detection method based on cross-domain knowledge transferring. Features are extracted from each packet. In detail, for each frame, the authors derived nine basic features (e.g., connection duration, protocol type, connection status) and 15 content features (e.g., number of failed login attempts, number of access to the control). The authors employ the TrAdaBoost algorithm to train a neural network using not only a part of the data of the Electra Datasets but also employing data from different domains, both from other ICS (e.g., SWaT Dataset [163]) or other CPS fields (e.g., KDDCup99 Dataset [240]). Then, they compared the error rate with respect to

a standard SVM and a standard LSTM, showing better results, especially when employing a small fraction (< 10%) of the Electra Dataset in the training phase.

HVAC_Traces by Ndonda and Sadre [208, 209] is a dataset recorded on a Heating, Ventilation, and Air Conditioning (HVAC) system powered by Honeywell and used to provide thermal comfort and acceptable indoor air quality on a university campus. The Building Management System is fully automated, and it is suited to monitor from 15 to 20 structures, each containing different PLCs and RTUs. Operators can access the system through the HMI. The protocols implemented are proprietary (e.g., DCE/RPC, NetBIOS, S7Comm) and use TCP/IP at the transport layer. The data capture was produced using *tcpdump*, at two routers via port mirroring. To obtain an accurate timestamp on each packet in two separate recording points, the authors synchronized the clocks using Network Time Protocol [241]. However, it was not sufficient to ensure good accurate timing. To overcome this problem, the authors introduced a correction factor calculated using ad-hoc ICMP messages sent periodically on the network. The anonymized dataset is publicly accessible in pcap files, where each file contains one hour of traffic. In total, there are about 7 days of collected data in normal conditions, without any attacks. Since the dataset does not contain attacks and is a novel collection, there are no IDS tested to the best of our knowledge.

Lemay et al. [191] present a dataset of a SCADAs network, also called **Lemay SCADA**, virtually implemented with SCADAs Sandbox. The simulations contain different MTUs and controllers connected with the Modbus/TCP protocol. The attacks are generated with an infected machine that launches various exploits to infect other devices. Then, the compromised machines launch different attacks by leveraging Metasploit [186] (e.g., Malware Injection, Reconnaissance). The authors give particular attention to the labeling process and to maintain normal intra-packet time properties. The captured data are divided into various collections with an explicit name indicating the types of implemented attacks. The authors also implemented a cover-channel attacks dataset presented as **Lemay Covert**. In these attacks, the least significant bit of the Modbus packets is used to carry information. To the best of our knowledge, this is the only available dataset containing side-channel attacks. Unfortunately, none of the attacks were designed considering Modbus protocol vulnerabilities. Instead, they are implemented with Off-the-Shelf Tools (i.e., Metasploit [186]). Data collection lasts about 6.25h, and the samples labeled as attacks are about 0.15% of the total for the first attack, while the covert channel packets are present in the whole capture. The datasets are shared in both pcap and csv format.

Schneider and Böttinger [242] proposed an unsupervised anomaly detection framework.

They employ deep autoencoders with pipelining parallel processing strategies to speed up the training. While the proposed framework performs well on the SWaT dataset [163], it shows very different results depending on the attack type when applied to the Lemay SCADA dataset. In particular, to correctly detect an attack, the framework requires a minimum duration of it. For attacks lasting longer than the minimum threshold, the Pr reaches 100%. Anton et al. [211] implemented different standard classification machine learning algorithms on Lemay SCADA. The authors extract 14 basic features from the packets and nine additional features derived from timing and frequency information. Algorithms are tested on three different batches of packets resulted from merging different Lemay SCADA datasets. Both RF and SVM result in an F1 and an Acc greater than 0.999 with all the batch, while k-means clustering report the lowest results. In a follow-up work of Anton et al. [243] data are considered as time series. Each second of network traffic was aggregated into a single data point. Three different algorithms were implemented to detect anomalies inside the three batches of captures defined in [211]. The first algorithm implemented was Matrix Profiles, and it performs well on data with periodic characteristics, requiring only one hyperparameter. Second, the Seasonal ARIMA-process performed well on periodical data and is more resistant to noise but requires a more complicated tuning of the three hyperparameters. Finally, the authors implemented LSTM, which requires a high training effort compared with the other two light-weight approaches. They tested the algorithms on a subset of the Lemay SCADA datasets containing seven attacks divided into three categories (fake command, executable upload, file moving). The attacks are almost all correctly classified with every algorithm. With LSTM, the accuracy is always greater than 0.90, while the F1 is really variable based on the threshold selection methodology.

Modbus SCADA #1 [212, 213] by Cruz et al. is a dataset containing data recorded from a small physical testbed simulating a liquid pump. The testbed comprises an HMI, an Adruino-based RTU, a PLC, a Variable-Frequency Drive, and a 3-phase motor. The protocols used are Modbus/TCP and Modbus/RTU. Data are divided into subfolders based on the attack deployed. Moreover, each pcap file is named with an intuitive strategy that includes the duration of both the capture and the attack. The attacks implemented range from MiTM to different flooding types: ping flooding, TCP SYN flooding, and Modbus Query flooding. All these flooding attacks are aimed at the generation of DoS. The data recording lasts for 24 days, containing 4.81% of data flagged as under attack.

This dataset was used by Radoglou et al. [214] to test an IDS. Firstly, the authors present an expansion of Smod [244], a penetration testing tool for Modbus/TCP, to enable the generation of DoS, MiTM, and replay attacks. Then, they deployed an IDS to detect DoS attacks and a

server for machine learning offloading computation. To train and test the models, the authors employed CICFlowMeter [245] to extract 83 features from each Modbus packets flow. Among the various algorithms tested, AdaBoost [229] and RF achieve the best results with a Pr of 0.96, an Acc of 0.81 and an F1 of 0.77.

QUT_DNP3 [64, 199] is a dataset presented in the Ph.D. dissertation of the author. The dataset contains data collected from a small section of a transmission substation SCADAs network. The testbed involves GOOSE and DNP3 protocols, enabling the communication between the Master, the Slave, the IED, and the attacker machine. All the communications pass through an industrial switch. The attacks are categorized into six categories: Injection, Flooding, Masquerading, Replay, MiTM, and all attacks. Each category also contains Reconnaissance packets. The attacks are launched by an attacker machine, which also generates a log providing information about each attack sequence's start and end. Each dataset file has a different duration based on the attack frequency during the capture creation since the authors implement a random time between two attacks. Moreover, the dataset is divided into two categories based on the attack frequency: frequent attacks (i.e., approximately an attack every half an hour) and infrequent attacks (i.e., approximately an attack every random time between one and four hours). For each frequency category, the authors provide two datasets, respectively, for training and testing. Furthermore, a control dataset with only legitimate communications (i.e., without any attacks) is available, and it covers 24 hours of recording. In total, the dataset contains 40 days of recording. It is worth mention that the labeling process was performed with particular care since it was the main topic of the thesis work. The dataset is available on Github [199]. However, to the best of our knowledge, there are no IDS tested on this dataset.

QUT_S7Comm [200, 201] developed by Rodofile et al. is an open-source dataset collected in a three time-based sub-processes testbed of a mining refinery plant. The plant testbed is composed of one Siemens PLC acting as Master and three PLCs actings as slaves, all connected with a switch to an HMI and communicating using S7Comm protocol. The attack dataset comprises 9 hours of data and 64 attacks from 13 different possible typologies. Data are provided with pcap files and four process logs: a master log, a conveyor log, a tank log, and a reactor log. The labels of the attack samples are contained in separated csv files. The control dataset comprises 8.5 hours of network traffic and process log data, with 32 different processes. This dataset's peculiarity is the particular division of the network traffic in separate files based on the node capture perspective: a file collected from the attacker's point of view, one from the HMI, and one from the master PLC. This particular composition could be initially complex to use, but on the other hand, it provides higher flexibility with respect to datasets with the entire

capture. The dataset is available on Github [201]. To the best of our knowledge, there are no IDSs implemented on this dataset.

4SICS [202] is a pcap dataset collected by Netresec from an ICS lab at the Industrial Cyber Security Conference. At this conference, there was an ICS testbed composed of heterogeneous devices such as PLCs, RTUs, servers, and industrial network equipment (e.g., switch, firewalls). It was available for hands-on testing by the conference attendees and, since the testbed was left almost uncontrolled, the data recorded are not labeled. Furthermore, it is impossible to know what the users have done and, eventually, what kinds of attacks are present. The dataset includes a wide variety of ICS protocol traffic such as S7Comm, Modbus/TCP, EtherNet/IP, and DNP3.

S4x15 ICS Village CTF Dataset [215] provided by Digital Bond, contains network traffic collected during a capture-the-flag (CTF) competition in the ICS Village. The system was composed of different interconnected PLCs, and the dataset contains, without labeling, the attacks launched by the players to the system. The dataset contains pcap files with Modbus/TCP and BACnet packets.

Basing on this dataset, Yu et al. [203] proposed an anomaly detection method based on TCP and UDP payload inspection. The detector's architecture comprises an offline module for the expected behavior model and an online module containing the actual anomaly detector and a packet signature generator. In the proposed work [203], the authors use 4SICS [202] dataset to model the normal traffic behavior for Modbus/TCP protocol, while the normal traffic behavior of BACnet protocol is based S4x15 dataset. Instead, malicious packets are retrieved from Quickdraw-Snort [246], a collection of Snort rules for ICS environments, which also provides some testing packets. Results show Acc and Re close to 100% and a very low false alarm rate.

WUSTL-IIOT-2018 [122, 164] is a dataset recorded from a testbed simulating a water tank control system. Network traffic was monitored for 25 hours, collecting 25 features. Then, the authors performed a data cleaning process to delete corrupted or missing values and outliers. In this phase, about 10k observations were erased, leaving the final version with about 7037k entries. Furthermore, only the six more relevant features are available in the provided csv file, together with a column indicating if the observations are related to an attack. Various attacks have been launched during the capture: port scanning using Nmap [247], address scan attacks, device identification, and unauthorized access to actuators status by using known exploits of the Mobus protocol. The final dataset contains 6.07% of data under attack.

On the same paper [122], the authors developed IDSs employing standard Machine Learning algorithms. The authors selected only six features from the datasets concerning the number

of packets, the packets' length, source and destination of addresses, and port numbers. Best results were achieved by Decision Tree (DT) and k-Nearest Neighbors (kNN) with an accuracy of up to 100% considering the offline evaluation. Instead, regarding the online phase, DT and RF obtain the best results with an accuracy of 0.999. Furthermore, these last two models performed well in terms of False Alarm Rate (i.e., percentage of the normal flows misclassified as abnormal flows) and Un-Detection Rate (i.e., the fraction of the abnormal flows misclassified as normal flows), which are close to 1.

Physical and Network Levels

Electric Power and Intelligent Control (EPIC) is a collection of data from 8 scenarios collected with the EPIC testbed [110, 111]. Each collection scenario lasts for 30 minutes under normal operations, resulting in more than 5000 readings of sensors and actuators, together with the corresponding Modbus network traffic. Blaq_0 [217] is a dataset obtained from the same testbed under different attacks. Blaq_0 contains network-level data collected from a three-day Hackaton 2018 where different teams attack the EPIC testbed. Both datasets are free upon request, but the second one is not widely used. The EPIC dataset contains both pcap and csv files. Both the datasets are free upon request on the iTrust website [217].

QUT_S7 (Myers) [218] is a dataset generated from a small scale ICS testbed. It is composed of a bi-directional conveyor belt system, a water pump system, and a "reactor" pressure vessel system. All these devices are connected to a power meter, and an HMI is used to collect logs. The protocol used is S7 Communication, the standard protocol for Siemens PLCs. The dataset contains device logs with information about each component's state and pcap files with network traffic. Data are divided into two folders containing control data (i.e., data in a normal behavior) and attack data. 21 attacks were launched, consisting of two major types: Injection Attacks and Flooding Attacks. Furthermore, the authors also provide an xlsx file containing pre-processed data. The dataset is freely available for the download [219].

In the same paper, Myers et al. [218] proposed a novel process mining based anomaly detection technique. The detector idea is to collect logs in order to produce a record of each device's status. From this data, the authors compute a model containing the expected behavior of the ICS. The model is designed to manage the entire process instance from start to finish with only acceptable events. Finally, process mining is used to perform conformance checking activity, calculating the fit of a given event log by replaying it on the model. Results show that only 16 attacks out of 21 were correctly identified with several false positives (i.e., Pr: 0.727;

F1: 0.744; Re: 0.762). The authors motivate that for most false positives, the starting condition was altered by previous attacks. It is a common problem that can be mitigated with a correct generation of the dataset, as will be discussed in Section 2.1.6, especially taking care of the labeling part.

SWaT [163, 121] is the most popular dataset in the ICS field. It contains monitoring data from a fully operational scaled-down water treatment plant. The testbed contains two separate networks: a level 1 star network that allows the communication between the SCADAs system and the six PLCs and a level 0 ring network that transmits sensor and actuator data to the corresponding PLC. The protocols employed for communications are CIP and EtherNet/IP. The first version (December 2015, described in [163, 217]) is the largest and most used. This version includes both network traffic and recordings from all 51 sensors and actuators for eleven days. Of these eleven days, seven days are under normal operation, and four days contain 36 different attacks, classified into four types based on the attack number of stage and devices affected. As reported in [248], the various SWaT releases are very different from the operational point of view, also implementing different actuators' control logic. It makes it difficult to transfer the detection framework among the dataset releases. are free upon request at the iTrust datasets webpage [217].

The SWaT dataset is probably the most used dataset on which researchers try their IDSs. Almost all the papers using the SWaT testbeds have no explicit mention of the dataset version used. However, from the data description, we can infer that the first version is used in almost all cases. Several innovative detection methodologies have been tested on this dataset, ranging from sensor noise fingerprint [249], a graphical-based detection approach [250] and a framework to generate invariants with association rules mining [251]. Kravchik and Shabtai [224] employ the SWaT physical data to test different detection approaches such as Dynamic PCA, 1D-CNN, and AutoEncoder, both in the time and frequency domains. Thanks to the linearity of many relations between sensors and actuators in SWaT, PCA performed very well, especially using a sliding-window approach, which results in 0.92 of Pr and 0.879 F1. Also AutoEncoder in the frequency domain reaches similar scores (i.e., Pr: 0.924; F1: 0.873). Similarly to WADI, excellent results on SWaT physical data were achieved by Abdelaty et al. [220]. This paper introduces DAICS, 2-branch neural network with automatic tuning mechanisms to update the system model. DAICS scores 0.9185 of Pr and 0.889 of F1. It is worth noting that, despite a not high Pr (i.e., 0.70) and F1 (i.e., 0.81), MAD-GAN [231] on SWaT achieved a Re of 0.954, the higher one to the best of our knowledge.

Instead, by relying on SWaT network traffic, Schneider and Böttinger [242] implement an

autoencoder-based unsupervised anomaly detection framework leveraging pipelining parallel processing strategies to speed up the training. To overcome the problem of unlabelled network packets and generate the ground truth, the authors developed a semi-automatic label estimation mechanism to detect the packets with a higher probability of being anomalous and then use a manual investigation to label them. Results reveal a Pr of 0.998 and an F1 of about 0.988 in scenarios like TCP session reset attack and SYN Flood attack, while other types of attacks as duplicate acknowledgments or TCP retransmissions are essentially not detected.

2.1.6 Lesson Learned: Good Practices

Based on the knowledge acquired by surveying the different works and analyzing the common mistakes and solutions implemented, in this section, we summarize concepts and good practices to consider when selecting a testing system. In particular, we summarize the good practices in creating an effective testbed in Section 2.1.6 and develop a dataset in Section 2.1.6. Furthermore, we also provide additional insight to help the standardization of the IDS results in Section 2.1.6. During the designing phase of each of the three resources, the designer must consider the final use of such resources and the other two resources' requirements in future integration. Figure 2.10 graphically represents the relation between the three resources. More precisely, a testbed should allow an efficient data collection to produce a well-representative dataset and integrate IDSs to validate the case studies in a real scenario. A dataset must be designed to be an exhaustive and precise representation of a testbed and easily allow data analysis tasks and implementing an IDSs. Finally, an IDS, which represents the higher-level products with respect to the other two resources, should generalize on different datasets to avoid construction biases. Moreover, the design of a dataset should consider an easy integration into a real-world scenario such as a testbed.

Good Practices: Testbed

An effective testbed development passes through various steps and challenges, each composed of a notable complexity level. These challenges should be considered during the design phase.
Scope Identification. During the design phase, the designer must consider the final application. The applications of a testbed can be [80]: i) Discovery, to study and obtain knowledge about a particular ICS field or system functioning; ii) Demonstration, to validate or experiment the research findings; and iii) Education, to use the testbed to educate students, researchers, and stakeholders. Every scope implies different requirements to deal with and different fund-

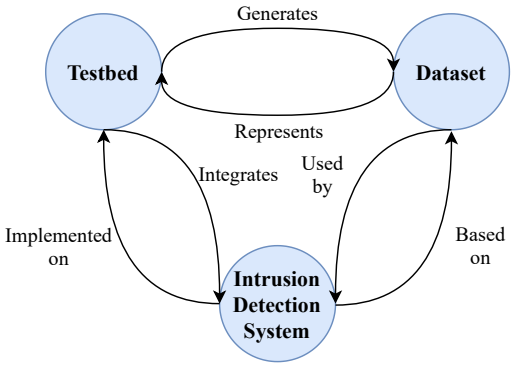


Figure 2.10: Relations between Testbed, Dataset, and IDS.

ing. For instance, if a testbed is specifically designed for IDS development, the authors must consider developing an attack chain and data collection accurately. On the contrary, the Educational testbeds do not have this requirement. Instead, they should be composed of an easily understandable and representative process. In this case, water systems are an excellent choice due to their immediate visualization. Once the scope is identified, the designer can give the system's specific layer adequate importance to satisfy the scope.

Fidelity. If the testbed is used for Discovery or Education, data's perfect fidelity is generally not needed. In these cases, Virtual or Hybrid testbeds are the preferable platforms to be used due to their flexibility and cheapness. Contrarily, in the case of validation tests, Physical testbeds are the best solutions since the smallest variation of measures is fundamental for the research. A complete work that can help researchers to identify the correct design criteria is [32]. The US NIST has recommended that a SCADAs testbed for security assessment should consider four general areas [13]: the control center, the communication architecture, the field devices, and the physical process itself.

Expensiveness. Expensiveness in the construction and the maintenance of Physical testbeds are the first barriers a research group will encounter when deciding to build one. Suppose a research group can deal with this limitation. In that case, it is useful to share with the community datasets collected from the Physical testbed and the related documentation, as the iTrust laboratory of SUTD [194] is doing with SWaT and WADI. Furthermore, provide a simple way for other researchers to access the testbed can be an added value not only for the community, which can take advance of it but also for the owner who can have a more critical view of the system.

Reproducibility and Comparability. Robust and innovative researches need to be reproducible and peer validated. Basing on this, testing on physical testbeds is not recommended

since it creates difficulties in reproducibility. An intelligent solution is to create one or more datasets capturing network traffic and physical measures from the testbed and share them with the community. In this way, the reviewer can easily verify the study, while the community will benefit from newly available datasets. On the other hand, if a virtual testbed is used, it is not required to provide a dataset to support the research. Instead, it is possible to provide the software with the entire simulation. However, it is fundamental to precisely indicate the architecture and the state of the ICS at the beginning of the experiment to avoid reproducibility errors due to a different scenario.

Missing Representation. We identified that the most common scenario represented with a testbed is relative to water management (e.g., Water Distribution, Water Treatment). This is probably because Water Systems are easier to implement regarding equipment and maintenance costs. Another very represented scenario is related to the Electric Plant (e.g., Power Grid, Power Power). However, IDSs rarely investigate this scenario. We believe this is due to the difficulty in developing detection systems that deal with high-dynamic environments such as Electricity. Also, the majority of the related dataset does not include attacks scenario. For future organizations that want to approach a testbed development, we identified a low contribution in scenarios such as large-scale manufacturing, Nuclear Plants, Transportation Systems, healthcare infrastructure, IIoT, or HVAC. Another interesting scenario missing in the testbeds analyzed is the cloud-based ICS. As discussed in [34], this scenario is always more adopted in real systems. Therefore, having a testbed related to cloud-based ICS would help secure future ICSs by studying the connection between the plant and the cloud, which inevitably opens new attack entry points.

Standardization. We noticed that a common problem in the analyzed testbeds is the lack of standardization according to the industrial security standards. The most common international standard for cybersecurity in the industrial system is the IEC 62443 [252]. However, different organization releases guidelines and requirements to securing ICSs. These organizations include NIST [13], North American Electric Reliability Corporation (NERC) [253], and European Union Agency for Cybersecurity (ENISA) [254]. In 2013 ENISA published a report providing a list of related works in the field of ICS security [254]. In this report, ENISA lists initiatives and groups working on ICS security worldwide. Among the presented projects, the most relevant and yet active, such as SNL Testbed at Sandia National Labs [105] or Joint Research Center's testbeds in Italy [86, 123], are discussed in this study. Furthermore, the report describes different standards, guidelines, recommendations, and practices for companies and manufacturers to help them to secure their installations by exploiting commercial solutions

and valuable policies.

Good Practices: Dataset

A well-designed dataset should exhaustively represent a testbed's behavior and allow easy implementation of research findings. To do this, the design process of an effective dataset must consider the following points.

Labeling. When designing a dataset, the labeling process must be precisely described in the documentation to allow researchers to process the data accordingly. Packets that are part of attacks must be carefully labeled to provide ground truth to researchers. Furthermore, in a valuable dataset, labels must also contain information about the attack type (e.g., Injection, Replay, DoS) and the attack phase. This last element is essential due to the recovery time of many ICSs: after an attack occurs, the system may need some time to stabilize itself. An inaccurate labeling process can wrongly consider this behavior part of the attack. Hence, a good strategy is to flag this kind of packet as *recovery*, leaving the decision on how to consider them to researchers. The work [192] explained that the authors of the SWaT dataset decided to label a process data sample as "Attack" when the attack was launched instead of when the system behavior started to change. This approach can lead to a ground truth problem if not correctly documented or managed. Furthermore, it is important to consider the label generation methodology accurately. A manual approach to flag each entry of an attack as malicious is costly, and if the data amount is large may be impracticable. On the other hand, fully automatic strategies are possible, and they work quite well if attacks are at the same time automatically generated. However, automatic labeling cannot provide high accuracy in case of complex attacks on highly distributed systems. Semi-supervised approaches provide a trade-off that efficiently spends an expert's work supported by a visualization platform such as RiskID [255].

Documentation. Many of the datasets surveyed lack in the documentation. To allow correct and easy use of the dataset, the designer should include detailed information about the dataset's characteristics or a description of the source testbed design. Exhaustive documentation should include the system's control logic, a description of the implemented attacks, and the configuration settings. In the SWaT case, as reported in [248], the recent versions of the dataset implement different control logic. However, the authors never mentioned such modifications.

Attacks Similarity. A complete dataset should also include attacks. Similarly, in a testbed, a researcher needs to be able to deploy attacks easily. However, the designer must approach

this phase with caution. Attacks should be as similar as possible to real cases. If a dataset is collected from a testbed, it is sufficient to launch the attacks following an adversary approach and various system information. The authors should also include a clear and complete analysis of the attacker model. On the other hand, it is important and challenging to capture traffic while the attack is occurring in order to generate the datasets. If not accurately managed, synthetic packets in the resulting capture could disrupt the fidelity of the dataset, making it unrealistic. Furthermore, if data are captured in different monitoring points inside the ICS, it is required a synchronization mechanism to provide consistent data.

Domain Shift. A common problem in a dataset is the so-called Domain Shift, i.e., the difference between the training entries and the testing data [220, 248]. To support researchers and IDS development, datasets should be released with complete documentation explaining the system's initial state. Another problem observed in [248] is related to the testbed remains unstable for a long time. More precisely, after the end of an attack, a system's behavior may need time to recover, remaining unstable. In this case, its behavior will be identified as anomalous by detectors even if flagged as Normal. To deal with this problem, when designing a dataset, the authors should consider adding another label to classify the dataset, e.g., "System Unstable". If the authors can directly interact with the testbed, another solution could be to restart the system after an attack. An imbalanced dataset is a collection of data that contains a significantly low number of samples from one class with respect to the other [256]. It is a critical issue that can influence the performances of Machine Learning based classifiers. Some datasets provided by researchers contain a low percentage of data classified as under attack, as reported in Table 2.4. This happens because attacks generally last for seconds or minutes, while the ICS is expected to run for much longer. There are different techniques to get better results from an imbalanced dataset [257]. One solution is to act at the data level by re-balance the data in a pre-processing phase using different sampling strategies (e.g., down-sampling). Another novel solution is Data Augmentation, recently introduced to improve Anomaly detection performances in [258]. This technique leverages generative models, such as GAN, to generate synthetic samples. In [256] the authors performed an experiment to understand the impact of an imbalanced dataset in the ICS security field. Different datasets were obtained from an extensive network traffic capture collected from a water control system testbed. To do this, the authors associated to a fixed number of attack samples a variable number of normal samples to create five different datasets with different imbalance ratios (i.e., the percentage of data under attack over the whole dataset). Ratios span from 0.1% to 10.0%. Results show a high Re variance with better results on higher ratios ($Re>0.99$ for ratio$\geq$0.70%; $Re<0.12$ for ratio$\leq$0.30).

At the same time, the Undetected Rate (UR) (i.e., the fraction of the attack samples classified as normal) shows near-zero values for high ratio and large misclassification on low ratio datasets (UR<0.01 for ratio≥0.70%; UR>0.88 for ratio≤0.30). Basing on these results, the authors conclude that it is advisable to generate datasets with at least 1% of data under attack to reduce imbalance problems when testing IDSs.

Good practices: Intrusion Detection System

Nowadays, the majority of IDS are based on Machine Learning and Deep Learning techniques. To build a model using such techniques is required a notable amount of well-organized dataset (e.g., balanced, labeled). Since these techniques are not always straightforward to understand and implement, researchers should implement a clear and well-approved pipeline. The following aspects can help the development of effective IDSs.

Results Baseline. While reading the different papers concerning the implementation of IDSs, we noticed the absence of an evaluation baseline in many cases. Defining a good baseline could help researchers evaluate if their proposed research is effective and improve the current state of the art. Furthermore, various works base their results on a subset of an available dataset. If not used for a specif reason (e.g., isolate and test a specific attack), this approach could cause a problem in understanding an IDS's effectiveness. For this reason, we believe that our Table 2.4 can support the future evaluation baseline. We also identified issues in the evaluation metrics. Many research not always use the same metrics, making it difficult to compare different approaches. We suggest using as many common metrics as possible, such as F1, Acc, Pr, and Re. Baseline problems are also mentioned in other fields, such as Review Helpfulness predictions [259], where the authors proposed to used the same features on the different models proposed by researchers. In this way, it is easier to compare the efficiency of a prediction model. In the ICS field, IDSs model features can be very different and based on diverse approaches. However, comparing a proposed model with the best IDSs of the art state could help future researchers identify the right research directions. Furthermore, to avoid the effect of design bias of the dataset, an IDS should be validated on multiple datasets.

Data Verification. Generally, researchers rarely investigate the causes of the weak performance of IDSs. Sometimes, the reasons may be due to a problem related to the distribution of the dataset. In [248], the authors showed that SWaT dataset behavior and data distribution between Training Set and Test Set significantly differ. However, even if the SWaT dataset currently represents the most used dataset to test IDSs, no previous works analyzed the statistical

distribution through statistical tests. Finally, to allow the community to validate the research approach and improve it, a good practice is to release the code repository source code (e.g., GitHub, Bitbucket).

2.1.7 Takeaway on ICS Security

In recent years the interconnection between IT and OT networks has opened up modern ICSs to new risks and novel vulnerability surfaces. These vulnerabilities were highlighted in many works but also by the dangerous malware targeting industrial companies. Therefore, it is vital to develop new security mechanisms to protect such systems.

In this survey, we provide a comprehensive overview of the ICS field by presenting the architecture and the typical devices employed. We then proposed an analysis of the industrial protocols used in ICSs, highlighting security measures offered by the protocols, their expansions, and analysis of their diffusion in the real world. Furthermore, we surveyed and analyzed the different platforms to test new security mechanisms in the ICS field. To do this, we categorized the testbeds as Physical, Virtual, or Hybrid based on their functioning and explained the various challenges and requirements to consider during the development or selection phases. Also, we presented the different ICS datasets dividing them based on the type of data provided and useful information that can help the reader choose the dataset (e.g., attack implemented, format, and various data information). To do this, we accessed every dataset and analyzed it separately. We also reported the IDS with the best performance present in the literature to offer a baseline for further work for each dataset. Finally, we depicted different good practices and suggestions for researchers who want to use this kind of testing method and institutions that want to build testbeds or collect datasets.

We believe this survey can help address future research on this field and new researcher approaching the ICS area. In the future, we aim to continue collecting new testbeds and datasets on the website to create a collection of useful information the research community can exploit for research and studies on this essential field.

2.2 ICS Exposure Measurement

According to the Purdue model [260], which represents the reference architecture model for the ICSs, these systems were supposed to operate on air-gap environments. However, due to the rise of the Internet and Ethernet technologies, the industrial business moved towards the connection to external networks, opening dangerous vulnerability surfaces. This digitization process requires the adaptation of legacy protocols that were not designed with security features (e.g., Modbus and DNP3) to the TCP/IP stack to allow operations such as remote communication. The lack of encryption and authentication can be exploited for malicious purposes such as data exfiltration, impersonification, and service disruption. The research interest and effort on ICS security increased with the number of incidents and the involvement of critical-infrastructures, from the most famous Stuxnet [3] that affected an Iranian nuclear power plant to the more recent Triton [4] that targeted Schneider Electric products. Many researchers estimated the vulnerable ICS landscape by looking for well-known ICS ports exposed to the Internet, by actively performing scanning of the IPv4 address space [261, 262] or leveraging public available information [263, 264] gathered by third-parties such as *Shodan* [265] and *Censys* [266]. *Shodan* and *Censys* are cloud services that allow the user to scan, discover, and query the devices connected to the Internet. However, many ICSs could not be indexed due to the presence of network devices like Network Address Translations (NATs) or Firewalls, leading to an incomplete estimation of the vulnerable ICSs. In [267] the authors analyzed the unprotected industrial traffic transmitted over the Internet, gathering information about the security status of the host systems.

In this study, we investigate the practical use of ICS protocols over the Internet, most importantly to learn about security issues. Moreover, we aim to determine if scanning activities (such as results from *Shodan*) provides accurate estimates of the use of ICS protocols. To achieve this, we present a framework to identify the ICS host transmitting industrial traffic over the Internet, based on traffic observed at an IXP. We compare and correlate the obtained results with the information from a prominent scanner: *Shodan*. We use *Shodan* due to its popularity in related works [263, 264], and because it supports significantly more ICS protocols with respect to other services (e.g., Censys). Our analysis shows that ICS hosts identified by *Shodan* are rarely actively using ICS protocols, and such measurements do not estimate the use of insecure protocols on the Internet. Instead, IXP (or Internet Service Provider (ISP))-based measurements are required. Our results also provide insights into the volume and type of legitimate and unprotected ICS communication found on Internet links.

2.2.1 Background on IXP

In this section we briefly recall the main concepts useful to understand our analysis.

ICS Traffic Analysis Through IXP

Our analysis relies on *VSIX* [268], a local IXP which manages the traffic circulating in the North East of Italy. This environment allows to collect packets circulating through the Internet in a secure and privacy-respectful way. In the following, Section 2.2.1 recalls the definition of Autonomous System (AS) and IXP, which represent the main entities of our system model. We detail our system model in Section 2.2.2 while in Section 2.2.2 we present the theory behind the packet sampling process. Then, Section 2.2.2 and Section 2.2.2 outline, respectively, the research questions of this study and the framework we implemented to address the proposed questions.

AS and IXP

An AS is a group of IP prefixes under the control of a single well-defined administrative authority that defines the routing policies, typically an ISP, uniquely identified by its Autonomous System Number. The routing within an AS is allowed via Interior Gateway Protocol, while the communication with other ASs relies on the Border Gateway Protocol. An IXP is a network facility that enables the interconnection and exchange of Internet traffic between more than two independent ASs according to their Border Gateway Protocol routing configurations. Its typical architecture consists of single or multiple switches connected to the border routers of the adherent ASs, ensuring benefits in terms of bandwidth, costs, and latency.

2.2.2 Related Work on Security Solutions for ICS

ICS security. The security of ICS is a widely studied research topic. Many research groups study new security solutions to implement in order to mitigate the threats to which ICSs are exposed. The Anomaly detection systems represent a low-cost and effective solution. Anomaly detection systems do not require any hardware substitution from the point of view of the company. Indeed they aim to monitor the state of the system passively, focusing, for example, on the physical state of the network [269], network traffic [270] or considering both of them [74], and raising an alert when the normal behavior of the system is violated.

Scan-based analysis. If, on one hand, there are many contributions in literature that present security solutions for the ICSs, on the other hand, there are not many contributions that analyze their current security implementation. Several works [263, 264] leveraged public available information, like *Shodan*, to identify internet-reachable industrial devices, while [261] and [262] manually performed an active scan of the IPv4 address space. For a better understanding of the threat landscape, Serbanescu et al. [271] deployed a low-interaction honeynet, also showing how the number of attacks increased after being indexed by *Shodan*.

Industrial traffic analysis. There is not much literature on industrial traffic analysis. In [267] the authors investigated the industrial communications passing through an IXP and an ISP. During their analysis, they were able, via correlation techniques, to identify scanning activities, possible firewall implementation other than unprotected industrial traffic. In our work, we are not interested in implementing any new tool for identifying industrial traffic, however, we wanted to exploit a wider range of protocols, so we proposed a slightly different packet filtering process to identify possible Industrial communications and more scanning activities. From a security point of view, we analyzed the critical issues deriving from implementing both industrial and non-industrial communication within the same network infrastructure and we argued what are the *Shodan* and *Shodan*-like services limits compared to a wide-view analysis of sampled traffic.

System Model

In our system model, we assume that different ICSs are connected to a local AS (e.g., as customers of an ISP), and the ICSs are using industrial traffic over the Internet for supervision and control (depicted in Fig. 2.11). The IXP provides the capability to sample traffic, according to the sFlow standard [272], to understand the used protocols and involved parties. However, our collector will not identify all the intra-AS communications and inter-AS communications that do not cross the IXP.

In the considered model, an attacker could be a malicious end host located at another AS, or a MiTM attacker. The former attacker needs to be active (e.g., scanning the well-known ICS ports), while the latter can be passive or active. In particular, this last attacker should be able to sniff the traffic from the IXP, therefore it could be a malicious operator of the IXP, an insider who obtained the access at the IXP (e.g., the government), an ISP, or, more in general, an IXP partner. The goals of the attackers are to learn process data (i.e., eavesdrop) and/or manipulate the ICS actively (e.g., by changing or injecting operational commands).

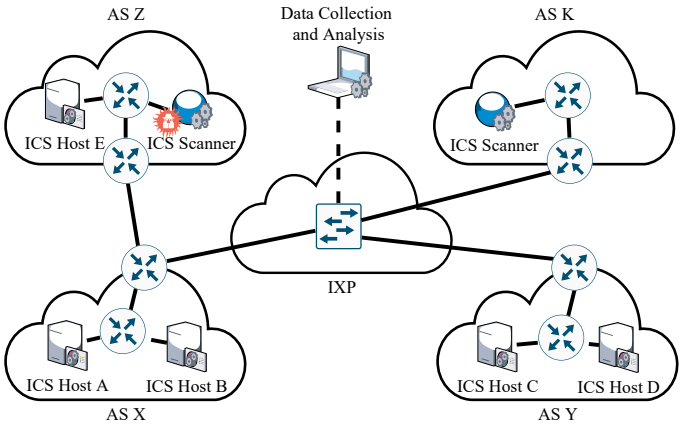


Figure 2.11: System model: ICS at *AS X*, *AS Y*, and *AS Z* communicate over the IXP. Scanners on the Internet (*AS Z*, *AS K*), which can be malicious or benign, look for exposed ICS services. The packets exchanged between two hosts belonging to *AS Y* and *AS Z* could still pass through the IXP, even if *AS Z* is not directly connected. We sat within the IXP network, where we are able to collect and analyze sampled packets.

ICS Packet Sampling

We assume that the ICSs connected to the IXP are continuously sending traffic over the Internet, with a rate of at least one packet per minute. This is a reasonable assumption in line with other works [273, 274], since the ICSs continuously monitor processes with hard real-time constraints. In fact, these systems require frequent and constant polling-time communications to monitor the physical processes.

We also assume that the number of sampled packets n is at least 10 times lower than the total number of packets N passing through the IXP over the observation period T. In this case, the sFlow sampling process can be modeled as a Binomial distribution $B(n, p)$ [275, 276] where p is the probability of a successful event. We will discuss this assumption in Section 2.2.5.

Under these assumptions, if an ICS is connected to the IXP and transmits 1 packet per minute, we can estimate the probability $\hat{p}$ that the sampled packet belongs to the ICS, as:

$$\hat{p} = \frac{1 \cdot 60 \cdot 24 \cdot T}{N}, \tag{2.5}$$

where T is the observation period, in days, and the numerator defines the overall number of packets sent from such ICS. In the limit case, we obtain $\hat{p} = 1$, if $N = 1 \cdot 60 \cdot 24 \cdot T$. This corresponds to the case that we collected only packets of the ICS host under consideration. The

probability that we observe at least one packet with the aforementioned characteristics is:

$$P(X \geq 1) = 1 - P(X = 0). \tag{2.6}$$

Generally, the probability to observe k industrial packets is:

$$P(X = k) = \binom{n}{k} \hat{p}^k \, (1 - \hat{p})^{n-k}, \tag{2.7}$$

where n is the total number of sampled packets and $\hat{p}$ is defined in Equation 2.5.

Research Questions, Challenges, and Goal

Our main goal is to investigate the practical use of ICS protocols over the Internet, in particular with respect to security. The main questions are: *RQ1: How often are (insecure) ICS protocols used over the Internet? RQ2: How often are ICS services exposed to third parties, in addition to the intended use by legitimate parties?*

The main challenge we faced is that third parties cannot directly observe legitimate use of ICS traffic unless they are routing the traffic (i.e., are in a MiTM position). Even in that case, efficiently filtering for ICS traffic out of large volumes of traffic can be challenging.

The outlined challenges raise the following additional research questions: *If ICS protocols are used or exposed, can we identify such hosts using active scanning (e.g.,* Shodan*), or IXP/based traffic collection? To which degree are the results of both complementing each other?* In other words, *RQ3: Is IXP active-scanning based enumeration of hosts a good estimator of (vulnerable) industrial traffic use on the Internet?*

For practical reasons, we focused our work on a geographical region in our country served by a specific IXP.

Proposed Framework

To address the research questions proposed in Section 2.2.2, we compared the properties of the industrial traffic passively collected at an IXP with the information gathered by *Shodan* which actively scans industrial ports. We summarized the steps performed in Fig. 2.12.The first step consists of data collection and processing. However, the different nature of the two data sources (i.e., the IXP traffic and *Shodan*) requires some consideration. In fact, while *Shodan* monitors all the IP address space, the traffic captured by us is restricted to a geographical region, which

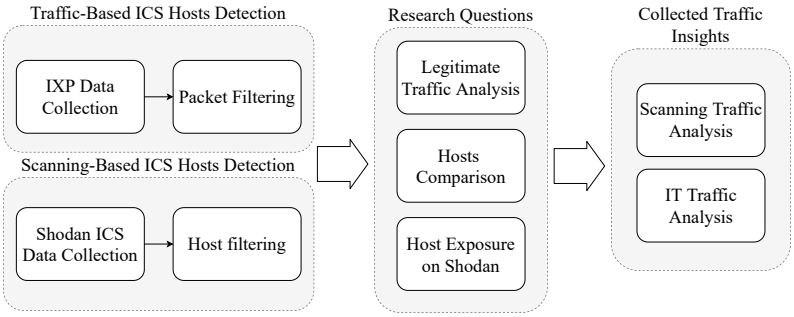


Figure 2.12: The proposed framework to compare the data collected at the IXP and the information available on *Shodan*.

makes the comparison unfair. To solve this, we defined a baseline to evaluate the two different approaches. The baseline is composed of the ICSs indexed on *Shodan* which are the most likely to be observed from our positions, that are the ones who belong to ASs directly connected to the IXP. Once the data were collected, we implemented a three-step filtering approach to extract the industrial traffic from the large dataset of network packets collected at the IXP. This procedure is based on well-known tools (e.g., Wireshark) and is able to identify both scanning and legitimate ICS activities.

In the second step, we analyzed the results of the first step to answer the research questions. We analyzed the legitimate industrial traffic, compared the hosts identified with the baseline, and investigated the exposure on *Shodan* of the hosts detected legitimately exchanging industrial traffic. Finally, to gather additional information about the current ICS security practices and threats, we investigated the data collected at the IXP to detect scanning activities and to deeply understand the network behavior and architecture (e.g., presence of a NAT) of the hosts. A port-scan approach cannot identify industrial services hidden behind a NAT, Firewall or VLAN. Instead, we can identify hidden industrial services by leveraging our IXP sampling-based approach. The presence of both Industrial and non-industrial traffic from a single host can indicate such hiding network mechanisms.

2.2.3 Implementation

In this section, we present the implementation of our framework of analysis. In particular, Section 2.2.3 defines the baseline that we use to compare our approach to a scan-based one, and Section 2.2.4 outlines the packet filtering approach used to identify the ICSs hosts.

Collection of the *Shodan* Baseline

We used *Shodan* to identify the ICS hosts exposed to the Internet within the IXP area as a baseline, due to popularity in related works [263, 264], and because *Shodan* supports significantly more ICS protocols with respect to other services such as Censys. We define the IXP area as the set of ASs directly connected to the IXP itself. Due to the limited access to the *Shodan* services, we collected all the Italian ICS exposed according to the *industrial-control-systems* category offered by the *Shodan* platform. However, the list of protocols offered in such category is incomplete compared to the list of protocol dissector implemented in Wireshark and reported in Table 2.5. In this table, we reported the complete list of the industrial protocols of our interest, the communication ports used and if the current version of Wireshark (currently version is 3.0.5-1) can dissect them. To address the limitations of the aforementioned category, for each of the missing protocol, we designed a specific query based on the reference ports and common terms (e.g., `port:10001 country:"us" I20100` to discover Automated Tank Gauge (ATG) tank monitoring systems) collecting the resulting hosts. We performed the queries by leveraging Shodan API. Finally, we selected the ICSs of our interest discarding the hosts that do not belong to an AS of the IXP area. We reported the results of the analysis in Table 2.6.

2.2.4 Packet Filtering

The large size of the traffic captured required the implementation of an automatic filtering approach. We applied a preliminary port-based filter to identify the ICS protocols, basing on the official documentation of the protocols and the ports list [277]. We reported the considered protocols in Table 2.5 followed by their port ranges and the Wireshark dissector availability. Then, due to the Wireshark dissector limitations, we applied the following three-step approach.

We reported the number of filtered packets at each step of the filtering process in Table 2.8.

1. Starting from the previously filtered packets, we identified the correctly dissected ICS packets by Wireshark, dividing scanning activities from legitimate activities.
   (a) We performed a deep packet inspection with Wireshark to identify all the Industrial Protocols packets supported and marked as not malformed;
   (b) We cross-validated the resulting data using *nDPI* [278], an open-source library able to dissect a wide range of industrial and non-industrial protocols, removing all the packets not tagged as non-industrial protocols;

Protocol	Port Ranges		Wireshark
	TCP	UDP	
Advanced Message Queuing Protocol (AMQP)	5671-5672	-	✓
ANSI C12.22	1153	1153	✓
ATG	10001	-	
BACnet/IP	-	47808	✓
CoAP	5683	5683	✓
Codesys	2455	-	
Crimson v3	789	-	
DNP3	20000	20000	✓
EtherCAT	34980	34980	✓
Ethernet/IP	44818	2222	✓
FL-net	-	55000-55003	
FF HSE	1089-1091	1089-1091	✓
GE-SRTP	18245-18246	-	
HART IP	5094	5094	✓
ICCP	102	-	✓
IEC60870-5-104	2404	-	✓
IEC61850	102	-	✓
Modbus/TCP	502	-	✓
MELSEC-Q	5007	5006	
MQTT	1883,8883	-	✓
Niagara Fox	1911,4911	-	
OMRON FINS	-	9600	✓
OPC UA	4840	-	✓
PCWorx	1962	-	
ProConOS	20547	20547	
PROFINET	34962-34964	34962-34964	✓
S7comm	102	-	✓
Zigbee IP	17754-17756	17754-17756	✓

Table 2.5: Industrial protocols with relative ports, Wireshark dissector availability (✓if wireshark is able to dissect it).

(c) Then, we filtered out the packets where the IP-source is tagged as scanner by *Greynoise* [279]. *Greynoise* is a security company that collects, labels, and analyzes Internet-wide scan and attack data and provides access to such information to users via API[2].

(d) We considered the difference between the results of b) and c) as *legitimate ICS traffic*;

2. In the second step, we further processed the initial port-based filtered packets. The goal is to identify additional scanning activities targeting the ports of the industrial protocols

[2]The classification methodology of Greynoise is not public, and this could affect the deduced results and reproducibility of the experiment.

	Italy		IXP Area	
Protocol	Hosts	%	Hosts	%
MQTT	1531	23.5	206	47.8
Niagara Fox	1382	21.2	50	11.3
Modbus/TCP	1346	20.6	70	15.9
Ethernet/IP	456	7	16	3.6
Siemens s7	417	6.4	37	8.4
PCWORX	373	5.7	2	0.4
CoAP	364	5.6	3	0.7
Codesys	183	2.8	10	2.3
BACnet/IP	170	2.6	26	5.7
AMQP	140	2.1	4	0.9
Omron FINS	83	1.3	6	1.4
ATG	32	0.5	7	1.6
OPC UA	13	0.2	1	0.2
DNP3	11	0.2		
IEC60870-5-104	6	0.09		
ProConOS	5	0.07		
GE-SRTP	5	0.07	1	0.2
Crimson v3	2	0.03		
MELSEC-Q	1	0.01	1	0.2

Table 2.6: List of the ICS per-protocol hosts exposed in Italy and IXP area according to and the corresponding percentage over the total number.

not supported by Wireshark.

(a) We used Wireshark to remove all the not malformed, industrial and non-industrial protocols (e.g., SSL, HyperText Transfer Protocol (HTTP)), keeping the packets generally tagged as TCP or UDP (e.g., SYN packets scanning industrial ports);

(b) We used *nDPI* to remove additional non-industrial traffic from the previous results;

(c) Then we used *Greynoise* to identify scanning activities;

3. Finally, we gather the results of the first two steps.

(a) We merged the scanners activities identified in 1.c and 2.c, and we tagged them as *ICS scanners*;

(b) We considered the sum of the legitimate ICS traffic (i.e., 1.d) and the ICS scanners (i.e., 3.a) as *ICS traffic*.

	MQTT	Zigbee IP	EtherCAT	PROFINET	IEC60870-5-104	ANSI C12.22	Ethernet/IP	Modbus/TCP	AMQP	OPC UA	CoAP	FF HSE	ATG*	BACnet/IP	Codesys	GE-SRTP*	ICCP/IEC61850/57	MELSEC-Q*	Niagara Fox	OMRON FINS	PCWorx*	HART IP	Crimson v3*	DNP3	FL-net*	ProConOS*	Overall
h	62	32	16	15	10	10	8	8	6	4	3	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	176
h_S	206	0	0	0	0	0	16	70	4	1	3	0	7	26	10	1	37	1	50	6	2	0	0	0	0	0	442
i	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2

Table 2.7: Comparison between the number of ICS hosts found with our approach (h) and by *Shodan* (h_S) within the IXP area. The value i represents the number of hosts common to the two approaches. * if Wireshark dissector is not available.

2.2.5 Results

We collected data for a period of 31 days from the 14th of January 2020 to the 14th of February 2020 at the considered IXP. According to the sFlow standard, the traffic was sampled with a sampling rate of 2^{-12} and packet truncation at 128 bytes. The sampling process estimates the effective traffic passing through the exchange point [275], while the truncation gives access to the full link layer, network layer, transport layer, and few bytes of the payload. The collection resulted in ~1.6B packets for more than 189GB of data. All the source and destination IPs were anonymized on-the-fly during the capture process. Furthermore, for privacy concerns, we excluded the entire payload from the analysis.

In the following section, we present the analysis results of the traffic capture obtained after the collection phase. In particular, Section 2.2.5 discusses the legitimate ICS traffic, Section 2.2.5 compares our approach with *Shodan*, while Section 2.2.5 reports an analysis of the *Shodan*-indexed hosts.

Legitimate ICS Traffic

We identified 168 different ICS endpoints and 12 different industrial protocols, with 8 hosts using two different ICS protocols. The percentage of legitimate ICS traffic identified over the total number of packets is $5 \cdot 10^{-5}$. Overall, MQTT and AMQP protocols count more than 55% of the legitimate ICS traffic. This result is not surprising since both protocols are widely used for IoT communication in non-industrial environments. ANSI C12.22 covers almost 21% of the total number of packets, followed by Modbus/TCP with 9% and Zigbee with 8%. Other ICS protocols are Profinet, Ethercat, IEC60870-5-104, and Ethernet/IP together count about 7%. Another interesting result is that despite MQTT official documentation [280] specifies port 8883 and AMQP port 5671 respectively for communicating over TLS, our results show that all the MQTT and AMQP communication rely on insecure ports, leading to known vul-

	Packets	%
Total	~1.6B	
After Port-based filtering	43584	0.0027
Step 1		
a) Wireshark filtering	3188	-
b) nDPI validation	2075	65.1
c) Scanning activities	1360	42.6
d) Legitimate ICS traffic	715	22.4
Step 2		
a) Wireshark filtering	32741	-
b) nDPI validation	26171	80
c) Scanning activities	3019	9.2
Step 3		
a) Total ICS scanners	4379	-
b) Total ICS traffic	5094	-

	Overall industrial traffic % w.r.t. total one
ICS traffic	~0.0003
ICS scanners	~0.0003
Legitimate ICS traffic	~0.00005

Table 2.8: Number of packets filtered step by step. The percentage represents the number of packets left compared o the previous step (i.e., the previous row).

nerabilities [281, 282].

Comparison with the *Shodan* Baseline

To address *RQ3*, we compared the hosts identified in Section 2.2.5 with the hosts identified in Section 2.2.3. We define H as the set of ICS hosts detected by us and H_S as the set of ICS hosts identified by *Shodan*. We also compute the number $i = |H \cap H_S|$ of ICS hosts detected by both approaches.

We reported the results in Table 2.7. Among the 176 hosts h identified in the Table, 12 hosts are duplicated (i.e., the same use more than one protocol), while among the 402 h_S, two hosts are duplicated. Although we detected an overall amount of 168 unique ICS hosts compared to 440 by *Shodan*, only 2 hosts were common to both the approaches, respectively an MQTT and a Modbus/TCP endpoint, meaning that the two methods are complementary. Our approach detected more hosts than *Shodan* for 8 protocol. The *Shodan* port-scanning approach, instead, detected more hosts for 12 protocols.

We further investigated why we detected only two endpoints in common with *Shodan*. To

do so, we extracted from the dataset collected at the IXP all the packets that come from, or are directed to, an element of H_S. Results show that 16.3% of $|H_S|$ (72 hosts out of 440) were captured during our analysis but were not leveraging industrial communication. This means that they correspond to false positives since they were not actually using industrial protocols or that their packet rate transmission was below our threshold of 1 packet per minute (defined in Section 2.2.2).

We relied on sampling rate the assumptions of Section 2.2.2, which are verified since $n << 10N$ (i.e., the sampling rate is 2^{-12} and $N \approx 2^{12}n$), to compute the probability that we miss all the packets of an ICS host (2.8) and the probability that we observe at least one packet of an ICS host (2.9) in the sampling period. In particular, given $T = 31$ days, $n = 1599431398$ sampled packets:

$$P(X = 0) = 1.848 \cdot 10^{-5}, \tag{2.8}$$

$$P(X \geq 1) = 1 - P(X = 0) = 0.999. \tag{2.9}$$

According to the previous results, the probability we missed an ICS host that respects our assumptions is negligible. A possible explanation is that the 72 hosts indexed as ICS by Shodan are false positive (e.g., they are general devices with exposed ICS ports) or ICS hosts but are not currently active.

Hosts Exposure on *Shodan*

Among the ICS hosts involved in legitimate ICS communication, we are interested in counting how many of them were also indexed by *Shodan*, and what kind of information *Shodan* was able to gather (i.e., *RQ2*). We found that 64.3% of such hosts were successfully identified by *Shodan* and 11% of them were found with ICS ports exposed, more specifically Modbus/TCP, MQTT, and AMQP.

In Table ,2.9 we reported the Top-5 exposed services and the Top-5 exposed ports based on the *Shodan* collected data. Due to the high percentage of Web Servers detected, we investigated deeply to understand what kind of services these devices were exposing. We found IP Cameras, Printers, Routers, and Network Attached Storage login pages other than energy monitoring and alarm systems. Note that in this work, due to the potential criticalities of the involved systems, we were not interested in performing any active penetration test to the identified devices for security and privacy concerns. For this reason, we analyzed the Common Vulnerabilities and Exposures (CVE) information provided by *Shodan* to identify possible vulnerabilities caused by unpatched systems or well-known critical services. We found that 10.2% of the sys-

Product	%	Port	%
1) MikroTik bandwidth-test server	25	1) 443	11.8
2) Apache httpd	12.5	2) 80	11.5
3) nginx	9	3) 2000	7
4) Open SSH	9	4) 8080	6
5) MQTT	7.9	5) 22	4.1

Table 2.9: Top-5 ports and services detected by *Shodan* and percentage of hosts exposing them.

tems indexed were affected by at least one CVE. We then associated with each vulnerability the corresponding Common Vulnerability Scoring System (CVSS) v2.0 score [283]. According to the NIST, the severity of a score between *0.0-3.9* is considered low, *4.0-6.9* medium, and *7.0-10.0* high. We found an overall amount of 207 CVEs, 30% of which have a score greater than 7.0 affecting 81% of the vulnerable hosts, 4.3% greater than 8.0, 2.9% greater than 9, and 2.4% equal to 10.0, all affecting 27.3% of the vulnerable hosts. We must note that all these vulnerabilities can be exploited remotely.

Summary of Main Results

RQ1: *How often (insecure) ICS protocols are used over the Internet?* We observed 12 different industrial protocols during our collection period. By analyzing these 12 protocols, 6 of them (i.e., EtherCAT, PROFINET, IEC60870-5-104, Ethernet/IP, Modbus/TCP, FF HSE) did not implement any encryption, authentication, or integrity protection features by design and were used by 59 hosts. In addition, protocols such as MQTT and AMQP support TLS (enabling confidentiality and authentication), but this was not implemented in practice. The use of insecure protocols and missing use of TLS affected an overall amount of 127 hosts, meaning that 75.6% of the hosts are using vulnerable ICS communications.

RQ2: *How often are ICS services exposed to third parties, in addition to the intended use by legitimate parties?* We found only a small subset of hosts that we identified as legitimately using ICS protocols (i.e., 7.1% corresponding to 12 hosts) also have ICS protocols ports exposed to the public Internet. Furthermore, for the hosts that we identified as legitimately using ICS protocols, we found that a good subset (i.e., 64.3% corresponding to 108 hosts) also has general IT ports exposed to the Internet. By analyzing these 108 hosts, 11 of them were affected by at least a CVE, instead, 9 of them were affected by different CVEs with a CVSS score greater than 7.0.

RQ3: *Is IXP active-scanning based enumeration of hosts a good estimator of (vulnerable) industrial traffic use on the Internet?* Table 2.7 shows that *Shodan* finds three times as many hosts as

our method, while the *i* value we calculated indicates that only about 1.2% of the hosts collected by our framework were also detected by *Shodan*. This means that *Shodan* missed most hosts that are actually implementing ICS protocols. We can conclude that while *Shodan* returns a higher number of hosts, it is largely unable to identify hosts that actually use industrial protocols for legitimate applications (identified instead by the traffic analysis) even with a forced manual direction to the right hosts. This may be due to the presence of NATs or other mitigation mechanisms which block the direct scan on the host port.

2.2.6 Additional Analysis and Insights

In the following section, we reported further results we obtained beyond the research questions defined in Section 2.2.2. In particular, Section 2.2.6 provides a detailed analysis of the origin of scanning campaigns. Section 2.2.7 reports an analysis of the hosts implementing both Industrial communication and non-Industrial communication. Finally, Section 2.2.7 presents a validation procedure for the IXP on which we relied on.

Scanning Activities

In this analysis, we associated any scanning activity identified during the packet filtering process to the relative ICS protocol, according to the targeted port.

Scanned protocols. In Fig. ,2.13 we reported an overview of the protocols scanned by malicious actors. We identified a total of 442 different IPs performing scanning activities. The most scanned port was 5683 used by CoAP with more than 50% of the scan packets, followed by BACnet/IP, ATG systems, and DNP3. Almost 30% of the scan packets were designed with protocol-specific requests, like *readProperty* function for BACnet/IP, *GET /.well-known/core* for CoAP, *List Identity* for Ethernet/IP, *Session Initiate Request* for HART IP, and *Controller Data Read* for OMRON FINS. The remaining 70% of the packets consisted of 61% of UDP packets, 32% of simple SYN packets, 35% of RST, and less of the 1% for SYN-ACK and other combinations, which confirms an established TCP connection.

Scanning actors. We used the *Greynoise* platform [279] to tag the scanners as Malicious, Benign, or Unknown, according to their logged activities. *Greynoise* reported the 3.5% of the host as Malicious, 37.4% as Benign, and the remaining 59% as Unknown. For instance, a malicious actor was associated with a behavior indicating a Mirai or a Mirai-like variant infection, while another one was found opportunistically scanning for Siemens PLC devices. We also identified well-known services that periodically scan the Internet address space, such as *Shodan* and *Cen-*

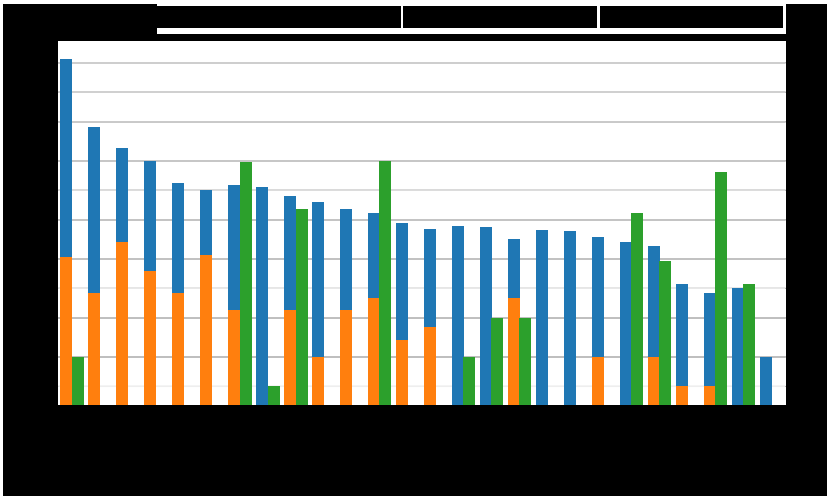


Figure 2.13: Protocols found among the legitimate and scanning ICS traffic. The Y-axis is log-scaled.

sys [266]. In particular, the Top-5 most frequent Scanner actors we identified are composed by the 35% by Censys, the 4.3% by *Stretchoid*, the 4.3% by *Shodan*, the 4.3% by *Net Systems Research*, and the 3.9% by *BinaryEdge*, while the 44.1% are unknown actors.

Malicious scanners. We observed that the ATG port is the most scanned by malicious actors, even more than well-known ICS protocols such as BACnet/IP, Modbus/TCP, and DNP3. Remote access to the control port of an ATG could provide an attacker the ability to reconfigure alarm thresholds, reset the system, and disrupt the operation of the fuel tank [284, 285]. However, since ATG is mainly used in the USA, this amount of scan traffic can be due to port 10001 being shared with other services. According to [286], several malware leverages this port to spread over the devices, furthermore, *Shodan* reports that almost all the devices with such exposed ports are network antennas. We reported in Figure 2.14 a heatmap with the origin of the IP associated with each scanning packet. The scanning activities come from 30 different countries. The 24.4% of the malicious actors come from China, 22% Netherlands, 12% Italy, 10% USA, and 7% Russia, while 8 other countries count for less than 5%. However, we must note that the authors of the scanning campaigns could also hide their source by using, for instance, a VPN. In this case, the IP origin represented in the map corresponds to the last VPN hop.

	Packet-based				Flow-based			
	Overall		ICS-to-ICS		Overall		ICS-to-ICS	
	Protocol	%	Protocol	%	Protocol	%	Protocol	%
Before IANA mapping	TLS	50.7	UDP	47.5	TLS	45	TCP	17.8
	HTTP	41.3	TCP	40.1	HTTP	23.7	TLS	15.8
	UDP	4.8	OpenVPN	8.8	TCP	18.4	UDP	8.2
	TCP	2.7	TLS	1.8	UDP	6.7	ICMP	4.1
	DNS	0.1	SIP	1.2	ICMP	3	RTCP	3.1
After IANA mapping	TLS	50.7	TCP	36.4	TLS	42.9	TLS	1
	HTTP	41.3	UDP	16	HTTP	22.6	TCP	0.6
	STUN	3.5	OpenVPN	8.8	XMPP	6.6	UDP	0.4
	XMPP	1.7	RSF-1 clustering	5.4	HP V.ROOM	5.5	Reserved	0.4
	UDP	0.8	ActiveSync	2.4	TCP	3.2	ICMP	0.3

Table 2.10: Top-5 non-industrial protocols.

2.2.7 IT Traffic

We identified that more than 91.6% of the industrial endpoints were also communicating via non-industrial protocols. This can be due to the use of more than one protocol by a single device, the presence of exposed IT services in ICS devices, or by the presence of a NAT on the network border that manages the traffic incoming or outgoing from the enterprise and manufacturing zone. In the first column of Table ,2.10 we reported that almost half of the traffic consists of encrypted TLS traffic, which could be due to the use of HTTP over TLS or to other secure communications such as VPNs. Moreover, HTTP covers 41.3% of the overall traffic, DNS covers 0.1%, while other interesting findings not mentioned in the table that represents less than 1% are OpenVPN, ESP, Wireguard, STUN, BitTorrent, FTP, and Telnet.

Due to the high amount of non-industrial traffic, we investigated if such behavior happened also between two endpoints of a legitimate ICS communication. We found out that 69.5% of the legitimate ICS endpoints also exchange non-industrial traffic. In order to have a clear view of the identified protocols, we applied a flow-based approach since the high amount of packets sent from a single host could significantly affect the statistics. As we can see in Table 2.10, this behavior is strongly evident in the amount of HTTP traffic, where just 23.7% of the hosts were using the HTTP protocol with respect to 41.3% found on the packet-based approach. Moreover, to reduce the number of the not precisely tagged TCP protocol, we mapped the lower port of each communication with the corresponding port registered by the IANA. This approach significantly changed the results of the flow-based approach for the two ICS-endpoints

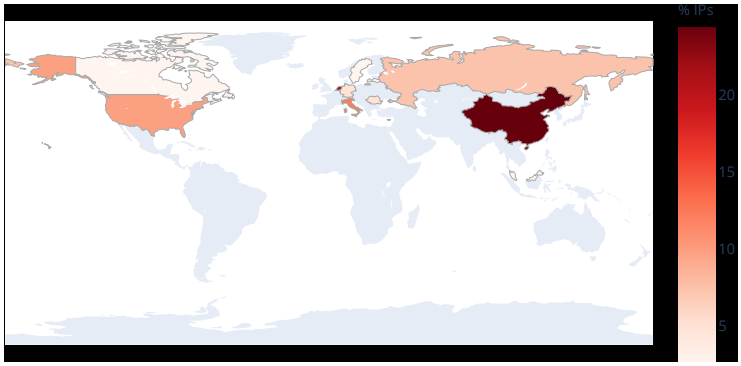


Figure 2.14: Heatmap of the malicious scanning activities origin.

analysis. We must also note that the low percentages obtained are due to the high frequency with which some ports changed within the same communication.

Validation

To verify the correct functioning of our environment, we injected self-crafted traffic into the IXP. To do this, we deployed a Modbus/TCP server and a *Mosquitto* MQTT broker in an Amazon EC2 server instance. Instead, within the IXP network, we deployed a synchronous Modbus/TCP and a MQTT client based on *pymodbus* and *paho-mqtt* python modules. During the communication between clients and servers, the Modbus client sends a *Write Single Register* request, and the MQTT client sends a *Publish Message* request. Considering that the 3-way handshakes were already accomplished, the overall amount of packets for each transaction is the following:

- Modbus/TCP:
  1. Client sends *Write Single Register* request;
  2. Server sends *Write Single Register* response;
  3. Client sends *TCP ACK* to server;
- MQTT:
  1. Client sends *Publish Message* request;
  2. Server sends *TCP ACK* to the client;

We sent 10 *Write Single Register* and 10 *Publish Message* requests per second for 24 hours. In this topology, the sFlow Agent samples just the traffic outgoing from the IXP network. It resulted in 2572852 packets exchanged, 1477915 outgoings, 346 of which were successfully sampled: 164 MQTT requests, 91 Modbus/TCP requests, and 91 Modbus TCP ACK. Furthermore, all the packets were correctly dissected by Wireshark. The sampling rate computed as the

ratio between the sampled packets and the overall outgoing traffic results in about $2.3411 \cdot 10^{-4}$ packets, which is what we expected, considering a sFlow sampling rate of 2^{-12}. This validation confirms the correct functioning of our IXP environment, the correct sampling functioning of sFlow, and the correct dissection function of the network packets by Wireshark.

2.2.8 Discussion

Comparison of Approaches. Our traffic analysis approach gives us a very different point of view with respect to *Shodan*. While *Shodan* collects data performing active port scanning and fingerprinting of the exposed services, having thus a wider overview of the exposed hosts, our approach allows identifying hosts currently active and communicating. Our analysis shows that *Shodan* identified just 2 hosts actually exchanging industrial traffic. There are three possible explanations for this result: the first one is that the systems were hidden behind a firewall, NAT, VLAN or were leveraging other port scanning mitigation techniques (e.g., filtering IP of scanning services), the second one is that the hosts indexed on *Shodan* were inactive during our measurement time window, and the third is that the packet transmission rate was low enough to avoid sampling detection (i.e., less than 1 packet per minute).

ICS Communications Security. The 98.8% of the ICS that we were able to identify were not recognized as ICS by *Shodan*. The 50% of the ICS protocols identified in the legitimate traffic were not implementing any encryption mechanism due to insecure protocols by design (e.g., Modbus/TCP) or bad configurations (i.e., MQTT, AMQP), exposing the whole systems to potential attacks. The 91.6% of hosts that exchange both industrial traffic and non-industrial traffic could be exploited by an attacker to investigate which protocols are used and what is the payload of the packets. For instance, the attacker could gather information about the servers that are commonly involved, analyze the DNS, FTP, and HTTP content to collect information about the employees, and use social engineering techniques to spoof them.

Limitations. Our approach presents some limitations. Therefore, as explained in Section 2.2.2, it is possible that sFlow was not able to detect some active ICS hosts since the accuracy of sFlow is dependent upon the sampling rate. The sampling rate also does not allow us to perform any encrypted traffic analysis since we could not record the entire communication flow. Moreover, the sFlow protocol truncates the packets at 128 bytes. This last limit, together with the fact that the Wireshark dissector does not support some protocols of interest, may lead to an incomplete analysis of the traffic. In our environment, the possible false negatives and false positives obtained are due to the third-party tool used in the extraction process (i.e., Wireshark, nDPI, and

Greynoise), and this makes it impossible to verify the ground truth. Finally, if a host implements a dynamic port range, instead of using the standard ports reported in Table 2.5, both our and *Shodan* approaches fail to correctly identify the host because both the approaches are port-based, leading to additional false negatives. More precisely, our approach relies on port-based inspection in two phases: i) during the preliminary port-based filtration; ii) during the Wireshark deep-packet inspection (i.e., Wireshark fails in identifying protocols that use non-standard ports).

2.2.9 Takeaway on ICS Exposure Measurement

The increasing using of insecure industrial protocols through the Internet exposed ICS and critical infrastructure to a wide range of cyber threats. Active scanning of the IP address space performed by services such as *Shodan* is a common practice to detect exposed ICS, however, it does not properly represent the real use of insecure industrial protocols. In this study, we addressed three research questions to investigate the current state of the art use of such protocols over the Internet by industrial systems. To do this, we proposed, implemented, and validated an analytic framework to detect legitimate industrial traffic communication and scanning activities based on a 31-day long sampled traffic capture collected at a local IXP. We compared our results with the information available on *Shodan*, proving that Shodan is not enough. In fact, while *Shodan* was able to identify a higher number of hosts, it detected only 1.2% of the hosts found by us. We also show that 64.3% of the hosts have IT services exposed, 11 of which have an alarming CVE vulnerability score, and that 75.6% of the industrial protocols implemented to communicate over the Internet do not implement any security feature.

Additional analysis, such as the analysis of the IT traffic, confirmed the convergence of the IT and OT networks in many systems (in our case the 91.6% of the identified hosts), providing a deeper point of view of the network architecture and security, such as the high rate of unencrypted IT traffic and insecure protocols. These network vulnerabilities could be exploited by malicious users, who constantly perform scanning campaigns, as our results show. Finally, we validated our system model by auto-injecting our traffic showing that it respects our assumptions.

2.3 ICS Honeypot Deployment

Most of the ICSs were designed decades ago to operate in an air-gapped environment, as recommended by the Purdue Mode [260]. Therefore, security practices, such as Encryption and Authentication, were not considered. However, due to the increasing digitalization and "smartification" of the processes, ICSs have been integrated with Internet connections to allow remote control or remote diagnosis operations. Different works measuring the current exposition and implemented protocols showed a dramatic security lack. In [10] the authors identify that 75.6% of the industrial protocols implemented to communicate over the Internet do not implement any security feature. Among the various security solutions proposed in the literature, honeypots represent an important protection mechanism that still needs contribution in the ICS field [77]. The goal of a honeypot is twofold: it can be used to fool the attacker, making them think they are interacting with the real system and collecting data about typical adversarial attack actions. However, developing an ICS honeypot is challenging since it should simulate both the network and the physical process with high fidelity. Honeypots are classified based on the level of interaction they offer to an attacker. *High-Interaction Honeypots* simulate all the services of the emulated machine and allow a high level of interaction with the emulated system. Instead, *Low-Interaction Honeypots* emulates the operating system and services provided by the simulated device, but due to the lower engagement it provides, it makes it possible to capture less information.

In this study, we present ICSpot, the first *High-Interaction* ICS honeypot that addresses the limitations of the physical process interaction of other ICS honeypots. Since ICSs are characterized by physical processes, the lack of such features can lead to an incomplete emulation of an industrial system. Once we developed the honeypot, we hosted it in an AWS server and a local IXP. We then analyze the results after one month of exposition, highlighting the differences between the two installation points. The source code of the honeypot is available on Github[3].

2.3.1 Related Works Limitations

The main challenge in designing an ICS honeypot is integrating a reliable physical process simulation to capture an attacker's attention. The majority of the proposed works lack the implementation of such a feature. ICSpot is built on top of the recent open-source HoneyPLC [287], which to the best of our knowledge, represents the most advanced ICS honeypot

[3] ICSpot' on Github: `github.com/ftrole/ICSpot`

Honepot/Features	Open Source	Network Simulation	Physical Interaction	Log	HMI
CryPLH [289]	○	◐	○	○	○
SHaPe [290]	●	○	○	●	○
Gaspot [291]	●	○	◐	●	○
GridPot [292]	●	○	◐	●	○
Conpot [293]	●	○	○	●	○
Antonioli et al. [294]	◐	◐	◐	●	○
HoneyPhy [295]	○	○	◐	○	○
DiPot [296]	○	○	○	●	○
S7commTrace [297]	○	○	○	●	○
Mimepot [298]	●	◐	●	○	○
HoneyNet [299]	●	●	○	●	○
HoneyPLC [287]	●	●	○	●	○
ICSpot	●	●	●	●	●

Table 2.11: Comparison among honeypots in literature. ● the feature is included. ◐ the feature is implemented but has limitations. ○ the feature is not implemented or not documented.

available. However, HoneyPLC does not include a physical process simulation. Starting from HoneyPLC, we extended its framework to include additional services and interaction with a MiniCPS-based [288] physical process together with an interactive HMI representing the physical process evolution.

ICS Honeypot Comparison. Table 2.11 reports a comparison between the different ICS honeypots. This table is an extension of the comparison presented in [287]. To the best of our knowledge, ICSpot is the first open-source honeypot that exposes an effective and interactive physical process. We also included an HMI representing the evolution of the process so that the attacker can interact with the physical process and look at the physics modification in real-time.

2.3.2 ICSpot Honeypot

We developed ICSpot by leveraging different existing ICS tools to obtain a complete ICS honeypot and address the current related works limitations. In particular, we built the ICSpot on the top of HoneyPLC [287]. HoneyPLC is a recent and effective honeypot which, as explained by the authors, implemented a complete set of functionalities, including multiple PLCs simulation, ladder logic capture capabilities, and low Honeyscore. We extended HoneyPLC to include an interactive physical process, additional industrial exposed services, and a log analysis interface.

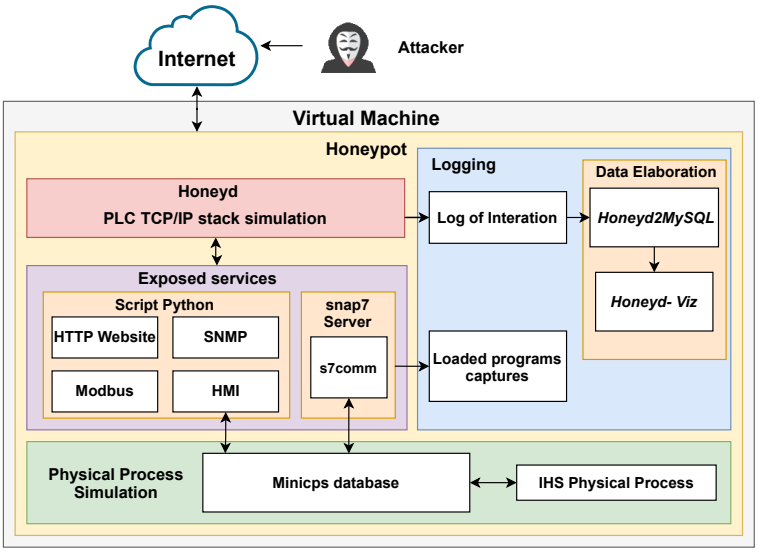


Figure 2.15: ICSpot Schema

Architecture

Figure 2.15 represents the components of ICSpot and how they interact with each other. The core tool we used to simulate the TCP/IP stack is Honeyd. Honeyd listens for requests addressed to the honeypots, simulates, and responds by imitating a list of services. Packets sent by Honeyd are modified by its personality engine: a component that simulates the behavior of a predefined operating system or a device chosen from the list of Nmap fingerprints. A configuration file lists the parameters defining the machine to be simulated, such as its MAC address, IP address, operating system, and services. Thanks to these features, Honeyd allows configuring and simulating complex honeypots without exposing the host machine to risks. ICSpot implements different industrial services described in Section 2.3.2. The most innovative is the S7comm server, and integrating the physical process (Section 2.3.2) has made it possible to create a highly interactive honeypot that aims to overcome the limitations of the work described in the previous chapter. Among the different PLCs offered by HoneyPLC, we decided to emulate Siemens Simatic S7-300. This choice is motivated by the lower Honeyscore obtained in [287] allowing, therefore, the best level of simulation and likelihood. The Shodan Honeyscore is a mechanism that flags a device as a honeypot with a confidentiality score. The score is calculated by analyzing features like open network ports or default known honeypot settings. All interactions with ICSpot are recorded, stored, and processed in a Logging module (Section 2.3.2).

Physical Process

The main contribution of ICSpot compared to its predecessors is integrating physical process simulation and the possibility of interacting with it. This is achieved by running a MiniCPS [288] simulation in the host machine. To this end, we leveraged the simulation proposed in the Industrial Hacking Simulator (IHS) project[4] by running it on the host machine of the honeypot in the background. IHS project is based on a simplified simulation of the SWaT water treatment process, provided by MiniCPS [288]. Two PLCs act on two pumps, which regulate the incoming and outgoing water flow in a tank, turning them on or off to maintain a constant water level. We integrated with ICSpot the possibility of using the S7comm protocol to write or read data in the MiniCPS database. In particular, we included the option of reading the PLC data blocks that include the current values on the MiniCPS database. Furthermore, the attacker can modify the state of the valves controlling the water flow. This allows interaction with the real-time undergoing water process. To avoid the crash of the process after disruptive interactions, we integrated a process adjustment when the water reaches too low or too high levels. The IHS project also offers a web interface showing data from the simulation. We modified such an interface to obtain a specific industrial HMI aspect and exposed it to deceive the attackers.

Services and Interaction

ICSpot includes a PLC website and a web interface reporting data about the simulated physical process, the interaction with the PLC memory exploiting the industrial protocols S7comm and Modbus, and device monitoring via SNMP protocol. In the following, we reported the details of the different services exposed.

HTTP. Generally, PLCs integrate HTTP web services to provide a list of functionalities. ICSpot implements a copy of the Siemens Simatic S7-300 website on port 80. On port 8000, ICSpot exposes an HMI interface that shows details of the physical process simulated: the tank water level, the status of the two valves, and the water ingoing and outgoing volume.

SNMP. The SNMP protocol, used to monitor devices connected to the network, is listening on port 161 UDP. The SNMP agent, when queried, responds with the Management Information Base (MIB), which reports information related to the PLC. This protocol is implemented with a python script from the SCADAs Honeynet project [299] and contains the MIB data of a real Simatic S7-300 PLC.

[4] IHS: github.com/CarlosLannister/IHS

Modbus. Modbus/TCP protocol enables the communication between industrial devices and supervisor computers in a master-slave paradigm. ICSpot implements Modbus/TCP using a python script from the SCADAs Honeynet project [299]. This implementation is considered low-interaction but allows connecting to the honeypot, which takes the role of slave, and reading and writing data, exactly as with a real PLC.

S7comm. The S7comm protocol was challenging to implement, mainly because Siemens does not release the official documentation. Following the example of HoneyPLC, we employed the snap7 library to implement an S7comm server listening on port 102 TCP. Based on HoneyPLC implementation, the S7comm server enables different functions. First, it simulates a Siemens Simatic S7-300 PLC, offering the possibility to read and write PLC memory blocks. Second, the S7comm server allows the honeypot to fool the Siemens Step7 Manager proprietary software. Since this proprietary software cannot detect the honeypot, it is reasonable to assume that no other application which can connect to it through the S7comm protocol can do it. Third, the S7comm server allows storing the programs injected by attackers in the virtual machine file system on which the honeypot is installed. This allows an accurate analysis of potential malware. These three functionalities derive from the HoneyPLC server implementation. Finally, we innovate the implementation of the S7comm server by offering the attacker data related to the physical process. The attacker can read the value of the volume of water present in the simulated tank by accessing the data blocks. The attacker can also modify the state of the valves in the process. This interaction allows the attacker to communicate with the physical process through the S7comm protocol. The S7comm server runs locally and is connected to the honeypot via the Honeyd configuration file.

Log Data Visualization

In order to make more usable and organized the data on the log generated by Honeyd, we leveraged Honeyd2MySQL[5] and Honeyd-viz[6] open-source tools. Honeyd2MySQL allows extraction of all the information from the Honeyd log and import them into a MySQL database. Then HoneydViz exploits the database created with Honeyd2MySQL to create a web interface showing useful statistics related to the collected data, such as the number of connections divided by ports, the IP addresses of the main attackers, their origin, and the connections established per day.

[5] honeyd2mysql: https://github.com/ikoniaris/honeyd2mysql
[6] Honeyd-viz: https://github.com/ikoniaris/honeyd-viz

2.3.3 Results

We installed two instances of ICSpot in two different environments. The first honeypot was installed in an AWS EC2 virtual machine, while the second honeypot was installed in a virtual machine in a network segment owned by VSIX [268] IXP.

We installed and exposed the virtual machines for one month. We report the data collected in the following.

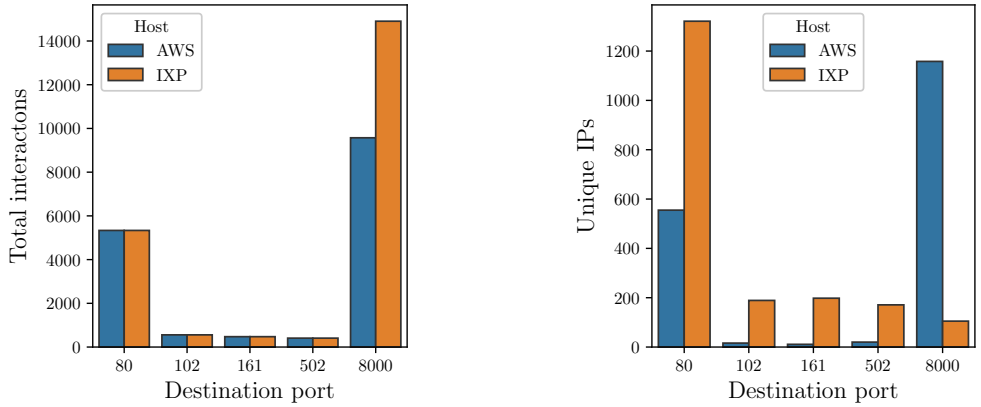
Interaction Analysis

Figure 2.16a compares the distribution of the interactions with the different exposed services for the two honeypot instances. We can see that the most requested were the website (port 80) and the HMI (port 8000). In particular, as reported in Figure 2.16b, we registered 2185 different IPs interacting with the port 8000 in the AWS instance and 105 in the VSIX case. Instead, we registered for port 80, 555 unique ISPs in AWS, and 1321 in VSIX. While such a high number of interactions was expected for port 80 since it is one of the most famous service ports, the high requests in port 8000 confirm the effectiveness of the HMI in attracting and increasing the attacker's engagement. Another interesting insight concerns the interactions through s7comm, attackers' second most exploited protocol to interact with ICSpot. These data indicate the effectiveness of ICSpot in implementing a service that faithfully reproduces that of an original PLC. We also analyzed the origin of the IPs used for interaction. We note that the scanning sources were identically distributed for both honeypot instances. This indicates that most malicious hosts that scan the Internet for vulnerable machines do not discriminate against associations or geographic areas but perform systemic network crawling. The nations from which the highest number of scans have been recorded are the United States and China, followed by Russia and India. Note that the original IP source represented may be hidden using VPN. In this case, the IP identified can represent the last VPN hop.

Interactions Origin

To analyze the presence of malicious actors among the IP sources, we leveraged GreyNoise[7]. Greynoise is a company that collects, labels, and diagnoses data and provides access to such information to users via API. Results between the two honeypot instances were slightly different. In both cases, 97% of the analyzed IP addresses are labeled in the GreyNoise database

[7] Greynoise, https://greynoise.io/



(a) Services targeted.

(b) Number of Unique IPs.

Figure 2.16: Services analyzed in the two ICSpot instances and number of different IPs connecting to each service.

"scanner". However, VSIX honeypot received a higher amount of different malicious scanning according to Greynoise classification (i.e., about 49%). Instead, 22% of the scanning was labeled as malicious in the AWS honeypot according to the Greynoise classification. This difference can be due to Shodan, which labels the AWS instance as a honeypot, reducing the interest of malicious scanners in analyzing such IP. The top-3 organizations with the highest number of malicious IP addresses in the VSIX case are Google LLC, Korea Telecom, and Chunghwa Telecom Co. Instead, in the AWS instance, the top-3 include DigitalOcean LLC, Google LLC, and WIND Telecom S.A. Both lists are followed by numerous ISPs from all over the world. Finally, GreyNoise can identify the actor related to an IP address, i.e., the entity actually using the IP address. The actor can differ from the organization to which the address belongs since it commonly happens that ISPs and companies offering cloud computing services or IP addresses block which are used for malicious activities. All the actors analyzed belong to hosts categorized as "benign" and represent legitimate organizations scanning the network to identify exposed and vulnerable services, sometimes also notifying the owners about the dangers they incur. The most frequent benign scanners in both the honeypot include Stretchoid.com, Censys, Bitsight, ShadowServer.org, BinaryEdge.io, and Shodan.io. All the malicious scanner actors are instead unknown by Greynoise.

2.3.4 Takeaway on Honeypot Deployment

We compared the feature of our honeypot with the related works proving its effectiveness and contribution. The comparison highlights the contribution of ICSpot in terms of physical pro-

cess simulation. We then exposed ICSpot on two different hosts and analyzed the interactions after a month of data collection. By leveraging Greynoise, we identified in the VSIX instance that 49% of the IPs interacting with the honeypots were labeled malicious, while in the AWS instance, 22%. The service installed on port 8000 has particularly attracted the attention of attackers labeled as malicious in the AWS instance. This demonstrates the interest of attackers in interacting with a physical process in an industrial system. We believe ICSpot can represent a building block toward implementing future honeypots with a higher degree of fidelity. In future works, we will extend the functions of ICSPot, like the PLC profiles, and compare the interactions received by ICSPot with other existing solutions.

3

Vehicles Security

In this chapter, we focus on a second application of CPS, the Vehicles. Vehicle security is a research area that spans from Intra-Vehicular Network (IVN) communication to vehicles-to-vehicles communication and vehicles-to-infrastructure communication. Modern vehicles have been integrated with hundreds of sensors and embedded systems to automatize the control process and to improve the users' driving experience. However, as in ICS these novel functions are built on top of legacy systems, inheriting all the underlying vulnerabilities and exposing the safety of the drivers to dangerous threats.

This chapter summarizes our research findings in the area of vehicle security. In particular, Section 3.1 overview the Controller Area Network (CAN) standard used for IVN communication and its vulnerabilities and proposes an authentication mechanism among the network nodes by exploiting the CAN bit-collision management mechanism and signal processing techniques [300]. Then, the second part of the emerging paradigm of EV. Nowadays, the climate change phenomenon is pushing toward a green transition, which has dramatically increased the sale of electric vehicles to replace traditional fossil-fueled vehicles. However, many works in the literature highlighted that the novelty of such EV systems led to new challenges from the security point of view [301, 302]. In this context, in Section 3.2, we present a novel relay attack on the EV charging process, which an attacker can exploit to charge its vehicle by charging the fee on a victim [303]. Lastly, in Section 3.3, we present a study on the possibility of profiling EV by analyzing their charging profile [304], opening to privacy exposure and leakage.

3.1 Internal Vehicle Network Security

The increase of interconnection and automation in automotive systems led to the development of new connection and communication standards among their components. The CAN, introduced in 1993, is the most common standard used for IVNs communication and other applications such as railways and industrial automation. A vehicular system uses the network's nodes, called Electronic Control Unit (ECU). These nodes communicate in broadcast through the CAN bus. Each ECU represents a single function in the machine, such as the engine control unit, airbags, or audio system. Modern vehicles include up to 200 ECUs, continuously exchanging messages to monitor the car's behavior. Due to the continuous integration of new vehicle services, the number of ECU will significantly increase.

CAN bus was initially created without considering security issues and was designed to operate in isolated LANs without any external interaction. In particular, CAN communication lacks encryption and authentication mechanisms, making it sensible to integrity, authenticity, and confidentiality attacks. Over the years, such lack of security properties was exploited by researchers to demonstrate the feasibility of spoofing attacks [305], command injection [306], eavesdropping [307], and replay attacks [308]. Despite that, in the last years, cars have been connected to external services such as GPS navigation systems, 5G connections, and Bluetooth. This open new vulnerability surfaces exploitable by hackers, also facilitated by the CAN bus's inherent vulnerability. One of the most famous events that attracted public attention on vehicle security happened in 2015 when two hackers showed that they could hack a Jeep [8] remotely. However, more recent studies also highlighted the exploitation of CAN bus with other attacks such as injection from outside the IVN via wireless channel [309] or ransomware injection through Over-The-Air updates (OTA) [310]. In 2019, the NIST issued the Special Publication 800-160 [311], which describes techniques and approaches to improve the cyber-resiliency of systems. NIST also considered self-driving cars as a system based on new emerging technologies among the examined use cases. The analysis showed that an adversary could subvert autonomous technology to divert and potentially crash the vehicle. Indeed, in the hypothesized threat scenario, the adversary installs malware and commands the CAN bus, thus gaining remote network control. One of the possible mitigations is validating data sent and received among network nodes.

To guarantee the system's safety and, consequently people's safety, new security mechanisms to protect CAN communication must be designed. The majority of the proposed solutions rely on modifying the CAN standard (e.g., cryptographic primitives [312, 313]) or integrating

additional nodes (e.g., Intrusion Detection Systems [314, 315, 316]). However, such solutions are sometimes impossible to implement due to market constraints and are difficult to integrate into current systems.

A common solution to hide information in communication is represented by covert-channels, introduced by Lampson in 1993 [317]. This technique exploits unintended channels to transmit information. In other words, in covert-channels specific data is embedded in a transmission medium that is not supposed to transfer such data. In literature, this technique has been implemented to exfiltrate information from networks without violating the expected behavior, thus without triggering IDS (e.g., in [318, 319]). Different works have employed covert-channel in recent years to implement security mechanisms, e.g., authentication [320, 321, 322, 323]. The strength of this approach is that by stealthy inserting the information within an existing medium, there is no need to modify the existing protocol.

In this study, we propose an authenticated key distribution system for CAN communication based on a combination of a watermark-based technique and jamming over a wired network. Indeed, most of the related work assumes the existence of a long-term secret shared among the manufacturer's different ECUs installed. We propose and validate a schema to dynamically and securely share or refresh this secret between the ECUs by considering the CAN network communication constraints. Similarly to [324], we define jamming over CAN bus the *intentional interference* operation on the signal aimed at destroying the communication between devices. CAN communication presents broadcast messages on a shared medium; therefore, jamming can block the reception of the message for all the listening nodes. Unlike previous works, our mechanism does not require any additional modification of the CAN standard or the CAN network architecture (e.g., adding new nodes). Our key distribution system exploits the traditional bit collision mechanism at the physical level to destroy part of the shared secret and reconstruct it at the receiver side. To do this, we use the Watermark Blind Physical Layer Security (WBPLSec) protocol which utilizes a jamming receiver in conjunction with a Spread-Spectrum (SS) watermarking technique [325]. The CAN specification (i.e., ISO 11898) considers three levels of the OSI model: physical, data-link and application layers. Figure 3.1 represents the standard layer according to the CAN specification and shows how we integrated the WBPLSec algorithm with it. The architecture we propose can be considered a Bump-In-The-Stack (BITS) [326], where *watermarking* and *jamming* are two atomic functions operating within the standard CAN stack. Instead, the architecture of the standard ECU is not modified; indeed, WBPLSec can be implemented at the software level in the ECU microcontroller to perform the watermarking while the CAN transceiver is used to perform the jam-

ming on the bus. By using this method to secure the communication, the original message to be transmitted is passed from the application layer (i.e., ECU micro-controller) to the WBPLSec function, which embeds an SS watermark in the information before passing it to the lower layers. More precisely, we use watermarking as a covert-channel. Instead, on the receiver side, the node selectively jams the transmitted message on the CAN bus using the jamming function added in the stack, making part of the communication unusable for the attacker.

As a result, our solution can be easily implemented in the existing CAN system by installing a transceiver on the ECUs.

WBPLSec algorithm was already applied with success on different communication means such as radio frequency [325], acoustic communications [327] and visible light communications [328]. To the best of our knowledge, this is the first work combining watermark-based communication with jamming over a wired network to implement a key distribution mechanism over the CAN network.

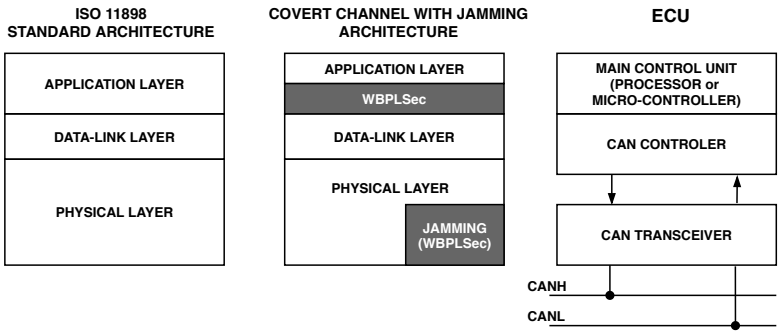


Figure 3.1: Integration of WBPLSec in the CAN protocol.

3.1.1 Background

In this section, we briefly recall the main concepts helpful in understanding the remainder of the study. In particular, in Section 3.1.1 we recall the CAN protocol, with a specific focus on the physical layer properties, which we exploit to perform the jamming. Then, in Section 3.1.1 we present the application of covert-channels and Watermarking techniques in the security field.

Controller Area Network (CAN)

CAN bus is the most common and widely diffused medium for IVN communication. CAN was initially released in 1986 by the Society of Automotive Engineers. However, nowadays is also implemented in other applications such as Building Automation and Transportation applications.

The CAN standard is defined in the ISO 11898, initially released in 1993 [329]. In particular, the physical layer characteristics of CAN are defined in the ISO 11898-2 [330], also known as *high-speed* CAN, and ISO 11898-3 [331], also known as *low-speed* CAN or *fault-tolerant* CAN. CAN enables communication among different nodes or ECUs on a differential bus. The communications rely on information broadcast, using a linear bus, star bus, or multiple star buses connected by a linear bus, terminated at each node by a resistance of 120 Ω. The base CAN frame consists of 108 bits, while the extended version of the can protocol has a length of up to 128 bits. Since the CAN protocol uses a broadcast transmission, every node receives every frame. To specify the receiver of a frame, there are two standard specifications. The CAN 2.0A defines 11 bit of identifier field, while the CAN 2.0B defines 29 bit of identifier. In this work, we consider the CAN 2.0A specification. The data field can contain up to 64 bits of information. Finally, the 16 bit CRC field identifies transmission errors and represents the protocol's only security feature. Figure 3.2 illustrates the CAN physical layer transmission model, which relies on dominant signals, or Controller Area Network High (CANH) (encoded as logical *0 s*) and recessive signals, CAN (encoded as logical *1 s*). An ECU comprises three elements: a processor, the CAN controller, and the CAN transceiver. According to ISO 11898-1 [332], which defines CAN data-link layer and physical signaling, the CAN controller and the CAN transceiver components of each ECU manage the transmission of data along the CAN bus. In particular, the CAN controller is responsible for assembling the frame at the data-link layer, while the CAN transceiver translates logical signals received from the CAN controller to the physical voltage level and vice versa. To avoid the frame collisions due to the broadcast transmissions and the shared medium, the CAN protocol implements a Carrier Sense Multiple Access/Collision Resolution (CSMA/CR) mechanism based on ID priority: the frame with the highest priority (lower arbitration ID) gain access to the bus, while all other nodes (higher priority arbitration ID) switch to a "listening" mode. Furthermore, if two nodes transmit a frame with the same priority level, if one node transmits a dominant bit and another transmits a recessive bit, in the resulting collision, the dominant bit "wins" between the two (i.e., logical AND combination). We leverage this collision mechanism to perform intentional jamming at the

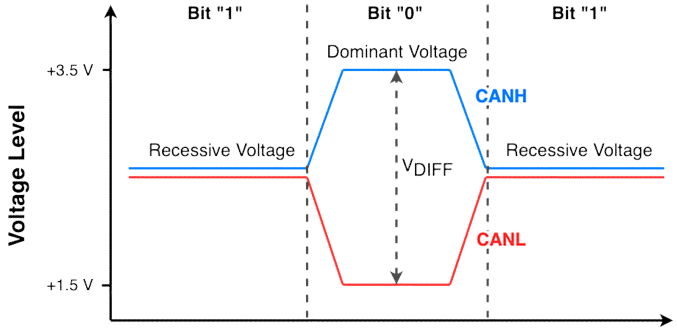


Figure 3.2: Collision management in CAN protocol (ISO 11898).

physical layer level. This will be detailed in Section 3.1.3.

Covert-Channels and Watermarking

Physical Layer Security (PLS) aims at securing communications by exploiting the physical properties of the communication channel. These techniques include processing the signal sent over a channel to obtain specific security properties without resorting to protocols or algorithms at upper layers than the physical one.

Protecting data from unauthorized access is indeed a fundamental aspect of communications. Cryptographic protocols answer this need. Only those authorized to participate in the communications have the credentials to interpret the protocol. Unfortunately, there are cases in which encryption is insufficient, and unconventional communication channels might solve the problem. In 1973, Lampson defined a *covert-channel* as a communication channel that is not intended for information transfer at all [317]. In literature, several contributions investigate how to implement effective covert-channels in different layers of the network. For instance, Hanspach et al. proposed the utilization of covert-channels to circumvent network security policies by establishing new communication paths [333]. Moreover, we can use hidden channels for various legitimate and non-legitimate purposes. While governments and companies can use these channels to protect their communications, cyber-criminals can also exploit this technology in the same way. In other words, covert-channels are just another way for digital communications through information hiding [334].

Watermarking is a technique used to hide or embed a signal into another signal, e.g., pictures and videos. PLS can be implemented with spread-spectrum watermarking techniques [335]. In particular, we implement the first equation described by Cox et al. [336]. We embed information in the original message through an SS watermark.

	CANAuth [338]	Car2X [339]	LCAP [308]	Libra-CAN [340]	CaCAN [341]	VeCure [342]	LeiA [313]	VatiCAN [343]	VuiCAN [344]	Fassak et al. [345]	TOUCAN [312]	TACAN [320]	CANTO [322]	Palaniswamy et al. [346]	Xiao et al. [347]	Mun et al. [348]	CAN-TORO [349]	Youn et al. [350]	S2-CAN [351]	AutoSec [352]
Omit long-term key distribution?	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Omit long-term key refresh?	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Requires additional nodes?		✓		✓	✓			✓			✓		✓	✓						

Table 3.1: Hypothesis in the long-term key distribution phase of the state of the art work on CAN authentication protocols.

Essentially, the SS watermark information is overlapped with the original information stream and travels within the CAN protocol frames. According to Lampson [317], a communication channel made in this way is a covert channel. Like other types of covert-channels, watermarking-based covert-channels allow embedding additional information without using any additional communication channel. In our case, this specific watermark (spread-spectrum watermark) encodes additional information within the original message without excessive extension of the payload. It is sufficient to extend the original message to accommodate the watermark. Without the watermark, we would have had to transmit the additional information as an additional message, which requires much more data to be sent.

Since confidentiality is one of the major concerns in any communication, researchers proposed many innovative solutions to improve it. Confidentiality through PLS was first considered in 1949 by Shannon, who presented the first application of information-theoretic secrecy [337]. On the other hand, cryptography, steganography, and watermarking are techniques that implement data secrecy using different paradigms than PLS. This work uses watermarking as a covert-channel that is not part of the CAN protocol to transfer information between two devices. Such covert-channel is an essential part of the PLS solution that we propose here.

3.1.2 Related Work

The insecure-by-design nature of the CAN protocol attracted the attention of many security researchers. Several works in literature proposed different approaches to increase the security of the transmission over the CAN bus ranging from the re-design of the protocol to the implementation of IDSs [353]. However, many challenges should be considered when designing a CAN solution. First, the ECUs have very low computation power, hard real-time constraints, and typically 8 bytes of payload. Thus, solutions with heavy computational requirements (e.g.,

Asymmetric Encryption) are not well suited. Second, the IVN architecture may sometimes be difficult or even impossible to modify after the product deployment. Therefore, solutions requiring additional nodes may be unsuitable and impossible to implement. These factors make the design of a new CAN security solution challenging. In the following, we briefly summarize the different security solutions proposed in the literature, grouping them by the implemented features.

Intrusion Detection Systems. IDSs represent a cost-effective solution to monitor the system's behavior passively or actively. This solution requires a preliminary training period and is specific to the particular system implemented since similar systems may differ in time constraints. Furthermore, the IDS approach requires installing the detector on a node with a high computational capability and access to the entire bus communication. IDSs are generally based on traffic analysis [354, 314]), physical invariants such as clock skew [355], bit-based [356] or frame-based [315], signal characteristics [357, 316]. If, on the one hand, IDSs represents a flexible solution able to prevent zero-day exploits, on the other hand, IDSs may require a modification of the CAN architecture and notable computing power.

Security-by-Design. The redefinition of the entire protocol allows for robustly protecting communication. However, this solution requires a profound modification of the existing protocol and, consequently, an adaptation of all the current systems currently implementing it. Different researchers attempted to introduce cryptographic primitives, such as authentication or encryption. The most common approach is to implement a Message Authentication Code in the CAN communication (e.g., [338, 308, 341, 313]). The two main approaches used in literature to send the Message Authentication Code in the CAN environment are to send it as a separate message or to truncate it. The drawback of the first approach is that it requires an additional frame for every message, slowing the entire communication. Instead, the second approach considerably reduces the already short payload available of the CAN frame.

Covert-Channel Authentication. The main idea behind these techniques is to embed secret and unique information in the CAN frame, exploiting protocol properties to authenticate the sender node or the message. The main advantage of this class of techniques is that authentication information is embedded in the frame that carries the data without requiring an additional authentication frame and without increasing the bus load. In [358], the authors present a security mechanism that exploits a covert-channel to implement a secure authentication between an ECU and the Monitor Node to generate shared session keys. Each ECU embeds unique authentication frames into CAN frames and continuously transmits them through covert-channels, which can be received and verified by an additional Monitor Node.

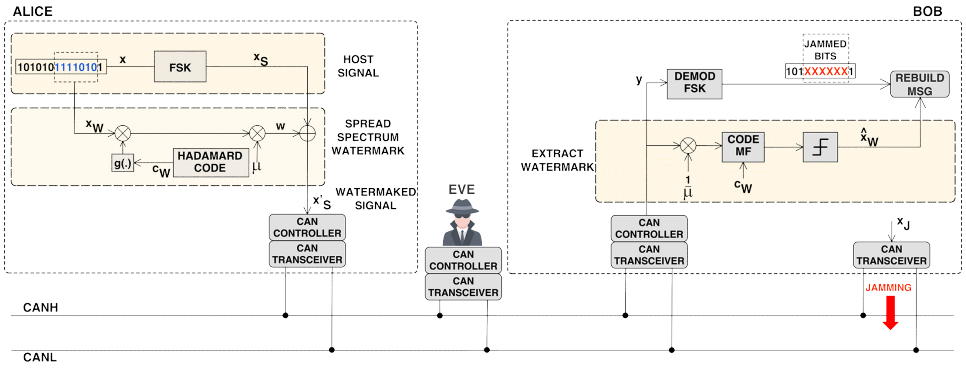


Figure 3.3: WBPLSec system model on CAN, the transmitter (Alice), the receiver with jammer (Bob) and the adversary (Eve).

Other works present covert-channel techniques to implement message authentication exploiting CAN transmission temporal features [321, 322, 323].

Limitation of Related Works

Although many algorithms introduce innovative authentication schemes for CAN networks, in most cases, these algorithms omit the implementation of a secure key exchange mechanism or assume that the key is installed in a secure storage partition at the vehicle production in *una tantum*. However, these assumptions are not flexible and do not consider the key refreshment during normal functioning in case of compromise. In Table 3.1 we report a summary of the assumptions on the key exchange of the various papers proposing authentication mechanisms over CAN bus (i.e., transmitter authentication or message authentication). We refer to *key exchange* as sharing the long-term secret that can be used, for instance, to validate the Message Authentication Code. To the best of our knowledge, only in Car2X the authors consider a dynamic key exchange phase at the vehicle boot time. However, this solution requires adding a central node to manage the entire sharing process, with a consequent modification of the CAN protocol communication phases.

Contrarily, our solution solves the lack of key exchange solutions by modifying the first phase of the communication at the vehicle boot time and without changing the existing architecture. Our methodology uses features already present in the CAN protocol and requires only modifying the source code in the ECUs.

Few works propose mechanisms to exchange long-term secrets in a CAN network. We recall these works and compare them with SENECAN in Section 3.1.7.

3.1.3 System Model

In this section, we present the system model for which SENECAN is conceived. In particular, in Section 3.1.3 we provide an overview of the key distribution mechanism based on jamming and watermarking. Instead, in Section 3.1.3 we describe the implementation of the WBPLSec algorithm in a real testbed to secure CAN communication. In general, the WBPLSec algorithm successfully applies to those media where jamming is possible. However, the effect is different between a wireless and a wired connection. In the former, the legitimate receiver can create a security region around him through jamming and watermarking [359]. In the latter case, the information is erased with jamming and then reconstructed with watermarking for all nodes connected to the bus. Furthermore, in wired communication, we no longer have the concept of a spatial region of security around the jamming node.

CAN Jamming Receiver

As previously described, CAN implements a CSMA/CR collision revolving mechanism based on ID priority. When a collision occurs, the dominant bit "0" wins over the recessive bit "1". What happens at the physical level is that, when transmitting a bit "0", CANH and Controller Area Network Low (CANL) are set respectively at $3.5V$ and $1.5V$, with a resulting $\Delta V = 2V$. Instead, when transmitting a bit "1", CANH and CANL are set respectively at $2.5V$ and $2.5V$, with a resulting $\Delta V = 0V$. This means that if both "0" and "1" are transmitted simultaneously, the bus will have a final $\Delta V = 2V$ and, therefore, all the nodes will see a "0". A node can transmit a series of "0" on purpose during another transmission to perform jamming by destroying a part of a frame.

Based on this principle, we develop an authenticated key distribution mechanism that exploits the jamming principles at the physical layer over wired communication to destroy part of the frame selectively. In particular, when Bob notes that Alice is trying to send him a frame, Bob starts jamming the communication so that only he can reconstruct the destroyed original message. To reconstruct the partially destroyed message, we implement the WBPLSec algorithm, which will be described in Section 3.1.3. Since this mechanism expands the size of the message, introducing an overhead on the transmission, it can be used to exchange a secret between the nodes (e.g., when the system starts) to encrypt future messages with traditional symmetric encryption algorithms (e.g., Message Authentication Code and AES), introducing security properties in the communications.

WBPLSec Algorithm applied to CAN

Since small sensors have limited computation power, in 2017, Soderi *et al.* developed the WBPLSec protocol for wireless communications as a valuable physical layer security standalone solution [325]. This technique combines watermarking as a covert-channel and a jamming receiver.

CAN is a robust vehicle bus for networking intelligent devices. Like the wireless sensors, the nodes of a CAN network have limited resources. The main intuition that inspired this work was to provide a new security solution for CAN network nodes fully compatible with the CAN standard stack. Indeed, in this study, we investigate the application of WBPLSec for the first time on a wired bus.

The scheme in Figure 3.3 depicts the proposed system model for the CAN protocol and is based on four main actions to transmit messages securely through the CAN: (1) spread-spectrum watermarking: part of the secret message is first modulated with a spreading sequence and then summed with the host signal; (2) receive jamming: Bob disrupts only part of Alice's message using an additional CAN transceiver; (3) message reconstruction: Bob knows the jammed part, he can reconstruct the clean message. Jamming does not affect the spread-spectrum watermark; (4) watermark extraction: we extract the watermark using a Code Matched Filter (CMF) that uses the same spreading code (i.e., c_W) used in transmission.

With reference to Figure 3.3, let's assume that the transmitter, i.e., Alice, and the receiver, i.e., Bob, would exchange a *secret message*, i.e., x, for supporting future confidential transmissions. WBPLSec transmits the information via two independent paths implementing a data splitting policy. Thus, the information is sent via a narrow-band signal and through the SS watermarked signal. Without loss in generality, in the rest of the study, we use the Direct Sequence Spread Spectrum (DSSS) for watermarking and a Frequency Shift Keying (FSK) as narrow-band signal[1].

Bob partially jams the watermarked signal, but this interference only affects the narrow-band signal because the SS watermark is immune to this type of interference. In this way, the watermark is used to recompose the entire signal [325].

The ECU transmitter combines the original modulated signal, x_S, with an SS watermark, w. In particular, we select the first equation for watermarking defined by Cox *et al.* [336] to build

[1] We can obtain similar results if we select Amplitude Shift Keying (ASK) for the host signal and DSSS for the SS watermarking.

the watermarked signal as follows

$$x_S^{'}(i) = x_S(i) + \mu w(i), \tag{3.1}$$

where $x_S(i)$ is the i-th bit of the continuous FSK transmitted signal [360], μ is the scaling parameter and $w(i)$ is the SS watermark. The signal watermarking is generated by using the traditional spread spectrum-based approach [361].

The host FSK modulated signal x_S can be expressed as

$$x_S(i) = A_a \sqrt{\frac{2}{T_{bs}}} \cdot cos(2\pi f_n i), \qquad \text{for } 0 \leq i \leq T_{bs}, \tag{3.2}$$

where A_a is the amplitude and T_{bs} is the symbol time. Then, $f_n = f_c + \frac{1-2b}{2T_{bs}}$ indicates the two frequencies needed to transmit two ($n = 1, 2$) binary digits (b). We assume equal to 0 the initial phase of FSK signal.

Recall that Alice and Bob want to exchange a secret x. This secret is modulated FSK to create the host signal while a part, x_W, is used to create the SS watermark. We select the last N_W over N bits from x to create x_W, which is given by

$$x_W(i) = \begin{cases} x(i), & \text{for } N - N_W \leq i \leq N, \\ 0, & \text{elsewhere.} \end{cases} \tag{3.3}$$

The DSSS watermark signal can be expressed as

$$w(i) = \sum_{k=-\infty}^{+\infty} \sum_{j=0}^{N_c-1} g(i - kT_b - jT_c)(c_W(i))_j (x_W(i))_k, \tag{3.4}$$

where $(x_W(i))_k$ is the k-th bit of the watermark signal. $(c_W(i))_j$ represents the j-th chip of the orthogonal Pseudo-Noise (PN) sequence. $g(i)$ is the pulse waveform, T_c is the chip time, and $T_b = N_c T_c$ is the bit time.

Next, the watermark embedding step in the host FSK signal, equation (3.1), occurs with w signal modulating a carrier frequency close to the range of the f_c used by FSK. The watermarking operation is performed by the ECU sender (Alice) before the transmission over the bus. Figure 3.3 shows how the transmitter embeds the original message into the host signal. Then, the processor passes the watermarked signal to the CAN controller that splits it into CAN

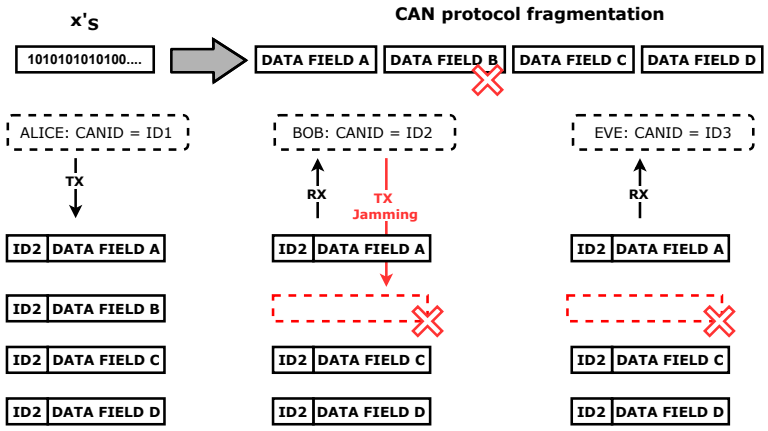


Figure 3.4: WBPLSec jamming operational diagram on CAN.

frames. According to the dimension of the frame allowed by the CAN protocol, each frame's data field contains up to 64 bits.

As shown in Figure 3.4, while Bob is receiving CAN frames with his arbitration ID, he can jam at most $M = N_W$ bits because these N_W bits are transmitted through SS watermark, i.e., through the covert-channel. *Jamming a signal in the CAN protocol* is equivalent to transmitting a dominant bit on the bus, i.e., the bit "0". For simplicity in our experiments, we destroy the entire frame, and therefore we destroy blocks of 64 bits of the data field. The effect of the jamming will be to destroy the whole frame (i.e., all the fields), canceling it from the bus for all the receivers, including the attacker (Eve), connected on the same bus. Note that the distance between Eve and the other nodes does not influence the attack scenario since the propagation speed of the information can be assumed as the velocity of light. The receiver has two *fingers* as shown in Figure 3.3. Bob demodulates the FSK signal with the first, whereas he recovers the SS watermark with the other. Unfortunately, part of the demodulated information is corrupted due to the jamming. However, Bob can get a clean signal by replacing corrupted bits with the information he conveys via the SS watermark that is immune to any jamming interference. In contrast, the eavesdropper cannot remove the interference because he does not know the jamming characteristics and which packets frames are affected.

Note that we assume perfect synchronization between Alice and Bob so that the two ECUs know when the secret is sent. The synchronization between the nodes will be discussed in Section 3.1.4. In addition, we know that the legitimate receiver should not jam more bits than those Alice embeds into the watermark, i.e., $M \leq N_W$. Indeed, the WBPLSec algorithm replicates a portion of the original information through the watermark, i.e., N_W of the total N bits.

Parameter	Value
FSK frequencies[1]	9.75 kHz, 10.25 kHz
DSSS carrier frequency	12 kHz
Samples[2] per FSK symbol (sl)	80
Samples[2] per DSSS symbol (sl)	80
Number of bits FSK payload (N)	128, 384
Number of bits FSK preamble[3]	128
Max. Number of jammed bits (M)	$2 \div 24$
Number of bits to create the watermark (N_W)	8, 24
Number of bits watermark preamble	8
Watermarking scaling parameter (μ)	0.3
DSSS Processing Gain (G_p)[4]	4, 8, 16

[1] Up-converted using 10 kHz carrier frequency.
[2] We assume the same symbol length for FSK and DSSS signals.
[3] It consists of the preamble and a synchronization sequence.
[4] Using Hadamard PN code.

Table 3.2: Parameters for WBPLSec experiments over CAN.

In this way, N_W will also be the maximum number of usable bits that we can use to reconstruct the information destroyed through jamming, i.e., M bits. So, in summary, to not lose information due to jamming, we must have $M \leq N_W$.

3.1.4 Protocol Implementation

In the following, we propose the first implementation of WBPLSec over the CAN bus in Section 3.1.4, discussing the implementation choices to prove the reliability of this algorithm in a wired bus. Then, in Section 3.1.4, we present an analysis of the performance of this algorithm in a virtual environment.

Functioning

The key distribution phase is performed entirely in a dedicated time slot. The total length of such a slot corresponds to the total number of single key distribution sessions, i.e., the unordered number of a couple of nodes communicating with each other (i.e., if Alice sends messages to Bob and vice versa, this is a distribution session since the shared key is symmetric) multiplied by the time of a single key distribution. This requirement will be discussed in Section 3.1.7. In this phase, every node has a set of defined time slots to send the keys to all the other nodes that are supposed to communicate within successive phases. This dedicated time slot is necessary to obtain a communication order and synchronization in the key exchange to avoid multiple simultaneous jamming from more nodes. The organization of this phase depends

on the specific number of nodes composing the network and their communication scheme. Therefore, this phase can be configured by the system manufacturers. We assume that each ECU stores a list to associate every other ECU to a particular KEY-ID number. The network manufacturers configured such a list and stored it in a secure and tamper-resistant memory area. The reason behind introducing the KEY-ID is that the standard CAN frame does not allow to specify the sender node. Thus, the receiving node cannot associate the corresponding decryption key. By including the KEY-ID in the message sent, every receiving ECU knows who is currently transmitting. Consequently, the receiving ECUs can associate the received symmetric key with the sender KEY-ID and use it for future communication. Future communication must reserve part of the data field (i.e., 8 bits) to include KEY-ID to authenticate the sender. Furthermore, since the CAN broadcast communication is based on an arbitration ID, each ECU with the same arbitration ID should also share the same key for a specific KEY-ID. More precisely, if a node sends a message to a specific arbitration ID, all nodes with such an arbitration ID will receive the message. Therefore, all the receiving nodes with the same arbitration ID must know the same key to decrypt the message.

At high-level, the steps required to perform the communication are reported in Figure 3.5 and described as follows.

1. Alice wants to securely send a key to Bob in the predefined time slot. The time slot length is fixed and corresponds exactly to the estimated time to complete the key transmission, i.e., complete the steps below.

2. Alice FSK modulates the original key and adds the signal's watermark. Furthermore, to prevent the retransmission due to the frames collision, she disables the default transmission queue capabilities (i.e., if the bus is busy, the ECU will not re-transmit the frame later). The KEY-ID of Alice is inserted into the original message to authenticate the nodes correctly. We decide to reserve the last 8 bits of the preamble to the KEY-ID of the transmitting node to support up to 256 different nodes on the bus. However, this parameter can be easily extended.

3. While Alice is transmitting frames, Bob, who knows by design that he is the receiver in this time slot, starts jamming using the additional transceiver to destroy a random number of frames. Since Alice is a node with no transmission queue, all the jammed frames will be lost and not re-transmitted once jamming is finished. While jamming, Bob stores the received frames in a queue.

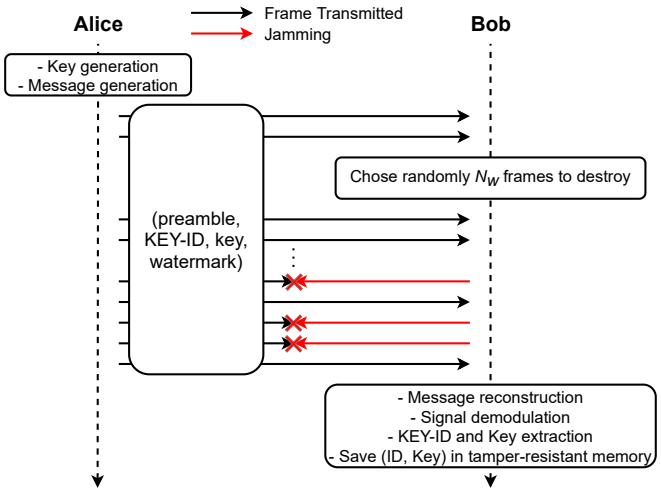


Figure 3.5: Protocol steps representation.

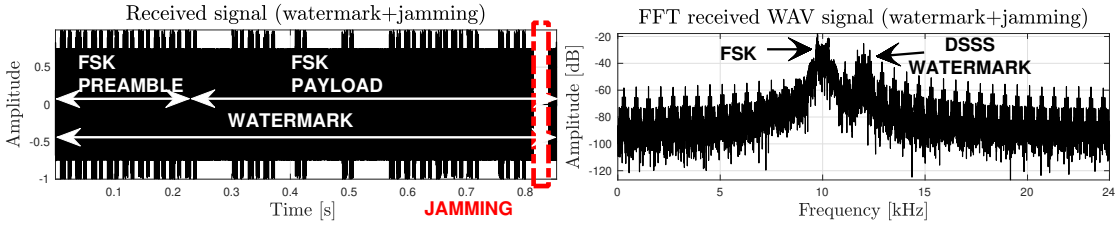
4. Bob lost frames during step 3), but he is the only one with the knowledge of the jammed bits position and consequently able to recover the key transmitted by Alice. By merging the frames in the queue in order, Bob can reconstruct the original message with the KEY-ID and the key thanks to an FSK demodulator and a watermark extraction.

5. Lastly, Bob extracts and saves the couple (KEY-ID, key) received from Alice in a tamper-resistant memory.

This procedure can be performed in the first phase of the network creation (e.g., vehicle pre-sale) or successive sessions to *refresh the keys* (e.g., during the vehicle revision or OTA updates), and it is repeated for every ordered couple of nodes. Again, this requirement will be discussed in Section 3.1.7. The procedure discussed allows the node to send or refresh the key securely. Furthermore, the schema enables the distribution of the keys in a one-way transmission way without requiring a response by the receiving node.

Simulation Performance

In this section, we evaluate the performance of WBPLSec over CAN bus in terms of Bit-Error-Rate (BER) of the watermarked signal x_S' and the watermark w. We simulate in Matlab 2021a the BITS architecture using the parameters in Table 3.2 to generate the signal with watermark and the relative jamming during transmission. We simulate the algorithm's key distribution behavior, including the jamming interference that produces the cancellation of CAN frames.



(a) Watermarked received signal with jamming in the TD ($M = 16$).

(b) Watermarked received signal with jamming in the FD ($M = 16$).

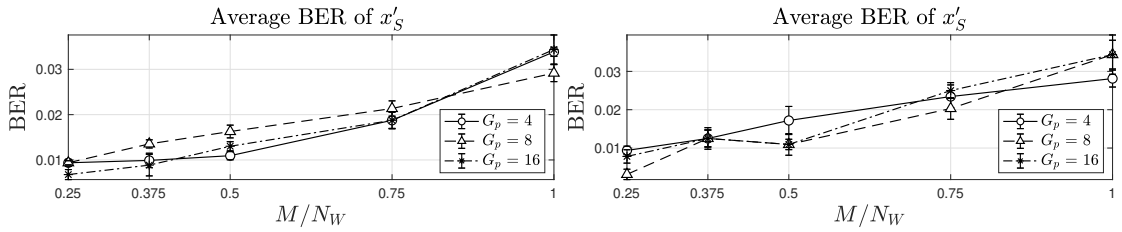
Figure 3.6: BITS architecture over CAN.

Considering the jamming on the bus, the simulations we perform are very similar to a real-world case. As described in the next section, the effect of jamming is to erase the frames on the bus, and therefore we can easily simulate it in a virtual environment. In this scenario, we can evaluate the effect of jamming on multiple transmissions by varying the length of the original message (N), the SS watermark (N_W and G_p), and the number of jammed bits (M).

The overall intuition of the BITS architecture is represented in Figure 3.6. There, we show the watermarked signal (x_S') in the Time Domain (TD) (Figure 3.6a) and in the Frequency Domain (FD) (Figure 3.6b). The TD white arrows denote the preamble and the payload, whereas the watermark is spread over the entire signal. Also, in red, we indicate the information canceled by the jamming interference produced by the legitimate receiver.

In a critical view of our contribution, we should note that with this number of samples per bit, we get up to 40960 samples when x_S' is 512 bits long. In fact, we compute x_S' with (3.1), where we represent each bit with 80 samples which include the oversampling rate needed for simulations. By default, Matlab stores all numeric values in a double-precision floating-point, representing each sample with 64 bits. Nevertheless, 64 bits is also the length of the CAN protocol data field. Therefore, we can transmit one sample at a time per CAN frame. This certainly increases transmission. To give an idea of what could be a real-world transmission time with this schema, we report Table 3.5 with the theoretical transmission times calculated with the nominal speeds supported by the CAN standard.

We verify the transmission quality through the WBPLSec algorithm on CAN in terms of BER. Figure 3.7 shows the average BER and its accuracy (i.e., the standard error of the average) of the watermarked signal x_S' for two different message lengths and different watermark settings. In particular, we consider for both scenarios the same ratio $\frac{M}{N_W}$. Under these settings, we can see that the more bits are destroyed, the greater the BER of the FSK signal. And this is the effect intended by Bob, i.e., disrupting the communication with his jammer so that the attacker



(a) $N = 384$ bits, $N_W = 24$ bits **(b)** $N = 128$ bits, $N_W = 8$ bits

Figure 3.7: Watermarked signal (x'_S) average BER when jamming, i.e., M and G_p change.

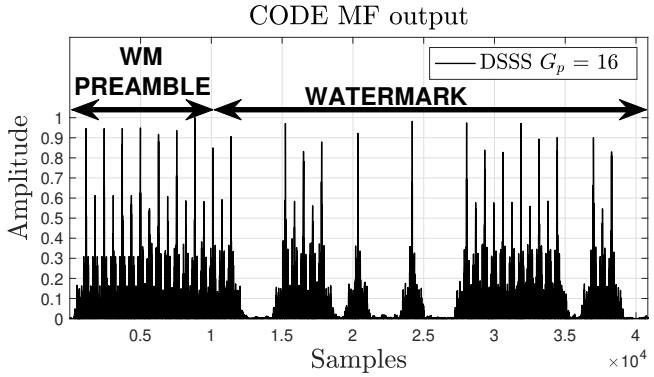


Figure 3.8: Extraction of the watermark using a code matched filter to the spreading code ($N = 384, N_W = 24, M = 16$).

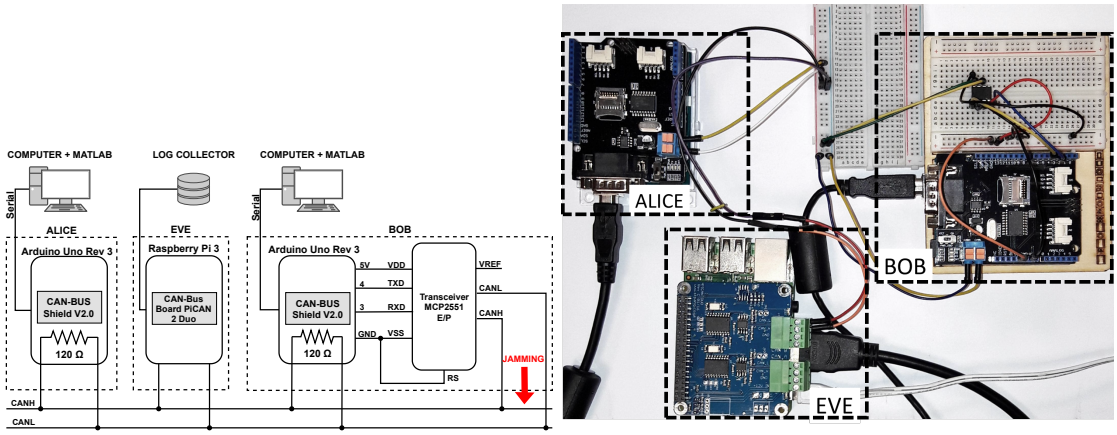
cannot receive the information sent on the channel by Alice.

Before evaluating the number of errors in the watermark after jamming, we extract it through a CMF. This process is performed by computing the normalized statistics [361, 325]

$$r \triangleq \frac{\langle y, c_W \rangle}{\langle c_W, c_W \rangle}, \tag{3.5}$$

where the y is the received signal by Bob, as shown in the system model represented in Figure 3.3, and c_W represents the Hadamard PN sequence. We assume $\langle c_W, c_W \rangle = 1$, i.e. PN sequences have unit energy. Figure 3.8 shows the output of the CMF, where we can observe the eight preamble bits used to synchronize with the start of the watermark. It is known how the matched filter maximizes the ratio at the output of the detector. And, the detector is the same one that was introduced with the traditional SS watermarking [361, 325], and the estimation of the embedded bit is given by $\hat{x}_W = sign(r)$.

Finally, we observe that the watermark also undergoes interference. Figure 3.10 shows the average BER and its accuracy (i.e., the standard error of the average) of the watermark w. Nev-



(a) Conceptual experimental setup.

(b) Real experimental setup.

Figure 3.9: Experimental setup.

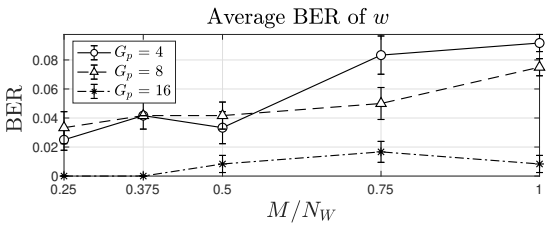
ertheless, in Figure 3.10, we can see how the information conveyed through the watermark is more immune to jamming. This is due to an inherent feature of spread-spectrum technology that is immune to narrowband interference, such as Bob's jamming. In particular, when the processing gain, or G_p, is at least equal to 16, the BER of w is less than 3%. This BER result is absolutely in line as it is obtained in other implementations of WBPLSec [327].

3.1.5 Testbed Validation

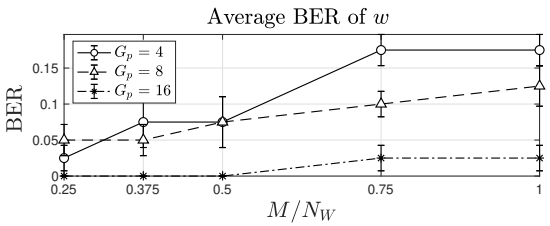
To test and validate the SENECAN functioning, we implement a scaled-down architecture composed of three nodes. In this section we describe the experiments we perform to reproduce the SENCAN distribution phase in a simplified real-world scenario. In particular, in Section 3.1.5 we describe our testing environment and the devices used. As an example scenario, to perform the tests we set $N = 512$ bits (i.e., 40960 samples in double precision, hence, 40960 frames), $G_p = 16$ and we jam 16 consecutive bits ($M = 16$), which correspond to jam $16 \cdot 80$ frames.

Equipment Setup

The architecture used to perform the experiments, represented in Figure 3.9a, is composed of three nodes: Alice, Bob, and Eve. Alice and Bob consist of an Arduino UNO board and a CAN shield from SeedStudio, respectively. Each CAN shield consists of a Microchip MCP2515 CAN controller and an MCP2551 CAN transceiver. Furthermore, Bob contains an additional

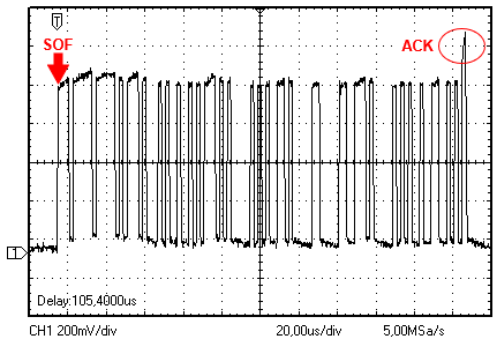


(a) $N = 384$ bits, $N_W = 24$ bits

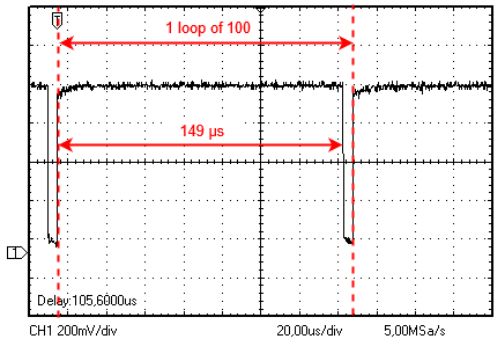


(b) $N = 128$ bits, $N_W = 8$ bits

Figure 3.10: Watermark (w) average BER when jamming, i.e. M, and G_p change.



(a) CANH and CANL under normal conditions.



(b) CANH and CANL under jamming condition.

Figure 3.11: Representation of the CANH and CANL under different conditions. The first bit is the start-of-frame that denotes the start of the frame. The images are collected from the oscilloscope and a differential probe.

MCP2551 CAN transceiver that we use to perform the jamming interference, as the implementation by Palanca *et al.* [362]. Instead, Eve consists of a Raspberry Pi 3 and a PiCAN2 DUO board that provides a CAN bus interface. The decision of using a Raspberry Pi to represent Eve is motivated by the necessity of easily collecting the traffic passing through the bus with a third node (i.e., Eve). To do this, we install Wireshark on Eve. Wireshark is a network protocol analyzer able to inspect and analyze CAN bus frames, and it gives us the possibility to monitor and collect traffic over the bus.

To perform the experiments, we set the transmission rate at 500 kbps, the common bitrate of high-speed CAN. Whereas the high-speed CAN (i.e., ISO 11898-2 [330]) network requires only two 120 Ω terminal resistors, we use the resistors embedded in the Arduino CAN shield.

Implementation

The implementation of the experimental architecture is depicted in Figure 3.9b. In our setup, Alice communicates with a base frame format over the bus and uses a fixed identifier for all

Algorithm 1: Jamming loop in Bob

Input: N; N_W; s_l
Output: Deletion of $N_W \cdot s_l$ CAN frames.
jam_f := Random(1, $N_W \cdot s_l$); ▷ Frames jammed
$jam_s := (N - N_W \cdot s_l - jam_f)$; ▷ Start of jamming
$c := 0$; $frame_{rx} := 0$;
while $c < jam_f$ **do**
    if $frame_{rx} \leq jam_s$ **then**
        $frame_{rx} = frame_{rx} + 1$; ▷ No Jamming
    else
        WriteBit(0, Port, Pin); ▷ Write Dominant Bit
        Wait(149 μ);
        WriteBit(1, Port, Pin); ▷ Write Recessive Bit
        $frame_{rx} = frame_{rx} + 1$;
        $c = c + 1$;
    end
end

the frames. To perform the *jamming*, Bob occupies the channel with the dominant bit to prevent communication with all the other nodes in the network. However, the default driver of the CAN transceiver MCP2551 prevents holding a dominant bit for more than 1.25 ms. In this case, the controller called TXD Permanent Dominant Detection inside the MCP2551 transceiver will disable the CANH and CANL output drivers to prevent data corruption on the CAN bus. We decide to address this issue by introducing recessive bits during the jamming. The CAN frame implements the practice of bit stuffing. An example of the experimental setup frame is shown in Figure 3.11a. The overshoot at the end of each frame depends on the acknowledgment (ACK) bit, which is dominant, and all the nodes drive it except for the one transmitting the frame. Because more nodes drive the bus dominant, a higher voltage is observed on the bus at the end of each frame. Communication of Alice is handled in order not to create a frame queue. In particular, when Alice cannot transmit the frame (e.g., jamming in progress), she will refuse to transmit the next frame. Moreover, the data field of each frame is transmitted to Alice via serial communication from Matlab. Implementing serial communication with Matlab arises from the need to easily store a significant number of samples and facilitate data processing. Each double value of the bitstream stored in Matlab is communicated to Alice in a little-endian representation, and an 8 ms delay is added between the serial writing of this value. For a correct reception of the bytes from serial, Alice will have to wait for 1 ms for each byte before forwarding the frame on the CAN network. Therefore, due to the delay introduced by the implementation setup in Matlab, frames are sent on the bus every 15 ms. Bob is also connected independently to a device with Matlab to store and process the values received from Alice. This time, the serial communication is directed from the Arduino board to Mat-

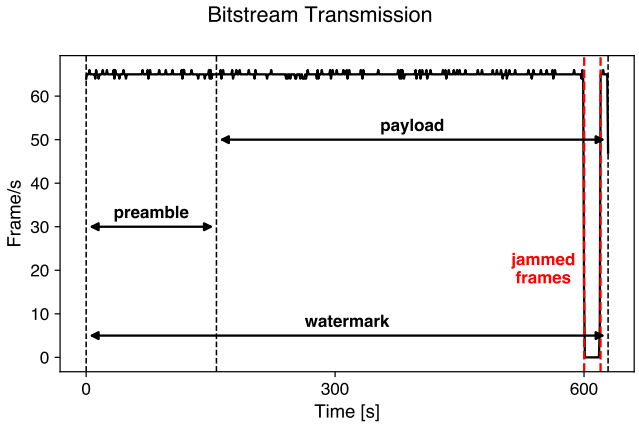


Figure 3.12: Bitstream transmission in terms of frame per second. The Figure shows the different fields of the bitstream and the jamming window ($N = 512$, $N_W = 24$, $M = 16$, $G_p = 16$).

lab. The Arduino board related to Bob can read the CAN frames through the dedicated CAN shield, count the frames to activate the jamming via the independent MCP2551 transceiver, and print the received frames on the serial line. Once Bob receives the predetermined frame before jamming, he starts to occupy the channel with the dominant bit for the entire time window necessary to destroy the predefined number of frames as sketched in Algorithm 1. As mentioned, Bob cannot write a dominant bit for more than 1.25 ms, but frames arrive every 15 ms on average. For this reason, Bob alternates recessive bits to cover the time window of a single frame. In our setup, we decide to keep the dominant bit for 149 μs a hundred times to destroy a single frame, as shown in Figure 3.11b.

3.1.6 Secuirty Analysis

In this section we analyze the security of the SENECAN key distribution mechanism. In Section 3.1.6 we discuss the possible threat model and attacker capabilities based on the most common assumption in the literature. Then, in Section 3.1.6 we analyze the security properties which our methodology ensures.

Threat Model

To evaluate the robustness of our approach, we assume that an attacker can access the communication over the CAN bus and perform different actions at every moment of the key distribution process. In particular, we consider the scenario where an attacker physically accesses

the network during the distribution process via the bus. In this setting, the attacker can both passively sniff or modify the communication. As previously described, we assume that a list to associate every ECU with a corresponding KEY-ID is stored in every ECU's secure and tamper-resistant memory. This assumption is consistent with the other works in this field, where the long-term secret is assumed to be securely stored during the production phase. Unlike the other works, we assume that the pre-shared secret consists only of a list of KEY-IDs used to associate the transmitting ECU with the corresponding KEY-ID. This corresponds to a *weak secret*. According to the literature, the common attacks that an attacker can perform in a CAN communication are the following.

- **Message Injection Attack**: The goal of this attack is to send a customized and malicious frame to an ECU to compromise the authentication phase.
- **Replay Attack**: This attack aims to reuse a previously transmitted and sniffed frame in successive communication to replicate the legitimate transmission.
- **Message Modification**: In this case, the goal is to modify the frame during transmission. This can be done in real-time by selectively jamming the communication to flip some bit from 1 to 0, or by collecting the entire message, modifying it, and re-transmit it.
- **Eavesdropping**: This attack aims at passively sniffing the communication to collect the traffic during the key exchange and, for example, to analyze it in a second moment to compromise future communication.
- **Masquerade Attack**: The attacker can modify and reuse the transmitted KEY-ID to impersonate an ECU during the exchange phase.
- **Adversarial Jamming**: An attacker can leverage a disturbing interference (e.g., jamming) similar to our mechanism to disrupt the communication, destroying one or more frames. In this way, an attacker can obtain a denial of service by removing part of the exchanged message.

Security properties

In the following, we discuss the security features of SENECAN. For each property, we then state which of the attacks described in the previous section can prevent. As previously mentioned, our solution can exchange a long-term shared secret or refresh it. This procedure ensures the following security properties.

	Car2X [339]	Mueller et al. [324]	Jain et al. [363]	TRICKS [323]	SENECAN
Previous Secret Independence		✓	✓	✓	✓
Replay Attack Resistance		✓	✓	✓	✓
Confidentiality	✓	✓	✓	✓	✓
Integrity			✓		✓
Authentication	✓		✓	✓	✓
One-Way Transmission		✓	✓		✓
No Additional Node		✓		✓	✓
Implementation	✓			✓	✓

Table 3.3: Different Key Distribution approaches for CAN.

Previous Secret Independence. If the key currently used is compromised, SENECAN allows to refresh the keys independently of the previous keys. Therefore the information obtained by an attacker could not affect the new secret exchange. The only requirement of the procedure is that the association of the KEY-ID with each other node is not compromised. This property allows preventing *Message Injection Attacks*.

Replay attack Resistance. It is probably the most challenging feature required for an offline authentication schema. Generally, this feature requires a nonce exchange and synchronization by the different nodes. This is a very complex constraint to obtain, as explained in [364]. Different works address this problem with an additional node and a constant synchronization protocol. In our case, replay protection is ensured because Bob chose and destroyed a random frame set. This information is known only by him, and the next time a new secret is shared, the jammed frames will be different. This will avoid any attempts by Eve to reuse an old frame. This property allows preventing *Replay* and *Message Injection Attacks*.

High Confidentiality. The confidentiality of the communication is achieved thanks to the jamming phase, which allows only Bob to know the jamming points and consequently reconstruct the original message. Since every bit of the original message is expanded to 80 frames, 64 bits payload each, to brute force the original message, the attacker requires to compute $2^{M \cdot 80 \cdot 64}$, where M is the number of the bit jammed during the frame transmission. The security level of the schema is consequently $M \cdot 80 \cdot 64$. This property allows preventing *Eavesdropping*.

Integrity. there are two possibilities for an attacker to modify the original message: in real-time or in a secondary moment. While the first approach is unfeasible due to the anti-replay property, the second would compromise the watermark, raising errors during the demodula-

Bitstream	Samples	Transmission Time	Time[1]	Optimised Time[2]
256 bits	20480[3]	7200 bps[4]	307.2 s	76.8 s
		500 kbps	4.42 s	2.21 s
		1 Mbps	2.21 s	1.11 s
		10 Mbps	0.22 s	0.11 s
512 bits	40960[3]	7200 bps[4]	614.4 s	153.6 s
		500 kbps	8.85 s	4.42 s
		1 Mbps	4.42 s	2.21 s
		10 Mbps	0.44 s	0.22 s

[1] One sample in double precision for each CAN frame.
[2] Two samples in single precision for each CAN frame.
[3] 80 samples for each symbol.
[4] Proof of concept with 15 ms delay for each frame.

Table 3.5: Transmission time comparison.

tion and watermarking verification. This property prevents *Replay*, *Message Injection Attacks*, and *Message Modification Attacks*. Furthermore, it can also identify *Adversarial Jamming* attempts.

Authentication. We assume that the list of the KEY-IDs of each node is stored in a secure memory area of the ECU and therefore cannot be modified. By adding this trusted and confidential information to the message exchanged, we ensure the authentication of the transmitter and store in the receiver the corresponding couple KEY-ID and Key. The KEY-ID could also be used in future Message Authentication Code mechanisms built on top of this solution. This property allows preventing *Masquerade* and *Message Injection Attacks*.

Malicious Interference Resistance. Thanks to the information in the watermark, SENECAN can partially mitigate *Adversarial Jamming* attacks. In the most favorable case, if the adversary would destroy the frames $N_W \cdot 80$ dedicated to transmitting the watermark, Bob can still reconstruct the original message using the Algorithm 1. On the contrary, if Eve jams other frames, SENECAN will still suffer from this type of attack. A manufacturer can increase the number of bits N_W for the watermarking process, increasing the attackable frame number for the attacker in exchange for an overhead due to more information to transmit.

3.1.7 Discussion

As described in the previous section, SENECAN allows obtaining all the security properties. Furthermore, there is no need to add nodes that act as a trusted third party differently from many previous works. Also, our approach implements a one-way communication rather than

bidirectional communication (i.e., challenge-response mechanism) differently from other works. This is a valuable property, especially in a broadcast environment as CAN, where the sender's identity is not declared in the message. Our work proposes a schema to securely exchange or refresh the long-term shared key. Due to the high computation requirements, we propose to perform this operation in a non-critical dedicated session (e.g., after the car turns on, in an OTA update, or during car maintenance). However, there can be other reasons to modify the long-term keys. For example, a manufacturer may require refreshing all the nodes' keys to increase their length or change the cryptographic primitives. The manufacturer can also refresh the keys in case of compromise. After the master secret is shared or refreshed, a car vendor can implement other security schemes on top of SENECAN for the successive phases of the communication. Therefore the specific key lifespan can depend on the security policies of the particular application.

Comparison with other schemes. As discussed in Section 3.1.2, most of the works assume that the long-term secrets are shared in the ECU during the production phase and do not consider the refresh of such secret in a successive phase. The only possibility to modify such a secret is physically replacing the ECU. To the best of our knowledge, few works proposed specific approaches to exchange and refresh long-term secrets in the CAN network. In Table 3.3, we report the comparison between the other works in literature that propose CAN-specific key distribution mechanism and SENECAN. The comparison reports the security features discussed in Section 3.1.6. It is important to note that not all the works discussed the mentioned properties. Therefore the analysis is based on our interpretation of the different works. We include in the comparison the type of communication (i.e., one-way transmission or not) and if the approaches require an additional node acting as a supervisor or a trusted third party for the distribution process. This last property is essential since it does not modify the network architecture and insert additional nodes. We also compare if the approach proposed was implemented and tested in a real environment or only discussed theoretically. In fact, in [324, 363], the authors proposed a collision-based approach similar to ours. However, their schema does not include integrity and authentication features but was not implemented and validated in a real scenario. Therefore, the requirement relative to the additional transceiver was not considered and discussed. We must note that even if SENECAN implements all the security properties mentioned and overcome their constraints, all the other approaches require fewer frames to transmit during the distribution, and consequently, they are much faster.

Limitations. We decide to transmit data through the serial to manage the bitstream with Matlab in the testing phase for implementation simplicity. However, this introduces a bottle-

neck in the exchange data between Matlab and Arduino, making each frame delivered every 15 ms equivalent to an estimated 7200 bps. This time is also due to the dominant bit-holding limitation. This overhead is even more impacting if considering the total number of key distribution sessions. By considering the KEY-ID field of 8 bits, in the limit case, we support up to $N_{ECU} = 256$ ECUs. Since every pair of ECUs must share a common key, there will be $(N_{ECU} \cdot (N_{ECU} - 1))/2 = 32640$ key distribution sessions. Considering the setting used in our experiments and reported in Table 3.5 if the bitstream is 256 bits, this would require $32640 * 307.2\,s \simeq 2785\,h$, which is clearly unfeasible. In particular, Table 3.5 reports the transmission time with our scenario and the estimated time with the typical transmission speed used in CAN environments. We believe that with a dedicated hardware implementation, i.e., the computation is entirely implemented in the ECU, without the necessity of the serial communication and the possibility to hold the dominant bit for the desired time, the transmission time can be reduced as shown in Table 3.5. In particular, with dedicated hardware, we can avoid the 115.2 kbps bottleneck due to the serial communication and the need to wait for 15 ms for every frame due to the Arduino driver. Furthermore, since the double-precision representation used by Matlab overestimates the representation of the signal to reduce the transmission load, every sample could be converted into a single-precision representation. This improvement allows transmitting two samples in a single frame, thus reducing by half the overall number of frames to transmit. With these simple optimization, we can cut the communication time as reported in Table 3.5. Considering a common CAN transmission speed of 500 kbps, the limit case transmission time would be $32640 * 2.21\,s \simeq 20\,h$. However, we believe that this number is still largely overestimated since not all the ECUs need to communicate with each other. Finally, our solution requires every ECU that implements SENECAN to embed an additional transceiver. This is because an ECU requires an interface to perform the jamming and another to read the bus simultaneously.

3.1.8 Takeway on IEV Network Security

In this study, we present and prove the effectiveness of SENECAN, an authenticated key distribution framework for devices communicating through the CAN bus network. The presented mechanism exploits the default protocol's bit collision properties to selectively destroy the frames by jamming over a wired connection. The experimental setup we use represents a proof of concept for future work in this direction and can be employed by all the devices operating in the CAN network to exchange a long-term secret among the different nodes. More-

over, in exchange for a long distribution phase, which can be executed in a defined phase of the system's life-cycle (e.g., after the production or during the vehicle revision), our approach allows obtaining more robust security properties compared to similar works and with a minimal network modification. Despite the reduced transmission speed in the SENECAN proof-of-concept implemented, we validate the key distribution based on the WBPLSec algorithm. As previously discussed, we strongly believe that the limitations introduced by Matlab and Arduino can be overcome by introducing ad hoc hardware and optimization. However, we do not exclude that the presence of dedicated hardware could guarantee Bob jam with a single transceiver assigned to the reception of frames in a normal traffic scenario. SENECAN is therefore helpful for systems with a limited number of devices communicating with and can be used as a building block for future CAN-based security systems.

3.2 EVEXchange: A Relay Attack on EV Charging System

The fast growth of EVs in the market led to the diffusion of new architectures to support the energy demeaning required by the vehicles' battery charging. Despite the global pandemic, the sales of EVs in the first quarter of 2021 were more than 2.5 times higher than in the same months of the previous year [365]. Furthermore, the International Energy Agency estimates that if governments agreed to encourage the so-called "Green Transition", EVs could reach 230 million by 2030. Vehicle vendors such as Honda plans to convert to electric its entire car production by 2040 [366]. This transition process is also facilitated by the global economic trend, pushing the adoption of renewable energies. The growing concern about the climate crisis leads to a worldwide movement to create a green and sustainable future. In 2018, The United States Environmental Protection Agency estimated that the 28.2% of Greenhouse Gas Emissions in the US is due to the transportation sector [367].

With such a forecast on the increase of EVs, the energy request from the electric grid will grow as well. This electric demand increase requires smart management of the charging process of each device to avoid overloads and local blackouts. The most common and upcoming paradigm employed to manage the charging of the EVs is the Vehicle-to-Grid (V2G). V2G systems manage the energy distribution from a Smart Grid to the vehicles (i.e., the final user) by providing a communication channel between the two parties [368]. It can be used for various features, from the charging schedule during off-peak hours to more advanced services such as automatic authentication and billing.

V2G is a novel paradigm and, for this reason, it still requires many investigations on security features. When designing such a complex and highly interconnected scenario, security aspects represent extensive and complex requirements, as highlighted by different works [369, 370]. For instance, by exploiting the unique MAC address of a vehicle and unshielded charging cables, it is possible to track a user across different stations [301]. Since V2G can provide a complete internet connection, the EV is exposed to various threats like malware, affecting the vehicle's internal components. The charging column can be attacked as well, for instance by a denial of service attack, blocking the delivery of the charge service to the users. Other exploits which have been proven to be effective in the V2G scenario include the profilation of the battery behavior [371] and the profilation of the vehicle charging process [304] based on the electric traces generated from the charging process.

In this study, we present *EVExchange*, the first relay attack specifically conceived for V2G communication. *EVExchange* allows the attacker to exchange the charging flows, accounting a victim for the energy consumed. We implemented *EVExchange* in both an emulated scenario employing MiniV2G [372] and in a physical testbed composed of different Raspberry Pi, proving its functioning and effectiveness. Finally, we propose an extension of the ISO 15118 protocol (i.e., the standard protocol in V2G communication) that utilizes distance bounding to identify relay attack attempts. We tested the distance bounding protocol in both scenarios under different conditions, proving its ability to identify the relay attack.

3.2.1 Background

This section overviews the basic concepts related to the electric vehicle charging system from a communication perspective. In Section 3.2.1, we introduce the V2G paradigm, while in Section 3.2.1 we analyze the most advanced standard in this field. Then, in Section 3.2.1 we recall the concept of relay attacks.

Vehicle-To-Grid (V2G)

The V2G concept refers to how an Electric Vehicle can communicate with the power grid. It is a feature reserved for Mode 3 and Mode 4 charges, while Mode 1 and Mode 2 have no communication at all since they employ standard and non-dedicated socket outlets [373]. The communication can range from simple signaling to high-level communication adopting most of the ISO/OSI layers. On the energy side, we can identify two different versions. Unidirectional V2G (also referred to as V1G) employs communication to manage the charging of the EV smartly. V1G can offer services to the grid, such as load leveling by shifting the power demand to off-peak hours, and the EV owners, by charging the EV when the energy price is lower. This strategy can impact the grid's performances avoiding overloads and local blackouts without requiring huge investments in the infrastructure [368].

The bidirectional V2G represents an advanced paradigm. In addition to offering smart management of the charging process, it enables the EV to create a bidirectional power flow with the grid. The discharge of a vehicle can be useful for the grid and the EV's owner in different contexts. The grid can benefit from ancillary services such as frequency regulation and balancing, load leveling, and voltage regulation. On the other side, EV owners can get revenues from the power sold to the grid [374].

To support the V2G paradigm, different players proposed different communication protocols. The most widely adopted protocols for the front-end communication between the vehicle and the Electric Vehicle Supply Equipment (EVSE) are ISO 15118, SAE J2847, and CHAdeMO. In the back-end communication between EVSEs and control centers, ISO 61850 and Open Charge Point Protocol (OCPP) are the most used [375]. Among them, c, developed by the Open Charge Alliance, is the most widely used protocol and represents the de-facto standard [376, 377, 378].

In this study, we uniquely focus on front-end communication. Nowadays, CHAdeMO can be considered the defacto standard. It enables communication through a Control Area Network and does not support any authentication method for the vehicle. However, it is available only on expensive DC chargers, not very suited for private owners. SAE J2847 was instead designed for homes. It supports AC and DC charging through Power Line Communication communication, and it is suited to manage different technologies, such as smart air-conditioning or smart refrigerators. However, with the expected increase of EVs in the next years, this integration can make it difficult to develop algorithms to manage all the devices smartly. The most advanced standard is ISO 15118 [379, 380]. It supports both AC and DC charging and shares the same communication means of SAE J2847, partially employing the same infrastructure. Since ISO 15118 can support many services, ranging from authentication to vehicle firmware updates [381], it aims to be implemented globally and become the standard for the future of electric mobility.

ISO 15118

Firstly released in 2013, ISO 15118 is a modern standard for regulating communications between the Electric Vehicle Communication Controller (EVCC) and the Supply Equipment Communication Controller (SECC). EVCC and SECC are, respectively, the endpoints that manage the transmission on the EV and EVSE [379, 380]. It defines a communication channel via Power Line Communication on the Control Pilot (CP) of the IEC 62196 connectors [382].

At the beginning of the connection, the Signal Level Attenuation Characterization protocol is employed to pair EVCC and SECC through a series of pulses. Then, the EVCC broadcasts a default number of UDP packets following the SECC Discovery Request (SDP) protocol to retrieve the IPv6 local-link address of the connected SECC. After that, the High-Level Communication Protocol starts using a TCP communication, generally ciphered using TLS. More information on the packets exchanged can be found in [372].

Unlike the oldest standards (e.g., CHAdeMO), which employ the communication channel only to exchange technical information about the battery and the recharge process, ISO 15118 leverages high-level communication to provide many services to the grid and the user. The authentication process is based on TLS protocol. The TLS certificate employed for the authentication can be obtained or updated during the connection phase. Payments are managed by the standard which supports External Identification Means such as credit cards, Radio-Frequency IDentification (RFID) cards, or QR codes. Furthermore, ISO 15118 provides a highly comfortable service called Plug-and-Charge (PnC). This mechanism allows the user to automatically account for the energy requested without using a card or other payment means at the recharge. In this way, the user only needs to insert the plug in his vehicle socket to start charging. The PnC authentication mechanism employs the TLS certificate installed in the vehicle and used by the charging system to identify the car [383]. The owner can obtain its personal certificate by registering with a charging service provider, as defined in the ISO 15118 standard [379]. However, as we will see in this study, PnC can expose the user to some security threats.

Relay Attacks

A relay attack is a technique through which an attacker can intercept communication between two entities and replay it in another place in space and time through a proxy [384]. It differs from a MiTM attack since there is no hypothesis that the attacker can understand or modify the information relayed (e.g., communication can be encrypted).

Relay attacks are powerful in many applications, generally in transmitting blocks of independent information or encrypted data. For instance, proximity cards (e.g., credit cards) are a profitable target for relay attacks. In this scenario, the card and receiver perform mutual authentication, and all subsequent traffic is encrypted. Using cryptanalysis to recover the keys might be unfeasible or may require tampering with the hardware with costly instrumentation. An attacker can exploit a relay attack to transfer the entire data flow (including the authentication) from the card to a remote reader. A practical attack consists of relaying the data flow from a victim's credit card to a reader near the attacker to bill the victim for the payment.

3.2.2 Related Work on EV Security

Although electrical vehicle charging systems are a novel topic, various research papers have examined various aspects of their security. Mustafa *et al.* [370] proposed a security analysis of

the charging system, highlighting different threats for charging at home, at work, or in public places. A similar investigation was conducted by Antoun *et al.* [369] showing possible countermeasures for ISO 15118 and OCPP. Other works addressed specifically the ISO 15118 standard [385, 386] proposing threats analysis and security mitigations. Different researchers also proposed to apply IDS approaches to identify attacks or anomalies in the EV to EVSE communication [387, 388]. However, none of these works analyzed the threats deriving from relay attacks in the charging process or tested the feasibility of the presented attacks in a real or emulated environment.

Few researchers conducted in-depth studies on aspects of the security of the ISO 15118 standard. Martinovic and Baker showed that it is possible to eavesdrop on the communication between a vehicle and a charging column by exploiting the electromagnetic emissions of the Power Line Communication on an unshielded cable [301]. Hofer *et al.* [389] focused on privacy aspects by presenting POPCORN, a protocol that enhances privacy on the ISO 15118 standard. To participate in V2G communication and especially to use PnC, EV should maintain keys and certificates stored inside the vehicle itself. To store these data safely, Fuchs *et al.* [390] designed HIP, a backward-compatible protocol extension for ISO 15118, which enables the generation and storing of keys in a Trusted Platform Module within the vehicle. Despite an increasing interest in these security aspects of the standard, to the best of the author's knowledge, there are no available solutions to protect against *EVExchange* or similar relay attacks.

There are many scenarios in which relay attacks are used. Its application on Near Field Communication (NFC), for instance, is analyzed in different works in literature [391, 392]. Recently, researchers have successfully proved the effectiveness of a relay attack on the SARS-CoV-2 contact tracking application, proposing a hashing-based countermeasure to secure the environment without losing privacy [393]. Also, the vehicular environment was interested in this kind of attack: examples in the literature show possible relay attacks conducted on the passive keyless entry [394]. In [395], the authors propose a solution to enforce the relay resilience of cryptographic protocols in such applications based on a crypto-chain framework. While numerous studies focus on the communication between vehicles and keys, to the best of our knowledge, this is the first study that highlights the threat of relay attacks on a V2G communication.

3.2.3 System and Adversary Models

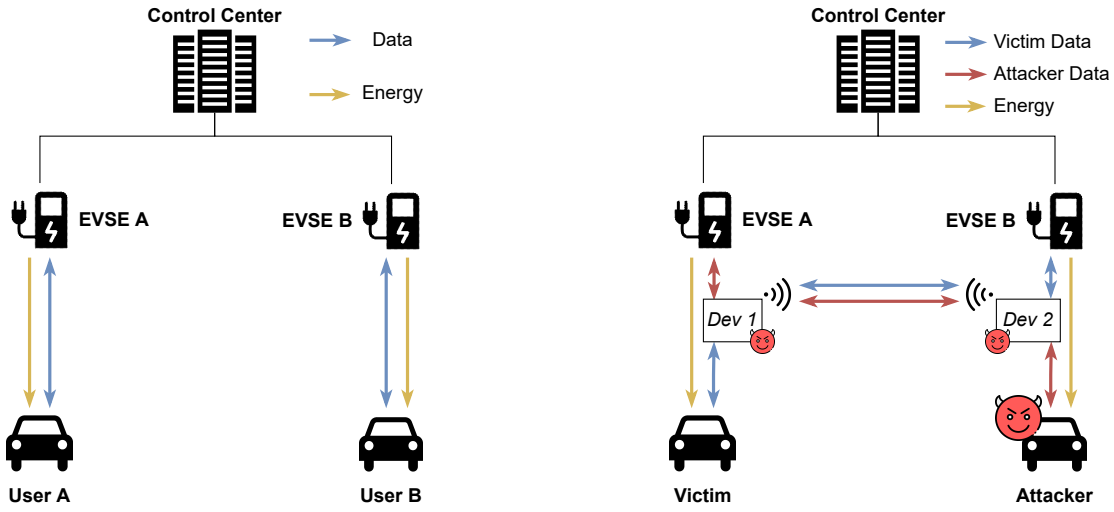
To be successfully implemented, *EVExchange* must be performed in a scenario that respects different assumptions from the system and attacker points of view. In this section, we outline the system model and detail the assumption an attacker must respect to implement *EVExchange*.

System Model. Figure 3.13a represents the scenario in which the *EVExchange* attack can be performed. As reported in the figure, two EVs are connected to two EVSEs, which are in turn, managed by the same back-end infrastructures. Since the victim will set the charging parameters used for the attacker's vehicle charge, the attacker must carefully choose two charging columns entirely supported by his vehicle. The attack can be extended if more than two EVSEs are available. However, this work focuses on the basic scenario with two EVs and two EVSEs. The front-end communication (i.e., between the vehicle and the charging column) employs the most common ISO 15118 standard using the PnC authentication method. Alternatively, this attack is also valid if other means for automatic billing based on a particular ID of the EV are used, such as Autocharge [396], which employs the MAC address of the EV and is commonly used in North Europe.

EV and EVSE are connected via wired cables, the most common setting for power and data travel in different cables. Examples of widely employed socket outlets are Type 1 or Type 2 for AC and Combo 1 or Combo 2 for DC [397]. There are no substantial differences for this study as soon as the communication is established and billing data are transmitted through the cable in the CP pin. It can also extend *EVExchange* when wireless communication is employed in the charging process between EV and EVSE. However, we do not consider wireless charging in this work since it is rarely used in the real world.

Adversary Model. As a preliminary phase, the attacker must tamper with the charging station to install two malicious devices (i.e., *Dev1* and *Dev2*) as depicted in Figure 3.13b. The two devices can be two simple microcomputers (e.g., Raspberry Pi) with two interfaces to demodulate the Power Line Communication in the CP pin and WiFi connection capabilities. A highly skilled attacker could design an ad-hoc device to minimize the device's size to remain undetected. Ideally, each device can be placed in the socket as an adapter, essentially invisible to an average user. Other solutions could be to cut the charging cable to extract the CP cable, cut it and connect it to the device's two Power Line Communication interfaces. The best solution depends on the charging column's type.

Furthermore, the two devices must be connected to each other. While a wired connection is the most reliable and fast solution, it can be visible and could create some suspicion in the user.



(a) Without attack.

(b) With *EVExchange* attack.

Figure 3.13: Scenarios with two EVs charging from two EVSEs connected to the same Control Center (a) and with the malicious devices (b). We represent the Unidirectional V2G scenario for simplicity.

A wireless connection is the most suited and straightforward approach to avoid this issue. In this work, we employed a standard WiFi connection (i.e., IEEE 802.11ac and IEEE 802.11g) with an intermediate Access Point and in an ad-hoc configuration. If the distance between the two devices is significant, high-range wireless connections (e.g., 4G/LTE) can also be employed.

Once installed and activated, the two devices must block the communication channel between each EV and its legitimate EVSE. Then, they must function as a relay by forwarding the communication coming from an EV to the other device (called *Dev1 to Dev2 relay*, or vice versa), which will recreate the data flow on the EVSE side. It is worth noting that the two devices do not need to read the content of the forwarding traffic. This is important because the security standard imposes the usage of TLS to encrypt the communication channel in public places, especially when using PnC [380]. However, as reported in [301], this security measure is often not implemented in practice, exposing the users to many security issues [372]. However, even if the traffic is encrypted, the relay process is still feasible, and *EVExchange* can be performed. In this work, we will assume that all the communications between EV and EVSE are always encrypted using TLS. The adversary does not have any valid certificate in addition to the one in the EV. Therefore, it is computationally infeasible for an attacker to decrypt and modify packets on the fly. The attacker is only able to stop and forward the communication flow.

The key concept to enable *EVExchange* attack is that, while the communication flows are

forwarded as described above, the energy provided from the two EVSEs is instead directed to the legitimate vehicle (Figure 3.13b). In this way, the attacker can control the energy supplied by the victim's EVSE and vice versa.

3.2.4 EVExchange Attack

After setting the two devices, the attacker can proceed with the *EVExchange* attack. We now describe the attack stages through which an attacker can make the victim pay for the energy consumed. We will use Figure 3.13b as a reference.

The attacker waits for a victim to arrive at the charging station. When the victim plugs the vehicle into the EVSE A, the attacker will follow by plugging his or her EV into EVSE B. At this point, both users must set the charging options they need (e.g., time of departure, and energy requirements). Since the two malicious devices are activated, each request made by a user will trigger an action in the EVSE of the other user.

At this point, to be stealthy, the attacker must replicate the victim's request. However, since the attacker has no clues on the victim's behavior, he can suppose with discrete confidence that the victim will require charging the vehicle since it is the most common operation at charging stations. While it is reasonable to assume that the user will look at the EVSE's display to verify the start of the charging process, the victim probably will not notice a minor difference in the charging parameters, provided that they are displayed in the EVSE. For example, the forecast duration of the charging process is variable based on the state of charge, the charger type, and the charging time. Therefore, it is improbable that an average user can precisely predict this parameter and spot the attack through it. After requesting the service, since the charging process can take longer, the victim will usually get away from the vehicle to spend time doing other things while the EV is charging. At this moment, the attacker, who controls the victim's EVSE, can require a stop of charging from the attacker's vehicle. The attacker will now trigger a stop in energy provision in the victim's EVSE (i.e., EVSE A). At the same time, the EVSE connected to the attacker's vehicles (i.e., EVSE B) will continue to follow the victim's request.

Then, when the attacker is satisfied with the charge of the vehicle, he or she can wait for the victims to come back and request a stop of charge for the attacker's EV. Alternatively, the attacker could stop the charging process before the end in his or her charging column to unlock the vehicle and go away, for instance, by using the Emergency Stop button.

Since PnC is employed by the two users in this scenario, the payment of the energy provided to the attacker's EVSE will be billed to the victim. In the same way, the energy supplied to the

victim's EV will be billed for by the attacker but, since the attacker has previously stopped the charge of the victim (at the moment the victim has moved away), he will pay virtually nothing. In contrast, the victim will be billed for a complete charge.

In the following, we summarize the steps of *EVExchange* . These steps are also illustrated in Figure 3.14.

0. The attacker places the two devices as depicted in Figure 3.13b;
1. The victim connects the vehicle to EVSE A; the attacker connects the vehicle to EVSE B;
2. The two vehicles start a communication with a charging request which is forwarded by the malicious devices;
3. The victim, unaware of the attack, goes away from the vehicle;
4. The attacker, while recharging by the victim's charging schedule, stops the victim's charge.
5. When the victim is back, he or she stops the charging process of the attacker.

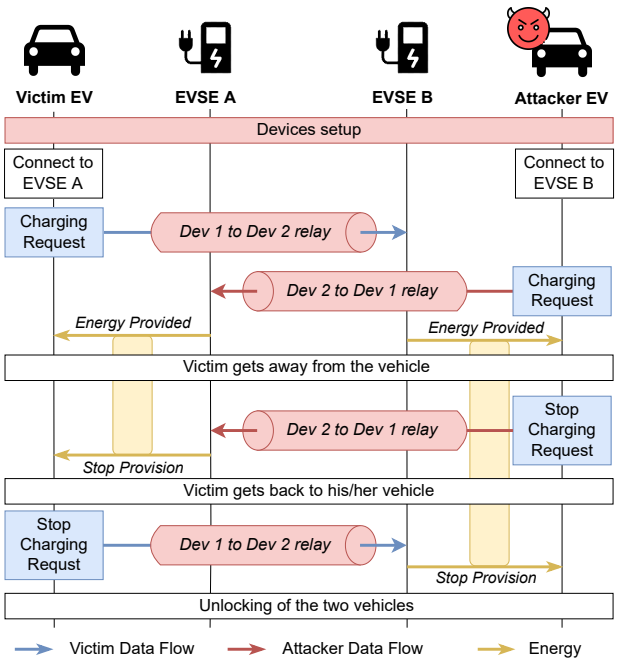


Figure 3.14: The different phases of the *EVExchange* attack. We represent the Unidirectional V2G scenario for simplicity.

Variations of the Attack

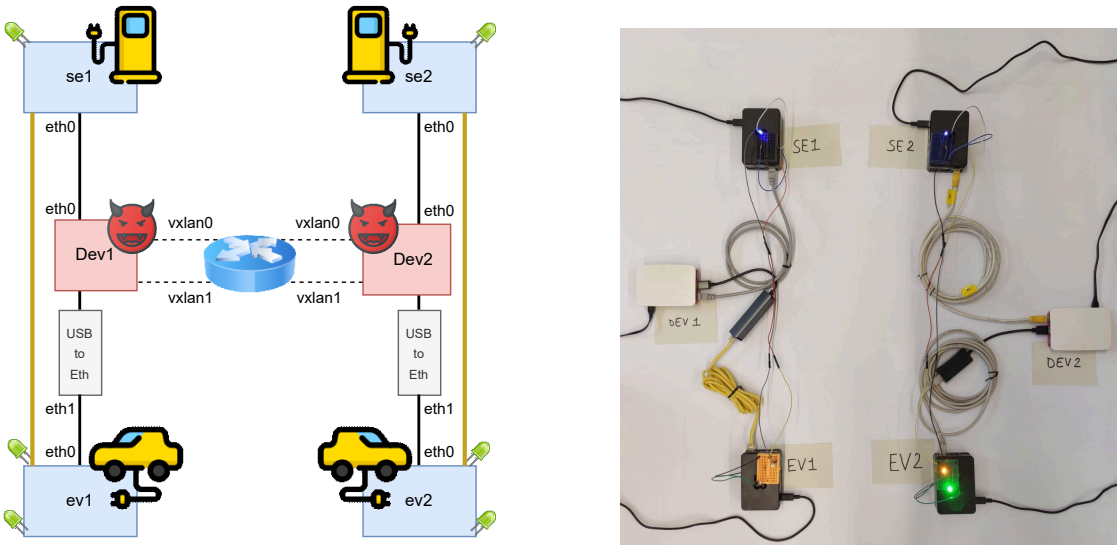
EVExchange attacks can be tailored to achieve different goals. We report here two examples, but many others could be possible.

Discharge Victim's Battery. We assume a system supporting the bidirectional charge (i.e., the vehicle can sell energy to the grid during peak hours and provide ancillary services to the grid [374]). In this case, since the attacker controls the victim's communication with the EVSE, he can decide to sell the energy to the grid the power in the battery. Furthermore, by doing so, the revenue will be billed for in the attacker's account.

Damage Victim's Battery. One of the most delicate components of the vehicle is undoubtedly the battery. It is subjected to fast degradation through usage, which is responsible for reducing the maximum capacity over time [398]. In [304] the authors demonstrate the possibility to profile a vehicle based on the battery charging profile. Some situations can speed up the degradation process, such as extreme operation temperatures, overcharging, and completely draining the battery [399]. Since the attacker controls the victim's charging parameters, he or she can overcharge the battery by requiring energy even if the battery is full. If the bidirectional charge is available, full discharge can be performed as well. Furthermore, an advanced attacker could modify the EVCC or, more simply, modify packets with battery status on the fly to send abnormal charging parameters to the victim's charging column requiring an amount of energy that may damage the battery.

Attack Validation

The EV charging infrastructure is complex to reproduce and manage since it involves different technical aspects, from energy to communication, and includes expensive components. The most common workaround to these limitations is the usage of simulators or emulators. We started our study by testing the attack on implementation of the scenario in MiniV2G [372], an open-source emulator able to simulate networks of EVs and EVSEs. MiniV2G is built on top of Mininet-WiFi [400], a popular software to create realistic virtual networks, running real kernel, switch, and application code. Furthermore, MiniV2G includes RiseV2G [401], an open-source simulator to implement the ISO 15118 communication. Currently, MiniV2G can only emulate the network communication between EVs and EVSEs without simulating the actual battery charging process. However, this limitation does not affect the implementation of *EVExchange* since it is entirely implemented at a network level. For space limitation, we will not discuss the MiniV2G implementation in this work, but we will focus on the development



(a) High-level architecture of the testbed.

(b) A picture of the testbed.

Figure 3.15: The testbed employed to test *EVExchange* attack and countermeasure.

of the physical testbed. However, the MiniV2G implementation and all the code related to this work can be found on Github[1].

We preliminary verified the feasibility of *EVExchange* on MiniV2G and then we implemented a more realistic scenario by using six Raspberry Pis to emulate vehicles, charging columns, and malicious devices. We used the Ethernet interfaces to simulate the Power Line Communication communication while we employed GPIO pins to emulate the energy exchange. We install LEDs to monitor the different stages (i.e., battery charging, energy delivered, authentication completed). As in MiniV2G, we employ RiseV2G in the physical testbed to perform the ISO 15118 communication, with a Python wrapper to turn on the LEDs. Figure 3.15a represents a high-level schema of the testbed, while Figure 3.15b illustrates a picture of the testbed developed.

To connect the malicious devices and allow the packets forwarding, we employ Linux `bridge` [402] command to create a channel between the two physical interfaces in each device. These settings do not alter the normal communication flow between EV and EVSE.

When the scripts to activate *EVExchange* are executed, bridges are deactivated, and the attack is set up by employing Virtual eXtensible Local Area Network (VXLAN) [403]. Generally, this tool addresses the need for overlay networks within virtualized data centers accommodating multiple tenants. In our case, we employ VXLANs to create two independent data flows over

[1] "EVExchange" on Github, github.com/donadelden/evexchange

the wireless network, which can transport packets from one interface of *Dev*1 to the opposite interface of *Dev*2. We employ this strategy to configure *EVExchange* by relaying data from each EV to the opposite EVSE.

3.2.5 Countermeasure

To prevent *EVExchange* and other potentially related attacks, in this section, we present an extension of the ISO 15118 protocol, which contains a countermeasure based on a distance bounding algorithm. In particular, in Section 3.2.5 we design the distance bounding protocol, while in Section 3.2.5 we discuss the security and the limitation of the proposed algorithm. Then, in Section 3.2.5, we describe an implementation of the protocol, providing some numerical results.

Distance Bounding Protocol

To create a countermeasure against *EVExchange*, we can exploit the temporal delay created by the relay process of the communication flows through a wireless channel. The strategy of measuring the distance between two devices by considering the Round Trip Time (*RTT*) is known as *distance bounding* [404]. As demonstrated in its applications in different contexts in the literature, this approach is the most simple and effective solution to relay attacks. Distance bounding is applied for instance in contactless smart cards [405], NFC devices [406, 407], and Passive Keyless Entry [408]. This protocol is well suited to work at the application layer in preventing relay attacks since these threats inevitably introduce a measurable delay in communication.

In general, the distance bounding enables one device (the *verifier*) to securely establish an upper bound on its distance to another device (the *prover*) [409]. In our case, the verifier is the victim's EV, which wants to check the authenticity of the charging column to which its connected. We consider the EVSE (from now on called supply equipment *SE* to avoid confusion) as the prover. Therefore, the algorithm's goal is to assess the EV is connected to the correct *SE* by verifying that the distance between them is no more than an expected value.

The phases of the proposed distance bounding protocol are similar to those proposed by Thorpe *et al.* [407], where the authors designed a protocol at the application layer of the NFC protocol. Our algorithm starts after the establishment of the IPv6 connection when the *SE* starts the listening mode. The core of the proposed solution resides in the fast packet exchange. In this phase, one entity will *immediately* respond to each packet sent by the other. It is possible

to compute the RTT precisely and estimate the distance between the two entities from each exchange. In the following, we explain the different phases of the algorithm in detail.

1. EV generates a random string $\alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_k\}$ with a fixed length k. Meanwhile, *SE* generates a random string $\beta = \{\beta_1, \beta_2, \ldots, \beta_k\}$ of the same length k. These two steps can be done beforehand.

2. The fast packet exchange starts for every $i = 1, 2, \ldots, k$ and the RTT_i is measured:

   - EV send a UDP packet to *SE* containing as data the symbol α_i;
   - *SE* receives α_i and *immediately* responds with an UDP packet including β_i.

3. After k exchanges, EV computes the mean μ and the standard deviation σ of the RTTs.

4. EV compares μ and σ with μ_{max} and σ_{max}, which represent the thresholds for μ and σ, respectively. If $\mu > \mu_{max}$ or $\sigma > \sigma_{max}$, an error is thrown, indicating an attack could be going on.

5. If no alert is raised, the secure communication between the two entities using TLS can start as depicted in ISO 15118. Before actually exchanging charging parameters and setting, *SE* sends to EV the string $S_{SE} = \{\tilde{\alpha_1}, \beta_1, \ldots \tilde{\alpha_k}, \beta_k\}$. Where $\tilde{\alpha_i}$ indicates the new string that will be compared with α_i previously stored.

6. EV computes $S_{EV} = \{\alpha_1, \tilde{\beta}_1, \ldots, \alpha_k, \tilde{\beta}_k\}$ and compares S_{EV} with S_{SE}. An alert is raised if the two strings differ since an attacker might have forged some packets. Where $\tilde{\beta}_i$ indicates the new string that will be compared with β_i previously stored.

7. Finally, if no alerts have been raised, the actual charging process can start following the ISO 15118 protocol.

Security Considerations

An attacker can employ a series of malicious devices placed in the middle between the EV and the EVSE. For visualization simplicity, in Figure 3.5, we represent this set of devices as one single entity called *relay* as a black-box. Considering the adversary devices as a black-box is a reasonable simplification since the legitimate user is unaware of them. We remark that the *relay* device can selectively or completely relay the traffic flow from two entities as for our hypothesis. Furthermore, the *relay* can eavesdrop on all the not-encrypted communication between the

two entities, but it is not equipped with a valid and signed pair of keys to initialize TLS sessions. We do not assume any restriction of the computational capabilities of the adversary. However, it is reasonable to assume that the attacker cannot decipher or modify communication encrypted with TLS.

The proposed distance bounding protocol performs two verifications on the communication. The first one is represented by the effective distance measurement provided by the RTTs. The attacker may try to tamper with it by reducing the latency generated by the relay. However, each strategy must be consistent and avoid failure in the second check during the verification of the transmitted data.

To lower the RTTs, an attacker can reduce the relay's complexity by employing, for instance, a faster transmission mode. We exclude the possibility of applying a wired connection since it will be easily spottable by an average user or the service provider. Furthermore, it is common for normal and semi-fast charging stations to be equipped with a detachable cable that must be carried by the driver [373], making it even more identifiable a wired relay. An alternative is to employ faster wireless communication modes with respect to the IEEE 802.11 standards, such as 5G, to reduce the protocol overhead and any protocol mode translation. However, this would, on the other hand, increase the system's cost and complexity. For short distances, Bluetooth can be considered, but it will lead to equal or lower performances as WiFi [410]. It is worth noticing that the Power Line Communication employs HomePlug Green PHY, which has almost no delay at the MAC layer when applied between two entities only [411], making it even harder to create a fast enough channel to avoid detection. Furthermore, it is important to recall that the implementation must be small enough not to draw the victim's attention.

The previous strategies represent attack optimizations to faster the packet exchange. Another strategy to reduce the RTT could be to tamper with the initial packet flows. Since the initial rapid packet exchange is performed without encryption, the attacker could potentially alter the transmission of the packets. For instance, an attacker can decide to send random β_i immediately after seeing an α_i to reduce the RTT. This process might bypass the first alert control assuring a lower μ and σ, but it will be detected during the second control when comparing S_{EV} and S_{SE}. By defining α_i and β_i values from an alphabet of N symbols, the probability for the attacker to correctly guess the entire string β is $\frac{1}{N^k}$. Assuming to employ only the 128 ASCII chars and a sequence of $k = 10$ exchanges, we obtain a probability of success for the attacker of $\frac{1}{128^{10}} \approx 10^{-22}$ which is negligible. We can further reduce this probability by implementing additional exchanges k and a larger alphabet N.

Note that the proposed protocol does not try to prevent *relay* from knowing both α and β.

Instead, it imposes bounds on the *maximum time* by which the information must be received. In other words, when *relay* reads the packet containing β_i, it introduces a delay that makes it too late for the forwarding of the packet to EV and the achievement of a low RTT. Furthermore, the transmission of S_{SE} secured by the TLS ensures that *relay* cannot be able to modify it. The only way it is possible to change S_{SE} by an attacker in possession of valid TLS certificates is to pretend to be EV and *SE* when sending messages to *SE* and EV, respectively. However, we can reasonably assume that the Public Key Infrastructure is solid, and the attacker cannot craft private keys and certificates. Nevertheless, it is essential that both legitimate entities check the validity of their counterpart's certificates before starting the charging process.

Evaluation

To implement the distance bounding algorithms, we wrote two Python scripts to be executed in the EV and the Supply Equipment (SE), respectively. The protocol starts with a pair of hello messages that enables the EV to get the IPv6 of the SE. Then, the EV starts the algorithm by sending a UDP packet to the SE that acts as a server and immediately responds. This process is iterated 100 times to account for channel variability. To evaluate, we compute the mean and the standard deviation of every set of measures. We perform 1000 executions of the described protocol for each scenario to validate the countermeasure.

To verify the feasibility and effectiveness of our countermeasure, we preliminary test it on the MiniV2G emulator under different propagation models and on the physical testbed with different distances between the devices. We report in the following the results related to the physical testbed, and space limitations.We create different configurations on the testbed in order to represent different possible scenarios:

1. A completely legitimate solution, without malicious devices in place (Wired);

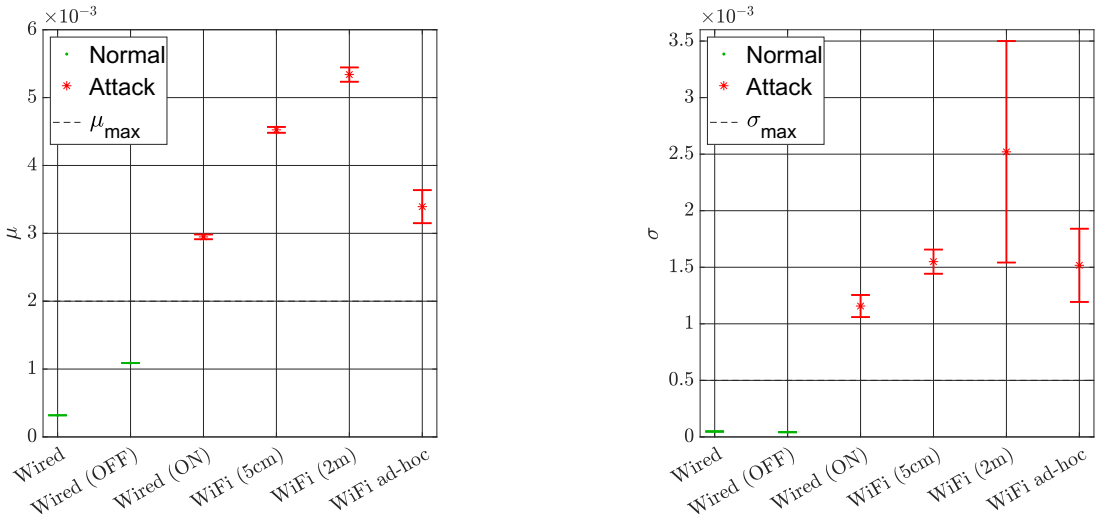
2. A legitimate scenario, with malicious devices, inserted but turned off (Wired OFF);

3. An attack scenario, where the two malicious devices are connected through a cabled Ethernet connection (Wired ON);

4. An attack scenario, where the two malicious devices are connected through a WiFi connection with a router in the middle, placed at 5cm (WiFi 5cm) or 2m (WiFi 2m) from the victim.

5. An attack scenario, where the two malicious devices are connected through ad-hoc WiFi connection (i.e., without any router in the middle). In this case, we avoid the extra hop between the two malicious devices given by the router (WiFi ad-hoc).

We represent the mean RTT in Figure 3.16a and the standard deviation of the RTT in Figure 3.16b. The error bar represents the 99% percentile. There is a clear separation between the wired data with respect to all the attack cases. This makes it simple to search for good threshold values for μ_{max} and σ_{max}, which are represented as a horizontal dashed line. Based on the data we have obtained during our tests, we can safely set $\mu_{max} = 2 \times 10^{-3}$ and $\sigma_{max} = 0.5 \times 10^{-3}$, without almost any risk of having false positives or false negatives.



(a) Mean RTT (μ)

(b) Standard deviation of RTT (σ)

Figure 3.16: 99% confidence interval for the mean (a) and standard deviation (b) in each scenario.

Note that the time needed for the distance bounding algorithms is generally less than $0.06s$ using 100 fast exchanges, with tops of about $0.3s$ when under attack, which is in practice a rare condition. Furthermore, sufficient security could be ensured even with a few exchanges, reducing the time requirements. Since a charge could last from half an hour to several hours, we can say that extra time added from this countermeasure is negligible and invisible to the end-user. We must underline that the experiments were performed in a controlled environment. A thorough evaluation of distance bounding should include a broader spectrum of devices and a wider range of environmental conditions. However, this is beyond the scope of this work.

3.2.6 Takeaway on Relay Attacks on EV Charging Systems

To support the ongoing diffusion of EVs, the charging process's cybersecurity must be considered to improve users' trust in the system. We demonstrated for the first time that *EVExchange*, a relay attack, is a potent threat against the electric vehicle charging environment against the ISO 15118 protocol. On one side, *EVExchange* can harm the victim, avoiding the charge of its vehicle. On the other side, *EVExchange* can damage the EV by exploiting wrong charging parameters and useless charging cycles. Furthermore, *EVExchange* allows the attacker to obtain a profit such as free energy and money from the victim.

To defend against relay attacks, we developed an effective countermeasure able to identify the relay attack in the early stages before sensitive data are shared. The security mechanism adapts distance bounding algorithms to work in the application layer of the ISO 15118 protocol. The countermeasure can always detect the attack in less than $0.3s$ without affecting normal communication if no attack occurs.

Since ISO 15118 is a novel protocol, we believe that our work can help the secure development of future versions (such as ISO/DIS 15118-20, under development at the moment of writing [412]), integrating countermeasures against relay attacks. In future works, the development of novel technology like Wireless Power Transfer could enable a possible extension of *EVExchange* to wireless communication between EV and EVSE.

3.3 Electric Vehicles Charging Profiling

EVSEs enable the charging process by bringing together multiple technologies. On the one hand, they allow the user to exchange data with the grid, providing means for authorizations to a central entity to negotiate the service and pay the associated fees. On the other hand, they allow for an exchange of information from the EV to the infrastructure, such that the charging process is conducted by providing safety towards EV's components. The former communication process (i.e., user to infrastructure) is secured using cryptographic procedures and secure network protocols. Their use mitigates all the well-known threats to the users' security and privacy from unprotected communications. However, the latter communication process (i.e., EV to EVSE) is not secure, as signals are exchanged without encryption or aggregation techniques. The signals exchanged during the charging process can hence be exploited as a side-channel to extract information peculiar to each EV, allowing hence for their profiling and successive recognition [413]. This represents a threat to users' privacy, as the connection of their EV to an EVSE monitored by a malicious user may lead to tracking their movement as well as information regarding their driving behavior. Since the majority of publicly available EVSEs are deployed without proper physical protection, they can be accessed by anyone and represent hence favorable spots for attackers targeting the charging infrastructure [414]. Therefore, an attacker can easily install devices to collect data regarding the charging process.

In this study, we propose *EVScout2.0*, an extension of EVScout [413], where we initially showed the feasibility of profiling EVs based on the current exchanged during the charging process. In particular, we extend the proposed framework by developing an enhanced feature extractor, which allows for higher classification scores than our previous work. We then exploit a novel Time Series (TS) for feature extraction by combining the current and pilots TSs. We will explain the reasoning behind this choice and provide the details of its computation. Furthermore, we consider a larger real-world dataset, comprising up to 300 TSs of the current and pilot signals exchanged by each of the 137 considered EVs, for a total of more than 7500 charging session. We perform a thorough evaluation of *EVScout2.0*, showing its profiling performance considering different training set sizes, as well as different unbalancing in the training-testing datasets. Compared to the previous work [413], we provide a more comprehensive analysis of the performance of the attack. In particular, we first show the performance of the novel feature extractor and investigate the performance of *EVScout2.0* for a varying number of features. We then compare the performance of the different classifiers that can be exploited by *EVScout2.0*. We also extend the number of classifiers and provide an in-depth description of the choice of

their hyper-parameters. We then investigate the dependencies of *EVScout2.0* on the number of training TSs needed for classifying EVs with sufficient confidence. We show that the attack is already successful considering 7 training examples. We then investigate the battery degradation over time and its impact on *EVScout 2.0* performances. Lastly, we show the superiority of *EVScout 2.0*, comparing its performance with those in [413], both on the old and new datasets.

3.3.1 Related Work Power Side-Channel

Power consumption can be exploited as a side-channel for different purposes [415]. For instance, an attacker may implement a laptop user recognition by exploiting the current drawn by a smart wall socket during users' activity given [416]. The same concept can be exploited for detecting the user's presence in a smart home, where raw data can be acquired and analyzed to detect activity and hence users' presence [417]. Raw power data also provides information regarding the actions a user is performing. For instance, by analyzing raw power data exchanged via a USB cable, an attacker may be able to obtain information regarding the victim's browsing activities [418]. The power exchanged during the charging process via USB can also leak more sensitive information, which an attacker can later exploit. For instance, the power analysis may leak information regarding digits composed on a touchscreen, allowing for the deduction of users' passwords [419]. However, no previous works investigate the feasibility of leveraging PSC to profile an EV during the charging process.

3.3.2 EV System and Threat Model

In this section, we introduce the scenario in which we conceived our experiments. In particular, in Section 3.3.2 we recall the EV charging system, which represents our system model, then in Section 3.3.2 we present the threat model designed for *EVScout2.0*.

EV Charging System

According to the V2G paradigm [420], the charging infrastructure for EVs is a network where a central controller (power distributor) distributes power based on EVSEs demand while accounting for the maximum supported load by the electric grid. We depicted in Figure 3.17 the typical architecture of a V2G system. EVSEs in the network may be deployed at different sites, e.g., private customer premises, public stations, or office buildings. Each EV is both physically and logically connected to the grid via the EVSE, which manages communications between the

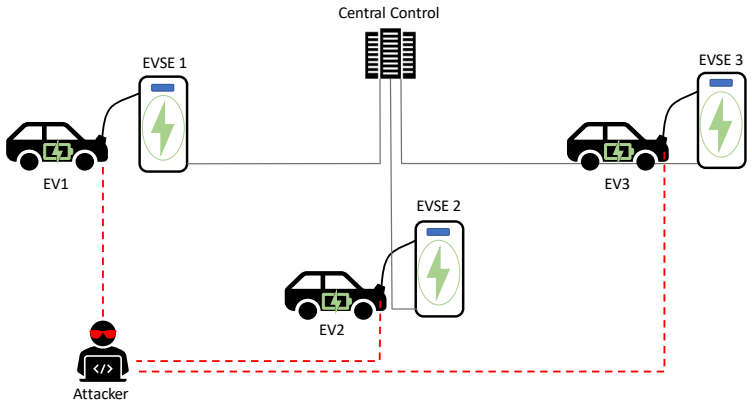


Figure 3.17: System and threat model. Multiple EVSEs are connected to the central control, which provides coordination and power distribution among them. A single EV is connected to each EVSE. The attacker has access to the physical quantities exchanged by multiple EVs during the charging phase.

user (i.e., the owner of the EV) and the power distributor. For public charging infrastructures and office stations, multiple EVSEs are connected to the power distributor through a Central Control that copes with the demand of a large number of connected users [414]. EVSEs are typically equipped with communication interfaces (wireless or wired) to allow communication with the user and the grid. Utilizing modules in the EV or smartphone, the user can communicate with the EVSE and, in turn, with the power supplier.

Current implementations of EVSEs are organized in three levels [421, 422]. Level 1 and 2 use a 5 lead connector based on SAE J1772 standard [423], where 3 leads are connected to the grid via relays in the EVSE. The remaining 2 pins, i.e., pilot and proximity lines, are used for signaling. The proximity line indicates whether a good physical connection has been established between the EV and the EVSE, blocking the initiation of the charging process in case devices are not properly attached and preventing hence damage to both the user and the involved devices. The pilot line provides a basic communication means between the EV and the EVSE. The combination of signals collected from all the pins is used to provide the main processing unit of the EVSE information regarding the charging process, allowing for metering used to assess the charging session state. If a problem arises at one of the two sides of the charging process, the EVSE computer hardware will remove power from the adapter to prevent injuries on both sides. Level 3 EVSEs are instead more complex, comprising bigger pins for power delivery and allowing power line communications via the pilot line.

Typical batteries employed for EVs belong to the class of Li-ion (Lithium-ion) [424, 425].

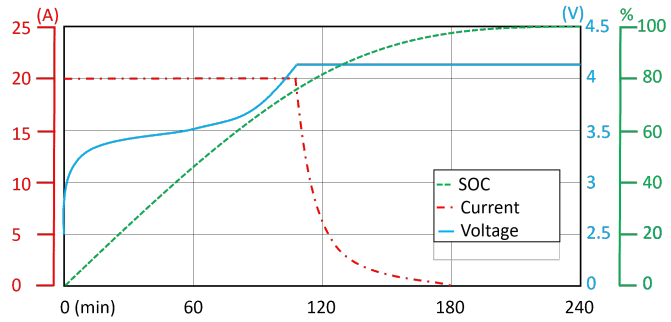


Figure 3.18: Charging profile of a Li-ion battery [427]: we see that, as the SoC increases, the charging mode switches from constant current to constant voltage. We further notice that the two phases are mutually exclusive.

Current and voltage values are exchanged during the charging process depending on the State of Charge (SoC) of the EV battery and can be divided into two classes: constant current/constant voltage and constant power/constant voltage [426]. In this work, we consider the first class, where the charging process can be further divided into two phases:

- *Constant current phase*, where the current level is constant while the voltage value increases;
- *Constant voltage phase*, where voltage is constant whereas current decreases.

The charging process starts with the constant current phase, and this operation mode is kept until the battery's SoC is above a certain value. After reaching the SoC switching point, the operation mode switches to constant voltage up to the full charge. Typical SoC switching values lie between 60% and 80% of the full charge. An example of a charging profile for an EV's Li-ion battery is shown in Figure 3.18. We here remark that constant current and constant voltage phases are mutually exclusive in time, as this will be exploited by *EVScout2.0*.

Threat Model

We consider two possible threat actors: an external attacker and the charging service provider. The first scenario considers a general malicious user who wants to target a specific vehicle. In this case, the attacker can compromise a specific EVSE that the user generally uses (e.g., in workplaces, public parking lots in industrial areas), reducing the number of EVSEs to compromise to succeed in the attack. The attacker may be equipped with a small measuring device that can be connected on one side to the EVSE plug and on the other side to the EV plug. This device measures the exchanged current at the connection point between the EV and EVSE, and we

assume that it is hard to notice by users. We assume that the device provides information to the attacker via either i) a wireless communication module or ii) storing the values of interest to be lately collected by the attacker. The device used to collect the power traces can be built in different ways. For instance, it can be composed of an Arduino board[2] and a standard power consumption monitoring module to sample both current absorbed and pilot. Furthermore, the device can be battery-powered so as not to impact the normal charging behavior. Even if the integration of an external device may impact the absorbed current, we can reasonably assume that this does not affect the profiling performance significantly, as demonstrated in similar works [413, 419, 418, 416].

The second case involves the charging column provider as a threat actor (i.e., the parking spot owner with the charging columns). In this case, the attacker has a higher possibility of manipulating the charging columns; therefore, the attack scales better on more vehicles. Indeed, the V2G infrastructure is exceptionally complex nowadays, and many actors participate in the energy distribution process (e.g., the energy provider, the energy plan contractor, the charging column provider, and the parking lot owner) without access to the same data. For instance, the provider of the charging column does not have access to the user information (data are encrypted), but she can instead easily access the power traces. Therefore, she can easily track and profile users based on their power requirements without being detected.

By employing one of the strategies mentioned above, the attacker has access to the TS of the signals exchanged between the EVSE and the EV during the charging phase. These values are hence recorded for each pin of the EV charger. In this study, we assume that the attacker trains a different classifier for each target EV. To accomplish this, a sufficient number of TSs of the target EV shall be collected. The attack is hence divided into i) the collection phase, where the attacker collects data regarding a target EV, and ii) the exploitation phase, where the attacker exploits the previously computed features to discriminate between different EVs based on the observed time series. To collect multiple traces of a single vehicle, the attacker may exploit one or more EVSEs in a public place with regular customers (e.g., workplaces, public parking lots in industrial areas). In this way, the probability of a vehicle going there many times, and thus the attacker having access to more charging traces, is higher. To build a set with sufficient features, we assume that the attacker collects the TS of the exchanged current values and the TS of the pilot signals. Notice that, to retrieve this data, the attacker does not need to perform elaborate V2G network intrusion schemes, as signals are exchanged outside the network. Furthermore, notice that the attacker is not modifying in any way the charging process. Hence,

[2] https://www.arduino.cc/

the system cannot automatically detect the attacker's presence via intrusion/anomaly detection techniques.

Our threat model is based on the fact that the majority of publicly available EVSEs are deployed without proper physical security and hence can be accessed by any malicious actor [414]. In this case, since there is no access regulation to the EVSEs, the attacker can freely attach the measuring devices. The attack is further facilitated by the fact that typical users will not modify the charging system, even though they may notice something unfamiliar. Therefore, the attacker may not only be represented by the company running the EVSEs network but can also be anyone interested in obtaining information on users' consumes and locations. On the other hand, we notice that the EVSE devices can be routinely checked by the staff of the running company.

Figure 3.17 shows the assumed system and threat model. In detail, multiple EVSEs communicate with the Central Control, providing coordination information and power distribution. A single EV is connected to each EVSE. As previously mentioned, the attacker gets access to the time series of the physical quantities exchanged by multiple EVs during the charging phase and exploits them for profiling. Note that if the attacker can remotely access the current exchanged in different network nodes, it can also locate users, leading to user tracking. The knowledge of the physical signal features associated with each EV (and hence the owning user) can also be exploited for impersonation attacks. Considering EVSEs, which are automated based on the specific user needs, an attacker could steal assets from a target user by generating a signal with the same physical features such that the EVSE recognizes the attacker as the victim. Scenarios that may harm the target user include billing and misbehaving users' exclusion from the system. Therefore, the motivation behind the attack can be multiple. As an illustrative example, consider advertising: the attacker has both information on a certain user's typical movements, and the amount of energy s/he consumes regularly. This information can be sold to EVSE owners, which will target their advertisement to the profiled user according to its demand. Notice that, although a single classifier is trained for each EV, the attacker collects information regarding multiple EVs, such that more than a single classifier can be implemented with the gathered data. Therefore, the attacker can also sell information about the collective use of the EVSE charging stations by EVs to EVSE companies. Although profiling can be implemented using cameras, this would not allow the collection of energy traces, therefore losing some of the information available with the proposed attack. Such information can be obtained utilizing *EVScout2.0* which may be used as an alternative or a complementary solution to cameras. The possibility of tracking a user gives a further threat. In fact, thanks to *EVScout2.0*, an attacker can detect

the presence of a target user in a certain place and time based on the fact that his/her EV is connected to a particular EVSE. We stress that the complexity of the overall charging infrastructure currently imposes several challenges that will be addressed in future implementations. Therefore, it is not possible to predict which actors will in the future be able to access power traces and use them for malicious purposes. Therefore, we aim to warn users and developers about the threat imposed by the cleartext exchange of power traces.

3.3.3 *EVScout2.0*

In this section we describe the *EVScout2.0* analysis methodology. First, we propose a high-level description of the attack configuration. Then we describe the preprocessing we apply to the dataset. In particular, we present the concept of tails and outline the method we designed to extract them automatically and the we present Delta TS, providing both the motivation behind our choice and the means to compute it. Lastly, we describe the novel and automatic feature extraction technique we employ in *EVScout2.0*.

Attack Description

As previously stated, in the context of EVs charging infrastructures, users' data are authenticated and secured. However, physical signals are generally not supposed to implement security measures and, therefore, can be easily exploited by malicious users. Since the exchanged current during the charging phase is a user's generated data, it comprises features and recurrent behaviors useful for profiling attacks. *EVScout2.0* identifies and extracts those physical features which are representative of every single EV, such that we can assert with sufficient confidence if and when a specific user is connected to the charging grid.

Figure 3.19 shows the block diagram of *EVScout2.0*'s steps. *EVScout2.0* starts with data collection. To profile EVs the attacker must collect multiple charging sessions for each target EV. We will discuss this requirement in Section 3.3.6, assessing the number of training examples the attacker needs to collect to profile an EV with sufficient confidence. Once collected the charging TSs (i.e., the dataset), *EVScout2.0* automatically computes the features that characterize each EV. To this aim, in the following, we propose a strategy to exploit the behavior of batteries during the charging process. In particular, as proposed in EVScout [413], assuming that the attacker has access only to the ampere-based electrical quantities, we exploit the current behavior during the constant voltage phase. Leveraging the nomenclature in [371], we

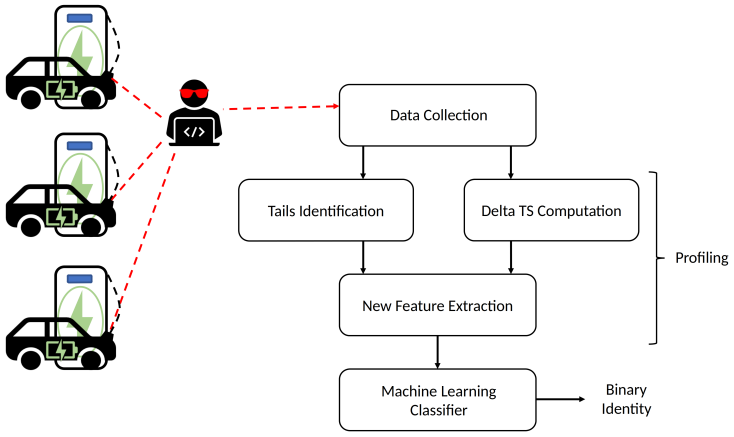


Figure 3.19: Block diagram of *EVScout2.0*'s steps.

name the current TS during the constant voltage phase as *tail*. In Section 3.3.3 we describe how *EVScout2.0* extracts the tails.

Notice that the choice of exploiting tails is due to the assumption that the attacker has only access to the ampere-based TS. If the attacker has access to voltage values, the corresponding features can not be extracted from the tail, as the tail corresponds to the constant voltage phase. Therefore features extracted from voltage values during constant voltage may be underrepresentative of the battery's behavior. If the attacker has access to both current and voltage TS, current features can be extracted from tails, whereas voltage features can be extracted during the constant current phase.

By noticing that each battery follows the current limits imposed by the pilot differently, we generate a further TS to be used to extract more features. Together with the tail, in *EVScout2.0* we exploit the Delta TS, i.e., the TS given by the punctual difference between the current TS and the pilot TS during the constant current phase. Delta TS hence includes all the data from the beginning of the TS up to the beginning of the tail. More precisely, since the first few seconds of the charging are generally noisy, we start our delta computation after the first few samples. We believe that this derived TS uniquely characterizes the behavior of each specific EV as we will explain in Section 3.3.3.

Tail Identification

Charging sessions are not necessarily comprehensive of the constant voltage phase, as a user may need to leave before the full charge is reached. In [371], the authors presented a framework to

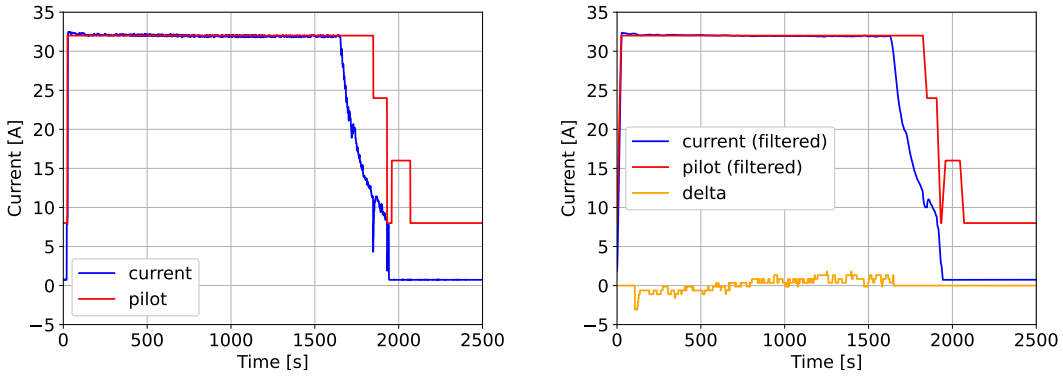
cluster similar charging behavior based on the charging tail. We exploit this portion of the TS to perform more detailed profiling. Since *EVScout2.0* exploits tails during the constant voltage phase, we adopt the algorithm we proposed in [413] to identify whether the considered session includes a constant voltage phase. The presence of a tail implies that the session terminates with full SoC, and eventually zero-current exchanged between EV and EVSE. Tails, however, can not be uniquely identified by the presence of zeros in the current TS, as this may be due to idle phases during the power scheduling process at the grid side. Furthermore, scheduling may cause shot noise in the TS also after full SoC, leading to spikes in the TS. Therefore, we designed a suitable tail reconnaissance algorithm.

To mitigate the effects of scheduling and highlight the trends in the considered TS, we propose applying a suitable filter. In particular, we filter both the current TS and the pilot TS with a length N_{avg} moving average filter. Given time instant t and denoting the electric current value at time t as $c(t)$, the output value $y(t)$ of the moving average filter at time t is given by

$$y(t) = \frac{1}{N_{\text{avg}}} \sum_{m=0}^{N_{\text{avg}}-1} c(t-m). \tag{3.6}$$

The effects of the moving average filter are shown in Figure 3.20. We see that, with respect to the non-filtered current TS in Figure 3.20a, the current TS in Figure 3.20b has a smoother behavior as the filter removes most of the noise and scheduling artifacts. Notice that different filter implementations can be considered, e.g., low pass filter. However, a low pass filter requires a more accurate design and leads to ringing effects, which may be misleading for trend, and hence tail identification.

If the filter has a sufficient length, its effects include spikes removal. This eases the identification of tails in the TS, as we can rely on the presence of steady zero values when the full charge is reached. In detail, if the current TS assumes zero values from t_{start} up to its end, then we can assume that full SoC has been reached. Tails are characterized by a descending trend in the TS, as shown from the current behavior during the constant voltage phase in Figure 3.18 and verified in Figure 3.20. By forward analysis of the current TS, it is difficult to identify the time instant corresponding to the beginning of the tail, as this would imply the overall TS analysis. Therefore, we propose to proceed backward from the point where full SoC is obtained. Proceeding from t_{start} backward, we identify the tail by accounting for the number of samples in the current TS reporting an ascending trend. Notice that, even though scheduling and noise could affect the trend of the TS, its effects are mitigated by the moving average filter (see Fig-



(a) Raw current and pilot profile.

(b) Current and pilot filtered with a moving average filter with $N_{avg} = 25$.

Figure 3.20: Normal current trace and filtered current trace with respect to the pilot during a sample charge. The delta is also plotted multiplied by a factor of 10 to increase understandability.

ure 3.20b). A perfectly backward-ascending trend is given by a negative difference between the values at time t and $t-1$, i.e., $y(t) - y(t-1) < 0$. However, we notice that tails do not always exhibit a perfectly backward-ascending trend. In fact, if the non-filtered TS is affected by heavy noise, its effects are still visible after filtering. Therefore, we relax the concept of perfect backward-ascending trend including samples for which $y(t) - y(t-1) \leq \varepsilon$, with ε being a small positive value. Furthermore, we also allow for short descending trends by accounting for $T_{\max}$ consecutive segments for which $y(t) - y(t-1) > \varepsilon$. If this is the case for $T_{\max}$ consecutive samples, the trend is considered fully descending and hence discarded.

Based on the considerations mentioned above, the steps of the tail extraction algorithm are shown in Algorithm 2. We denote as $\mathcal{I}$ the set of EV indexes. Notice that each current TS is associated with a unique pilot TS. We denote as $\mathcal{C}(i)$ and $\mathcal{P}(i)$ the sets of the current and pilot TS associated with ID $i \in \mathcal{I}$, respectively. We assume that the two sets are ordered such that the current and pilot TS associated with a certain charging session are associated with the same index in the two sets. We define the set $\mathcal{W}(i) = \{\mathcal{C}(i), \mathcal{P}(i)\}$, whose elements (c, p) are the couples of current and pilot time series respectively taken from sets $\mathcal{C}(i)$ and $\mathcal{P}(i)$. For each TS $c \in \mathcal{C}(i)$, we calculate the filtered current and pilot TS respectively denoted as $\tilde{c}$ and $\tilde{p}$ by applying the filtering function (3.6). We then search for the time instant t_{start}, from which $\tilde{c}$ is composed only by zero values. If t_{start} is found, then the current time series might contain a tail, and we proceed to identify the number of tail samples. Given our definition of backward ascending trend, we compute the number of samples for which the filtered time series is such

Algorithm 2: Tail extraction algorithm.

Data: $\mathcal{W}, \mathcal{I}, C_{\max}, T_{\max}, N_{\text{avg}}$
Result: $\mathcal{T}_c, \mathcal{T}_p$
for $i \in \mathcal{I}$ **do**
  for $(c, p) \in \mathcal{W}(i)$ **do**
    compute $\tilde{c}$ and $\tilde{p}$ via (3.6);
    compute t_{start};
    if t_{start} *found* **then**
      $n = 0, s = 0$;
      for $t = t_{\text{start}}, t_{\text{start}} - 1, \ldots, 1$ **do**
        if $\tilde{c}(t) - \tilde{c}(t-1) > \varepsilon$ **then**
          $n = n + 1$;
          ▷ increase counter of backward-ascending samples
        else
          $n = 0$;
          $s = s + 1$;
          ▷ reinitialize counter after descending trend
        end
        if $n = T_{\max}$ **then**
          exit loop;
          ▷ maximum length reached
        end
      end
      $\mathcal{T}_c(i) = \mathcal{T}_c(i) \cup \tilde{c}(t_{\text{start}}, s)$;
      $\mathcal{T}_p(i) = \mathcal{T}_p(i) \cup \tilde{p}(t_{\text{start}}, s)$;
      ▷ add the detected time series to the sets
    else
    end
    go to next c;
    ▷ move to the next time series
  end
end

that $\tilde{c}(t) - \tilde{c}(t-1) \leq \varepsilon$. As short descending trends are also allowed, we account for the number n of consecutive descending samples. However, if this trend is persistent, we should discard these samples. Therefore, we set a threshold value $T_{\max}$ for the number of descending samples, after which we stop the counter. Instead, the counter is reset if an ascending segment is found after a descending samples series. Given the number S of tail's samples, the tail $\tilde{c}(t_{\text{start}}, s)$ is obtained from the filtered current TS, starting from $t_{\text{start}} - s$ up to t_{start}. The tail $\tilde{p}(t_{\text{start}}, s)$ associated to the pilot TS is analogously obtained, starting from $t_{\text{start}} - s$ up to t_{start}. Both current and pilot TS are eventually added respectively to the set $\mathcal{T}_c(i)$ and $\mathcal{T}_p(i)$ of current and pilot TS tails associated with EV ID i.

Delta TS computation

Aside from current tails, we compute another TS to extract features for the classifiers. Since different batteries' charging sessions have different charging parameters, the maximum current which can be absorbed is variable and characterize the specific vehicle [413]. Furthermore,

pushed by advanced charging algorithms [428], during some periods, the EV can be forced into charging at a lower current, e.g., to deal with peak leveling during peak hours. However, as visible in Figure 3.20, generally, the battery does not charge at the exact amount of energy expected from the pilot signal. Furthermore, the absorbed current often exhibits small variations around the maximum current deliverable by the charging column. It is particularly true when considering the behavior of the two TSs is the time preceding the tail. To capture these changes, we compute *Delta TS*, i.e., the TS given by the combination of the current TS and the control pilot TS during the constant current phase (i.e., the period preceding the tail).

To compute Delta TS, we calculate the difference between the current and pilot TSs at each time instant during the constant current phase. While the pilot TS generally is not affected by noise, the current TS instead exhibits some tiny positive and negative spikes, as shown in Figure 3.20a. While the tails generally follow a decreasing trend, the values assumed by the TSs before the tail (i.e., the ones used to compute the Delta TS) are generally more constant. Since the moving median filter provides better performance in removing noise and spikes when the data in the neighborhood of the peak are quite constant [429], we use it instead of the moving average filter used in Section 3.3.3.

Given time instant t, we denote the electric current value at t as $c(t)$ and the correspondent pilot value as $p(t)$. The resulting point at time t in the Delta TS can be expressed as:

$$z(t) = p(t) - median\left(c\left[t - \left\lfloor \frac{N_{\text{avg}}}{2} \right\rfloor, t + \left\lceil \frac{N_{\text{avg}}}{2} \right\rceil\right]\right). \tag{3.7}$$

where $c[a, b]$ represents the array of values of the TS c from $t = a$ to $t = b$, N_{avg} is the filter length, and $median(x)$ is the median value of array x (i.e., the middle value separating the grater and lower halves of x).

Improved Feature Extraction

Segmentation represents a classical approach for feature extraction in TSs [430, 431, 416]. Unfortunately, segmentation is not a viable solution since the TSs we consider here are generally short with no stationary components. Therefore, we do not further process tails before extracting features. In our previous work [413], we computed the mean, mode, median, max value, standard deviation, auto-correlation, length of the tail, and the slope of the linear approximation for each tail. Furthermore, we used as a feature the total kW delivered and the overall session time duration, leading to a total of 18 features considering both pilot and current TSs

independently. Instead, in this work, we adopted a more sophisticated feature extraction process that can extract several more features.

We analyze widely used feature extraction tools that can automatically extract features from a given TS [432, 433, 430]. We use Time Series Feature Extraction based on Scalable Hypothesis tests (`tsfresh`) [434], which is available as a Python package easily integrable with other tools such as scikit-learn [435]. `tsfresh` exploits the power of 63 TSs characterization methods to extract hundreds of features from a TS. Moreover, it contains functions to slightly reduce the number of features to remove the less meaningful ones. We use `tsfresh` to extract features from the current tails and the delta current-pilot TSs. It is worth mentioning that, with respect to the previous work [413], we decided not to employ features extracted from the tails of the control pilot TS since the pilot tail is generally based on the optimization algorithm [428] and not on the behavior of the battery. For example, even if the battery has reached its maximum SoC, the control pilot could remain at a high value if the grid has energy available. In the same way, if many vehicles start requesting energy, the control pilot will reduce its value independently of the SoC of our target EV. Furthermore, we removed the needs for the duration of the charge and the total energy absorbed by the battery (kW) since they can depend on the user behavior and the SoC of the battery at the beginning of the charging process.

Since `tsfresh` can generate around 800 features for each TS both from the time and frequency domains. We removed those that are not relevant to our classification problem. To reduce the number of features, from now on denoted as *NoF*, we select the most significant ones by using `SelectKBest` of scikit-learn [435] with the `chi2` function, which is suitable for classification purposes. Since the chi-squared measures the dependence between stochastic variables, we can highlight and select only the features that offer more information for the classification. We employ this strategy with respect to other more complex methods such as Random Forest feature reduction, since `SelectKBest` can achieve approximately the same results with lower computational complexity. As expected, due to the high variance in the charging tail, the most meaningful features are related to the tail TS. For example, high feature importance score is assigned to the values that are more than r times sigma distant from the mean of x, with different $r \in [3, 5, 6, 7]$, indicating the number of outliers that the moving average filtering has not eliminated. Other significant important features include the number of peaks, the standard deviation, the quantiles, and the c_3 statistics [436] (a coefficient used to measure non-linearity). The features computed from the delta TS are instead less relevant, but the standard deviation has a significant importance score. More details on the features extracted can be seen in the `tsfresh` documentation [434].

3.3.4 Evaluation Framework Description

In the following, we present the experiments we use to test *EVScout2.0*. In particular, in Section 3.3.4 and Section 3.3.4 we describe respectively the ACN Infrastructure and Dataset on which we based our analysis. We then explain how the new version of the dataset differs from the one in the previous work. Then, in Section 3.3.4 we outline the machine learning classifiers we use to profile EVs.

The ACN Infrastructure and Dataset

In order to test *EVScout2.0*, we exploit the Adaptive Charging Network (ACN) proposed in [428]. It consists of level 2 EVSEs connected with a central controller that regulates power exchanges in the grid. Employing an online optimization framework, the ACN allows adapting the power exchanged in the grid, satisfying users' power demand while coping with the grid's capacity limits. The dataset comprises $50kW$ DC charging sessions from different ACNs sites, each reporting user-specific measurements such as the arrival and departure time, the kW/h delivered, current and pilot TSs collected between the EV connection and disconnection time. Notice that, although the user may have planned for a full recharge during the selected period, this may not be reflected in the TS. In fact, due to the variable number of connected EVs, the upper power limit of the grid, and the premature departure of the user, the battery may not be fully charged at disconnection time. Notice also that, in the ACN dataset, not all TS are sampled with the same period. However, we avoid upsampling with filtering because it can introduce statistical features that are not representative of the analyzed battery. Each user in the dataset is identified by a unique ID associated with the owned EV. We specifically focused on the biggest site, Caltech, which contains charging sessions collected from 54 different EVSEs.

New Dataset

Like in the previous work [413], we decided to use the ACN dataset containing time series sampled with an average period of 3.8s. However, since the dataset was recently enlarged, we expanded the number of EVs considered from 22 to 187. To generate the dataset, we selected all the available EVs up to June 18, 2021, from the `caltech` site (one of the three locations available in the dataset). We also excluded by default EVs without charges and charges without any EV assigned (i.e., anonymous charges). To download the dataset, we employ the Python APIs provided by the ACN Dataset [428]. The number of charges associated with each EV

ranges from one to over 300. However, not all the charges are performed up to the full SoC, and thus, not always a tail is available. We, therefore, remove these TSs and the corresponding EVs, because our method is based on the presence of the tails to compute features on both current and delta. Furthermore, we consider all the EVs with more than 8 employable charging processes associated. We made this choice to be able to effectively use cross-validation even for those EVs with a small number of charges. After this cleanup, 137 EVs remains in the dataset, resulting in a dataset more than six times bigger than the one used to test EVScout in [413].

Classification Algorithms Comparison

The first step of *EVScout2.0* is to build a suitable dataset to be exploited for profiling, as explained in Section 3.3.4. It is worth mentioning that the dataset does not provide any information regarding the brand and the model of the EVs. Since the energy behavior highly depends on the chemical reactions of the single battery, we believe that *EVScout2.0* could be able to distinguish EVs of the same model. Furthermore, the batteries employed by the analyzed EVs all belong to the same class, i.e., constant current/constant voltage. However, since both classes discussed in Section 3.3.2 show particular behaviors in time, we believe that our attack could be easily extended to the constant power/constant voltage class. The effectiveness of *EVScout2.0* in these cases will be investigated in future works.

For each session, *EVScout2.0* first identifies whether a tail is present and discards all the other sessions. Then it builds a feature vector for each tail, associating it with the ID of its corresponding EV. We test *EVScout2.0* across all EVs in the dataset by averaging the performance obtained with every single classifier. In particular, we implement a binary classifier (One-vs-Rest strategy) for each EV, and we associate each feature vector of the *target* EV with label 1, otherwise with label 0 all the other traces (i.e., *non-target* vehicles). The overall performance of the obtained classifiers is averaged considering 100 randomly created training and testing sets, except RF and ADA classifiers for which, for timing reasons, we consider 25 iterations. The overall performances of *EVScout2.0* are obtained by averaging the results obtained for each ID's classifier in order to mitigate too high or low results caused by a particular vehicle.

Let us denote as Q the ratio between the number of feature vectors associated with the target EV and the number of feature vectors associated with other EVs. Hence, Q measures the amount of unbalancing in the considered dataset. To further assess the performance of *EVScout2.0*, each classifier is tested for multiple Q values. Regardless of the value Q, the 80% of the dataset has been used for training and the remaining 20% for testing unless otherwise specified.

As the number of feature vectors of a single EV is smaller than the overall number of feature vectors, when considering small Q values, the set of feature vectors associated with other EVs is randomly created from the overall set. Another value that can lead to different results is the number of features *NoF* employed in the classification. Since our feature extraction strategy returns about 1500 features, using them all can lead to overfitting. We show in the next sections how different *NoF* values affect the classification performance and provide a justification for choosing a suitable *NoF* value that provides a good threshold to balance classification performance and overfitting.

3.3.5 *EVScout2.0* Performance: Vehicle Profiling

We first assess the performance of *EVScout2.0* in terms of profiling every vehicle based on its charging behavior. To this aim, we implement and compare common machine learning algorithms for classification. We describe in Section 3.3.5 the classifiers, and we provide a discussion on their hyper-parameters setting. Then, in Section 3.3.5, we present the results obtained via *EVScout2.0* with the different classifiers.

Classification Algorithms

Once features have been identified and selected, *EVScout2.0* feeds them to a binary machine learning classifier. The profiling task can be formulated as a supervised classification problem, where a two-class classifier is trained with both features from the target EV and features from all other EVs. In particular, we assume that a classifier whose input is the features vector from the target EV shall return output value 1, otherwise it shall return output value 0. In literature, this approach is also called *One-vs-Rest*, and more precisely, it aims at creating a specific model for a single device by using the other class, composed of other different devices, to create a decision bound around the class under consideration. If, on the one hand, this approach requires a different model for each class, on the other hand, it allows focusing on a single class, leading to a more robust model for the specific class.

We evaluate our pipeline by using and comparing six different common machine learning models which are often used in the field [437, 438], namely:

- SVM classifier [439];
- kNN classifier [440];
- DT classifier [441];

Model	Parameter	Values
SVM	kernel	['rbf']
	regularization (c)	$[1, 10, 10^2, 10^3]$
	gamma (γ)	$[10^{-4}, 10^{-3}]$
kNN	n_neighbors	$[1, \ldots, 10]$
	weights	['uniform', 'distance']
	metric	['euclidean', 'manhattan']
DT	criterion	['gini', 'entropy']
	max_depth	$[8, 10, 14, 30, 70, 110]$
LR	max_iter	$[5000]$
	regularization (c)	$[10^{-2}, 1, 10^2]$
RF	n_estimators	$[50, 200, 1000]$
	max_depth	$[10, 100, \textit{None}]$
ADA	n_estimators	$[10, 100, 500, 1000, 5000]$

Table 3.6: Grid search cross validation parameters for each model.

- Logistic Regression (LR) classifier [442]
- RF classifier [443];
- ADA classifier [444].

Hyperparameters optimization is obtained via grid search with cross-validation to fine-tune our models and extract the best results. Table 3.6 indicates the different parameters employed in the grid search for each model. The training set is suitably divided into training and validation sets, which we test on a grid of possible hyper-parameters. Notice that all six classifiers are standard machine learning algorithms without deep architectures. Although deep learning automates the feature extraction process, a large number of samples shall be used to train deep architectures effectively. The use of non-deep structures allows us to show the feasibility of *EVScout2.0* over our currently available dataset and, simultaneously, control the feature relations during the classification process. The same motivation resides behind the choice of binary classifiers. In fact, a single multi-class classifier can be designed to have a single class for each EV. However, multi-class classifiers require a larger dataset for training purposes than binary classifiers. Although we consider a larger dataset than our previous work [413], it is not big enough to provide interesting results with deep models or a multi-class scenario.

Profiling Results

We exploit the implementation of the classification algorithms employing the scikit-learn library [435]. Results are assessed in terms of Pr, Re, and F1. We present numerical results assessing the validity of *EVScout2.0* as a function of Q, the amount of unbalancing in the dataset. This measure provides evidence of how the model is resistant in constrained scenarios where the attacker has few charging traces available for the target vehicle. Since traditional performance measures such as F1 may be misleading when considering highly unbalanced datasets, the geometric mean (G-Mean) between recall and specificity has been proposed as a suitable performance metric [445]. Therefore, we consider G-Mean an accurate indicator of the validity of *EVScout2.0* for large Q values. By denoting as *TP*, *TN*, *FP*, and *FN* respectively the number of true positive, true negative, false-positive and false-negative outcomes, we can express the Re as $R = \frac{TP}{TP+FN}$, and Specificity as $\alpha = \frac{TN}{FP+TN}$. Geometric Mean (Gm) is hence obtained as G-Mean $= \sqrt{\alpha R}$. We recall that we present each score as the average over different vehicle-specific trained models to obtain more robust scores and avoid biases due to specific vehicles with a highly diverse charging profile.

Since the tail extraction process is automated, the number of extracted tails also depends on the parameter values. Based on our previous work [413], we set N_{avg} (i.e., the size of the moving average filter) to 25, which provides the best value in terms of classification scores. A smaller filter length would fail to remove noise on the TSs, while a bigger one would filter out important data characteristics. We used the same value for the median filter to maintain coherence in the TSs since, albeit, with the necessary differences, the two filters are quite similar. Notice that the filtering process aims to improve tail identification and classification performance. Therefore, the optimal filter length is obtained by a trial and error process instead of selecting a length based on, e.g., correlation analysis.

We employ the classifiers provided by the scikit-learn Python library [435]. For each model, we perform a `GridSearchCV` to tune the model using grid search with cross-validation. In the following, we present an analysis of the results with respect to different parameters and values.

Number of Features *NoF*

Firstly, we analyze the scores based on the number of features *NoF* maintained after the feature extraction phase. In Figure 3.21 we plot the F1 scores for different ratios Q using the same model (kNN) but varying the numbers of features *NoF* from 10 to 200. We can see a significant increase in the score going from 10 to 25 features, especially for higher ratios Q. From 100

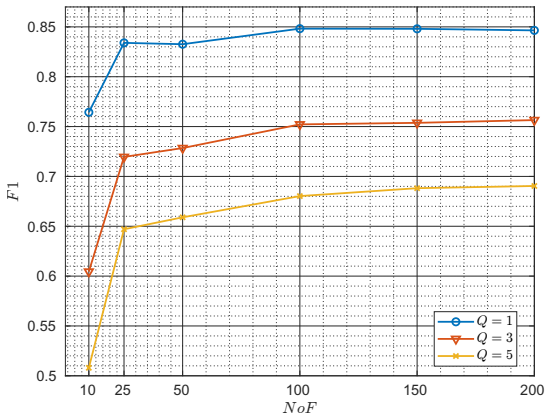


Figure 3.21: F1 score of kNN classifier with different numbers of features NoFs.

features up, the increase is instead negligible, meaning that the features already capture almost all the entropy available. For this reason, we selected $NoF = 100$ for the other experiments in this study unless otherwise specified.

Unbalance of the Dataset Q

To understand how our models deal with the unbalance of the dataset (i.e., the ratio between the number of feature vectors associated with the target EV and the number of feature vectors associated with other EVs), we test all the six models against different values of Q. We test Q values ranging from 1 (i.e., the same number of feature vectors for the target and the other EVs) to 5 (i.e., five times more features vectors not related to the target EV). We crafted the *non-target* class in the training set to obtain more robust scores by randomly sampling the charging traces among all the *non-target* vehicles. Furthermore, to make the classification problem more "open-world", we inserted in the test set, as *non-target* class, traces of vehicles without occurrence in the training set.

Figure 3.22 shows the results obtained by *EVScout2.0* for different Q values. It is possible to notice how all the scores decrease with Q for all six models. As the number of considered EVs increases with Q, the chance of two users having the same EV model or having EVs with similar charging profiles increases. This is reflected in a worsening of classifiers' performance. With no unbalancing (i.e., $Q = 1$), we reach the highest precision of 0.88 with the RF classifier. On the other hand, almost all the classifiers reached at least 0.86 in the recall, except for DT, which has lower performances almost for every indicator. We notice that the RF classifier is the most resistant to changes in Q if we look at the precision, while LR offers the best recall scores. If

we look at both scores, even if the absolute values are generally lower, kNN and ADA seem to be the most resistant to higher Q. Furthermore, we can observe that for the maximum $Q = 5$, precision and recall are respectively 0.77 (with RF) and 0.71 (with LR), meaning that EVs can still be profiled with sufficient confidence.

As for the other scores, also F1 degrades for increasing values of Q for all the models. We can see that RF is the best one for almost all the ratios Q, while for the highest two (i.e., $Q = 4.5$ and $Q = 5$) ADA performed slightly better. However, this is not the case regarding G-Mean, confirming its validity for unbalanced datasets. In particular, ADA presents a large variance in terms of G-Mean, a sign that is not a suitable algorithm for highly unbalanced datasets. Other algorithms, such as RF, kNN, and LR, are instead able to maintain high values of G-Mean for every value of Q. This shows that profiling can be achieved with good results irrespective of the amount of unbalancing in the dataset, i.e., a single user can still be profiled based on its charging profile also in largely populated networks.

All these considerations must be considered when designing *EVScout2.0*. The best algorithm for each case can be selected if the dataset distribution is known. In particular, RF is advisable for perfectly balanced datasets, LR for highly unbalanced datasets. Instead, if the dataset is unknown and a resilient model is needed, kNN can be a good choice. In fact, we employed kNN in many experiments from now on, also thanks to its fast training time with respect to other models such as RF or ADA.

3.3.6 *EVScout2.0* Performance: Additional Performance Analysis

In addition to the classification-based analyses, we included additional scenarios based on the training set characteristics. Since the data conditions may be different in a real-world scenario, we propose an analysis considering different data properties in the following. In particular, in Section 3.3.6, we examine different training set size values to assess the minimum number of charges needed to obtain sufficiently high classification scores. In fact, a large number of labeled traces in a real-world attack may be challenging to obtain. In Section 3.3.6, we investigate if and how much the Li-ion battery's degradation impacts the performance of *EVScout2.0*. This analysis can be useful to understand how a model can still be precise when dealing with natural phenomena like physical battery degradation.

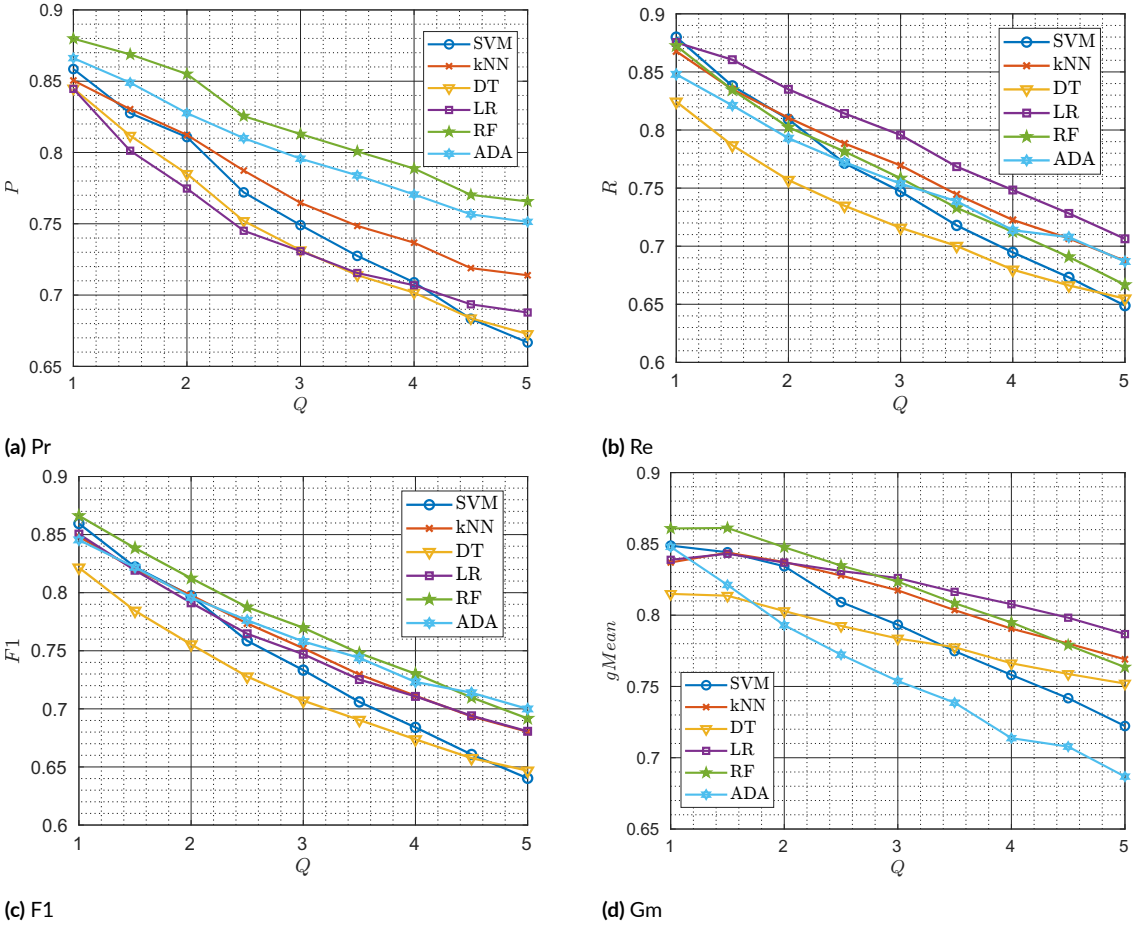
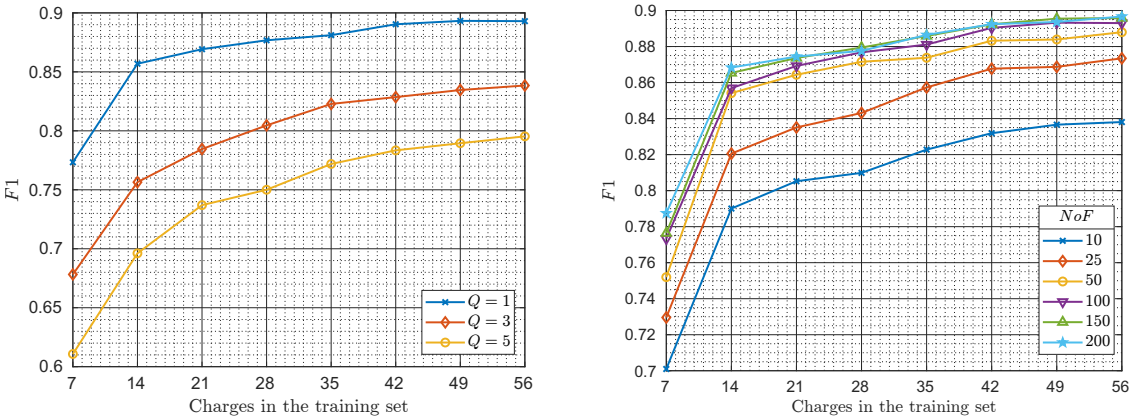


Figure 3.22: Performance of *EVScout2.0* for a different amount of unbalancing in the dataset. Results are shown for the different classifier algorithms and $NoF = 100$. We see that good classification performance are obtained for all classifier. As Q increases, some classifiers are more robust than others to increasing unbalancing. Results on G-Mean show that profiling can also be achieved in largely populated networks.

Training Size Variation

In a real-world scenario, an attacker may be limited by the number of labeled charges s/he can get for a target vehicle. For instance, the attacker may not want to leave its malicious device in the field too much to reduce the possibility of being detected. To simulate this scenario, we analyze the performance of *EVScout2.0* while varying the train set size. We avoid using as target EVs with less than 70 charges with tails, while to create the group with the others EVs we employed the whole dataset. As a testing set, we always use the last 20% of the available charges. For the training set, we set fixed values from the 80% to the 10% of the min number of charges for each EV (i.e., 70). In other words, the training set size ranges from 7 to 56 feature vectors



(a) Training set size changing.

(b) Changing NoF and training set size for $Q = 1$.

Figure 3.23: F1 scores of kNN classifier while reducing the training set size.

for each EV, with steps of 7 charges.

In Figure 3.23a, we can see how the F1 decreases while decreasing the training set size. However, we can appreciate a significant fall only for the smallest values of the training set, while the increase is less relevant for training set sizes bigger than 14. Above 42 charges, the performance increase is almost negligible, especially considering the smaller Q ratios. We can also look at the absolute values of the F1. It is greater than 0.69 for the most unbalanced dataset when using 14 feature vectors, showing discrete classification performance even with small training sets. By considering the total number of vehicles per training set size and the imbalance ratio, we can compute the number of vehicles of the target class in the different scenarios. In particular, we can notice that with 7 charges (i.e., the training set size 14 if $Q = 1$, training set size 21 if $Q = 3$, and training set size 35 if $Q = 5$), we can obtain an F1 Score of at least 75%. With only 7 charges, we can achieve a good classification precision for the target vehicle.

Furthermore, we also analyzed the impact of both training set reduction and feature reduction. In Figure 3.23b, we see the different behavior of the F1 score while varying the number of features and the training set size (we show the data only for $Q = 1$ for clarity). As expected, there is a clear drop in the performances for the lower training set sizes. However, this can be partially compensated by employing a bigger number of features, up to 100. Over this threshold, the gain is negligible, coherently with what is presented in Section 3.3.5 and Figure 3.21.

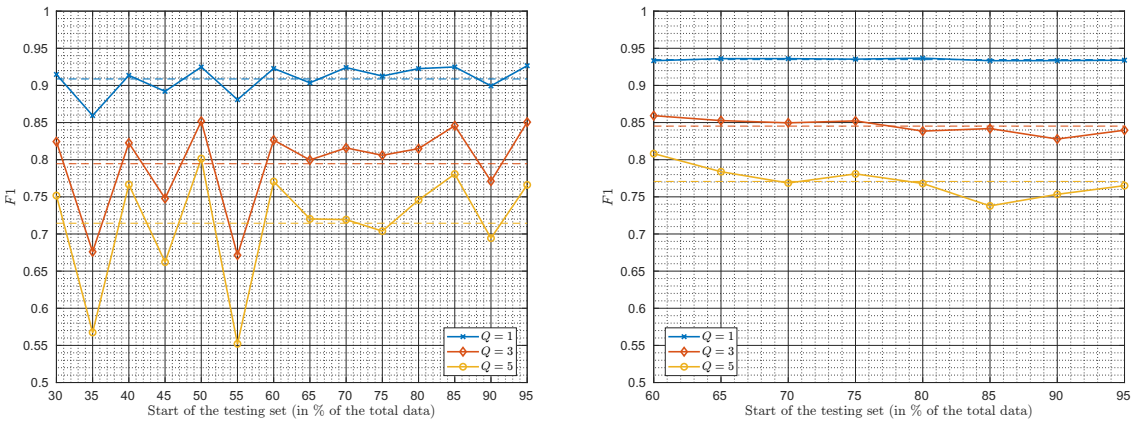
Battery Degradation Performance

The degradation of a Li-ion battery used in an EV is widely discussed in literature [398]. During a battery life, many aspects can adversely affect its performance depending on the number of charge cycles, aging, operating, and storing conditions. Degradation on EV batteries can lead to a shorter traveling distance and a reduced battery's available power output [446]. Generally, a battery is considered at the end of its life when it has already lost 20% of its initial capacity. Modern Li-ion batteries should have a five to ten-year calendar life depending on the materials and how they are managed. They should be able to supply between 1000 and 2000 cycles of charge. However, many behaviors, such as keeping the battery at a high SoC or high temperature, or often employing fast chargers, can reduce even more the lifetime of the battery [447, 446].

Being aware of this problem, we investigate if the battery degradation affects our profiling model. Since the charge profile of an EV will always be different due to the variation of the physical properties of the battery, we analyzed how the extracted features degrade the performance of *EVScout2.0* in time. To test how our model behaves in this context, we selected the 10 EVs with the higher number of charges (i.e., more than 150 charges for each EV) spanning about two years. We train the kNN classifier in the first part of the data, and we then test it over multiple consecutive test set batches. In particular, the training set accounts for the TS measured in a specific period and represents the baseline to measure the successive battery degradation. Each testing set is given by batches of Z chronologically consecutive TSs, with $Z = 5\%$ of the total available testing data. In other words, we considered as each test set a sliding time window of consecutive TSs.

Initially, we train employing only the first 30% of the data for each EV. These results are shown in Figure 3.24a, where it is possible to see a small difference in scores while shifting the testing set. However, we cannot blame the physical battery degradation for these results for a series of motivations. Firstly, the reduction is not important in absolute terms (i.e., less than 0.1 for $Q = 1$) and can be due to different behaviors of the users in different periods (e.g., detaching the EV before full SoC). Secondly, it is not the monotonous decrease that we would have expected. Instead, the scores seem not to follow any pattern, showing good classification results also in the last steps (i.e., testing set given by more recent TSs). Third, the small size of the training set employed can be a partial motivation for the random behavior of the scores.

To remove the train set size as a variable to create strange behavior in the results, we perform a second train using the 60% of the data. By doing so, we obtain a chart with fewer points, as



(a) Train set size 30% of the data.

(b) Train set size 60% of the data

Figure 3.24: F1 score for different ratios while varying the start of the testing set. Each value in the horizontal axes represent the score on a testing set composed of the 5% of the data starting from the point forward. Dashed horizontal lines represent the mean of the F1 for each ratio.

shown in Figure 3.24b. In this case, scores are almost constants with tiny variations which are expected in the context.

This demonstrates how *EVScout2.0* can perform well, also considering a time distance of almost two years between the training and the testing data. Although not affected by the small battery degradation that could happen in this time frame, it could still be affected by unusual and unpredictable behaviors of the EVs owners or by the ACN scheduling algorithm, especially when considering highly unbalanced datasets and a small training set. Furthermore, this experiment can be seen as a remark on the need for a big training set as presented in Section 3.3.6. We will provide more analysis on the degradation in future work, considering datasets that span more than two years.

3.3.7 EVScout2.0 Performance: Comparison with EVScout

In our previous work [413], we performed experiments similar to those discussed in previous sections but employing a different pipeline, a different feature extraction process, and a smaller dataset composed of only 22 EVs. Since it was one of the first works on the privacy of the EV charging systems, we use the results of [413] as a comparison baseline. In particular, we propose two different comparisons: one using the previous EVScout on the new and extended dataset, and one using the new *EVScout2.0* on the small dataset employed in [413]. It is worth mentioning that we employed the exact dataset used in [413] and not simply the same EVs

updated with the new charges.

EVScout on the new dataset

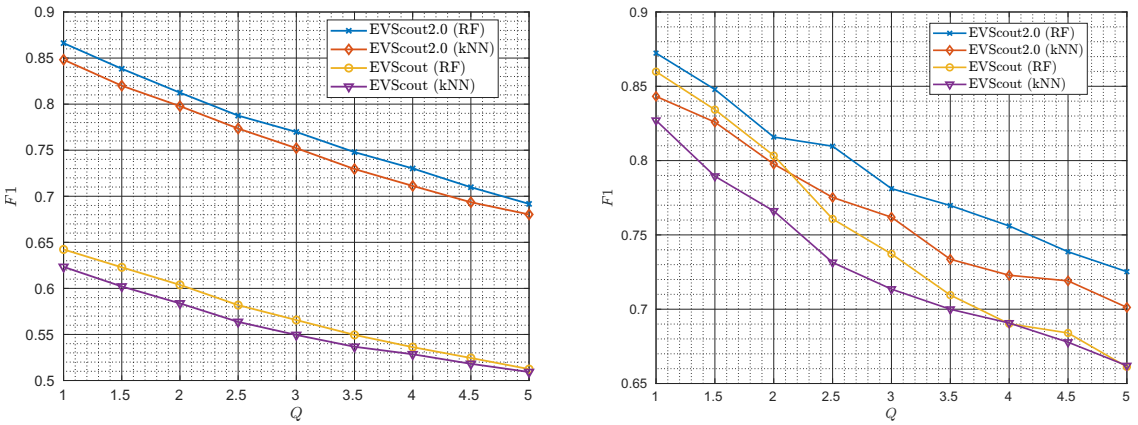
We tested EVScout [413] in our new dataset. To generate results comparable with those presented in this study for *EVScout2.0*, we employed the same dataset with 137 valid EVs, even if EV with less than eight charges could also be used in EVScout. We performed the same pre-processing phases and assessed the performance for $N_{avg} = 25$. Results are shown in Figure 3.25a for different values of Q. We can see a clear dominance of *EVScout2.0* with respect to its previous version in the new dataset composed by 137 EVs. The difference in the classification performance ranges from about 0.15 for $Q = 5$ to more than 0.20 for $Q = 1$. We remark that the different charging traces (also for the same vehicle) belong from different EVSEs in the same site. Therefore, the results confirm that the models implemented are resistant to multiple EVSEs and that different charging stations do not significantly impact the charging behavior.

EVScout2.0 on the previous dataset

Furthermore, we test our new algorithm *EVScout2.0* on the same dataset employed for the testing of EVScout. Since *EVScout2.0* needs at least 8 charging session comprising tails for each EV, we have to discard four EVs, resulting in a total of 18 EVs. We show the results of this experiment in Figure 3.25b. Even if the enhancement is less pronounced with respect to the new dataset scenario, we can see a clear improvement in the performance of *EVScout2.0* with respect to its previous version, especially considering high unbalancing Q.

3.3.8 Possible Countermeasures

To deal with information leakage from smart meters [448] proposes to obfuscate the communication between user and supplier through rechargeable batteries. This solution is effective in modifying the demand-response correlation in the measured data. This approach can be leveraged to mitigate the effects of *EVScout2.0*, where the EV's battery drains current from a secondary battery which communicates with the EVSE, masquerading the original battery's tail behavior. However, as the number of involved batteries doubles, this approach incurs a higher implementation cost at both EV and EVSE sides. Furthermore, the attacker may be able to extract features from the secondary battery and still perform profiling. A similar concept is exploited in [449], where noise is added to smart meters data via an adversarial learning



(a) Comparison on the new dataset.

(b) Comparison of the original dataset.

Figure 3.25: F1 values of the comparison with the previous work [413].

framework. This idea can be exploited to mitigate the effects of *EVScout2.0* by adding a suitable amount of noise to the current required by the EV's battery during the tail phase. However, this would imply redesigning how EVSEs manages the current required by EVs, as the added noise may mislead both the attacker and the EVSE. The risk is, therefore, that the recharging process's efficiency drops. In [419], the authors successfully applied a low-pass filter to remove the information leakage due to the high-frequency components from the signal. However, the work targets mobile devices, which are very different from vehicles, with tiny batteries and different charging patterns. Other non-technical countermeasures are also possible. For instance, EV owners can be educated to check the presence of suspicious devices attached to the EVSE. Furthermore, running companies should often inspect their equipment for illegitimate devices and install closed-circuit TV to detect the presence of such devices. As described above, several solutions in the literature prevent the leaking of power side-channel information. However, most of them have been studied for the smartphone environment and are rarely implemented in reality. In future work, we will focus on identifying ad-hoc solutions to prevent the inference of sensitive information via power-side channels, specifically in communications between EVs and EVSEs.

3.3.9 Takeaway on EV Profiling

As we demonstrated, introducing a charging system that utilizes personal information can lead to privacy leakage and profiling attacks. In this study, we extended our previous work proposing *EVScout2.0*. In particular, we extended the work in [413] by employing a bigger dataset, and

a novel feature extraction technique, and compared more algorithms for the classification task. We also show how real-world constraints such as limited training test size and battery degradation over time impact the classification quality. With respect to the previous work, we employed a bigger dataset going from 22 to 137 EVs. We employed six models capable of reaching precision and recall of 0.88 for the balanced datasets while still offering good results (precision 0.77, recall 0.71) with highly unbalanced datasets. Furthermore, we evaluate the performance loss generated by a training set size reduction, showing how *EVScout2.0* can reach good performances even with a small training set. In addition, we assess that the proposed algorithm can correctly identify EVs even if the model is trained with data two years early. Finally, we showed that *EVScout2.0* is capable of attaining good classification performance, even in challenging scenarios such as highly imbalanced datasets or small training sets, proving to be a viable and effective solution for EV profiling. We believe this work can warn all the parties involved (i.e., the users, the manufacturers, and the scientific community) about the feasibility of profilation attacks in the growing V2G infrastructure.

4

Cross-domain Cyber-Physical Systems Applications

In the previous chapters, we focused on the security issues of two widely studied CPS applications. As discussed, CPSs expose a wide range of services to dangerous threats due to their complexity and the intertwining between the physical and IT worlds. Therefore, it is important to approach the CPS security analysis from multiple angles. In this chapter, we leverage the knowledge of the previous studies to investigate other CPS application domains. In particular, in Section 4.1, we present a survey on the usage of PSC in the literature, focusing on both the existing attacks and countermeasures [415]. As will be discussed, PSCs have been proven effective in reversing and profiling the functioning of many embedded devices (e.g., smart cards, vehicles, and laptops). Then, in Section 4.2, we extend the approach applied in EV in Section [450], and we show how it is possible to fingerprint USB devices. This funding can be used, for instance, to securely authenticate a personal device and avoid malware delivery injection in critical applications (e.g., Stuxnet). Finally, in Section 4.3, we present the first security analysis of the emerging Hyperloop transportation technology [451]. Hyperloop merges the concepts of ICS since it consists of a critical, distributed, and sensing infrastructure, and the concept of vehicle, due to the pod communication management. As a result, Hyperloop inherits all the vulnerabilities and risks of the two systems.

4.1 Power Side-Channel Overview

The wide spread of smart devices such as smartphones and IoT stand-alone devices led to the diffusion of new standards to include them in daily life. According to Statista [452], by the end of 2020, 44.9 percent of the world's population is projected to own a smartphone. Most of such devices rely on the USB standard for two main functions: data transfer and power charging. While USB protocol security is a topic widely studied, less importance is given to the charging application.

Nowadays, most high-end smartphones have an average duration of around 11 hours with a normal user usage [453]. Users rely on power bank suppliers or charging stations in public places to deal with this limitation. However, generally, common users do not take seriously the possible threats that can occur when connecting their mobile device to untrusted USB ports. A survey [454] conducted in 2016 involving 1, 439 volunteers from multiple industries revealed that most volunteers were unaware of any risks associated with charging their phones in public places. Worse, many trusted mobile charging stations and wall plugs interact with their phones while using them. In 2018, another survey [455] in the form of a questionnaire involving 2, 500 participants showed that users are less concerned about smartphone charging threats compared to mobile malware. These results underline that a malicious attacker may exploit users' unawareness to compromise charging tools, steal sensitive information from users, or compromise the device plugged in.

Several works showed that it is possible to leak sensitive user information through unintentional physical and electromagnetic phenomenons, such as vibration [456], noise [457] or power consumption [458]. These attacks are generally referred to as side-channel attacks. The strength point of such attacks is that they rely on the system's physical uncontrolled characteristics, which are difficult to avoid considering during the design phase. Furthermore, while traditional attacks need to take control of the targeted device actively, side-channel attacks are passive and hard to detect. For instance, an attacker may compromise a power bank or a public charging station by adding an integrated module to deliver malicious payloads [459] (i.e., active attack) or to collect power consumption traces (i.e., passive attack). In both cases, the attacker would directly or indirectly steal private user information from mobile devices.

This study provides a broad overview of the side-channel attacks that exploit a USB connection. This class of attacks is generally underestimated by users, even if it can raise significant privacy threats. Once presented and analyzed the main side-channel attacks in the literature, we also survey the countermeasure implemented to prevent such attacks. Furthermore, we present

the different tools employed in research to discuss attack feasibility and countermeasure development. To the best of our knowledge, we are the first work that surveys this topic.

4.1.1 Related Work

The literature includes several surveys that collect countermeasures for USB attacks. However, different analyses and aspects are taken into consideration. Most of those surveys focus only on the software and hardware vulnerabilities of the two devices involved in the USB communication. This survey analyzes the side-channel effects of USB powering and communication.

In [460], the authors present a detailed survey related to the main attacks which exploit the USB standard. The authors specifically focused on malware injection via USB devices and software attacks on USB drives with hack tools. However, this work is relatively outdated (i.e., 2011), so it does not consider recent attacks using malicious USB peripherals or the BadUSB attack family and side-channel attacks. Most attacks are based on AutoRun and AutoPlay vulnerabilities, which are already solved and deprecated in the recent Operative Systems version [461]. Moreover, this work provides a multilayered security mechanism based on software implementations in the operating system. However, since 2011, several new attacks were developed, and the proposed solution may be ineffective, mostly on recent side-channel attacks.

In 2017, Nissim et al. [462] reported 29 different USB-based attacks analyzing the goal of such attacks and the corresponding weakness exploited. The attacks were divided based on the USB hardware required for executing the attacks: i) programmable microcontrollers, ii) the common USB peripheral devices that can be found in most organizations and households, and iii) crafted devices composed only from electrical hardware component. The authors also present different tools for detecting USB attacks. However, none of the presented attacks considers using USB for charging purposes or the communication side-channel effects.

In [463], the authors present an accurate Systematization of Knowledge (SoK) to survey and categorize USB attacks and defense mechanisms by also considering the recent USB *Type-C* standard. The authors classified the USB vulnerabilities dividing them into different communication layers (i.e., Physical Layer, Transport Layer, Application Layer, and Human Layer). The survey also includes a description of the different vulnerabilities at each layer, the corresponding countermeasure, and the proposal of an authentication system for USB *Type-C* standard. Although the survey covers all security issues related to the USB standard, it gives little space to side-channel attacks, leaving out the most recent ones, such as the Juice Filming Charging (JFC) attack.

The literature also includes different surveys about side-channel attacks. In 2009, Cai et al. presented one of the first works [464] collecting and explaining the classes of threats related to mobile phone sensors' information. However, this study does not consider modern machine learning-based attacks. A recent and complete survey was proposed by Spreitzer et al. [465] in 2017. In this work, the authors proposed a systematic classification of side-channel attacks according to the information extracted, the attacker's position, and the type of attack (i.e., active modification or passive sniffing). The authors proposed the classification of side-channel attacks as the main contribution. In our survey, we follow this classification, but we consider a threat model that no other work has not previously explored in the literature. Moreover, in [465], the authors analyzed only one USB electric power-based side-channel attack. Therefore, all more recent machine learning-based power trace classification techniques are not included. More recently, Conti et al. [466] presented an analysis of the literature on network traffic analysis techniques on mobile devices and related datasets. However, the authors focused only on the side-channel attacks related to the network traffic analysis.

To sum up, this survey focuses on the side-channels rising from the physical connection between the two devices via a USB cable. By passively observing and analyzing these side-channels, attackers can extract sensible information from the involved devices. Furthermore, we provide an overview of the countermeasures to prevent such attacks and the tools to support the research in this field. To the best of our knowledge, this is the first survey to cover such an emerging class of attacks on USB connections.

4.1.2 Background

In this subsection, we recall the main concepts about USB useful to understand the remainder of the study. In particular, subsection 4.1.2 will briefly recall how USB environment and how it works while subsection 4.1.2 will recall the concept of side-channel.

USB Ecosystem

Universal Serial Bus is the most popular standard used to wired connect peripheral devices (i.e., USB devices) to a central host and transmit information. The first USB version 1.x was introduced in 1996 [467] by a group of seven companies to standardize the connection between the different external peripherals and the PC. This first version supports a transmission velocity of up to 12 Mbps. During the years the USB standard evolved, passing through versions 1.1 [468], 2.0 [469], 3.0 [470], 3.1 [471], 3.2 [472] up to the latest version 4.0 [473] which

support a transmission velocity up to 40 Gbps. Different plugs, transmission velocity, and power delivery capabilities characterize each standard. Therefore the devices supporting USB communications are heterogeneous and not always backward compatible. Over the years, the evolution of the USB standard allowed the integration of new functions in addition to transmitting information, such as the ability to power or recharge stand-alone devices. Therefore, the evolution of the USB standard and the use of USB for multiple functions allowed to increase in the spectrum of compatible peripheral devices, opening, on the other hand, new dangerous vulnerabilities surfaces. Implementing new features in an insecure standard by design forced the research community to find alternative solutions to secure the security lacks.

As reported in [463], security problems related to the USB standard are present at many communication levels, and during the evolution of this standard, security was rarely considered. Despite that, in 2014, the Universal Serial Bus Implementers Forum (USB-IF) explicitly stated that security falls outside the USB specification scope. According to [463], USB communications rely on four main layers, as depicted in Figure 4.1. In this categorization, the higher level consists of the Human Layer. This layer involves all the actions and interactions between the host and the end device. There could be communications where both entities rely on humans, e.g., data transfer between PC and smartphone, or a human can control only one of the two sides, e.g., smartphone charging. The action performed at this layer affects all the underneath layers creating unique functioning patterns. Attackers can exploit this layer through social engineering attacks or human errors. The Application Layer comprises the user-level programs and software on the host and the device sides. Generally, at this layer, the main attacks are Code Injection (e.g., Stuxnet [3], Duqu [474], and Conficker [475]) and Code Exfiltration (e.g., USBee [476]). The Transport Layer regulates the data transmission policies between the two entities involved in the communication. The main attacks at this layer are Protocol Masquerading and Protocol Corruption (e.g., Kernel level bugs [477] and USBProxy [478]). Finally, the physical layer comprises signals representing communications occurring at the lower level over the USB bus. Typical attacks at this level include Signal Eavesdropping (e.g., USB snooping [479]), and Signal Injection (e.g., USB Killer [480]) This survey will focus on USB Communication's physical layer, which is the vector of the so-called Side-Channel attacks.

In version 1.0, the USB standard supported only two plugs type: Type-A and Type B. The wires used in the cable are depicted in Figure 4.2. USB 1.1 introduces other four plugs: Mini-A, Mini-B, Micro-A, and Micro-B. These versions rely on the same transmission wires introduced in version 1.0 but introduce new plugs to integrate devices with smaller dimensions. In 2016, the USB 3.0 Promoter Group and the USB-IF introduced the USB Type-C Authentication

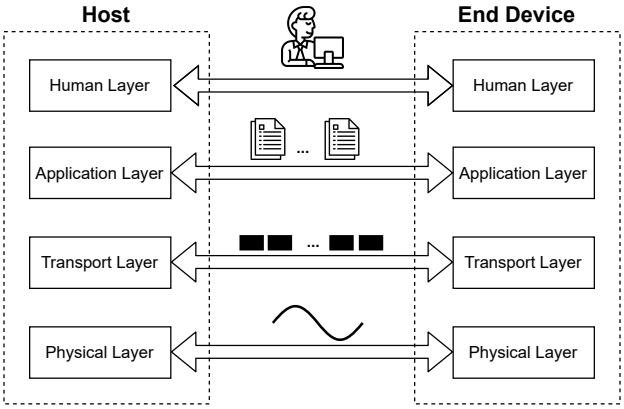


Figure 4.1: USB Communication Layers

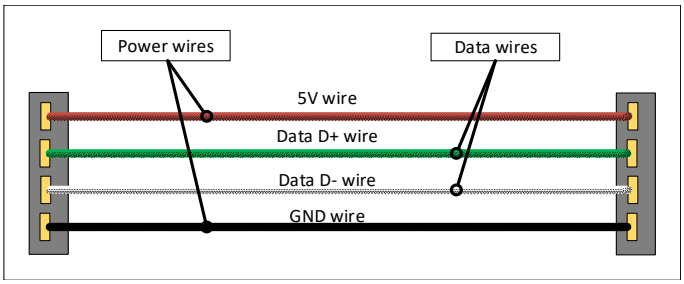


Figure 4.2: USB type-A and type-B wiring

specification [481] to Type-C products. Unlike type-A and B, Type-C or USB 3.0 generally has additional data and power wires, thus making both charging and data transmission faster. Furthermore, from version 3.1, USB standard allows transmitting also video signals. USB Type-C Authentication is based on a Self-Signed Certificate, which end-devices use to authenticate with hosts. However, as shown in [463], USB Type-C Authentication protocol reveals many security problems when formally verified. More recently, in 2019, USB-IF released a new version of Type-C Authentication specification [482]. However, no related security study by the research community has been published so far. Instead, in the more recent standard USB 4.0 [473], the words "security" and "secure" are never mentioned in this 549 pages document. The only security aspect mentioned is "authentication".

In recent years USB standard was also introduced in the smartphone world. In fact, USB On-The-Go (OTG) standard [483] is used to allow recharging the device and to connect peripheral devices (e.g., USB flash drives, digital cameras, mouse or keyboards). In this scenario, the host is the power supplier (i.e., power bank, wall socket, a smartphone, or a general device with charging capabilities), and the end-device is the device powered by the host (i.e., a

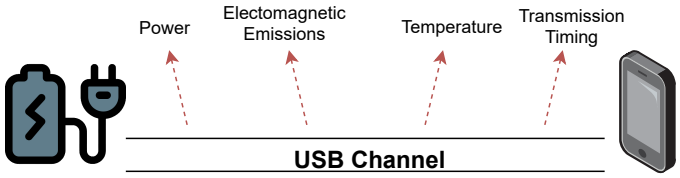


Figure 4.3: Example of possible side-channels during devices charging process.

peripheral device or the smartphone in a recharging scenario). Also, in this case, there is no reference to security features in the OTG standard specification [483]. Smartphones can also be connected via USB to a host running software to update or restore the device. Generally, the software used to perform critical operations requires user authentication only from the device side (i.e., password confirmation). In contrast, the software can perform any function without authentication, enabling potential malicious threats.

As described so far, the USB standard is designed without considering security. However, USB is used in a wide range of applications, ranging from critical applications (e.g., ICSs) to home applications (e.g., Internet of Things (IoT) devices). Furthermore, the devices relying on such communication are very heterogeneous with different requirements and specifications, opening also a problem of standardization of security measures. For these reasons, it is important to identify security threats arising from the insecure USB standard design and develop new security solutions to prevent potential malicious events.

Side-Channel Attacks

Side-channel attacks refer to those attacks exploiting uncontrolled systems' physical characteristics to compromise the confidentiality or the integrity of the system. Side-Channel attacks were originally implemented to break cryptographic primitives exploiting common patterns in the power consumption to reverse the algorithm (e.g., Simple Power Analysis and Differential Power Analysis). However, in recent years side-channel attacks have also been implemented to profile actions in different encrypted network traffic scenarios such as smartphone user actions [484, 485], IoT actions [486], and internet traffic [487]. A side-channel attack's main idea is to leverage uncontrolled physical characteristics produced by the system during its functioning (e.g., power consumption, noise, timing) to identify confidential information and functioning patterns. Figure 4.3 shows different possible side-channels derived from Smartphone charging process.

According to the categorization in [465] side-channel attacks can be categorized differently.

Generally, side-channel attacks are *passive* since they aim to passively listen and leak information to learn confidential information or model the system's behavior. However, an *active* attack can rely on side-channels to corrupt the system's physical characteristics (e.g., injecting a signal) to modify the system's normal function or tamper with it. Based on the information extracted, the side-channel attacks can target *physical properties* (e.g., power consumption or electromagnetic emanation) or *logical properties* (e.g., statistics provided by the operating systems). Finally, we can also categorize the attacker's position according to its target. An attacker can be *Local* if it is in direct control of the device. *Vicinity* attackers are nearby the target device and can, for instance, wiretap or eavesdrop on the communication. *Remote* attackers only rely on software execution on the targeted device, for example, employing back-doors.

4.1.3 Threat Model

As modern mobile devices' applications provide more and more advanced functionalities and personalized user experiences, they are also increasingly becoming energy-demanding. The intensive usage of those apps, combined with their need for energy, often causes batteries to be quickly depleted. For this reason, users need to charge their devices multiple times a day due to their long usage time and limited battery capacity [488]. The increasing demand for ways for users to charge their devices led to the growth of the mobile charging industry [489] and the installation of mobile charging stations in public places where everybody can access them, such as airports, coffeehouse, or libraries [490]. While public mobile charging stations provide users with a convenient solution, plugging smart devices with unknown charging ports can expose the user to potential threats [491].

In this survey, we consider the security and privacy research work that considers a threat model where a mobile device is connected to a USB port via a cable. In particular, we consider an attacker that can observe one or multiple side-channels by controlling or by accessing the surrounding of the USB charging port. In the threat model considered by this survey, we define the device that provides a USB port (e.g., public charging station, power adapter, USB hub, desktop PC) as *host device*, the portable device (e.g., laptop, smartphone, tablet computer, portable hard drive, keyboard) powered by such USB port as *end device*, and the USB connection cable as *channel*. Figure4.4 depicts the various components of the threat model.

Considering an attacker that has control over a host's USB port, the USB cable, or the USB hub, we can identify three attack scenarios as proposed in [492]: public charging stations, semi-charging stations, and chargers borrowed from others. We refer to public charging stations as

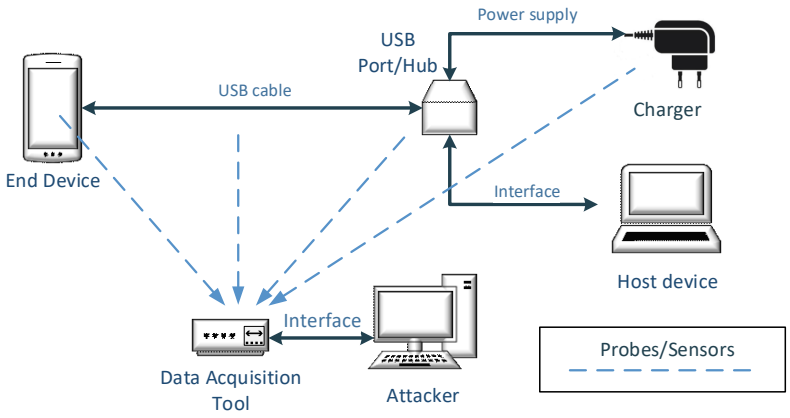


Figure 4.4: The threat model considered in this work and the possible sensor/probe placements for side-channel measurements (i.e., the source of the dashed arrows).

the USB ports on appliances available in public places such as airports, shopping malls, and entertainment venues. In this scenario, the attacker tampers with charging stations and exploits users' default trust in the person providing the charging station. Semi-public charging stations are devices that USB ports cannot be fully attended by users all the time. An example of a semi-public charging station in a hotel, a user has no control over nor can inspect the assigned room before check-in. During this time, an attacker can access the room and deploy its attack tools on the available USB ports. In the third scenario, users can borrow a charger from another person to charge their devices. In this case, an attacker could have previously tampered with such a charger and deployed its attack tool.

Considering an attacker that does not control the host's USB port, we assume that the attacker can access the immediate surrounding of such a USB port, the end device, or the channel. In this situation, an attacker can place a probe (e.g., antenna, sensor) and observe wireless side-channels such as electromagnetic, radio-frequencies, or sound emission from either the end device, the channel or the host.

4.1.4 Attacks

The wide diffusion of the USB standard in daily life and the low cost required to reproduce a usage scenario allowed researchers to study in-depth the vulnerabilities associated with the USB environment. This subsection categorizes the work according to the type of side-channels an attack relies on. The side-channel-based attacks considered in this survey are summarized in

Year	Work	Target device	Targeted Information	Analysis	Side-channel	Location & components
2006	Oren [493]	Host device (PC)	Key extraction	Frequency spectrum	Voltage variance	USB port, end device
2009	Vuagnoux [494]	End device (keyboard)	Keystrokes	Matrix Scan Technique	Electromagnetic (EM) emissions	Magnetic probe
2015	Genkin [495]	Host device (laptop)	Key extraction	Signal processing	Voltage variance	USB port, end device
2015*	Meng [492, 454, 496, 455]	End device (smartphone)	Screen and user inputs	Machine Learning, OCR	Mobile High-definition Link (MHL) standard	Modified USB charger
2016	Belgarric [497]	End device (smartphone)	Elliptic curve encryption	Machine learning	EM emissions	Triggering app, magnetic probe, electric probe
2016	Genkin [498]	End device (smartphone)	Elliptic curve encryption	Machine learning	EM emissions, Electric current	External Coil, USB pass-through
2016	Sim [499]	End device (keyboard)	Keystrokes	Signal processing	EM emissions	Oscilloscope
2017	Spolaor [458, 485]	End device (smartphone)	Personal data	Signal processing	Electric current	Malicious app, USB port
2017*	Yang [500, 501]	End device (smartphone)	Web pages, under Tor	Machine learning	Electric current	USB port, Host
2017	Su [479]	End device	Transfer data	Signal processing	Voltage variance	USB hub, Host device
2021	Cronin [502]	End device (smartphone)	Screen and user inputs	Machine Learning	Electric current	USB port, Host device

Table 4.1: Side-channel based attacks on USB powered devices by year (* year of the first original proposal).

Table 4.1.

Electromagnetic Emissions

EM emissions (or radiations) [503] are waves that can propagate in space, and they include visible light, radio waves, microwaves, etc. For example, such emissions are generated by electrons' flow from the data transfer on a USB cable that acts as an antenna. Attackers can exploit this phenomenon to extract cryptographic keys and keystrokes recognition.

Attacking Elliptic Curve-based Encryptions. The Elliptic Curve Digital Signature Algorithm (ECDSA) security relevance is meaningful since important companies employ elliptic curve encryption in their payment systems. Genkin et al. [498] demonstrated that ECDSA implementations on mobile devices are vulnerable to electromagnetic and PSC attacks. Based

on this, the authors perform such attacks by proposing a complete and inexpensive framework prototype. In the paper, the authors consider two similar threat models: an end device lying over a coil or connected to a USB cable. In the former threat model, the coil senses the electromagnetic emissions from the end device's processor while performing. These traces can be acquired by the attacker's PC via an inexpensive device (i.e., a sound card). With this setting, the authors can extract the ECDSA signature key used by the OpenSSL library running on iOS and Android devices. In the latter threat model, the authors perform the same attack by relying on power analysis over the USB cable (see subsection4.1.4).

In a similar work, Belgarric et al. [497] show that, for elliptic curves over the prime domain, private keys on smartphones can be recovered using electromagnetic side-channels and lattice simplification techniques. In this study, the USB cable is assumed to be the only I/O interface of the end device (i.e., a smartphone), and it is used as a trigger for the oscilloscope via self-written software installed on the targeted device. Such software sends a trigger signal through the end device's USB interface (connected to the oscilloscope's trigger port) and immediately invokes the Bouncy Castle signature function. The electromagnetic radiations are then acquired and analyzed by manually observing additive operations in the collected traces. Similar to the previous attack, the authors apply lattice-based cryptanalysis on such traces to obtain the cryptographic keys.

Keystrokes Recognition. Considering a USB wired keyboard as the end device, researchers show how to infer keystrokes from electromagnetic emanation side-channel. In 2009, Vuagnoux et al. in [494] conducted an extensive study on this kind of attack by testing its feasibility on different keyboard technologies, among which PS/2, USB, laptop, and wireless. In particular, the authors assess Matrix Scan Technique as the most effective method for keystroke inference on the USB cable. Under similar settings, a brief study by Sim et al. in [499] aims at inferring keystrokes from wired keyboards only via signal processing.

Electric Power Analysis

This class of attacks consists of measuring the electric energy provided by a USB port to the end device. Since a USB port delivers energy to an end device using direct current, an attacker can rely on the measurement of electrical quantities: current (in Ampere), electric resistance (in Ohm), the difference of potential (in Volt), and power (in Watt). In this subsection, we explore the electric power analysis-based attacks that have been exploited to retrieve cryptographic keys, exfiltrate data, and infer content from either host or end devices. We underline that the attacks

targeting host devices typically rely on voltage variance measurement since they assume an open circuit scenario (i.e., no electric current flows from the 5V to GND pins). Besides, the attacks targeting end devices measure the electric current since the circuit is closed on a load (i.e., end device).

Cryptographic Key Retrieval. One of the first works that use energy consumption as a side-channel on USB is proposed by Oren et al. [493]. In this work, the authors measure the voltage variance on the host's USB port to reveal ongoing system activities. For example, it is possible to apply power analysis to infer operation signatures such as the execution of OpenSSL RSA decryption (the RSA acronym comes from the authors' family name of the original algorithm: Ron Rivest, Adi Shamir, and Leonard Adleman). This attack could be perpetrated by measuring the voltage variance via a connected end device. A similar scenario is considered by Genkin et al. in [495]. This work aims to retrieve RSA and ElGamal keys from a laptop via cryptanalysis and signal processing. Although the main focus of such a paper is electromagnetic emissions from the laptop's chassis, the authors also explore a use case involving USB, VGA, and Ethernet cables. In particular, the authors show that it is possible to perform the attack on the measured shield potential via a simple voltage-meter. It is worth noticing that both these works target the host device as the attack victim since they aim to extract its cryptographic keys.

Data Exfiltration. To exfiltrate private information from an Android device, Spolaor et al. in [458] propose to use the power consumption during the charging process as a covert channel. In particular, this attack is carried out while the end device is in an idle situation (i.e., no user interaction, screen off) to achieve stealthiness. When the battery level is high enough, the authors observed that part of the unused electric power supplied by a USB port is directly used to power other mobile devices' functionalities. Based on this observation, the attack encodes a message (i.e., the targeted private information) as temporized energy over-consumption on the end device, which can be observable (then decoded) from the host device. Implemented for the Android operating system, this attack is composed of two components:

- A malicious application named PowerSnitch, installed on the end device, converts the information to exfiltrate in binary code. According to the value of the bit to transmit, the PowerSnitch app generates a Central Processing Unit (CPU) burst, which causes the over-consumption for a pre-determined time interval (i.e., energy pulses). To remain undetected, this app also checks whether certain transmission conditions are satisfied (e.g., the screen is off, Android Debug Bridge is not active, high battery level). Moreover, the PowerSnitch app does not require any special permission except to access the information to exfiltrate.

- On the host device, the attacker controls the power supplier (i.e., USB port) and can measure the electric current provided to the end device. The attacker can process the signal from the energy traces, decode the binary transmission, and reconstruct the original information.

Since this attack does not use the data transfer feature of the USB cable, it also works when software (i.e., charge-only mode) or hardware (e.g., the so-called USB condoms) countermeasures are in place. Due to this covert channel's hostile nature, the data transfer rate is quite limited (i.e., between 1 and 2*bits* per second), but it has a low BER. However, the authors improved the original attack by providing a better transfer rate and a more compact and cheaper solution for measuring the electric current, which could be embedded in a power-bank or a common USB charger [485].

Crosstalk Leakages. After testing, Su et al. [479] found that most computers and external USB hubs suffer from crosstalk leakage effects. This side-channel allows malicious peripherals located outside the communication path to capture and observe sensitive USB traffic. The authors also assess that crosstalk leakage effects could be observed on the USB power cable. Therefore, physically cutting the USB data cable and charging-only USB cable are ineffective as countermeasures against this attack.

Interaction Inference. More recently, Cronin et al. [502] found that while a smartphone is charging, the power trace via the USB charging cable leaks information about its screen content. Based on this discovery, the authors propose Charger-Surfing, a PSC-based method. It uses the power leakage from the smartphone to infer the animation's location playing on the smartphone's touch screen and thus obtain sensitive information about the user. A more prominent feature of Charger-Surfing is that its effectiveness is victim-independent. An attacker can use data acquired from any end device's touch-screen as training data for the neural network. This means that the attack can be carried out without knowing the specific model of the victim's end device, which adds to the danger of Charger-Surfing.

Web Pages Identification. One of the strongest side-channel analysis points is that it reveals important information or patterns, even if the communication is encrypted. This concept was used by Yang et al. [500] to attack data privacy during the charging process. The authors demonstrated that an attacker could steal user information while a user is using a mobile charging station to charge a smartphone. The attack works even without using the data transfer function of the USB cable. Based on this finding, the authors proposed a side-channel attack that leverages USB power consumption correlation, analyzing the relevant variables that may cause power consumption changes. The attack leverages a malicious charging station and can identify the

web pages loaded by the user while charging with the malicious station.

Browsing web pages using Tor is considered secure, but it is not immune to PSC attacks on USB cables. In an extended version of the attack [501], Yang et al. can identify the websites the user visited through Tor, even hidden services, with a higher accuracy rate than ordinary websites. This study was performed by relying on the measurement of the electric current supplied when charging the smartphone.

Juice Filming Charging Attacks

An interesting class of attacks that leverages the threat model considered in this survey is the JFC attack [492]. These attacks rely on MHL technology, which activates automatically when an Android device or iPhone is connected to a projector via the USB cable. In other words, the JFC attack exploits the MHL standard to turn the USB cable into a video cable (VGA or HDMI). As a result, the end device's screen and user inputs can be obtained by the host device (e.g., modified malicious charger) via the USB data connection. Despite not relying on a side-channel, the JFC attack is included in this survey since it is transparent to the user, unaware of the active MHL connection between the end device and host device. As reported by Meng et al. in [492], when an end device is connected to a malicious charger that enables MHL, the latter one can record the user's input (i.e., taps on a touch-screen) and screen activity. To make this attack even more dangerous, JFC cannot be detected by anti-malware software, and it works on both iOS and Android end devices.

An extension of the JFC attack proposes implementing a prototype system called JuiceCaster [454]. JuiceCaster automatically segments the video feed from MHL into images, and it applies the Optical Character Recognition (OCR) technology to extract private textual information. JuiceCaster can automatically start when a new end device is detected, record the video in MP4 format, and store it in a folder. To further enhance the efficacy of JFC attacks, Meng et al. also extend JuiceCaster system in [496] to automatically identify meaningful private information via machine learning techniques from a long-lasting video feed.

The effectiveness of JFC attacks is also assessed in three real-world environments [455]: a company with more than two hundred employees, a university, and a business hall. Deploying modified chargers under these settings, the authors collected personal user information, including user keys, email accounts, and social network chat logs. These results further confirm that the JFC attack seriously threatens user privacy.

4.1.5 Countermeasures

Through the years, researchers also investigated methods to counter the leakage of private information via side-channels. In most cases, the authors of an attack also propose possible software or hardware countermeasures against such an attack. In addition to that, side-channel analyses can also be relied on to detect ordinary attacks on USB. Such attacks generate a resource usage overhead (e.g., CPU usage, data transfer) on host or end devices, which produces side-channels. In this subsection, we report countermeasures against side-channels attacks or countermeasures that rely on side-channels for detecting ongoing attacks via USB interfaces. Table 4.2 summarizes the countermeasures in the state-of-the-art. In what follows, we categorize the countermeasures according to the type of side-channel involved.

Countering EM Emissions

Vuagnoux et al. [494] propose different countermeasures to cope with keystrokes inference attacks, which can also be valid against other attacks relying on EM emissions side-channels. As a first solution, the authors suggest properly shielding keyboards to reduce electromagnetic radiation without affecting their normal function significantly. However, this solution dramatically increases production costs since all internal components of a keyboard and cable have to be shielded and the host device. Therefore, a second solution is to secure the surrounding area around a vulnerable keyboard. This can be achieved by shielding the room (which may still be costly) or establishing a physically secure perimeter (e.g., one hundred meters). Both these solutions would stop attackers from observing EM emissions. Thus both these solutions can also counter other EM emanations-based attacks. A third solution is to input high-frequency filtered matrix signals [504] into the keyboard, which limits the electromagnetic radiation to a certain extent. However, with this solution in place, electromagnetic emanations are noisy but still present, and they can further be analyzed with the aid of machine learning. As a fourth solution, using two-ways encryption for communication is a possible software-based countermeasure to protect user data from EM emissions side-channel attacks. The attacker will not retrieve the encrypted communication content even if EM emissions are still measurable. In the specific case in [494], keyboards' communication chips should be modified to implement encryption. However, encryption alone may not be enough: 1) weak encryption schemes can be broken; 2) an attacker can still leverage other side-channels information (e.g., keystrokes timing) to profile the user behavior. It is worth noticing that users can use only the second countermeasure, despite not always being possible, which can be implemented by users while

Year	Work	Attack	Countermeasure	Principle	Disadvantages	Advantages
2009	Vuagnoux [494]	EM emission on keyboards	Shield cables/keyboard	Reduce EM emanations	High cost	Effective
			Create a safe space	Designation of signal shielded areas	Valid only within a certain range	Low cost,effective
			Encrypted USB communication	Modified keyboard chip	Encryption may be cracked	Improves communication security
			High frequency filtered matrix signal	Limiting EM emission	Does not eliminate EM emission	Improves security
2015	Meng [492]	JFC attack	USB Condom	Inhibit USB data transfer	Disrupting USB data transfer	Effective
			System Alerts	Reminder to users	Burden on users	None
			Special unlocking method	No password input on screen	May hinder usability	Input undisclosed
			Own the charger	Avoid unfamiliar chargers	Not always practical	Effective
2017	Su [479]	Crosstalk leakage	Session key encryption	Increase confidentiality level	May be cracked	No additional Hw
			Dedicated 5V power supply	Decouple data and power USB wires	High cost, special hardware	Effective
2017*	Jiang [505], Meng [455, 506]	JFC attack	Detecting CPU utilization	JFC attack increases CPU usage	Hinders user experience	Highly targeted
2019	Farhi [507]	Malboard	side-channel detection module	Detection via three side-channels	Additional hardware required	Effective
2020	Barankova [508]	Keyloggers	USB Keylogger detection	Measure keyboard energy consumption	Additional hardware required	Effective
2020	Ibrahim [509]	Malicious end devices	Detection of deceitful end devices	EM emissions fingerprinting	Additional hardware required	High accuracy,Zero false positives
2020	Matovu [510]	Activity inference	Randomize current draw	Change load resistors	Additional hardware required	Effective
				Random usage patterns via service	May affect user experience	Effective
2021	Cronin [502]	Charger-Surfing	Randomize keyboard positions	Adding burden to deciphering	Hinder usability	Position not predictable
			Delete input animation	user input cannot be captured	Feedbackmissing	Effective
			Increased noise	Input difficult to capture	Noise can be filtered	Low cost
			Eliminate leakage channels	Low-pass filter into charging circuit	Additional hardware required	Effective

Table 4.2: Proposed countermeasures against Side-channel attacks on USB powered devices by year (* year of the first original proposal).

the other ones have to be implemented by the manufacturers (i.e., they need special hardware modules). EM emission side-channel is not only used to carry out attacks but also to detect malicious end devices disguised as USB flash drives (e.g., Rubber Ducky, Keylogger). Ibrahim et al. in [509] assess that end devices generate unique EM emissions during boot operation when connected to a host device via USB. The authors leverage this observation to propose *MAGNETO*, a framework that fingerprints the EM emissions from USB devices (i.e., USB flash drives). When an end device is connected via USB, *MAGNETO* can measure the EM emissions of such device and identify its manufacturer and model with high accuracy and zero false positives. By continuously monitoring EM emissions, the authors can also detect potentially malicious activity initiated by USB devices even after the boot process.

Electric Side-Channels

Several works propose methods to protect devices from side-channel attacks that exploit power consumption. To cope with Charger-Surfing and other similar side-channel attacks, Cronin et al. in [502] propose both software- and hardware-based solutions that take into account the specific characteristics of such attacks. The PSC produced by button animations can be addressed with the first proposed software-based countermeasure: randomizing the key positions on touchscreens. This kind of countermeasure has already been widely adopted in many financial savings applications. However, applying this method to lock screen password/PIN negatively affects the usability of end devices. Users would have to assess the new key positions every time they are randomized, which is inconvenient in time-critical and repetitive operations, such as screen unlocking and small payment authorizations. A second possible software-based countermeasure is to remove the button input animations. However, this solution would not provide feedback to the user on whether the button was correctly pressed. A third software-based countermeasure consists of adding noise on the screen, for example, by using an animated background. Yet, this solution has two downsides: 1) animated backgrounds can only be used on the lock or home screens, so they cannot secure the use of on-screen keyboards (e.g., messaging); 2) if the animation is predictable (e.g., in a loop), the generated noise can be filtered out with enough samples. As a hardware-based countermeasure, the authors propose to mitigate the power leakage by inserting a low-pass filter into the charging circuit. This solution filters out the frequencies used to carry out Charger-Surfing attacks.

Matovu et al. in [510] proposed one software and one hardware solution to counter PSC attacks for smartphones. The hardware countermeasure consists of a scrambler unit to be de-

ployed on the host side. Such a countermeasure relies on a microcontroller randomly switching several load resistors to variate the current draw, thus masking the signal properties. Similarly, the software countermeasure perturbs the current draw by the user activities via a background service that generates random patterns of CPU and memory resources usage. The authors prove that both solutions effectively counter user activity inference attacks.

Su et al. [479] propose one software-based and one hardware-based countermeasure against crosstalk leakage attacks. On the software side, the authors argue that session encryption (e.g., the Diffie-Hellman key exchange protocol) can effectively mitigate such attacks. However, an excessively lasting session may give enough time to attackers to break the encryption. On the hardware side, a solution that eliminates crosstalk leakage on USB hubs consists of two components: 1) optically decouple the USB data lines; and 2) deploy a dedicated 5V power supply for each downstream port. An alternative effective but less expensive solution is to decouple the USB power and data lines using an LC low-pass filter in a low-dropout (LDO) voltage regulator.

Farhi et al. [507] propose a Malboard: a malicious hardware USB device. Such a device can automatically inject keystrokes (e.g., malicious commands) with a target user's behavioral characteristics into a host device. By mimicking the user's behavior, Malboard can elude detection mechanisms that continuously verify the user's identity via her keystrokes. The authors also propose a method to detect Malboard, which leverages three side-channels from the keyboard device: power consumption, sound, and the user's reaction to input errors. This method achieves perfect detection accuracy for all three side-channels.

Barankova et al. in [508] develop a method to identify the presence of keyloggers hidden inside keyboards, keypads, or cables. The authors propose an effective countermeasure based on the electric voltage-current side-channels on USB interfaces to identify these malicious devices.

Countermeasures against Juice Filming Charging Attack

Together with the first proposal of the JFC attack in [492], Meng et al. also suggest some methods to defend against such an attack. On the operating system side, the user should be prompted with a notification regarding possible privacy threats from utilizing unfamiliar charging devices. On the USB interface side, users can rely on secure hardware to prevent USB data transfer (e.g., USB Condom) to deactivate the MHL connection. However, this would undermine other USB functionalities. To prevent secret information from being disclosed via MHL (e.g., PIN code, unlocking pattern), the authors suggest relying on unlocking methods that are

not recorded on the screen, such as fingerprint, iris recognition, and face recognition. On the user side, only using personal chargers is a simple and effective solution but also not always convenient. However, most of these methods either transfer the burden of security on the user or disrupt the USB standard's functionality.

To cope with these inconvenient, further research on JFC attack [505, 511] assess that such attack increases the CPU and Graphics Processing Unit (GPU) usage on end devices, thus their energy consumption. In [505], Jiang et al. first investigated the energy consumption caused by JFC attacks by surveying more than 500 participants. However, the increased CPU usage by JFC attack has minimal effect on the user experience, being imperceptible by the average user. Based on these findings, Meng et al. in [511] further studied the impact of JFC attacks on both CPUs and GPUs usage to develop JFCGuard. This security mechanism monitors and analyzes CPU and GPU usage during the end device's charging.

JFCGuard is further improved to detect JFC attacks by relying on CPU usage only (not on GPU usage) [506]. First, the authors identify unique patterns from users while charging their end devices by investigating the habits of more than a hundred mobile device users from China and Denmark. Then, a machine learning-based detector (i.e., SVM) identifies anomalies by monitoring CPU usage. Detecting an anomaly will trigger an alert and notify the user of a potential JFC attack, suggesting taking some actions (e.g., stopping charging).

4.1.6 Tools for Side-Channel Measurements

The surveyed works rely on different tools to measure side-channels under our threat model. Most of these works provide a detailed description of such tools and experimental settings. To foster new researchers approaching side-channel analysis, we list and discuss the various approaches in the state-of-the-art. In Table 4.3, we summarize the hardware and software tools categorized by side-channel.

Electromagnetic Emissions

The research that considers EM emissions relies on several solutions for sensing/probing and data acquisition. Considering keystroke inference attacks, Vuaxgnox et al. in [494] use a Universal Software Radio Peripheral (USRP, which ranges from Direct Current to 2.9GHz, 64M Samples per second, 12 bits resolution) and GNU Radio for wide-band spectrum analysis and process modulations. To capture a wide-band signal, the authors rely on Analog-to-Digital Converters (ADC) that acquire EM emanations between 25MHz and 300MHz via an

Side-channel	Work	Use	Tool	Function & Model	Type	Notes
EM emissions	Vuagnoux [494]	Attack	Data acq.	USRP SDR	Hw	Range DC - 2.9GHz, 64MSps, 12bits
			Analysis	GNU Radio	Sw	Signal processing and modulation
			Probe	Antenna	Hw	Obtain the raw signal
			Data acq.	Tektronix TDS5104 O-scope	Hw	Assess the triggering signal, 5GSps
	Sim [499]	Attack	Probe	Near-field	Hw	Measure EM emanation from USB cable of keyboard
			Data acq.	O-scope	Hw	Convert analog signal to digital data
	Genkin [498]	Attack	Probe	Coil	Hw	Collect EM emissions
			Data acq.	Creative sound card	Hw	Amplify and analyze signals
	Belgarric [497]	Attack	Probe	Magnetic probe	Hw	Measure the magnetic field of end device
			Probe	Electrical probe	Hw	Measure trigger from on D+ wire of USB cable
			Data acq.	O-scope	Hw	Convert analog signal to digital data
	Ibrahim [509]	Counterm.	Probe	Aaronia PBS2 EMC	Hw	25mm magnetic field probe PBS-H3
			Data acq.	HackRF One SDR	Hw	Convert analog signal to digital, Low-cost
			Data acq.	Rohde&Schwarz FSW8 Spectrum Analyzer	Hw	Convert analog signal to digital, High-cost
Electric voltage/ current	Spolaor [458, 485]	Attack	Data acq.	Monsoon powermeter	Hw	Measuring electric current on USB cable
			Data acq.	PowerSnitch	Sw	Encodes information into CPU bursts on the end device
			Sensor	ADS712 low-current	Hw	Measure on GND wire of USB cable, Hal sensor
			Data acq.	Arduino Nano board	Hw	ATmega328 micro controller
	Yang [500, 501]	Attack	Power	Rigol DP832	Hw	Supply power to the end device
			Bypass	200Ω resistor	Hw	Connect D+ and D- wires on USB cable
			Probe	0.1Ω resistor	Hw	Shunt resistor on GND wire of USB cable
			Data acq.	NI USB-6211 DAQ	Hw	Measure voltage on the shunt resistor
	Barankova [508]	Counterm.	Sensor	INA219 current shunt	Hw	Measure on 5V wire on USB cable, I2C bus
			Data acq.	Raspberry Pi 3	Hw	Get data via GPIO, Automated inspection
	Cronin [502]	Attack	Probe	0.3Ω resistor	Hw	Shunt resistor on GND wire of USB cable
			Sensor	AD7813 ADC	Hw	Analogic to ditital converter, 62.5KSps, 10bit
			Data acq.	Espressif ESP32	Hw	Tensilica Extensa LX6, WiFi & BT
	Genkin [495]	Attack	Amplifier	Bruel&Kjaer 5935	Hw	Provide 40db gain to the signal
			Data acq.	NI MyDAQ	Hw	Measure voltage at USB power wires, 16bits, 200KSps
	Su [479]	Attack	Data acq.	Agilent MSO6104A O-scope	Hw	Convert analog signal to digital data, 4GSps
			Probe	Agilent 10073C	Hw	Passive, measure voltage, up to 500MHz
		Counterm.	Circuit	LC filter & LDO	Hw	Low-pass filter, decouple USB power and data wires
	Genkin [498]	Attack	Probe	0.33Ω resistor	Hw	Shunt resistor on GND wire of USB cable
			Data acq.	Creative sound card	Hw	Measure the voltage from shunt resistor
MHL standard	Meng [492, 454, 496, 455]	Attack	Data acq.	USB to VGA/HDMI	Hw	Embedded into a USB charger
Processor usage	Jiang [505], Meng [455, 506]	Counterm.	Data acq.	App to monitor CPU/GPU usage	Sw	Detect ongoing JFC attacks

Table 4.3: List of Hardware (Hw) and Software (Sw) tools used for side-channel measurements.

antenna (i.e., bi-conical antenna, 50Ω VHA 9103 Dipol Balun) and an oscilloscope (i.e., Tektronix TDS5104). Continuously capturing a wide-band signal is not feasible due to the limited amount of memory of the oscilloscope. To cope with this shortcoming, the authors use an ADC to identify specific triggers (i.e., peaks) and collect only the interesting portions of the signal. In their experimental settings, Sim et al. in [499] use a near-field probe and an oscilloscope and measure and acquire the EM emitted from the USB cable of keyboards.

Attacks that aim to infer cryptographic keys also rely on similar experimental settings: a probe to sense EM emission and a data acquisition device. In particular, Belgarric et al. in [497] also use an oscilloscope for data acquisition from a magnetic probe placed near the end device's processor. To detect triggers from ongoing cryptographic processes, the authors rely on an electrical probe connected to the D+ wire of the USB cable. In the attack in [498], Genkin et al. propose two threat models: one that senses EM emissions via a coil and one that measures the electric current via a shunt resistor on the GND wire of USB cable. In both scenarios, the authors propose an inexpensive Creative sound card as a device for signal amplification and data acquisition.

To identify fake USB flash drives, Ibrahim et al. in [509] profile end devices by measuring their EM emissions when connected to a host device's USB port. In their experiments, the authors measure such emissions with a magnetic probe (i.e., Aaronia PBS2 EMC PBS-H3). For the data acquisition from this probe, the authors propose a low-cost HackRF One SDR or a more reliable but expensive Rohde&Schwarz FSW8 signal and spectrum analyzer.

Electric Voltage/Current

Considering the side-channels related to electric properties, we can distinguish between two typical approaches:

- Current measurement on a circuit closed by a shunt resistor (i.e., voltage drop) or Hall-effect sensor (i.e., on the 5v or GND wires of a USB cable).
- Voltage variance measurement from an open circuit (e.g., from 5v and GND of a USB port);

Yang et al. in [500, 501] and Cronin et al. in [502] both measure the electric current flowing through a shunt resistor (i.e., of 0.1Ω and 0.33Ω, respectively) on the GND wire of a USB cable. While the former method uses a high-precision data acquisition tool (i.e., National Instruments USB-6211 DAQ) to measure the voltage drop on the shunt, the latter relies on a

high-speed analog to digital converter (i.e., AD7813) connected to a dual-core microcontroller board (one core for measurements and one core for wireless data transmission). It is worth noticing that, to allow charging currents above $500mA$, the authors in [500, 501] connect the D+ and D- wires of the USB cable with a 200Ω resistor. In their second threat model, Genkin et al. in [498] also consider a shunt resistor. Still, they acquire the data via a Creative Tracker Pre sound card, which amplifies the voltage drop signal (60dB gain) and converts it to digital data (at 192K samples per second).

In the preliminary version of their covert channel attack, Spolaor et al. in [458] develop the PowerSnitch app for Android devices to encode the target information in CPU bursts. The authors measure the electric current overhead produced by such bursts with a Monsoon power monitor[1]. Since this device is very precise but quite expensive (around 800 USD), in the improved version of the attack in [485], the authors propose a low-cost solution (about 20 USD) composed of an ADS712 low-current Hal-effect-based sensor and an Arduino nano board. Since such a solution is also designed to be compact, it can fit into power banks and USB chargers.

The countermeasure against keyloggers proposed by Barankova et al. in [508] measures the electric current with a shunt resistor on the USB 5v wire used by a keyboard. To measure such current, the authors rely on an INA219 current sensor that provides the data via an I2C communication protocol to a Raspberry Pi 3.

To assess the feasibility of a crosstalk leakage attack in [479], the voltage variance is measured via a passive probe (i.e., Agilent 10073C) on a USB port from the same USB hub of the target one. Subsequently, the authors use an oscilloscope (i.e., Agilent MSO610A with 4G samples per second) to acquire the measurements from such a probe. As a countermeasure to this attack, the authors implement an improved USB condom that embeds an LC low-pass filter (with a ferrite bead inductance of 27 NH) and a low-drop voltage regulator (i.e., LDFM50).

To correctly measure the voltage variance for extracting cryptographic keys, Genkin et al. in [495] attach a probe at the power wires at the far-end of a cable (e.g., USB, VGA, and Ethernet). Such voltage variance is first amplified with a high-input-impedance low-noise amplifier (i.e., a customized Bruel&Kjaer 5935 with 40dB gain), high-pass filtered at 10 kHz, and then measured by a voltmeter (i.e., National Instruments MyDAQ, 16 bits resolution, and 200 KSps).

[1] High-voltage Monsoon power monitor - https://www.msoon.com/high-voltage-power-monitor

Enabling and Detecting JFC attacks

Juice Filming Charging attack needs to enable MHL via micro USB by modifying a USB charger. In particular, Meng et al. in [492, 454, 496, 455] use an HDMI/VGA to USB converter(e.g. VGA2USB[2]) to acquire the video feed from the victim Android and iOS end devices. Since an ongoing JFC attack generates a not negligible processor usage, an app installed on the end device [505, 455, 506] can monitor the CPU/GPU usage to detect such attack via machine learning techniques.

4.1.7 Takeaway on Power-Side Channel

We can identify some trends and common approaches from the analysis of the work surveyed. In the first instance, we can notice that researchers focused their efforts on electric power and electromagnetic emissions side-channels to carry out their attacks and develop countermeasures. While the former side-channel needs probes to be physically attached or modify a USB port, cable, or charger (i.e., shunt resistor for electric current and probe on two USB wires), the latter can be measured wirelessly with magnetic or near-field probes at a short middle distance from the target device. In both cases, such attacks are passive. Therefore they are challenging to be detected via software countermeasures.

The countermeasures proposed in the state-of-the-art mostly focus on preventing specific traditional and side-channel attacks. Unfortunately, some countermeasures may be costly or inhibit useful functionalities. For example, shielding vulnerable hosts and end devices can be expensive unless done by manufacturers' design. Also, monitoring processes to detect a specific attack (e.g., JFC attack) can be computationally expensive and hinder user experience. As another example, a USB condom prevents exploits and malware attacks by physically cutting off the data communication (i.e., D+ and D- wires) and stops legitimate data transfer and not protecting against PSC attacks.

In general, users can protect their host, and end devices can be protected from these attacks in two ways. The first solution consists of being aware of such threats and adopt security-oriented behaviors. In particular, avoid using untrusted hardware, public charging stations, power banks for rent, or not owned USB chargers. If this cannot be avoided, the second solution is to use hardware-based countermeasures. To mitigate electric PSC, users can use frequency filters (e.g., LC filter and LDO) on the USB power wires or decoupled power sources (e.g., on USB hubs). Avoiding EM emissions can be particularly challenging, and it requires

[2]VGA2USB - https://www.epiphan.com/products/vga2usb/

joined efforts from manufactures and users. On the one hand, manufacturers should produce hardware secure by design and properly shielded to prevent or mitigate EM leakages. On the other hand, users can make sure to use such hardware and software that enables encrypted USB communications. However, these solutions may be costly (additional/customized devices) or hard to implement (specialized communication protocols).

We also noticed that the most used analysis methods are signal processing and machine learning techniques. Researchers apply the signal processing technique when they know the specific protocol or communication they are targeting (e.g., serial communication). In some cases, signal processing relies on additional triggers that determine when interesting information can be observed. On the other hand, machine learning techniques are leveraged when the attack requires discovering unknown patterns or recognizing previously seen patterns (classification) produced by a target entity (e.g., web page, process, keystroke).

We also identify several unexplored aspects of our threat model that may lead to possible future research directions. First, the two main physical side-channels (i.e., electric power and EM emissions) may be leveraged to infer additional information about the users and end devices. For example, a user can be uniquely identified from the side-channel leaked by the set of background processes running on an end device. Second, other physical side-channels (e.g., acoustic, temperature, timing) and their combination with other ones (i.e., electric power and EM emission) are only marginally investigated for USB keyboard devices. Future research could investigate the feasibility of carrying out the attacks or improving USB communications security via such side-channels. Third, to the best of our knowledge, side-channels from IoT devices powered via USB have not been investigated yet.

Another promising research direction is the investigation of emerging charging technologies' side-channels, among which wireless charging technology seems to be the most promising [512]. Manufacturers of mobile devices (e.g., Huawei, Xiaomi, Apple) have made this technology a selling point for their flagship products in recent years [513]. While widely used wireless charging devices adhere to Qi standard [514], Apple invested efforts in AirPower [515], a wireless technology that charges devices of different power. Another similar technology, Xiaomi's spaced charging technology Air Charge [516], has achieved long-range charging of 5W for a single device within a few meters radius. These technologies project power beams to end devices via electromagnetic induction and mmWaves, and they do not involve data transmission. Since wireless charging technologies have an individual power source, a possible side-channel attack could monitor end devices' real-time position from a single location.

4.2 USB Power Fingerprint

The default trust on USB ports on a host device (e.g., workstation, public charging station, power bank) can be exploited by hackers to exfiltrate private information from USB devices [517], such as smartphones, tablets, and flash drives (i.e., host-to-guest attack). An attacker can disguise a malicious USB peripheral [518, 519] (i.e., guest-to-host attack) as a legitimate one to inject harmful commands (BadUSB, Mousejack, Rubber Ducky), deploy malware [3, 474], steal private user information (OMGCable, BadUSB2.0), spy on a user (tiny microphone/camera, cottonmouth, GSM spy bug), or destroy host's hardware (USBKill). Hence, malicious USB peripherals can cause severe harm to a host device if not promptly identified and blocked.

In recent years, several research works have investigated the feasibility of fingerprinting USB devices to protect host devices by relying on Physical Unclonable Functionss (PUFs) [520], or side-channels such as timing [521, 522] and electromagnetic emissions [509]. To the best of our knowledge, no work in the literature considers PSC information to fingerprint USB peripherals.

Threat Model

This study investigates the feasibility of inferring coarse- and fine-grained information about a USB peripheral from its power traces measured at a USB port. Our system aims to protect the host device by identifying unauthorized or malicious USB peripherals, thus detecting attacks such as malware injection or private information exfiltration. In what follows, we describe possible use case scenarios, the attacker capabilities, and the PowerID preparation.

Use Case Scenarios. We conceive the threat model by considering three use case scenarios described in the following, where users and system administrators can rely on PowerID to enhance the security of the USB ecosystem.

1. End-user Personal Protection: Aside from malware-infected storage drives, dangerous threats to the user security and privacy (e.g., data exfiltration, command injection, credential theft) are perpetrated via USB attack tools disguised as flash drives or even concealed within USB cables. To protect from these attacks, users may want to assess the legitimacy of a USB peripheral. Hence, users can deploy PowerID on a USB port to build power trace-based fingerprints for all their legitimate peripherals creating a whitelist of allowed personal devices.

2. Organization Assets Protection: An organization has a strict policy on the peripherals that its members are allowed to use. For instance, an organization does not allow storage drives to

prevent the theft of intellectual property or WiFi adapters that may expose its network to external threats. Hence, an organization's security team may enforce access control on its workstations to avoid potential attacks or human errors in connecting rogue USB peripherals. To this end, an organization can deploy PowerID to allow its members to connect only USB peripherals with pre-approved characteristics (e.g., type, model) or permitted actions (e.g., read-only) while alerting the security team whether anomalous power traces are detected.

3. Infrastructure Access Control Enforcement: In a critical infrastructure scenario (e.g., nuclear plants, refineries, or water distribution systems), stringent access control rules regulate access to field devices (e.g., Programmable Logic Controllers, and control centers). Tampering with an individual field device can lead to catastrophic consequences for both the environment and the population surrounding the infrastructure. For example, Stuxnet [3] was an example of malware delivered to critical infrastructure via an infected USB storage drive. As in the previous use case, PowerID can enforce the list of authorized USB peripherals allowed to connect to field devices. To build this list, the security team trains PowerID on the power trace fingerprinting only to allow a set of specific peripherals and block malicious connection attempts.

Attacker Capabilities. We assume an attacker aims to compromise the host device of a user, an organization, or a critical infrastructure with a malicious USB peripheral. The attacker's objective may include delivering malware, exfiltrating sensitive information, or corrupting the host device. Therefore, the attacker replaces the legitimate USB peripheral with a compromised one (with the same appearance but concealing an attack tool) inducing the user to connect it to the host. Moreover, the attacker may obtain physical access to a user's host device (e.g., lunchtime attack) and attempt to connect its attack tool to the USB port.

PowerID Preparation. We assume that the adversary cannot interfere with power traces collection (e.g., sensor tampering) or compromise the model training phase (e.g., poisoning attack [523]) since such processes are crucial in the PowerID preparation. To obtain optimal fingerprints, we assume that the hardware settings during these processes are the same (or similar) as the ones of the final deployment (i.e., testing phase). We believe this is a reasonable assumption since organizations typically purchase many identical host devices with the same hardware components. PowerID framework relies on an external standalone device to collect the power traces from an electric current sensor between the host and the peripheral under test (more details in subsection 4.2.1). Traces processing and analysis can be done locally on such a device or remotely on a server (i.e., such a device streams the collected traces via networking technologies).

Goals of the Analyses

In our analyses, we assess whether we can infer information about a USB peripheral from the power traces analysis. In particular, the objective of our analyses is to answer the following questions:

① **Type**: Can we recognize the *type* of a USB peripheral during its states *Boot* and *On*?

② **Model**: Can we distinguish the specific *model* of a peripheral during its states *Boot* and *On*?

③ **Device**: Given peripherals of the same model, can we assess the identity of a specific *device*?

④ **Action**: Considering a peripheral in *On* state, can we recognize an ongoing *action* given a device type? E.g., reading from an flash storage drive (Fd), downloading from a WiFi adapter.

⑤ **Device via action**: Given an ongoing action for a type of peripheral, can we identify a specific *device*?

⑥ **Bad vs. Good**: Can we discriminate between malicious USB-based tools and legitimate peripherals?

4.2.1 PowerID System Design

In this subsection, we present our system and describe its components. In Figure 4.5, we provide an overview of PowerID. As a preliminary parameter, the *inference target* [1] defines the specific information about the connected peripheral as the target of the inference. Excluding the power traces acquisition [2], such parameter influences all the other components [3]-[6] and outcome [7] of our system. It is worth noticing that we can use the same power trace as input of several instances of our system with different inference targets. In what follows, we describe in detail the components of PowerID system.

Power Traces Acquisition

This component acquires the power traces of a connected peripheral via a sensor deployed between the port and such a peripheral (step [2]). Such a sensor provides reliable measurements

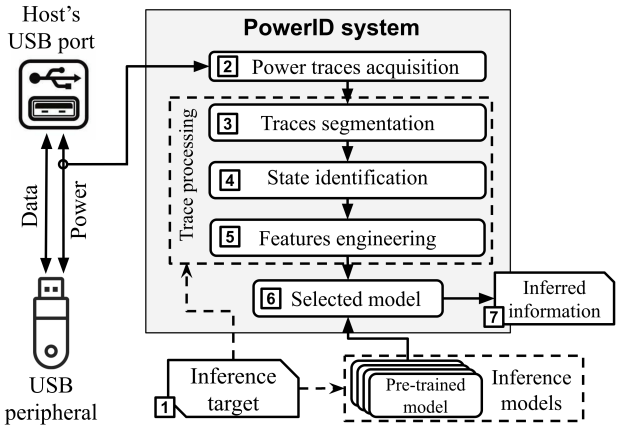


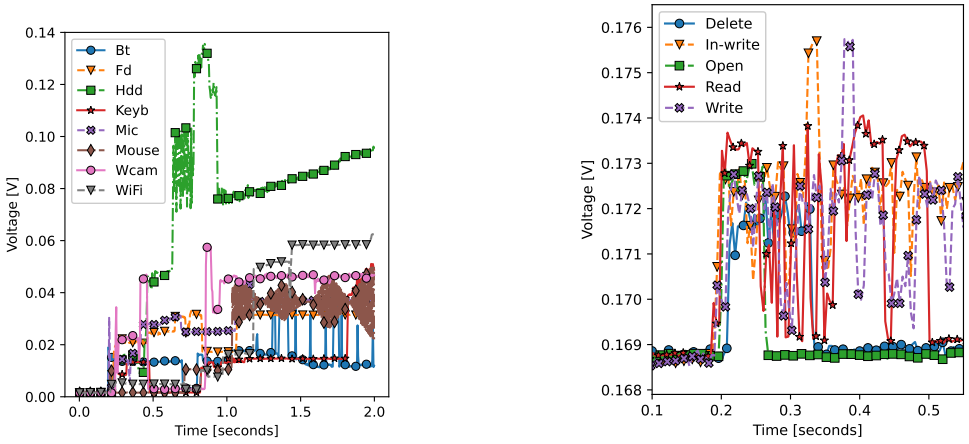
Figure 4.5: System design.

of the electric current supplied by the port. The resolution and sampling rate of a sensor determines the quality of power traces acquired as well as their size. While a low resolution and sampling rate negatively affect the informativeness of a power trace, a high sampling rate requires more computational resources and time for processing. For this reason, it is important to find a trade-off between trace quality and the required processing and resources available. A sensor needs to be calibrated to provide readings within zero and maximum current provided by the considered USB port safely under the full-scale value to avoid saturation. As an additional requirement, the sensor deployment should not affect the performance of the peripheral. With the above requirements in mind, we consider an ADC as a sensor that measures current in terms of voltage drop on a shunt resistor (see subsection 4.2.2). Finally, this component delivers the power traces for further processing. In Figure 4.6 we report some examples of power traces collected from our experiments.

Traces Processing

We process the acquired power traces to obtain viable data for a machine learning-based model. The goal of this process is two-fold: preserving the information within power traces and identifying the current state of the peripheral. To attain this goal, we apply three methods: trace segmentation [3], state identification [4], and feature engineering [5]. It is worth noticing that we apply the same process to obtain the datasets for both the model training and testing. However, while the state of a peripheral is known in the training data, during the testing, we identify the state with step [4].

Traces Segmentation. In this step, we divide the power traces into segments by applying a



(a) State *Boot* for different types of USB peripherals.

(b) State *On* of a Flash drive device.

Figure 4.6: Example of power traces of states *Boot* and *On* for different devices.

sliding window. Such a method considers two parameters: the *window duration* and *overlap ratio*. In selecting values for these parameters, we consider several insights obtained by the observations of the power traces of USB peripherals. Since some activities span for a brief time, a window with a long duration may produce segments that include an excessive amount of data unrelated to a state or action. Hence, such segments may not contain enough meaningful information for a considered inference target. The overlap allows us to obtain more segments for training our models and make them more robust to noise and avoid overfitting. However, a big overlap can produce excessive segments and information redundancy, leading to high computational overhead in the processing phase. From the observation of the obtained traces, we set a duration window of 1 second with a 75% overlap.

State Identification. In our system, we focus on the states *Boot* and *On* of a USB peripheral. The analysis of segments from state *Boot* allows us to early achieve the inference target [1] within a few seconds from the connection of a peripheral to the monitored USB port. PowerID also relies on state *On* segments to infer all the considered target information, especially the ones related to the actions on a peripheral (i.e., ④ and ⑤). Due to low variability, we do not consider the power traces during the state *Sleep*. While identifying the starting time of a state *Boot* is trivial (i.e., change between open to close circuit), the identification of the transition time between states *Sleep* and *On* requires a refined approach since different peripherals produce heterogeneous power consumption in these states. Hence, we need a general approach independent of the considered peripheral (rather than setting a specific threshold for each USB peripheral type and model). To do this, we apply on groups of four consecutive segments a Changing Point

Detection algorithm based on cumulative sum [524]. In particular, we consider a valid changing point if: (1) the next one does not occur within less than 250*ms*, (2) there is at least 25% of value increase. We consider as the start of a state *On* the changing point identified by at least three segments among the four in the same group.

Features Engineering. We consider the segment of a power trace as a univariate time-series, i.e., sequential single data points over a constant time increment. Therefore, we apply the feature extraction method for time series provided by the *tsfresh* libraries [434]. Such libraries allow the extraction of 740 features from each segment, such as statistical features (e.g., mean, standard deviation, variance), linear trend, coefficients of Fast Fourier, and Continuous Wavelet Transform. The feature extraction depends on the inference target 1 and differs between the model training and testing phases. During the training phase, we select the k most significant features that effectively characterize the inference target. As a general approach, we aim to minimize the number k for two main reasons: (1) to not incur the curse of dimensionality and (2) to reduce the time and computational resources required by the feature extraction process. In the testing phase, we only extract the k meaningful features for the inference target and the ones required by regression for state *On* identification.

Selected Model

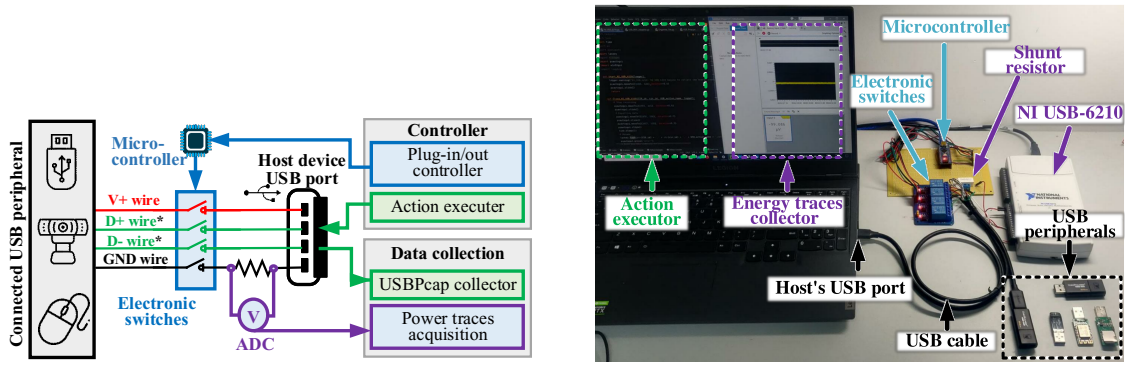
For each inference target 1 considered, we train a classification model on the selected features for such a target. Upon the preliminary analyses, we select the Random Forest classifier as it achieves a higher performance among the other considered learners. While side-channel analyses via deep learning techniques [502, 525, 521, 526] can automate the feature selection process, the training of a deep neural network requires a huge number of examples and high computational and time resources due to the repeated training (i.e., epochs) for weights and parameters optimization. By performing the classification via non-deep learning-based techniques, we can pre-select the important features, thus reducing the complexity of the problem (i.e., the number of features to extract from the time series) and, consequently, the model size in memory. In the testing phase, we load the pre-trained model (step 6) related to the inference target and test the previously unseen power trace segments. As a result of the classification, the model provides in the output the inferred information 7.

4.2.2 Experimental Setup

In this subsection, we describe the implementation details of the power traces collection framework in subsection 4.2.2 and the dataset collection and processing setup in subsection 4.2.2.

Power Traces Collection Framework

We design and implement a framework to collect the power traces for our analyses. In Figure 4.7a, we overview the logical components of our framework[1], while in Figure 4.7b we show our experimental setup.



(a) Logical schema of the framework.

(b) Equipment setup.

Figure 4.7: Framework for power traces collection.

Power Traces Acquisition. In our experimental framework, we measure the electric current supplied by a USB port to a peripheral in terms of the voltage at the extremities of a shunt resistor (i.e., 0.01Ω) in a series of the GND (ground) wire of a USB extender cable. We measure such voltage with the ADC of a *National Instruments USB-6210* Data AcQuisition (DAQ) and acquire the measurements via DAQExpress tool (depicted purple in Figure 4.7a). Despite such ADC can acquire up to $125ksps$ (samples per second), we set the sampling rate to $10ksps$ to achieve a faster feature extraction.

Actions Controller. For our analyses, collected power traces for states *Boot*, *On*, and *Sleep* of the USB peripherals. To automatize this process, we rely on two components: *Plug in/out controller* for *Boot* and *Action executor* for *On* and *Sleep*.

From the hardware perspective, the *Plug-in/Out controller* uses a series of electronic switches driven by a micro-controller (i.e., Arduino Nano). Data and power wires of the USB cable

[1] For the sake of simplicity, we report a USB 2.0 pinout while our experimental setup fully supports the USB 3.1 standard

Type	Action	Description
Flash Drive / Portable Hard Drive	Write	Transfer files from host to guest
	In-Write	Copy files locally guest to guest
	Open	Open a file inside guest device
	Read	Transfer files from guest to host
	Delete	Delete files from guest device
WiFi Adapter	Connect	Connect to a WiFi network
	Download	Download files via WiFi network
	Upload	Upload files via WiFi network
	Disconnect	Disconnect from a WiFi network
Bluetooth	Active	Transfer files via Bluetooth
Microphone	Active	Audio recording from the guest
Webcam	Active	Video acquisition from the guest
Mouse	Active	User activity (click, move, scroll)
Keyboard	Active	A user typing textual contents

Table 4.4: List of considered actions for device types.

are connected to an electronic switch. From a software perspective, we can open or close the switches by sending the related command to the micro-controller. In particular, we implement such a function to replicate the insertion of a USB peripheral into a USB port (i.e., first connecting the power and then the data wires), thus triggering the state *Boot*.

Once the peripheral is connected to the host, the *Action executor* performs a list of actions (i.e., state *On*) according to the type of peripheral currently under test. In Table 4.4, we report the list of considered actions for each peripheral type. For actions based on file transfer (e.g., Write, Download), the action executor involves a file randomly selected from a pool of files with assorted sizes (i.e., from 10MB to 200MB).

To obtain a reliable ground truth for our data, the scripts of the *Action controller* log the timestamp for each plugin/out and action. To assess whether a script is properly triggered and action on the peripheral, we also monitor the USB data traffic via USB sniffer (i.e., USBPcap collector).

Dataset Collection and Analyses Setup

We collected the power traces of the USB peripherals listed in Table 4.5. In total, we collected 8 different device types, 35 different device models and 82 unique devices. In total, we collect more than $6k$ traces for state *Boot* (i.e., around $46k$ segments) and more than $14k$ traces for state *On* (i.e., around $108k$ segments with $AR \geq 0.5$). We obtain the power traces from the USB 3.1 ports of a laptop Lenovo Legion AMD Ryzen 7 5800H 16-core CPU 3.2GHz with 16GB RAM running MS Windows 10 64-bit.

For the power traces processing and model training, we use a Desktop PC AMD Ryzen 9

Table 4.5: List of the USB peripherals involved in our analyses (# indicates the number of individual peripherals)

Type	ID	Brand	Model	USB v.	#
Flash Drive	Fd1	Kingston	DT100 G3	3.2	6
	Fd2	Sandisk	3.2Gen1	3.2	6
	Fd3	Aigo	U310 pro	3.1	6
	Fd4	Aigo	U310	3.1	1
	Fd5	Kingston	DTKN	3.2	1
	Fd6	Kingston	MicroDu03 G2	3.2	1
	Fd7	PNY	TA4-064	3.2	1
	Fd8	Sandisk	Ultra	3.2	1
Portable Hard Drive	Hdd1	WD	My Passport	3.1	2
	Hdd2	WD	Black P10	3.1	1
	Hdd3	Seagate	One Touch	3.2	1
	Hdd4	Toshiba	DTB420	3.0	1
WiFi Adapter	WiFi1	TP-Link	WN726N	2.0	6
	WiFi2	TENDA	U6 N300	2.0	6
	WiFi3	D-Link	DWA-171	2.0	3
	WiFi4	ASUS	USB-AC57	3.1	1
	WiFi5	ASUS	USB-AC68	3.0	1
	WiFi6	Ugreen	AC650 11ac	2.0	1
	WiFi7	Mercury	UD6H	2.0	1
Bluetooth Adapter	Bt1	Lenovo	BT5 LX1815	2.0	4
	Bt2	Ugreen	BT4 US192	2.0	1
	Bt2	TP-Link	TL-UB240	2.0	1
Microphone	Mic1	Ugreen	Desktop Microphone (Mic).	1.1	4
	Mic2	Soaiy	L28	1.1	1
	Mic3	Depusheng	T7	1.1	1
Webcam	Wcam1	Logitech	C270	2.0	2
	Wcam2	Logitech	C920pro	2.0	1
	Wcam3	Philips	P506 HD	2.0	1
Mouse	Mouse1	Dell	MS116c	2.0	6
	Mouse2	Logitech	G102	2.0	2
	Mouse3	Logitech	M546	2.0	1
	Mouse4	Logitech	MX Master	2.0	1
Keyboard	Keyb1	Dell	KB216d	2.0	6
	Keyb2	Logitech	K845	2.0	2
	Keyb3	DURGOD	TAURUS K320	2.0	1

5900X 12-core 3.7Ghz CPU with 64GB RAM running MS Windows 10 64-bit. As libraries, we utilize the *tsfresh* for the time series feature extraction, *Scikit-learn* for the machine learning model and evaluation metrics implementations, and *imblearn* to deal with dataset unbalance.

4.2.3 Experimental Evaluation

We analyze the performance of PowerID to answer the research questions in subsection 4.2. Upon a preliminary comparison across several learners, we select RF classifier as a classification method. In each analysis, we split the dataset into 80% training set and 20% testing set using a stratified approach to maintain the same class proportions. By relying on a validation set (10% of the training set), we study the performance of a model trained by varying the number of features k. In particular, we select the k top features ranked by their ANOVA F-value. Since the analyses have different goals, the best feature set can change considerably. For this reason, we perform a different feature selection for each analysis. To mitigate the possible unbalancing in the training set, we employ SMOTE algorithm [527] to balance the elements in each class.

Analysis	Target	Approach	States: Boot	States: On	Types: Fd	Types: Hdd	Types: WiFi	Types: Other
①	Type	Multiclass	✔	✔				✔
②	Model	Multiclass	✔	✔				✔
③	Device	Binary	✔	✔	✔			
④	Action	Multiclass		✔	✔			
				✔		✔		
				✔			✔	
⑤	Device	Binary		✔	✔			
				✔		✔		
				✔			✔	
⑥	Bad vs. good	Multiclass	✔	✔				✔

Table 4.6: List and description of the analyses.

We tune the hyper-parameters of RF by running a five-fold cross-validated grid-search over different sets of parameter values. We use the above-described pipeline in all the considered analyses unless explicitly mentioned.

In Table 4.6, we summarize the analyses we present in the remainder of this subsection. In particular, we report the states considered for every analysis and the device models employed for the classification. We refer as *multiclass* to a classification task involving more than two classes. Instead, we refer as *binary* to a specific binary classifier with the focus on a single target class at the time (i.e., we apply a One-vs-All strategy). This second approach allows the model to create a decision bound around the target class to discriminate it with respect to all the other classes. In this type of classification, we report the aggregated results as the average of the binary classification of all the considered classes. Interested readers can find more details on the difference between these two approaches in [528].

To attain an open-world scenario in a multiclass approach, we consider an additional class *Other* composed of a random sample of segments (10% of the considered dataset) unrelated to the current analysis but with the same AR. Similarly, we attain an open-world condition for a binary approach by removing the 10% of the non-target classes from the training set but not from the testing set.

We evaluate our classification performance with several standard metrics: Pr, Re, F1, Gm, and Area Under the receiver operating characteristic Curve (AUC). For each analysis, we present the experimental results, highlight the meaningful insights and discuss the limitations.

Types	State = **Boot**, k=100				State = **On**, k=50			
	F1	**Pr**	**Re**	**Gm**	**F1**	**Pr**	**Re**	**Gm**
Fd	0.98	0.99	0.98	0.99	0.99	0.99	0.99	0.99
Hdd	0.97	0.97	0.98	0.99	1.0	0.99	1.0	1.0
WiFi	0.99	1.0	0.98	0.99	1.0	1.0	0.99	1.0
Bt	0.93	0.93	0.93	0.96	0.98	0.97	1.0	1.0
Mic	0.95	0.93	0.96	0.98	0.98	0.97	1.0	1.0
Wcam	0.98	0.98	0.99	0.99	0.99	0.98	1.0	1.0
Mouse	0.95	0.94	0.95	0.97	0.99	0.99	1.0	1.0
Keyb	0.94	0.94	0.94	0.96	0.98	0.97	1.0	1.0
Avg.	0.96	0.96	0.96	0.98	0.99	0.98	1.0	1.0
Std.	0.02	0.02	0.02	0.01	0.01	0.01	0.0	0.0

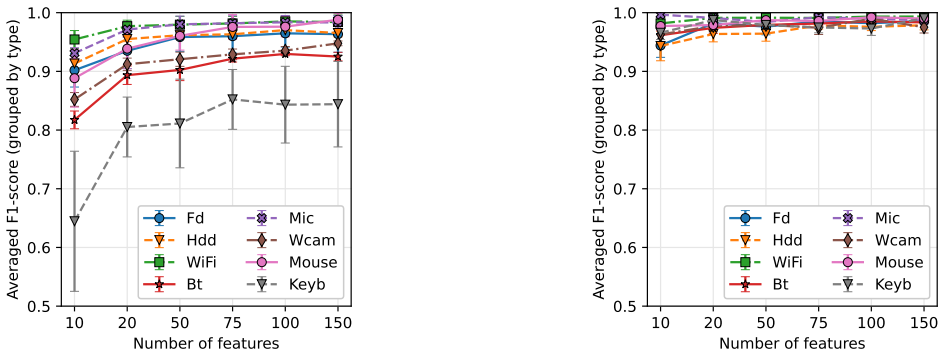
Table 4.7: Results of the analysis ① for device type recognition in terms of F1, Pr, Re, and Gm.

Device Type Classification ①

In this analysis, we aim to classify the device type from the power traces during the states *Boot* and *On*. Considering the state *Boot*, we can assess the peripheral within a few seconds from its connection to a USB port. However, we also classify the type of a peripheral during its activity (i.e., state *On*) to continuously verify that its type would not change (e.g., an Fd turns into a spy camera/microphone after some time).

Method and Dataset. We collect power traces from peripherals and grouped them by device type, obtaining eight in different classes. We also consider an additional class *Other* where in the state *Boot* analysis corresponds to random traces from the state *On*, and in state *On* analysis corresponds to random traces in the state *Sleep*.

Results. We first analyzed the multiclass classification performance on the validation set, varying the number of features selected. By inspecting the Mouse and Keyboard (Keyb) traces on the state *Boot*, we observe that most of them follow a sequence in terms of power draw: an initial peek at the connection (below 0.5 second), flat low, moderate, and stabilize in the state *Sleep*. Hence, we can assume that the model requires more information (features) and segments to classify them correctly. Hence, by selecting the best 100 features for the state *Boot*, we achieve the performance plateau with all the device types. For state *On*, we can achieve high accuracy even with a small number of features as we reach the plateau with 50 features for all considered types. Considering the best feature sets for the two states separately, we report the result of the classification on the testing set in Table 4.7. PowerID models can discriminate with high accuracy between the device types for both the states *Boot* and *On*. Therefore, every device type has a unique fingerprint.



(a) State *Boot*.

(b) State *On*.

Figure 4.8: Performance for the analysis ② varying the number of considered features.

Device Model Classification ②

In this analysis, we delve a further level deep into the device identification by assessing whether the states *Boot* and *On* can discriminate the device model. Therefore, we perform a multiclass classification by considering as classes all the different device models in Table 4.5.

Method and Dataset. We group power traces by the device model, obtaining the 35 different classes. We consider an additional class *Other* that for the state *Boot* analysis corresponds to random traces from the state *On* while for state *On* analysis corresponds to random *Sleep* traces.

Results. We report in Figure 4.8 the F1 on the validation set, varying the feature number. The results indicate the average F1 and the standard deviation of the device models belonging to the same device type. Considering the state *Boot*, we can notice in Figure 4.8a that the performance plateau for every device model is reached at around 75 features. As in the previous analysis, we also note that the Keyb models are hard to classify when using information from a few features. This is again due to the quick handshake between the host and the device during the *Boot* phase. Regarding the state *On* in Figure 4.8b, the classification performance is high. Thus, we can conclude that this state presents a clear fingerprint for all device models.

By selecting the best 75 features for both states *Boot* and *On*, the results on the testing set underline that most of the device models present a unique power fingerprint, also among devices of the same type. However, the device model fingerprints from the state *On* perform better than the ones on state *Boot*. In particular, the devices which perform worst are the *Keyb3* and *Fd8*, From a visual inspection of the traces for such models, we observe that the state *Boot* lasts a short time (below 0.5 second), thus more difficult to fingerprint.

Authentication of Individual Device ③

In this analysis, we aim to discriminate individual devices of the same models. This fine-grained analysis detects whether a device has been tampered with or substituted.

Method and Dataset. We consider power traces of the device models with at least four individual devices, i.e., the device models with $\# \geq 4$ in Table 4.5. For the same model devices, we perform a binary classification using one device as a target class at a time. Given a target device, we remove the segments of one random non-target device from the training set, and we added them to the testing set. We iterate this process for every device of a given model considering the states *Boot* and *On*.

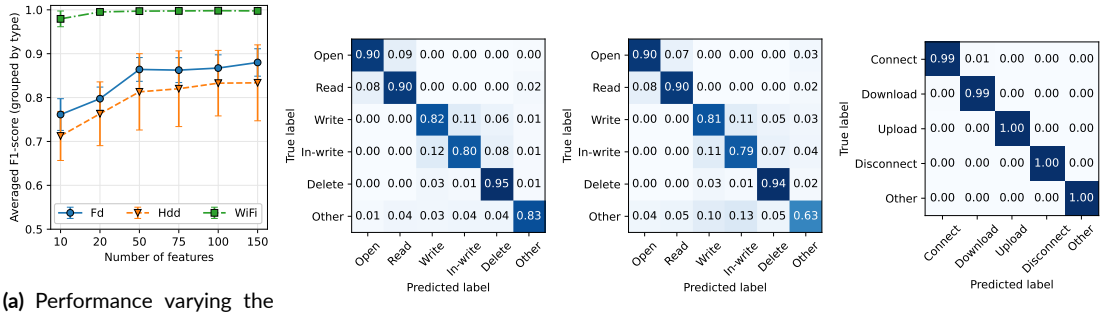
Results. We analyzed the averaged F1 by device model varying the number of considered features. As expected, due to the high specificity of this analysis, we generally achieve lower performance than analyses ① and ②. For both the states, we require at least 100 features to reach an F1 higher than 0.95 for most models. The detailed results of the testing set are reported in Table 4.8. Confirming the results from the previous analysis, PowerID achieves a F1 higher than 0.9 for most models on state *On* but it cannot correctly discriminate a few individual *Mouse1* and *Keyb1* devices for the state *Boot*. As a noticeable exception, the *WiFi1* model has the lowest score on the state *On*. Upon further inspection, we confirm our findings by observing that the traces of the action are very similar among the devices of such a model.

Model	State = **Boot**, k=100					State = **On**, k=100				
	F1-score	Precision	Recall	Gmean	AUC	F1-score	Precision	Recall	Gmean	AUC
Fd1	0.97 ±0.04	0.97 ±0.05	0.97 ±0.03	0.98 ±0.02	1.00 ±0.00	0.98 ±0.01	0.98 ±0.02	0.98 ±0.02	0.99 ±0.01	1.00 ±0.00
Fd2	0.98 ±0.02	0.98 ±0.03	0.98 ±0.04	0.99 ±0.02	1.00 ±0.00	0.96 ±0.02	0.96 ±0.01	0.97 ±0.02	0.97 ±0.01	1.00 ±0.00
Fd3	0.98 ±0.04	1.00 ±0.00	0.96 ±0.08	0.98 ±0.04	1.00 ±0.00	0.91 ±0.07	0.91 ±0.07	0.91 ±0.07	0.95 ±0.04	1.00 ±0.00
WiFi1	0.98 ±0.01	0.99 ±0.01	0.98 ±0.03	0.99 ±0.01	1.00 ±0.00	0.79 ±0.02	0.72 ±0.02	0.88 ±0.05	0.90 ±0.01	0.97 ±0.02
WiFi2	0.99 ±0.01	0.99 ±0.01	0.99 ±0.01	0.99 ±0.00	1.00 ±0.00	0.97 ±0.02	0.97 ±0.03	0.98 ±0.02	0.98 ±0.02	1.00 ±0.00
Bt1	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00
Mic1	1.00 ±0.01	1.00 ±0.00	0.99 ±0.01	1.00 ±0.01	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00
Mouse1	0.82 ±0.12	0.79 ±0.15	0.87 ±0.10	0.90 ±0.07	0.96 ±0.03	0.95 ±0.05	0.95 ±0.06	0.96 ±0.05	0.97 ±0.03	1.00 ±0.01
Keyb1	0.90 ±0.12	0.92 ±0.07	0.89 ±0.17	0.92 ±0.10	0.98 ±0.03	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00
Avg.	0.96 ±0.06	0.96 ±0.06	0.96 ±0.05	0.97 ±0.03	0.99 ±0.01	0.95 ±0.06	0.94 ±0.08	0.96 ±0.04	0.97 ±0.03	1.00 ±0.01

Table 4.8: Results of the analysis ③ for device fingerprinting. For each row, we report the averaged scores (and its standard deviation) across the devices of the same model.

Actions Classification per Device Type ④

In this analysis, we focus on the state *On*, and we investigate whether we can infer type-specific actions across all models. After identifying the device type (analysis ①), PowerID aims to identify unexpected action a device performed (e.g., unauthorized file transfer).



(a) Performance varying the number of features. **(b)** Confusion matrix for Fd. **(c)** Confusion matrix for Hdd. **(d)** Confusion matrix for WiFi.

Figure 4.9: Results for analysis ④ for action identification per device type.

Method and Dataset. We consider three device types of power traces: Fd, external hard drive (Hdd), and WiFi. We focus on these types because they have a broader set of actions to analyze (see Table 4.4). For every device type, the class *Other* is composed of random segments of actions by the other types.

Results. Figure 4.9 reports the results of the analysis. In particular, Figure 4.9a shows that the actions of *WiFi* type have a clear fingerprint while Fd and Hdd require a higher number of features to reach an average F1 higher than 0.8. For a better understanding, in figures 4.9b, 4.9c, and 4.9d we report the confusion matrix with the 75 best features selected for Fd, Hdd, and WiFi, respectively. In figures 4.9b and 4.9c, we can observe that most of the miss-classification of Fd and Hdd are between *Write* and *In-Write* actions. This is probably because *In-Write* is derived by the combination of *Read* and *Write*. Therefore, *In-Write* generates power traces similar to the *Write* actions. Moreover, the Hdd type suffers from the *Other* class mainly due to the similarity with traces of the Fd type.

Individual Device by Actions ⑤

The previous results demonstrate that PowerID identifies with suitable accuracy types, models, the individual device of a specific model, and actions. Building on top of analyses ① and ④, we investigate whether PowerID can discriminate individual devices from the actions they are performing.

Method and Dataset. Similarly to analysis ④, we focus on Fd, Hdd, and WiFi types. In particular, we consider *Read* and *Write* for Fd and Hdd, and *Download* and *Upload* for WiFi. These actions are the most commonly performed for legitimate and malicious purposes (e.g., data exfiltration, malware injection). We select the power traces of the actions from all device

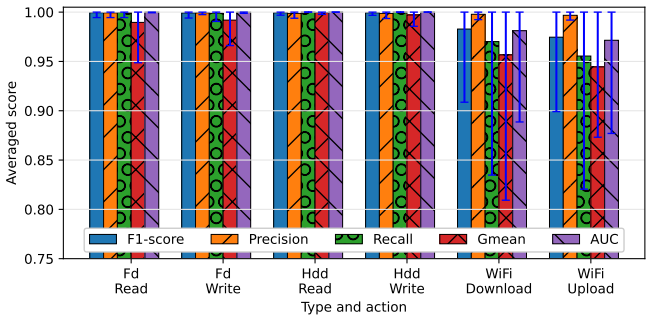


Figure 4.10: Performance of analysis ⑤ for device fingerprinting on different actions.

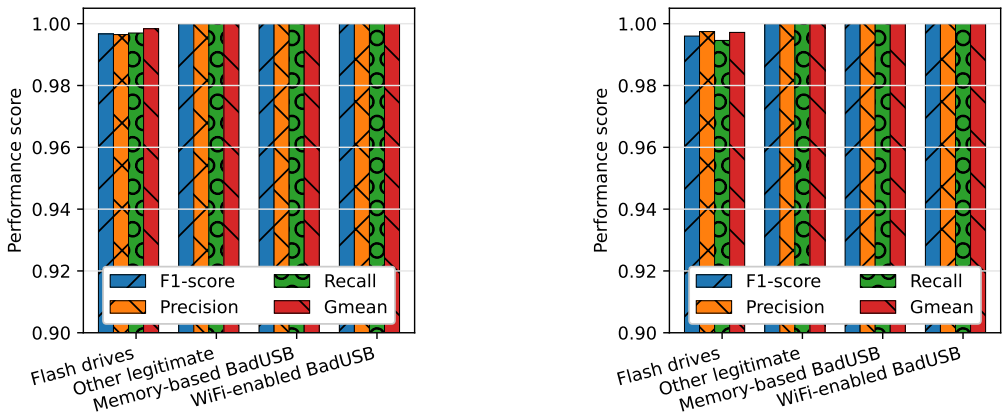
models for each type. Considering a class is the pair (*Device*, *Action*), we obtain 46, 10, and 38 classes for Fd, Hdd, and WiFi, respectively. For each device type, we perform binary classification on such classes.

Results. We report the results in Figure 4.10. We represent with the low and high caps of error bars the best and worst scores among the individual devices, respectively. Overall, PowerID achieves good classification for all the types and actions, despite the high number of classes per type. In particular, we observe that in the case of Fd and Hdd, the different actions are distinguishable from one device to another. However, the WiFi type obtains slightly lower performance and higher variability. From a detailed analysis, we assess that some devices (of model WiFi1, WiFi2, and WiFi6) are miss-classified due to similar behavior in several power trace segments. Despite the variability in the WiFi type, PowerID can correctly discriminates most of the devices. Hence, it is possible to fingerprint an individual device from its actions.

Malicious Device Identification ⑥

To assess whether it is possible to detect malicious devices from their power traces, we perform a multiclass classification between legitimate peripherals and BadUSB devices, focusing on states *Boot* and *On*.

Method and Dataset. In this analysis, we collect power traces from four BadUSB devices (two WiFi-enabled and two memory-based BadUSB devices). While collecting the traces, we run several common attacks, such as local (memory-based) and remote (WiFi-enabled) command injection on the command prompt (varying the types and number of commands) and WiFi scanning and connection. For the state *Boot*, we execute the attack at the device connection, while for state *On*, we delay its execution. The class *Other legitimate* is composed of traces from legitimate peripherals other than Fd type (e.g., Bluetooth (Bt), Mic, WiFi). Moreover, we



(a) State *Boot*.

(b) State *On*.

Figure 4.11: Performance of analysis ⑥ on the discrimination between legitimate and malicious types.

use the features set of the states *Boot* and *On* identified in ①.

Results. From the results in Figure 4.11, we observe that PowerID discriminates the BadUSB devices from Fd and other legitimate peripherals with perfect accuracy.

4.2.4 Takeaway on USB Power-Side Channel Fingerprint

This study presents PowerID, a framework to fingerprint the USB peripherals based on their power consumption during different working conditions. Based on the fingerprints of authorized peripherals, PowerID can protect the host from USB-based threats, identify unauthorized peripherals, and detect illicit actions. We extensively evaluated the performance of PowerID with an exhaustive power traces dataset composed of more than eighty unique devices spanning 35 models and eight types. Results highlight that PowerID achieve a high accuracy in inferring peripheral type, model, activity, and identity.

4.3 Hyperloop

Hyperloop is a radically new concept of transportation system introduced by Elon Musk in 2013 [529], where he described a preliminary version (Figure 4.12) of the Hyperloop project connecting Los Angeles and San Francisco. The vision Musk is to provide fast transportation means by mitigating air resistance and friction. Besides the speed objective, Hyperloop also allows for reduced operating costs and, hence, for energy savings. After the first official document, the project continued through university contests organized by SpaceX and later with the foundation of two startups: Virgin Hyperloop and Hyperloop Transportation Technologies. Today, different companies are working on Hyperloop systems, and Hyperloop testbeds are deployed in different countries like Europe, America, the East, and Middle-East [530, 531, 532].

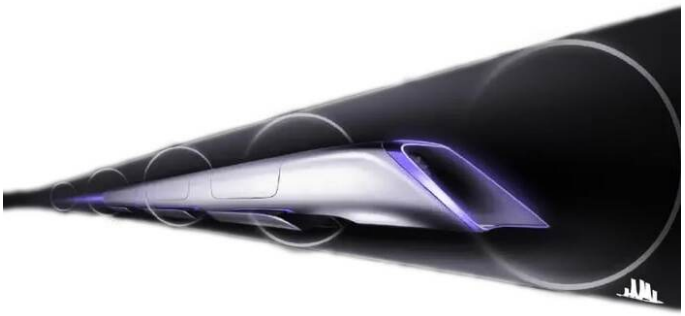


Figure 4.12: First Hyperloop conceptual design rendering [529].

Hyperloop is characterized by two main elements: a tube with vacuum capabilities and a capsule that travels along the tube with passengers or cargo, exploiting air pressure and electromagnetic forces. Hyperloop promises a set of advantages compared to available transportation systems. First of all, as already mentioned, it provides high-speed transportation. Airlines generally fly at 980km/h, and special trains reach a speed of 570km/h, expected for Hyperloop is 1220km/h. Secondly, it allows for reduced operational costs. By placing solar panels properly over the tube carrying the capsule, generated energy can be used to run the system and stored for later use [529].

In recent years, cybercriminals are increasingly targeting CPSs. Examples are countless and interest a wide range of categories, from attacks against nuclear plants [3] to vehicles and transportation systems [8, 533]. Therefore, assessing the system's weaknesses is necessary to prevent dangerous consequences.

This subsection analyzes the different components of the current state-of-the-art Hyperloop

technology and the challenges they pose in cybersecurity. Despite the lack of official documentation, we collected information from publicly available sources and identified common elements with other transportation systems, such as aviation, railways, and automobiles, to create a high-level description of the Hyperloop system. Based on similar domain knowledge, we focus on the different components of Hyperloop and their interconnection, identifying the key security and privacy vulnerabilities they imply. We focus on their interaction via signaling and communication by considering the overall Hyperloop ecosystem as a group of interconnected sub-systems. This provides us with means to analyze the cyber-physical security of Hyperloop at different levels, starting from in-pod threats and going to the complex vehicle to infrastructure interactions. Finally, we propose different security practices to prevent possible attacks. To the best of our knowledge, this is the first work focusing on the cybersecurity challenges related to Hyperloop technology.

4.3.1 Overview of the Hyperloop Infrastructure

In this section, we overview the components of the Hyperloop system, focusing first on the physical infrastructure and then on the network infrastructure.

Physical Infrastructure

We represent in Figure 4.13 the overall physical infrastructure of Hyperloop. We divide the physical system into three macro components: the tube, the station, and the capsule.

Tube. The tube is a pipe with a near-vacuum environment inside, reaching an air pressure equivalent to 60, 960m above sea level, where the Hyperloop capsule transits [531, 534]. Thanks to this environment, the tube significantly reduces the aerodynamic drag to guarantee high speeds while keeping energy consumption at a minimum level. The tube exploits a series of venting valves deployed along its length to rapidly pressurize and depressurize the internal environment. In case of emergencies, the capsule will stop at a predefined emergency station deployed along the route length. The tube is equipped with emergency egresses every 75m to allow passengers to exit the vehicle in case of need [534]. The tube implements valves specifically built to isolate and re-pressurize part of the tube to ease the repairing process. The Hyperloop system aims at producing more energy than it consumes thanks to solar panels placed over the tube [529, 534]. The energy consumption of the Hyperloop mainly depends on the journey's distance and the number of seats, but it is generally lower than aircrafts [535].

Capsule. Hyperloop trains are called pods or capsules. According to to [532, 536], the capsule has the size of a small commercial aircraft with a capacity of 28-50 passengers each, allowing for the transportation of more than 160, 000 passengers daily from Los Angeles to San Francisco (561km) [532]. The capsule's size depends on the purpose (i.e., passenger or cargo), and the tube-to-pod size ratio varies from 1.4 to 4 [536]. Different levitation mechanisms are under investigation, ranging from air beaming employed in the first concept [529], to more efficient electromagnetic or electrodynamic suspension systems [537]. Four propulsion engines flank pods to support any possible levitation implementation high-speed movements. Thanks to the sound bulkhead, the capsule can accelerate without impacting the quality of experience of the passenger. Capsules are designed for ultra-high-speed using various technologies. In particular, pods can be wrapped with materials to monitor and transmit critical information such as temperature, stability, and integrity [532]. Two or more lithium-ion battery packs will power the capsule life support system, making it unaffected by power outages. Battery packs will be charged at the base station before the start of the journey. Otherwise, Musk mentions the possibility of storing compressed air to reverse as energy with fans in second moments [529].

Station. According to [534], the Hyperloop portal will be a central hub that integrates all nearby transportation systems to enable a seamless end-to-end journey. The station will offer digital ticketing, biometric check-in, wayfinding, and on-demand boarding services. The Hyperloop station will include portals from which passengers may depart and arrive. To foster the availability of on-demand transportation and remove the need for timetables, the Hyperloop system will allow multiple pods to convoy or autonomously break away from a convoy to reach their destination. This will be possible thanks to the high-speed switching technology, which combines the benefits of on-demand transportation with higher efficiency and train throughput [534]. Furthermore, this system will allow pods to avoid stopping at every station.

Network Infrastructure

The network architecture to manage Hyperloop is proposed in different works [538, 539]. As depicted in Figure 4.14b, we divide the network infrastructure into four layers.

Hyperloop Network. This Hyperloop-specific layer contains the first communication link with the field network. This includes communications between capsules (originating from the pod itself or from user services), between capsules and the station, or between capsules and the Internet. To support the fast communication between internal antennas, Remote Access Unitss (RAUs), and stations, we can assume the existence of a wired bus (e.g., optical fiber) in-

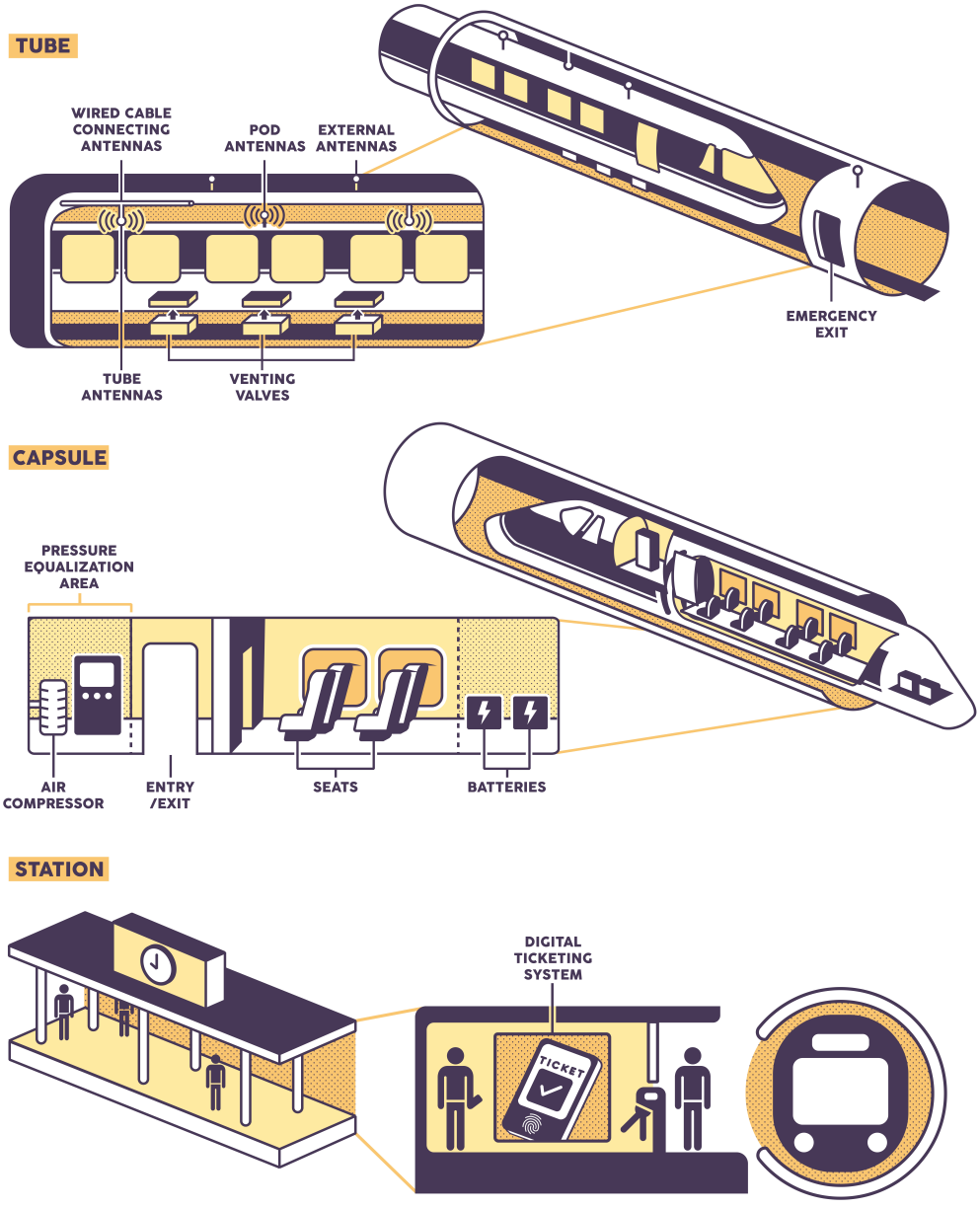


Figure 4.13: An overview of the Hyperloop physical infrastructure and its details.

side the tube, which interconnects these entities. RAUs are placed outside the tube to provide an Internet connection to the communication inside the tube. The connection between the Hyperloop Network and the Access Network can be wireless (e.g., WiFi, 5G) or wired (e.g., optical fibers) to meet the requirement of fast communication. The tube is internally equipped with several antennas used to communicate with the pods. More antennas are installed on capsules to alleviate the timely expensive handover process [538]. The communications originated by the users inside the pods and from the instrumentation on the capsule are captured by the internal antennas and then forwarded to the station (via a wired connection) or the Internet (via RAUs).

Access Network. Connects the Hyperloop Network and the Aggregation Network. The Access Layer is also responsible for gathering the information between the various RAUs placed along the tube and connecting them with one or more ISPs. In fact, since the tube can be hundreds of kilometers long through different countries, the ISPs can differ based on the geographical area. An ISP is an organization that provides the Internet connection to the requestor by assigning it a public IP address. Besides communication coming from the pods, this network allows the ground station of the Hyperloop to connect to the Internet, enabling services such as remote connection.

Aggregation Network. It connects the Access Network to the Core Network by gathering and routing the information between the different ISPs. ADSL, WiFi, Ethernet, and optical fiber are among the technologies used in this layer.

Core Network. The purpose of this layer is to interconnect the different components of the network, enabling access to the various facilities and services, such as databases and clouds, for computations. It must provide low congestion, short delays, high availability, and strong adaptability to support future applications.

4.3.2 Hyperloop Communication Network

In the following, we divide the network into the different communication channels employed for the system's operation. Figure 4.14a represents a schema of the various communications in the Hyperloop network.

We divide communications into Pod-to-Pod (P2P), Pod-to-Tube (P2T), User-to-Pod (U2P), Tube-to-Station (T2S), In-Pod (InP), and User to all the devices available in the stations, User-to-Station (U2S), e.g., ticketing service.

Pod-to-Pod Communication

P2P communication exchanges information in a model similar to vehicle platooning. Pods are expected to automatically merge and detach based on their routes [532]. To avoid collisions, a pod must detect the presence of another pod and its relative location. To this aim, pods exchange messages containing information on their location and speed with other pods. Data transmission between high-speed pods could be supported by ad-hoc protocols similar to Dedicated Short Range Communications used for vehicle-to-vehicle communication.

Pod-to-Tube Communication

The P2T link is used to control the pods. Thanks to this link, the tube can control and manage the magnetic or air forces needed to handle the pod's movements and maneuvers. Viceversa, the pod can apply magnetic or air forces for the movement generated by the onboard engine. It demands great attention to guarantee the safety of infrastructure and passengers.

The P2T link can also be used to exchange messages about the status of the capsule and its location. This information will then be delivered via a T2S link to the central station for managing purposes, such as pressure regulation and motor management. Thanks to RAUs deployed along the tube, the pod can also use the P2T link to access the Internet to guarantee users' connectivity or exchange information with external entities [539].

As discussed in [538], there are only two wireless standards to support communication at the Hyperloop speed: 802.11ax networks and 5G NR. They are therefore likely to be employed in the existing Hyperloop infrastructure between internal antennas and pods.

User-To-Pod Communication

To provide an infotainment system to users during transportation, the pod is equipped with an internal wireless network. Hence, users can retrieve information regarding the trip status, their current location, and expected arrival time. Furthermore, users have access to entertainment material and more generally to the Internet. This connection is guaranteed by the P2T communication link that provides access to the Internet thanks to the RAUs located along the tube. The various access points are connected to the local ISP, which enables the connection to the core network providing Internet access.

Tube-to-Station Communication

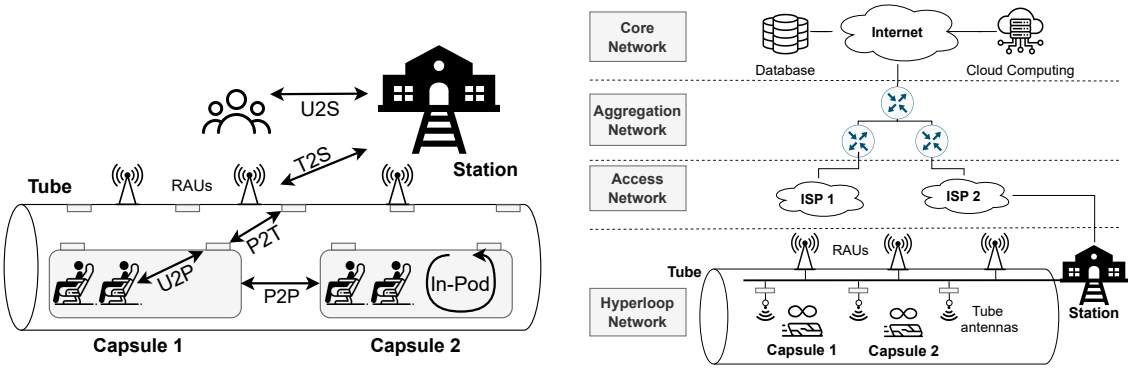
The operations that the tube needs to handle to guarantee that pods can safely travel from one station to another need to be managed by a central entity. These operations include, for instance, operating valves that create the near-vacuum environment and opening emergency exits in case of need. Furthermore, the tube needs to actuate the pods' motion according to the scheduling and report to the station information about the pod's status for managing purposes. These connections can be either wired (e.g., optical fiber) or wireless (e.g., LTE, 5G), depending on the requirements in terms of latency and reliability. Furthermore, a dedicated power line will deliver the energy collected via solar panels to the accumulator deployed at the central station. A second power line will provide energy for operations related to the tube. These operations include the application of power for maglev and pods direction.

In-Pod Communication

Each pod is equipped with an internal network where controllers and actuators regulate the basic pod functioning. This concept is similar to that exploited in modern vehicles, where a Controller Area Network bus connects many ECUs. The pod might use a similar technology to connect different internal areas and manage operations such as air conditioning, door operations, fire alarms, and battery management system [540]. All these units are connected to the pod safety controller that continuously monitors data to guarantee safety. This network is also used to deliver power to the different components of the pod. For instance, suitable controllers handle the flank propulsors that allow for high-speed movements. In-vehicle networks are usually implemented via cable, and different ECUs manage different pod parts.

User-to-(Devices in the) Station Communications

For a complete analysis of the Hyperloop system, we have to consider the communications occurring inside the station between the users and the infrastructure. Many smart devices are available to users. Tickets can be bought from ad-hoc terminals, which have to verify the available seats on the pods in real-time. Turnstiles manage the user's access to the boat area and could use the user's biometrics information to facilitate the access. Displays show delays and departure times for pods and help users find the proper gate. Furthermore, even if not physically located in the stations, we can consider the smartphone apps and web services that enable users to buy tickets remotely in advance to avoid long queues.



(a) Communications types inside the Hyperloop Network Layer.

(b) Different network layers composing the Hyperloop Network Infrastructure.

Figure 4.14: Network composition of Hyperloop.

4.3.3 Cybersecurity Analysis

Based on similar domain knowledge (e.g., ICSs and vehicular systems), we present an analysis of the possible security and privacy issues for each communication category. Table 4.9 reports the different attacks affecting each communication link, the corresponding threats, and their effect on the system.

Pod-to-Pod Security

Several cyberattacks can affect P2P communication. MiTM attacks can modify or craft packets to include false information about the pod and generate a wrong reaction in the neighboring pods. For instance, one pod under attack can claim to be close to the preceding pod and make it speed up. Alternatively, an attacker may launch a Sybil attack by sending data from an inexistent capsule to create inconsistencies between different sensors. Furthermore, an attacker might capture P2P communication packets in a blackhole attack preventing packet forwarding. These attacks, especially if coordinated on a certain number of capsules, can potentially be very destructive and create disorder in the network. MiTM can also be used to send malicious messages to disconnect a pod from a convoy. Another way to force the disconnection of a pod could be a flooding attack leading to a DoS. This could make a capsule unable to receive messages from other pods, inhibiting early reaction to the Hyperloop system's problems. Even if a complete disconnection of the target pod is not possible due to the high capacity of the communication channel, a slight delay in the packet transmission could be dangerous due to the hard real-time constraints on location sharing to avoid incidents.

Supposing pods periodically broadcast their position to alert all nearby pods, spoofing a false location could make all the other pods take wrong actions such as break or speed-up/down even if not needed, leading to inefficiency (e.g., delays) or crashes.

Pod-to-Tube Security

P2T communication has several purposes ranging from critical aims (e.g., managing the levitation of the pod) to secondary scopes (e.g., forwarding the traffic from the infotainment system of the pod). A MiTM attack can tamper with the levitation mechanism to increase or decrease its power leading to the derailing of the pod (i.e., slamming to the edges or the floor of the tube). Hijacking a pod may also be possible. In fact, due to high-speed switching, an attacker may trigger a switch to detach a pod from a convoy and redirect it in the wrong direction.

Location data is transmitted not only on the P2P channel but also with the ground station, through the tube, for central safety management. An attacker can spoof this communication to send messages with false locations of the pods to generate inconsistencies in the system. Furthermore, hiding the presence of a pod near the station via spoofing could make another capsule leave the station before the hidden pod's departure, causing sudden breaks or incidents. Although the pod's location might be helpful for scheduling purposes and estimating arrival times, real-time location sharing might represent a privacy threat. Indeed, a user may target a specific pod and hijack it to a particular location. Although the pod travels at a very high speed, an attacker may predict when a targeting pod will reach a specific point in the tube based on the shared location information. Therefore, it is fundamental to protect the privacy of the pod's location. Furthermore, an attacker might infer information on the users traveling in the pod due to the exchange of data from U2P and P2T. Without traffic encryption or aggregation, an attacker may track and profile users.

The pod must forward the data through the tube to reach the ground station and the ISP. An attacker could disable or delay this exchange of messages by performing a flooding attack from a pod. DoS could lead to the disruption of the infotainment system and the delay of safety-critical data flows.

It is possible to imagine an exchange of energy between the tube and the pod (via, e.g., wireless power transfer) to charge the pod's batteries. This kind of energy flow must be managed by the tube following the requests coming from the pod. An attacker could tamper with this communication to stop the energy request and avoid charging the pod, or it may require more power than needed to damage the batteries. Furthermore, a DoS could prevent the packet

exchange, lead to an unpredictable energy transfer behavior, or limit the number of pods to benefit from it.

User-to-Pod Security

The infotainment system is a central purpose of the U2P network. A malicious user can generate a DoS attack to prevent other users from using the system. Since the pod is shielded inside the tube, the stop of this communication channel may imply the stop of every communication outside the tube. To access the infotainment system, the user is requested to authenticate using credentials linked to the ticket to assign bandwidth and fairly keep track of users' misconduct. A malicious actor can hijack another user to circumvent restrictions or can DoS the authentication system to avoid other users connecting to the access point. It is reasonable to think that users will purchase a ticket that grants them access to the pod. The pod is in charge of checking the ticket validity. A malicious user may use a relay attack to steal another user's ticket or jam the pod receiver to prevent all users from accessing the pod.

Given the number of users connected to the same access point, a malicious entity could try to attack other users, for instance, to eavesdrop on communications and steal sensitive information. MiTM could also be possible for non-encrypted communications, while traffic analysis could be applied to infer users' activities by monitoring encrypted data. With a view to the future, a router can also cache content to serve popular user requests faster [541]. These systems can be vulnerable to cache poisoning attacks, leading to phishing attacks and credentials leakages. A malicious user can also eavesdrop on U2P communication to track and profile a specific user's pod access.

Tube-to-Station Security

T2S communication must be reliable since it has to manage the high traffic of pods and possible issues regarding trips. A DoS attack can mine the system's availability and prevent the exchange of messages in the T2S communication.

Again, even a small delay in data communication can be devastating in systems with hard-real time constraints if proper mitigation techniques are not in place. Similarly, manipulation of the data transmitted can damage the system. Sending malicious messages can affect the tube's functioning, such as forcing depressurization. An evil entity who obtained control of the channel could also insert false anomalies or spoof pods' information, for instance, to make the system stop for useless maintenance. On the contrary, if the system suffers in case of minor malfunc-

tions, an attacker may spoof sensor values to hide anomalies and prevent the proper monitoring of the Hyperloop system. Another possible effect of an injection attack could be the complete shutdown of tube parts' by, e.g., opening a closing depressurization valve, creating an interruption of the service and potential accidents.

Since the tube covers long distances, possibly in different states, it is also important to consider physical attacks which can damage the communications. A malicious actor can tamper with the wired connections to stop communication if cables are exposed to external entities. All the devices, such as RAUs, which are placed along the tube, need constant monitoring because the manumission of the T2S communication could potentially have catastrophic consequences.

In-Pod Security

Even if most of the management and computations occur outside the pod, the capsule has to provide sensor measurements and manage the actuators to execute the station's orders. A DoS in the in-pod network could lead to a consumption of resources and prevent or delay the reception of critical messages. These urgent messages control critical systems such as emergency stops or fire detectors. Hence even a small delay in the packets could lead to accidents. The capsule controls many non-critical devices, such as ventilation and infotainment systems. Tampering with these technologies can create discomfort to the users (e.g., losses of connection, too high-/low air temperature, turning off lights) or compromise user privacy by eavesdropping on the communication between the users and the access point. An attacker gaining MiTM capability in the in-pod network could propagate false messages, creating inconsistencies in the pod's status and hence false alarm messages. Alternatively, the malicious user could lead to dangerous behaviors of the pod's components (e.g., opening the door while the pod is in movement) or tampering with the battery management system to remove the power source.

User-to-(Devices in the) Station Security

A malicious user can exploit different attacks to impact U2Ss security and privacy. An evil user can steal sensitive data thanks to MiTM and eavesdropping. Information on a specific user's habits may lead to profiling or user tracking, representing a privacy violation. An attacker may also spoof the U2Ss communication to create false messages. For instance, an attacker may be able to bill another user for a ticket or modify the information shared with a user on the availability of free seats in a pod, thus preventing purchases. Furthermore, the attacker may

Communication	*Attack*	*Threat*	*Effect*	*Countermeasure*	*Impact*
Pod-to-Pod	MiTM, Spoofing, Relaying, Replaying	Impersonation, Information Gathering, Vulnerability Exploitation, Manipulation, System Corruption, Loss of Service	Sending messages with malicious information to make the preceding or following pod break or speed-up/down to cause collisions; Disconnect a connected pod, or connect a disconnected pod	Authentication, Encryption	High
	DoS, Flooding	System Corruption, Loss of Service	Pod unable to receive messages from other pods; Generated delays make received information useless	Resource management recovery, IDS	Medium
	Location Spoofing	Manipulation, System Corruption	Injecting false information on the location of nearby pods to create collisions	Distance bounding, Digital Signature	High
Pod-to-Tube	MiTM, Relaying, Replaying	Impersonation, Information Gathering, Vulnerability Exploitation, Manipulation, System Corruption, Loss of Service	Sending fake commands to a pod or to the tube; Derail a pod; Hijack a pod	Authentication, Encryption, IDS	High
	DoS, Flooding	System Corruption, Loss of Service	Disable or delay the exchange of messages between tube and pod (both ways); Prevent exchange of energy between the tube and the pod	Resource management recovery, IDS	Medium
	Location Spoofing	Manipulation, System Corruption	Create inconsistencies between pod's real and claimed locations	Distance bounding, Digital Signature	Medium
	Privacy violation	Private Data Leakage	Track the location of a pod, together with identifiers of onboard users	Data Anonymization, Encryption	Low
Tube-to-Station	MiTM, Spoofing, Relaying, Replaying	Information Gathering, Vulnerability Exploitation, Manipulation, System Corruption, Loss of Service	Sending malicious control messages that affect the tube's functioning; Generate anomalies or spoofing the pod information; Shutdown parts of the tube; Hide running anomalies	Authentication, Encryption, IDS	High
	DoS	System Corruption	Prevent exchange of messages from T2S and vice-versa	Firewall, IDS	Medium
	Physical Tampering	System Corruption, Loss of Service	Cut the wired communication; Tamper with the RAUs to interrupt the service	Enforcement with anti-tamper materials, Anomaly detector	Medium
User-to-Pod	MiTM, Eavesdropping, Attacks to authentication system	Private Data Leakage, Loss of Service	Intercept and modify users communications to the external network; Steal sensitive information; Prevent users to access the pod using the purchased ticket	IDS, Encryption, Authentication, Data Anonymization	Low
	DoS, Attacks to authentication system	System Corruption, Loss of Service	Prevent user from exchanging information with the pod (mostly related to infotainment), and so with the Internet	Resource management recovery, Authentication, Firewall	Low
	Privacy violation, Phishing	Private Data Leakage	Steal private information of users in the pod	Encryption, Authentication	Low
In-Pod-Communication	MiTM, Spoofing, Relaying, Replaying	Information Gathering, Vulnerability Exploitation, Manipulation, System Corruption, Loss of Service	Compromise in-pod critical and non-critical systems (e.g., infotainment, ventilation, light, emergency exits, fire detectors and extinguishers)	IDS, Integrity check	Medium
	DoS, Flooding	System Corruption, Loss of Service	Consume resources and prevent or delay the reception of critical messages (e.g., malfunctions, emergency stops)	Firewall, IDS	High
User-to-(Devices in the) Station	MiTM, Eavesdropping	Private Data Leakage	Steal sensitive information from the user	Encryption, Authentication, VPN	Low
	Spoofing, Attacks to authentication system	Manipulation, System Corruption, Loss of Service	Bill another user for a ticket; Prevent users from booking a ticket; Prevent user from reaching the legitimate pod	Encryption, Digital Signature	Low
	Privacy violation	Private Data Leakage	Tracking users pod's accesses; Profiling users	Data Anonymization	Low

Table 4.9: Possible attacks and threats affecting Hyperloop and the consequent effect on the system.

modify the information on the pod for the purchased ticket, causing passengers to miss their pods. Also, an attacker may modify the information that the station shares with the user on scheduling and pod tracking, thus causing traveling issues.

4.3.4 Attacks Countermeasures

The first line of protection of the system should follow the best security practice and security standards from the system's design phase. These standards are applied by industries but also by governments and researchers. Since Hyperloop is a novel technology, no security standards support its design phase. However, Hyperloop inherits different properties from other sectors, such as automotive, railway systems, and aviation, therefore, the designer can rely on the existing standards. For instance, automotive security policies are collected in different standards such as AUTOSAR and ISO/SAE 21434. Instead, industrial security standards are defined, for instance, in IEC 62443 and NCSC secure design principles.

Besides security standards, security mechanisms can be applied to increase the overall security level of the system and prevent attacks. Table 4.9 shows the list of the possible countermeasures related to each attack previously described.

To prevent MiTM attacks, the most common solution is introducing encryption and authentication techniques in communication to avoid data manipulation from unauthorized parties. Instead, DoS attacks are very challenging to prevent. Common strategies include resource managers correctly allocating the memory to the critical processes and redistributing the resources if a DoS attack is detected. To prevent network attacks, a common solution is the implementation of Intrusion Detection Systems to identify malicious behaviors early and make accurate decisions to protect the system. It is also possible to implement a good firewall in strategic points of the network (e.g., the station or the different access points) to increase the perimetral protection of the network.

An important factor for preserving the system's function is ensuring every pod's correct localization. For this reason, an attacker may try to spoof the pod's location to alternate and compromise Hyperloop. Other than the previous measures to avoid spoofing, like encryption and authentication, a common solution to prevent spoofing of communication is the introduction of a distance bounding protocol. This protocol measures the physical distance of an entity by analyzing the expected response time. In this way, the delay added by manipulating data or by a spoofed position will be detected by the receiver. This protocol has been widely used to prevent relay attacks and GPS spoofing.

Due to the system's distributed nature, the physical infrastructure of the Hyperloop also opens potential vulnerability surfaces. The most common solution to prevent physical tampering is integrating anti-tampering material equipped with sensors that identify attempts. Another solution is to employ anomaly detector software to identify the alteration of a process related to a compromised device.

To prevent user privacy leakages, a standard solution is to provide correct anonymization in all communications involving sensible data. Possible solutions include encryption and data aggregation. In the case of devices with limited capacity or time-constrained operations, differential privacy represents a viable solution.

4.3.5 Takeaway on Hyperloop

Hyperloop represents an innovative and promising technology that is still under development. Indeed, currently, there are no official documents related to the system's technical implementation, which does not allow precise analysis of the infrastructure. However, cybersecurity is a continuous analysis process, which must be seriously considered from the design phase. This requirement is emphasized since Hyperloop exhibits a large attack surface comprising network, communications, and physical processes. Compromising the security of a transportation system can dramatically impact the safety of the passengers and the surrounding environment.

This study represents the first work in the direction of the analysis of the cybersecurity of the Hyperloop system. In particular, we identified the main type of communication characterizing the Hyperloop system and investigated possible attacks and threats on the different parts of the infrastructure. Then we discussed the best practices, countermeasures, and standards to prevent potential catastrophic attacks. We reported how traditional attacks could compromise the system's safety. Furthermore, we discussed the privacy issues related to users and the infrastructure. For each of the identified vulnerabilities, we also proposed possible countermeasures.

This analysis should be deepened in future works using more precise technical details of the Hyperloop. In addition, we expect that academia and industries will develop more reliable communication and efficient protocols to support the Hyperloop high-speed communication shortly. Future security researchers must also consider and include such novel technologies.

5 Conclusion

CPSs are nowadays employed in a wide spectrum of services, ranging from industrial operations to transportation systems. Their distributed and complex nature makes them intrinsically difficult to secure against attacks, exposing the safety of the involved users to dangerous incidents. For this reason, in this thesis, we investigated three RQs aimed at analyzing the modern threats affecting these systems, and the possible countermeasures to them.

In the first part of the thesis, in Chapter 2, we focused on ICS security, gathering all the knowledge in this field from the literature and proving a systematic analysis of the testing platform and the IDS solutions operating on them. To motivate the necessity of improving the security of current industrial systems, we performed a measurement study highlighting the dramatic exposure of the communication protocols and services of more than 50 industrial endpoints. Then, we developed an innovative ICS honeypot. While measuring the honeypot exposure, we noted that industrial systems are still highly targeted by malicious actors over the internet on specific vulnerable industrial services. Therefore, ICSs still represents a profitable target for attackers due to their vulnerabilities and the business they produce. For this reason, future studies must provide design guidelines to secure the design of future systems and develop effective detection and sanitization mechanisms to mitigate possible attacks and subsequent catastrophic consequences.

In the second part of the thesis, we focused on the security of vehicular systems. In particular, we highlighted, in Chapter 3, the intrinsic design vulnerabilities on the IVNs and the emerging EV ecosystem, and we proposed innovative security mechanisms. More precisely, we focused

on the design issues in the CAN protocol, and we identified that the proposed solution relies on strong assumptions on the pre-shared secret. To this end, we proposed a key distribution algorithm based on watermarking and jamming to alleviate such assumptions, which can be integrated into current vehicle systems without hardware modifications. Then, concerning the novel EV environment, we showed the feasibility of a relay attack on the charging process, which allows charging of the recharging fees on a victim, and a privacy attack on the charging process. Like ICS, also vehicles suffer from a weak design in terms of security properties since they rely on legacy components. Therefore, vehicle manufacturers should re-design the car's communication system or build security technologies on top of the existing ones. However, as we have shown, recent technologies like EVs also suffer from common vulnerabilities like relay attacks or privacy leakage exposure.

As final study cases, in Chapter 4, we leveraged the knowledge of our studies on ICS and vehicles domain to analyze cutting-edge state-of-the-art of cross-domain CPS applications. In particular, we first systematically surveyed and analyzed the PSC-based attacks and countermeasures on USB application. Then, we propose a framework to fingerprint common USB peripherals in order to authenticate legitimate devices and prevent the connection of malicious ones and the subsequent delivery of malware, like in the Stuxnet case. Finally, we performed the first security analysis of the emerging Hyperloop technology by leveraging the common points with other CPSs. The analysis highlight that this novel technology imposes new challenges from the security point of view, like the necessity of using specific lightweight protocols to handle the train speed or the exposure of critical components.

To sum up, CPSs still needs contribution in the security field. Indeed, the systematic analysis of the various systems highlights the presence of many gaps from the security perspective and remarks on the requirement for additional studies and secure design standards in this field.

Future Work and Directions

Driven by the finding of this thesis, in future works, we will leverage the gathered experience to discover and shed light on novel threats affecting CPS both from the design and implementation perspectives. This thesis has shown that many vulnerabilities similarly afflict CPSs of different belonging from different domains. Studying how CPSs operate, interact with each other and identify common design patterns is essential for identifying risks and vulnerabilities and preventing attacks with dramatic consequences. At the same time, we will focus on identifying innovative and effective security solutions to prevent or mitigate potential attacks by also

considering the constraints and limitations that affect these systems. The security solutions must consider the requirements that characterize the CPS domain, such as real-time operating constraints, limited computational capacity, and long lifespan. Because in many attacks, the human factor is the entry vector (e.g., phishing, social engineering), it is necessary to keep the human being at the center of solutions by designing inclusive and user-friendly solutions for human operators and end users. In the following, we summarize possible research directions that the community .

- *Secure communication protocols*: Develop robust and secure communication protocols that protect against cyber-attacks such as man-in-the-middle attacks, packet sniffing, and eavesdropping.
- *Resilient and fault-tolerant systems*: Create cyber-physical systems that are resilient and fault-tolerant to mitigate the risk of catastrophic failure due to cyber-attacks.
- *Intrusion detection and prevention*: Develop effective intrusion detection and prevention mechanisms that can detect and prevent cyber-attacks in real-time.
- *Data security and privacy*: Enhance the security and privacy of sensitive data transmitted within cyber-physical systems. This could include developing secure data transmission protocols, encryption, and access control mechanisms.
- *Security testing and verification*: Develop innovative tools and techniques for testing and verifying the security of cyber-physical systems.
- *Collaborative security*: Develop collaborative security mechanisms that can enable cyber-physical systems to work together to detect and mitigate cyber-attacks.
- *Secure supply chain*: Develop secure supply chain mechanisms to ensure that components and software used in cyber-physical systems are free from malicious code or vulnerabilities.

Acknowledgments

I sincerely thank all the amazing people who contributed to this achievement and supported me during these intense but amazing three years.

I thank the Cariparo foundation and Yarix s.r.l, and particularly Nicola Bressan and Roberto Chiodi, for supporting my Ph.D.

I thank my supervisor, Prof. Mauro Conti, for the illuminating supervision and all the opportunities.

I thank my family for always supporting me and bearing with me.

I thank all my colleagues for making this path less heavy.

I thank all my friends for always being present.

References

[1] M. Conti, D. Donadel, and F. Turrin, "A survey on industrial control system testbeds and datasets for security research," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2248–2294, 2021.

[2] A. Humayed, J. Lin, F. Li, and B. Luo, "Cyber-physical systems security—a survey," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1802–1831, Dec. 2017.

[3] Falliere, Nicolas, Murchu, L. O, Chien, and Eric, "W32. stuxnet dossier," *White paper - Symantec Corp. - Security Response*, vol. 5, no. 6, p. 29, 2011.

[4] A. A. Di Pinto, Y. Dragoni, and A. Carcano, "Triton: The first ics cyber attack on safety instrument systems," in *Blackhat USA*, 2018, pp. 1–26.

[5] S. Shrivastava, "Blackenergy-malware for cyber-physical attacks," *iTrust - Analysis Report*, vol. 74, p. 115, 2016.

[6] A. Hobbs, "The colonial pipeline hack: Exposing vulnerabilities in us cybersecurity," in *SAGE Business Cases*. SAGE Publications: SAGE Business Cases Originals, 2021.

[7] Jenni Bergal. (2021) Florida Hack Exposes Danger to Water Systems. [Accessed: 12-12-2022]. [Online]. Available: https://www.pewtrusts.org/en/research-and-analysis/blogs/stateline/2021/03/10/florida-hack-exposes-danger-to-water-systems

[8] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," *Black Hat USA*, vol. 2015, no. S 91, 2015.

[9] K. Wolsing, L. Thiemt, C. v. Sloun, E. Wagner, K. Wehrle, and M. Henze, "Can industrial intrusion detection be simple?" in *European Symposium on Research in Computer Security*. Springer, 2022, pp. 574–594.

[10] G. Barbieri, M. Conti, N. O. Tippenhauer, and F. Turrin, "Assessing the use of insecure ics protocols via ixp network traffic analysis," in *2021 International Conference on Computer Communications and Networks (ICCCN)*, 2021, pp. 1–9.

[11] M. Conti, F. Trolese, and F. Turrin, "Icspot: A high-interaction honeypot for industrial control systems," in *2022 International Symposium on Networks, Computers and Communications (ISNCC)*. IEEE, 2022, pp. 1–4.

[12] C. Alcaraz, "Secure interconnection of it-ot networks in industry 4.0," in *Critical Infrastructure Security and Resilience*. Springer, 2019, pp. 201–217.

[13] K. Stouffer, V. Pillitteri, S. Lightman, M. Abrams, and A. Hahn, "Guide to Industrial Control Systems (ICS) Security NIST Special Publication 800-82 Revision 2," NIST, Tech. Rep., 2015. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf

[14] B. Zhu, A. Joseph, and S. Sastry, "A taxonomy of cyber attacks on scada systems," in *2011 International conference on internet of things and 4th international conference on cyber, physical and social computing*. IEEE, 2011, pp. 380–388.

[15] B. Filkins, D. Wylie, and A. J. Dely, "SANS 2019 State of OT / ICS Cybersecurity Survey," *SANS Institute*, no. June, 2019.

[16] L. Neitzel and B. Huba, "Top ten differences between ICS and IT cybersecurity," *InTech Magazine*, no. June, 2014.

[17] B. Miller and D. Rowe, "A survey scada of and critical infrastructure incidents," in *Proceedings of the 1st Annual conference on Research in information technology*, 2012, pp. 51–56.

[18] Israeli Test on Worm Called Crucial in Iran Nuclear Delay. Accessed: 25-01-2021. [Online]. Available: https://www.nytimes.com/2011/01/16/world/middleeast/16stuxnet.html

[19] Kaspersky Lab, "Threat landscape for industrial automation systems in the second half of 2016," *AO Kaspersky Lab, 1997 - 2017*, pp. 1–12, 2016.

[20] M. Nawrocki, T. C. Schmidt, and M. Wählisch, "Uncovering vulnerable industrial control systems from the internet core," in *NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2020, pp. 1–9.

[21] N. Networks. The Cost of OT Cybersecurity Incidents and How to Reduce Risk. Accessed: 02-02-2021. [Online]. Available: https://www.nozominetworks.com/solutions/challenge/cost-of-ot-cyber-security-incidents/

[22] IBM. IBM Study: Businesses More likely to Pay Ransomware than Consumers. Accessed: 02-02-2021. [Online]. Available: https://www-03.ibm.com/press/us/en/pressrelease/51230.wss

[23] Coverware. Ransomware Costs Double in Q4 as Ryuk, Sodinokibi Proliferate. Accessed: 02-02-2021. [Online]. Available: https://www.coveware.com/blog/2020/1/22/ransomware-costs-double-in-q4-as-ryuk-sodinokibi-proliferate

[24] Dragos. EKANS Ransomware and ICS Operations. Accessed: 02-02-2021. [Online]. Available: https://www.dragos.com/blog/industry-news/ekans-ransomware-and-ics-operations/

[25] H. Holm, M. Karresand, A. Vidström, and E. Westring, "A Survey of Industrial Control System Testbeds," *NordSec 2015: Secure IT Systems*, vol. 9417, pp. 213–230, 2015.

[26] H. Christiansson and E. Luiijf, "Creating a European SCADA security testbed," *IFIP Advances in Information and Communication Technology*, vol. 253, pp. 237–247, 2008.

[27] S. McLaughlin, C. Konstantinou, X. Wang, L. Davi, A.-R. Sadeghi, M. Maniatakos, and R. Karri, "The cybersecurity landscape in industrial control systems," *Proceedings of the IEEE*, vol. 104, no. 5, pp. 1039–1057, 2016.

[28] M. H. Cintuglu, O. A. Mohammed, K. Akkaya, and A. S. Uluagac, "A Survey on Smart Grid Cyber-Physical System Testbeds," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 1, pp. 446–464, 2017.

[29] Y. Geng, Y. Wang, W. Liu, Q. Wei, K. Liu, and H. Wu, "A survey of industrial control system testbeds," *IOP Conference Series: Materials Science and Engineering*, vol. 569, no. 4, 2019.

[30] S. Choi, J. H. Yun, and S. K. Kim, *A comparison of ICS datasets for security research based on attack paths*. Springer International Publishing, 2019, vol. 11260 LNCS. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-05849-4_12

[31] S. Ghosh and S. Sampalli, "A survey of security in scada networks: Current issues and future challenges," *IEEE Access*, vol. 7, pp. 135 812–135 831, 2019.

[32] U. P. D. Ani, J. M. Watson, B. Green, B. Craggs, and J. R. Nurse, "Design considerations for building credible security testbeds: Perspectives from industrial control system use cases," *Journal of Cyber Security Technology*, pp. 1–49, 2020.

[33] B. Green, R. Derbyshire, W. Knowles, J. Boorman, P. Ciholas, D. Prince, and D. Hutchison, "{ICS} testbed tetris: Practical building blocks towards a cyber security resource," in *13th {USENIX} Workshop on Cyber Security Experimentation and Test ({CSET} 20)*, 2020.

[34] D. Bhamare, M. Zolanvari, A. Erbad, R. Jain, K. Khan, and N. Meskin, "Cybersecurity for industrial control systems: A survey," *computers & security*, vol. 89, p. 101677, 2020.

[35] D. Pliatsios, P. Sarigiannidis, T. Lagkas, and A. G. Sarigiannidis, "A survey on scada systems: secure protocols, incidents, threats and tactics," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1942–1976, 2020.

[36] C. Alcaraz and J. Lopez, "Digital twin: A comprehensive survey of security threats," *IEEE Communications Surveys & Tutorials*, 2022.

[37] F. Khorrami, P. Krishnamurthy, and R. Karri, "Cybersecurity for Control Systems: A Process-Aware Perspective," *IEEE Design and Test*, vol. 33, no. 5, pp. 75–83, 2016.

[38] L. Obregon, "InfoSec Reading Room Secure Architecture for Industrial Control Systems," *SANS Institute InfoSec, GIAC (GSEC) Gold Certification*, vol. 1, pp. 1–27, 2014.

[39] Promotic Software. Accessed: 10-02-2021. [Online]. Available: https://www.promotic.eu/en/index.htm

[40] IANA. Service Name and Transport Protocol Port Number Registry. Accessed: 02-02-2021. [Online]. Available: https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

[41] Modbus - IDA, "Modbus Application Protocol Specification," Modbus, Tech. Rep., 2006. [Online]. Available: http://www.modbus-ida.org

[42] Modbus, "MODBUS/TCP Security, Protocol Specification," Modbus Organization, Tech. Rep., 2018. [Online]. Available: https://modbus.org/docs/MB-TCP-Security-v21_2018-07-24.pdf

[43] A. Shahzad, M. Lee, Y.-K. Lee, S. Kim, N. Xiong, J.-Y. Choi, and Y. Cho, "Real time modbus transmissions and cryptography security designs and enhancements of protocol sensitive information," *Symmetry*, vol. 7, no. 3, pp. 1176–1210, 2015.

[44] G. Bernieri, S. Cecconello, M. Conti, and G. Lain, "Tambus: A novel authentication method through covert channels for securing industrial networks," *Computer Networks*, vol. 183, p. 107583, 2020.

[45] GE Harris, "Overview of DNP3 Security Version 6," DNP3, Tech. Rep., 2020. [Online]. Available: https://www.dnp.org/LinkClick.aspx?fileticket=hyvYMYugaQI%3D&tabid=66&portalid=0&mid=447&forcedownload=true

[46] S. Bagaria, S. B. Prabhakar, and Z. Saquib, "Flexi-dnp3: Flexible distributed network protocol version 3 (dnp3) for scada security," in *Recent Trends in Information Systems (ReTIS), 2011 International Conference on*. IEEE, 2011, pp. 293–296.

[47] M. Majdalawieh, F. Parisi-Presicce, and D. Wijesekera, "Dnpsec: Distributed network protocol version 3 (dnp3) security framework," in *Advances in Computer, Information, and Systems Sciences, and Engineering*. Springer, 2007, pp. 227–234.

[48] A. Kleinmann and A. Wool, "Accurate Modeling of the Siemens S7 SCADA Protocol for Intrusion Detection and Digital Forensics," *Journal of Digital Forensics, Security and Law*, vol. 9, no. 2, 2014.

[49] C. Lei, L. Donghong, and M. Liang, "The spear to break the security wall of s7commplus," *Blackhat USA*, 2017.

[50] PROFIBUS, "PROFINET System Description - Technology and Application," PROFIBUS, Tech. Rep., 2014.

[51] Profibus International, "Security Extensions for PROFINET - PI White Paper for PROFINET," Profibus, Tech. Rep., 2019.

[52] V. Schiffer, "Common Industrial Protocol (CIP™) and the Family of CIP Networks," *ODVA, Inc.*, no. February, pp. 1–134, 2016.

[53] ODVA. CIP Secuirty. Accessed: 8-10-2020. [Online]. Available: www.odva.org/technology-standards/distinct-cip-services/cip-security

[54] I. González, A. J. Calderón, J. Figueiredo, and J. M. Sousa, "A literature survey on open platform communications (OPC) applied to advanced industrial environments," *Electronics (Switzerland)*, vol. 8, no. 5, pp. 1–29, 2019.

[55] H. Renjie, L. Feng, and P. Dongbo, "Research on OPC UA security," *Proceedings of the 2010 5th IEEE Conference on Industrial Electronics and Applications, ICIEA 2010*, pp. 1439–1444, 2010.

[56] G. Clarke, D. Reynders, and E. Wright, *Practical modern SCADA protocols: DNP3, 60870.5 and related systems*. Newnes, 2004.

[57] P. Maynard, K. McLaughlin, and B. Haberler, "Towards understanding man-in-the-middle attacks on iec 60870-5-104 scada networks," in *ICS-CSR*, 2014.

[58] M. Adamiak, D. Baigent, and R. Mackiewicz, "Iec 61850 communication networks and systems in substations," *The Protection & Control Jurnal-Smart Grid*, pp. 61–68, 2010.

[59] H. Yoo and T. Shon, "Challenges and research directions for heterogeneous cyber-physical system based on IEC 61850: Vulnerabilities, security requirements, and security architecture," *Future Generation Computer Systems*, vol. 61, pp. 128–136, 2016.

[60] R. American Society of Heating and A.-C. Engineers. BACnet Website. Accessed: 11-10-2020. [Online]. Available: http://www.bacnet.org/

[61] W. Xu, Y. Tao, and X. Guan, "The landscape of industrial control systems (ics) devices on the internet," in *2018 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (Cyber SA)*. IEEE, 2018, pp. 1–8.

[62] T. Carlsson. Industrial network market shares 2020 according to HMS Networks. Accessed: 29-01-2021. [Online]. Available: https://www.hms-networks.com/news-and-insights/news-from-hms/2020/05/29/industrial-network-market-shares-2020-according-to-hms-networks

[63] Á. L. P. Gómez, L. F. Maimó, A. H. Celdran, F. J. G. Clemente, C. C. Sarmiento, C. J. D. C. Masa, and R. M. Nistal, "On the generation of anomaly detection datasets in industrial control systems," *IEEE Access*, vol. 7, pp. 177 460–177 473, 2019.

[64] N. R. Rodofile, "Generating attacks and labelling attack datasets for industrial control intrusion detection systems," Ph.D. dissertation, Queensland University of Technology, 2018.

[65] N. R. Rodofile, K. Radke, and E. Foo, "Extending the cyber-attack landscape for scada-based critical infrastructure," *International Journal of Critical Infrastructure Protection*, vol. 25, pp. 14–35, 2019.

[66] J. Giraldo, E. Sarkar, A. A. Cardenas, M. Maniatakos, and M. Kantarcioglu, "Security and privacy in cyber-physical systems: A survey of surveys," *IEEE Design & Test*, vol. 34, no. 4, pp. 7–17, 2017.

[67] MITRE, "Mitre att&ck," https://attack.mitre.org/, (Accessed on 03/01/2023).

[68] Kaspersky, "Ics cert," https://ics-cert.kaspersky.com/, (Accessed on 03/01/2023).

[69] NIST, "National vulnerability database (nvd)," https://nvd.nist.gov/, (Accessed on 03/01/2023).

[70] M. Roesch *et al.*, "Snort: Lightweight intrusion detection for networks," in *Lisa*, vol. 99, no. 1, 1999, pp. 229–238.

[71] Suricata Intrusion Detrction System. Accessed: 25-05-2021. [Online]. Available: https://suricata-ids.org/

[72] R. Mitchell and I. R. Chen, "A survey of intrusion detection techniques for cyber-physical systems," *ACM Computing Surveys*, vol. 46, no. 4, 2014.

[73] Y. Hu, A. Yang, H. Li, Y. Sun, and L. Sun, "A survey of intrusion detection on industrial control systems," *International Journal of Distributed Sensor Networks*, vol. 14, no. 8, 2018.

[74] G. Bernieri, M. Conti, and F. Turrin, "Kingfisher: An industrial security framework based on variational autoencoders," in *Proceedings of the 1st Workshop on Machine Learning on Edge in Sensor Systems*, 2019, pp. 7–12.

[75] H. R. Ghaeini and N. O. Tippenhauer, "HAMIDS: Hierarchical monitoring intrusion detection system for industrial control systems," *CPS-SPC 2016 - Proceedings of the 2nd ACM Workshop on Cyber-Physical Systems Security and PrivaCy, co-located with CCS 2016*, pp. 103–111, 2016.

[76] M. Caselli, E. Zambon, and F. Kargl, "Sequence-aware intrusion detection in industrial control systems," in *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security*, ser. CPSS '15, 2015, p. 13–24.

[77] J. Franco, A. Aris, B. Canberk, and A. S. Uluagac, "A survey of honeypots and honeynets for internet of things, industrial internet of things, and cyber-physical systems," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2351–2383, 2021.

[78] T. Alves, R. Das, and T. Morris, "Embedding encryption and machine learning intrusion prevention systems on programmable logic controllers," *IEEE Embedded Systems Letters*, vol. 10, no. 3, pp. 99–102, 2018.

[79] Y. Xie, W. Wang, F. Wang, and R. Chang, "VTET: A Virtual Industrial Control System Testbed for Cyber Security Research," *2018 3rd International Conference on Security of Smart Cities, Industrial Control System and Communications, SSIC 2018 - Proceedings*, 2018.

[80] N. O. Tippenhauer, "Design and realization of testbeds for security research in the industrial internet of things," in *Security and Privacy Trends in the Industrial Internet of Things*. Springer, 2019, pp. 287–310.

[81] M. Almgren, W. Aoudi, R. Gustafsson, R. Krahl, and A. Lindhé, "The nuts and bolts of deploying process-level IDS in industrial control systems," *ACM International Conference Proceeding Series*, pp. 17–24, 2018.

[82] H. G. Aghamolki, Z. Miao, and L. Fan, "A hardware-in-the-loop scada testbed," in *2015 North American Power Symposium (NAPS)*, 2015, pp. 1–6.

[83] T. Alves, R. Das, and T. Morris, "Virtualization of industrial control system testbeds for cybersecurity," in *Proceedings of the 2nd Annual Industrial Control System Security Workshop*, 2016, pp. 10–14.

[84] T. Cruz, L. Rosa, J. Proença, L. Maglaras, M. Aubigny, L. Lev, J. Jiang, and P. Simões, "A cybersecurity detection framework for supervisory control and data acquisition systems," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 6, pp. 2236–2246, 2016.

[85] R. C. Borges Hink and K. Goseva-Popstojanova, "Characterization of Cyberattacks Aimed at Integrated Industrial Control and Enterprise Systems: A Case Study," *Proceedings of IEEE International Symposium on High Assurance Systems Engineering*, vol. 2016-March, pp. 149–156, 2016.

[86] C. Siaterlis, B. Genge, and M. Hohenadel, "EPIC: A testbed for scientifically rigorous cyber-physical security experimentation," *IEEE Transactions on Emerging Topics in Computing*, vol. 1, no. 2, pp. 319–330, 2013.

[87] Epic testbed website. Accessed: 08-05-2020. [Online]. Available: http://sourceforge.net/projects/amici/

[88] H. Gao, Y. Peng, K. Jia, Z. Dai, and T. Wang, "The design of ICS testbed based on emulation, physical, and simulation (EPS-ICS Testbed)," *Proceedings - 2013 9th International Conference on Intelligent Information Hiding and Multimedia Signal Processing, IIH-MSP 2013*, pp. 420–423, 2013.

[89] R. E. Gillen, L. A. Anderson, C. Craig, J. Johnson, A. Columbia, R. Anderson, A. Craig, and S. L. Scott, "Design and Implementation of Full-Scale Industrial Control System Test Bed for Assessing Cyber-Security Defenses," *Proceedings - 21st IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM 2020*, pp. 341–346, 2020.

[90] H. Hui, P. Maynard, and K. McLaughlin, "Ics interaction testbed: a platform for cyber-physical security research," in *6th International Symposium for ICS & SCADA Cyber Security Research 2019 6*, 2019, pp. 89–96.

[91] G. Bernieri, F. Del Moro, L. Faramondi, and F. Pascucci, "A testbed for integrated fault diagnosis and cyber security investigation," *International Conference on Control, Decision and Information Technologies, CoDIT 2016*, pp. 454–459, 2016.

[92] Hydra testbed repository. Accessed: 11-12-2020. [Online]. Available: https://github.com/hydra-testbed/Part-list

[93] J. Jarmakiewicz, K. Maślanka, and K. Parobczak, "Development of cyber security testbed for critical infrastructure," in *2015 International Conference on Military Communications and Information Systems (ICMCIS)*, 2015, pp. 1–10.

[94] M. Kaouk, F.-X. Morgand, and J.-M. Flaus, "A testbed for cybersecurity assessment of industrial and iot-based control systems," in *21è Congrès de Maîtrise des Risques et Surete de Fonctionnement*, 2018.

[95] J. Kim, K. Kim, and M. Jang, "Cyber-Physical Battlefield Platform for Large-Scale Cybersecurity Exercises," *International Conference on Cyber Conflict, CYCON*, vol. 2019-May, pp. 1–19, 2019.

[96] G. Koutsandria, R. Gentz, M. Jamei, A. Scaglione, S. Peisert, and C. McParland, "A real-time testbed environment for cyber-physical security on the power grid," in *Proceedings of the First ACM Workshop on Cyber-Physical Systems-Security and/or PrivaCy*, ser. CPS-SPC '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 67–78.

[97] P. Čeleda, J. Vykopal, V. Švábenskỳ, and K. Slavíček, "Kypo4industry: A testbed for teaching cybersecurity of industrial control systems," in *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, 2020, pp. 1026–1032.

[98] J. Rubio-Hernan, J. Rodolfo-Mejias, and J. Garcia-Alfaro, "Security of cyber-physical systems," in *International Workshop on the Security of Industrial Control Systems and Cyber-Physical Systems*. Springer, 2016, pp. 3–18.

[99] LegoSCADA Testbed. Accessed: 11-01-2021. [Online]. Available: http://j.mp/legoscada

[100] F. Guo, L. Herrera, M. Alsolami, H. Li, P. Xu, X. Lu, A. Lang, J. Wang, and Z. Long, "Design and development of a reconfigurable hybrid Microgrid testbed," *2013 IEEE Energy Conversion Congress and Exposition, ECCE 2013*, pp. 1350–1356, 2013.

[101] W. Xu, Y. Tao, C. Yang, and H. Chen, "MSICST: Multiple-scenario industrial control system testbed for security research," *Computers, Materials and Continua*, vol. 60, no. 2, pp. 691–705, 2019.

[102] R. Candell, T. Zimmerman, and K. Stouffer, "An Industrial Control System Cybersecurity Performance Testbed," *National Institute of Standards and Technology, U.S. Department of Commerce*, no. November, 2015.

[103] T. Edgar, D. Manz, and T. Carroll, "Towards an experimental testbed facility for cyber-physical security research," *ACM International Conference Proceeding Series*, pp. 4–7, 2011.

[104] C. Queiroz, A. Mahmood, J. Hu, Z. Tari, and X. Yu, "Building a SCADA security testbed," *NSS 2009 - Network and System Security*, pp. 357–364, 2009.

[105] V. Urias, B. Van Leeuwen, and B. Richardson, "Supervisory Command and Data Acquisition (SCADA) system cyber security analysis using a live, virtual, and constructive (LVC) testbed," *Proceedings - IEEE Military Communications Conference MILCOM*, no. Lvc, pp. 1–8, 2012.

[106] D. C. Bergman, D. Jin, D. M. Nicol, and T. Yardley, "The virtual power system testbed and inter-testbed integration," *2nd Workshop on Cyber Security Experimentation and Test, CSET 2009*, no. August, 2009.

[107] I. Ahmed, V. Roussev, W. Johnson, S. Senthivel, and S. Sudhakaran, "A SCADA system testbed for cybersecurity and forensic research and pedagogy," *ACM International Conference Proceeding Series*, pp. 1–9, 2016.

[108] P. Blazek, R. Fujdiak, P. Mlynek, and J. Misurec, "Development of cyber-physical security testbed based on iec 61850 architecture," *Elektronika ir Elektrotechnika*, vol. 25, no. 5, pp. 82–87, 2019.

[109] E. Korkmaz, A. Dolgikh, M. Davis, and V. Skormin, "Industrial control systems security testbed," in *11th Annual Symposium on Information Assurance*, 2016.

[110] S. Adepu, N. K. Kandasamy, and A. Mathur, "EPIC: An electric power testbed for research and training in cyber physical systems security," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11387 LNCS, no. November, pp. 37–52, 2018.

[111] Singapore University of Technology and Design (SUTD). Electric Power and Intelligent Control (EPIC) testbed Webpage. Accessed: 13-01-2021. [Online]. Available: https://itrust.sutd.edu.sg/itrust-labs-home/itrust-labs_epic/

[112] H. K. Shin, W. Lee, J. H. Yun, and H. C. Kim, "Implementation of programmable CPS testbed for anomaly detection," in *12th USENIX Workshop on Cyber Security*

Experimentation and Test, CSET 2019, co-located with USENIX Security 2019, 2019. [Online]. Available: https://www.usenix.org/conference/cset19/presentation/shin

[113] B. Green, A. Le, R. Antrobus, U. Roedig, D. Hutchison, and A. Rashid, "Pains, gains and PLCs: Ten lessons from building an industrial control systems testbed for security research," *10th USENIX Workshop on Cyber Security Experimentation and Test, CSET 2017, co-located with USENIX Security 2017*, 2017.

[114] F. Sauer, M. Niedermaier, S. Kießling, and D. Merli, "Licster–a low-cost ics security testbed for education and research," in *6th International Symposium for ICS & SCADA Cyber Security Research 2019 6*, 2019, pp. 1–10.

[115] hsainnos. Low-cost ICS Testbed Github Repository. Accessed: 10-02-2021. [Online]. Available: https://github.com/hsainnos/LICSTER

[116] T. Morris, R. Vaughn, and Y. S. Dandass, "A testbed for SCADA control system cybersecurity research and pedagogy," *ACM International Conference Proceeding Series*, 2011.

[117] A. Hahn, B. Kregel, M. Govindarasu, J. Fitzpatrick, R. Adnan, S. Sridhar, and M. Higdon, "Development of the PowerCyber SCADA security testbed," *ACM International Conference Proceeding Series*, pp. 1–4, 2010.

[118] N. Sayegh, A. Chehab, I. H. Elhajj, and A. Kayssi, "Internal security attacks on scada systems," in *2013 Third International Conference on Communications and Information Technology (ICCIT)*. IEEE, 2013, pp. 22–27.

[119] Cx-270322: Idaho national laboratory (inl) smart grid test bed revision 2. Accessed: 30-04-2020. [Online]. Available: https://www.energy.gov/sites/prod/files/2017/10/f38/CX-270322.pdf

[120] A. P. Mathur and N. O. Tippenhauer, "SWaT: A water treatment testbed for research and training on ICS security," *2016 International Workshop on Cyber-physical Systems for Smart Water Networks, CySWater 2016*, no. Figure 1, pp. 31–36, 2016.

[121] Singapore University of Technology and Design (SUTD). Secure Water Treatment (SWaT) testbed Webpage. Accessed: 13-01-2021. [Online]. Available: https://itrust.sutd.edu.sg/itrust-labs-home/itrust-labs_swat/

[122] M. A. Teixeira, T. Salman, M. Zolanvari, R. Jain, N. Meskin, and M. Samaka, "SCADA system testbed for cybersecurity research using machine learning approach," *Future Internet*, vol. 10, no. 8, 2018.

[123] I. N. Fovino, M. Masera, L. Guidi, and G. Carpi, "An experimental platform for assessing SCADA vulnerabilities and countermeasures in power plants," *3rd International Conference on Human System Interaction, HSI'2010 - Conference Proceedings*, pp. 679–686, 2010.

[124] C. M. Ahmed, V. R. Palleti, and A. P. Mathur, "WADI: A water distribution testbed for research in the design of secure cyber physical systems," *Proceedings - 2017 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks, CySWATER 2017*, pp. 25–28, 2017.

[125] Singapore University of Technology and Design (SUTD). Water Distribution (WADI) testbed Webpage. Accessed: 13-01-2021. [Online]. Available: https://itrust.sutd.edu.sg/itrust-labs-home/itrust-labs_wadi/

[126] Y. Yang, K. McLaughlin, S. Sezer, T. Littler, E. G. Im, B. Pranggono, and H. F. Wang, "Multiattribute scada-specific intrusion detection system for power networks," *IEEE Transactions on Power Delivery*, vol. 29, no. 3, pp. 1092–1102, 2014.

[127] F. Zhang, H. A. D. E. Kodituwakku, J. W. Hines, and J. Coble, "Multilayer Data-Driven Cyber-Attack Detection System for Industrial Control Systems Based on Network, System, and Process Data," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4362–4369, 2019.

[128] C. M. Davis, J. E. Tate, H. Okhravi, C. Grier, T. J. Overbye, and D. Nicol, "SCADA cyber security testbed development," in *2006 38th Annual North American Power Symposium, NAPS-2006 Proceedings*, 2006, pp. 483–488.

[129] M. Krotofil and J. Larsen, "Rocking the pocket book: Hacking chemical plants," in *DefCon Conference, DEFCON*, 2015.

[130] DVCP-TE. Accessed: 12-01-2021. [Online]. Available: https://github.com/satejnik/DVCP-TE

[131] A. A. Farooqui, S. S. H. Zaidi, A. Y. Memon, and S. Qazi, "Cyber security backdrop: A scada testbed," in *2014 IEEE Computers, Communications and IT Applications Conference*. IEEE, 2014, pp. 98–103.

[132] T. H. Morris, Z. Thornton, and I. Turnipseed, "Industrial Control System Simulation and Data Logging for Intrusion Detection System Research," *Seventh Annual Southeastern Cyber Security Summit*, 2015. [Online]. Available: http://www.ece.uah.edu/$\sim$thm0009/icsdatasets/cyberhuntsvillepaper_v4.pdf

[133] B. Genge, C. Siaterlis, I. Nai Fovino, and M. Masera, "A cyber-physical experimentation environment for the security analysis of networked industrial control systems," *Computers and Electrical Engineering*, vol. 38, no. 5, pp. 1146–1161, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.compeleceng.2012.06.015

[134] A. Giani, G. Karsai, T. Roosta, A. Shah, B. Sinopoli, and J. Wiley, "A testbed for secure and robust SCADA systems," *ACM SIGBED Review*, vol. 5, no. 2, pp. 1–4, 2008.

[135] D. Formby, M. Rad, and R. Beyah, "Lowering the barriers to industrial control system security with {GRFICS}," in *2018 {USENIX} Workshop on Advances in Security Education ({ASE} 18)*, 2018.

[136] ——. Grfics. Accessed: 11-01-2021. [Online]. Available: https://github.com/djformby/GRFICS

[137] D. Jin, D. M. Nicol, and G. Yan, "An event buffer flooding attack in dnp3 controlled scada systems," in *Proceedings of the 2011 Winter Simulation Conference (WSC)*. IEEE, 2011, pp. 2614–2626.

[138] V. S. Koganti, M. Ashrafuzzaman, A. A. Jillepalli, and F. T. Sheldon, "A virtual testbed for security management of industrial control systems," in *2017 12th International Conference on Malicious and Unwanted Software (MALWARE)*, 2017, pp. 85–90.

[139] S. Lee, S. Lee, H. Yoo, S. Kwon, and T. Shon, "Design and implementation of cybersecurity testbed for industrial iot systems," *The Journal of Supercomputing*, vol. 74, no. 9, pp. 4506–4520, 2018.

[140] P. Maynard, K. McLaughlin, and S. Sezer, "An Open Framework for Deploying Experimental SCADA Testbed Networks," *5th International Symposium for ICS & SCADA Cyber Security Research 2018: Proceedings*, 2018.

[141] ——. Ics testbed framework. Accessed: 08-05-2020. [Online]. Available: https://github.com/PMaynard/ICS-TestBed-Framework

[142] D. Antonioli and N. O. Tippenhauer, "Minicps: A toolkit for security research on cps networks," in *Proceedings of the First ACM workshop on cyber-physical systems-security and/or privacy*, 2015, pp. 91–100.

[143] Minicps: a framework for cyber-physical systems real-time simulation, built on top of mininet. Accessed: 08-05-2020. [Online]. Available: https://github.com/scy-phy/minicps

[144] B. Reaves and T. Morris, "An open virtual testbed for industrial control system security research," *International Journal of Information Security*, vol. 11, no. 4, pp. 215–229, 2012.

[145] M. Almgren, P. Andersson, G. Björkman, M. Ekstedt, J. Hallberg, S. Nadjm-Tehrani, and E. Westring, "Rics-el: building a national testbed for research and training on scada security (short paper)," in *International Conference on Critical Information Infrastructures Security*. Springer, 2018, pp. 219–225.

[146] A. Ghaleb, S. Zhioua, and A. Almulhem, "Scada-sst: a scada security testbed," in *2016 World Congress on Industrial Control Systems Security (WCICSS)*. IEEE, 2016, pp. 1–6.

[147] SCADA-SST - scada security testbed. Accessed: 12-01-2021. [Online]. Available: https://sourceforge.net/projects/scada-sst/

[148] C. Queiroz, A. Mahmood, and Z. Tari, "SCADASim - A framework for building SCADA simulations," *IEEE Transactions on Smart Grid*, vol. 2, no. 4, pp. 589–597, 2011.

[149] T. Z. Queiroz Carlos, Mahmood Abdun. scadasim on github. Accessed: 26-07-2020. [Online]. Available: https://github.com/caxqueiroz/scadasim

[150] A. Almalawi, Z. Tari, I. Khalil, and A. Fahad, "SCADAVT-A framework for SCADA security testbed based on virtualization technology," *Proceedings - Conference on Local Computer Networks, LCN*, pp. 639–646, 2013.

[151] P. Singh, S. Garg, V. Kumar, and Z. Saquib, "A testbed for scada cyber security and intrusion detection," in *2015 International Conference on Cyber Security of Smart Cities, Industrial Control System and Communications (SSIC)*. IEEE, 2015, pp. 1–6.

[152] M. Mallouhi, Y. Al-Nashif, D. Cox, T. Chadaga, and S. Hariri, "A testbed for analyzing security of SCADA control systems (TASSCS)," *IEEE PES Innovative Smart Grid Technologies Conference Europe, ISGT Europe*, pp. 1–7, 2011.

[153] C. Wang, L. Fang, and Y. Dai, "A simulation environment for SCADA security analysis and assessment," *2010 International Conference on Measuring Technology and Mechatronics Automation, ICMTMA 2010*, vol. 1, pp. 342–347, 2010.

[154] J. Gardiner, B. Craggs, B. Green, and A. Rashid, "Oops I did it again: Further adventures in the land of ICS security testbeds," *Proceedings of the ACM Conference on Computer and Communications Security*, pp. 75–86, 2019.

[155] E. Eide, L. Stoller, and J. Lepreau, "An experimentation workbench for replayable networking research," in *NSDI*, 2007.

[156] E. Eide, "Toward replayable research in networking and systems," *Position paper presented at Archive*, no. May, 2010. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.170.9948&rep=rep1&type=pdf

[157] S. Choi, J.-H. Yun, B.-G. Min, and H. Kim, "Poster: Expanding a programmable cps testbed for network attack analysis," in *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*, 2020, pp. 928–930.

[158] Industrial control system (ics) cyber attack datasets by morris et al. Accessed: 27-04-2020. [Online]. Available: https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets

[159] I. Moreno-Garcia, A. Moreno-Munoz, V. Pallares-Lopez, M. Gonzalez-Redondo, E. J. Palacios-Garcia, and C. D. Moreno-Moreno, "Development and application of a smart grid test bench," *Journal of Cleaner Production*, vol. 162, pp. 45–60, 2017.

[160] J. Goh, S. Adepu, K. N. Junejo, and A. Mathur, "A dataset to support research in the design of secure water treatment systems," in *International Conference on Critical Information Infrastructures Security*. Springer, 2016, pp. 88–99.

[161] Y. Chen, C. M. Poskitt, and J. Sun, "Learning from mutants: Using code mutation to learn and monitor invariants of a cyber-physical system," in *IEEE Symposium on Security and Privacy (SP)*. IEEE, 2018, pp. 648–660.

[162] SWaT_Simulator. Accessed: 12-01-2021. [Online]. Available: https://github.com/yuqiChen94/Swat_Simulator

[163] J. Goh, S. Adepu, K. N. Junejo, and A. Mathur, "A dataset to support research in the design of secure water treatment systems," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10242 LNCS, no. October, pp. 88–99, 2017.

[164] WUSTL-IIOT-2018 Dataset for ICS (SCADA) Cybersecurity Research. Accessed: 11-01-2021. [Online]. Available: https://www.cse.wustl.edu/~jain/iiot/index.html

[165] M. Zolanvari, M. A. Teixeira, L. Gupta, K. M. Khan, and R. Jain, "Machine Learning-Based Network Vulnerability Analysis of Industrial Internet of Things," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6822–6834, 2019.

[166] Sutd-mit international design centre (idc), water distribution (wadi) testbed. Accessed: 06-05-2020. [Online]. Available: https://itrust.sutd.edu.sg/itrust-labs-home/itrust-labs_wadi/

[167] M. Liljenstam, J. Liu, D. Nicol, Y. Yuan, G. Yan, and C. Grier, "Rinse: The real-time immersive network simulation environment for network security exercises (extended version)." *Simulation*, vol. 82, pp. 43–59, 01 2006.

[168] P. R. Lyman and C. Georgakis, "Plant-wide control of the tennessee eastman problem," *Computers & chemical engineering*, vol. 19, no. 3, pp. 321–331, 1995.

[169] L. C. Argenta, M. J. Morykwas *et al.*, "Vacuum-assisted closure: a new method for wound control and treatment: clinical experience," *Annals of plastic surgery*, vol. 38, pp. 563–577, 1997.

[170] Emulab: A time- and space-shared platform for research, education, and development in distributed systems and networks. Accessed: 14-12-2020. [Online]. Available: https://www.emulab.net/

[171] Matlab/Simulink: Simulation and Model-Based Design. Accessed: 14-12-2020. [Online]. Available: https://mathworks.com/products/simulink.html

[172] Oracle VM VirtualBox: a powerful x86 and AMD64/Intel64 virtualization product for enterprise as well as home use. Accessed: 15-12-2020. [Online]. Available: https://www.virtualbox.org/

[173] K. Kaur, J. Singh, and N. S. Ghumman, "Mininet as software defined networking testing platform," in *International Conference on Communication, Computing & Systems (ICCCS)*, 2014, pp. 139–42.

[174] Z. Thornton and T. Morris, "Enhancing a virtual scada laboratory using simulink," in *International Conference on Critical Infrastructure Protection*. Springer, 2015, pp. 119–133.

[175] CRATE - Cyber Range And Training Environment. Accessed: 11-01-2021. [Online]. Available: https://www.foi.se/en/foi/resources/crate---cyber-range-and-training-environment.html

[176] OMNeT++ Discrete Event Simulator. An extensible, modular, component-based C++ simulation library and framework, primarily for building network simulators. Accessed: 15-12-2020. [Online]. Available: https://omnetpp.org/

[177] CyberCity allows government hackers to train for attacks. Accessed: 10-01-2021. [Online]. Available: https://www.washingtonpost.com/investigations/cybercity-allows-government-hackers-to-train-for-attacks/2012/11/26/588f4dae-1244-11e2-be82-c3411b7680a9_story.html

[178] Matlab/Simulink Coder: Generate C and C++ code from Simulink and Stateflow models. Accessed: 14-12-2020. [Online]. Available: https://www.mathworks.com/products/simulink-coder.html

[179] Summit - Oak Ridge National Laboratory's 200 petaflop supercomputer. Accessed: 22-12-2020. [Online]. Available: https://www.olcf.ornl.gov/olcf-resources/compute-systems/summit/

[180] November 2019 - TOP500 Supercomputer Sites. Accessed: 22-12-2020. [Online]. Available: https://www.top500.org/lists/2019/11/

[181] M. Rollins, *Beginning Lego Mindstorms Ev3*. Apress, 2014.

[182] Teach, Learn and Make with Raspberry Pi. Accessed: 14-12-2020. [Online]. Available: https://www.raspberrypi.org/

[183] OPNET Network simulator. Accessed: 14-12-2020. [Online]. Available: https://opnetprojects.com/opnet-network-simulator/

[184] Pnnl control testbed. Accessed: 03-08-2020. [Online]. Available: https://controls.pnnl.gov/testbed/

[185] Arion: simplifying models for controls research. Accessed: 14-12-2020. [Online]. Available: https://arion.labworks.org/

[186] D. Maynor, *Metasploit toolkit for penetration testing, exploit development, and vulnerability research*. Elsevier, 2011.

[187] PowerWorld: The visual approach to electric power systems. Accessed: 15-12-2020. [Online]. Available: https://www.powerworld.com/

[188] T. Morris and W. Gao, "Industrial control system traffic data sets for intrusion detection research," *IFIP Advances in Information and Communication Technology*, vol. 441, pp. 65–78, 2014.

[189] WEKA – The workbench for machine learning. Accessed: 15-01-2021. [Online]. Available: https://www.cs.waikato.ac.nz/ml/weka/

[190] EPANET. Application for Modeling Drinking Water Distribution Systems. Accessed: 15-12-2020. [Online]. Available: https://www.epa.gov/water-research/epanet

[191] A. Lemay and J. M. Fernandez, "Providing SCADA network data sets for intrusion detection research," *9th USENIX Workshop on Cyber Security Experimentation and Test, CSET 2016*, p. 8, 2016.

[192] G. Bernieri, M. Conti, and F. Turrin, "Evaluation of machine learning algorithms for anomaly detection in industrial networks," in *2019 IEEE International Symposium on Measurements Networking (M N)*, 2019, pp. 1–6.

[193] R. Taormina, S. Galelli, N. O. Tippenhauer, E. Salomons, A. Ostfeld, D. G. Eliades, M. Aghashahi, R. Sundararajan, M. Pourahmadi, M. K. Banks, B. M. Brentan, E. Campbell, G. Lima, D. Manzi, D. Ayala-Cabrera, M. Herrera, I. Montalvo, J. Izquierdo, E. Luvizotto, S. E. Chandy, A. Rasekh, Z. A. Barker, B. Campbell, M. E. Shafiee, M. Giacomoni, N. Gatsis, A. Taha, A. A. Abokifa, K. Haddad, C. S. Lo, P. Biswas, M. Fayzul, B. Kc, S. L. Somasundaram, M. Housh, and Z. Ohar, "Battle of the Attack Detection Algorithms: Disclosing cyber attacks on water distribution networks," *Journal of Water Resources Planning and Management*, vol. 144, no. 8, 2018.

[194] iTrust, Centre for Research in Cyber Security, Singapore University of Technology and Design, iTrust Labs Dataset Info. Accessed: 24-04-2020. [Online]. Available: https://itrust.sutd.edu.sg/itrust-labs_datasets/dataset_info

[195] Batadal datasets. Accessed: 08-05-2020. [Online]. Available: http://www.batadal.net/data.html

[196] R. Taormina, S. Galelli, H. Douglas, N. O. Tippenhauer, E. Salomons, and A. Ostfeld, "A toolbox for assessing the impacts of cyber-physical attacks on water distribution systems. environmental modelling software," *Environmental Modelling Software*, vol. 112, pp. 46–51, 02 2019.

[197] A. Erba, R. Taormina, S. Galelli, M. Pogliani, M. Carminati, S. Zanero, and N. O. Tippenhauer. Constrained concealment attacks on reconstruction-based anomaly detectors in industrial control systems. Accessed: 07-12-2020. [Online]. Available: https://github.com/scy-phy/ICS-Evasion-Attacks

[198] ——, "Constrained concealment attacks against reconstruction-based anomaly detectors in industrial control systems," in *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, 2020.

[199] N. Rodofile. Scada network attack datasets and process logs. Accessed: 08-05-2020. [Online]. Available: https://github.com/qut-infosec/2017QUT_DNP3

[200] N. R. Rodofile, T. Schmidt, S. T. Sherry, C. Djamaludin, K. Radke, and E. Foo, "Process control cyber-attacks and labelled datasets on s7comm critical infrastructure," in *Australasian Conference on Information Security and Privacy*. Springer, 2017, pp. 452–459.

[201] N. Rodofile. Scada network attack datasets and process logs. Accessed: 08-05-2020. [Online]. Available: https://github.com/qut-infosec/2017QUT_S7comm

[202] Ics lab: 4sics ics lab pcap file. Accessed: 27-04-2020. [Online]. Available: https://www.netresec.com/?page=PCAP4SICS

[203] T. C. Yu, J. Y. Huang, I. E. Liao, and K. F. Kao, "Mining anomaly communication patterns for industrial control systems," *Australasian Universities Power Engineering Conference, AUPEC 2018*, 2018.

[204] CyberCity SANS Holiday Hack 2013 Dataset. Accessed: 10-01-2021. [Online]. Available: https://assets.contentstack.io/v3/assets/blt36c2e63521272fdc/bltff8e7c1232f3bcbc/5fbd7be072a3526f28dbed75/sansholidayhack2013.pcap

[205] J. M. Beaver, R. C. Borges-Hink, and M. A. Buckner, "An evaluation of machine learning methods to detect malicious SCADA communications," *Proceedings - 2013 12th International Conference on Machine Learning and Applications, ICMLA 2013*, vol. 2, pp. 54–59, 2013.

[206] K. Demertzis, L. Iliadis, and S. Spartalis, "A Spiking One-Class Anomaly Detection Framework for Cyber-Security on Industrial Control Systems," in *EANN 2017: Engineering Applications of Neural Networks*, 2017, vol. 2, no. August, pp. 122–134. [Online]. Available: http://link.springer.com/10.1007/978-3-319-65172-9_11

[207] Dataset for cybersecurity research in industrial control systems. Accessed: 06-05-2020. [Online]. Available: http://perception.inf.um.es/ICS-datasets/

[208] G. K. Ndonda and R. Sadre, "Network trace generation for flow-based IDS evaluation in control and automation systems," *International Journal of Critical Infrastructure Protection*, vol. 31, p. 100385, 2020.

[209] S. R. Ndonda Gorby Kabasele. Hvac traces. Accessed: 21-05-2020. [Online]. Available: https://github.com/gkabasele/HVAC_Traces

[210] A. Lemay. Modbus dataset from cset 2016. Accessed: 08-05-2020. [Online]. Available: https://github.com/antoine-lemay/Modbus_dataset

[211] S. D. Anton, S. Kanoor, D. Fraunholz, and H. D. Schotten, "Evaluation of machine learning-based anomaly detection algorithms on an industrial modbus/tcp data set," in *Proceedings of the 13th International Conference on Availability, Reliability and Security*, 2018, pp. 1–9.

[212] I. Frazão, P. H. Abreu, T. Cruz, H. Araújo, and P. Simões, "Denial of service attacks: Detecting the frailties of machine learning algorithms in the classification process," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11260 LNCS, no. 700665, pp. 230–235, 2019.

[213] I. Frazão, P. Henriques Abreu, T. Cruz, H. Araujo, and P. Simoes. Modbus tcp scada #1 dataset. Accessed: 30-04-2020. [Online]. Available: https://github.com/tjcruz-dei/ICS_PCAPS/releases/tag/MODBUSTCP%231

[214] P. Radoglou-Grammatikis, I. Siniosoglou, T. Liatifis, A. Kourouniadis, K. Rompolos, and P. Sarigiannidis, "Implementation and Detection of Modbus Cyberattacks," in *2020 9th International Conference on Modern Circuits and Systems Technologies (MOCAST)*. IEEE, 2020, pp. 1–4. [Online]. Available: https://ieeexplore.ieee.org/document/9200287/

[215] Peterson, d., wightman, r.: Digital bond s4x15 ics village ctf pcap files. Accessed: 27-04-2020. [Online]. Available: https://www.netresec.com/?page=DigitalBond_S4

[216] R. C. Borges Hink, J. M. Beaver, M. A. Buckner, T. Morris, U. Adhikari, and S. Pan, "Machine learning for power system disturbance and cyber-attack discrimination," *7th International Symposium on Resilient Control Systems, ISRCS 2014*, 2014.

[217] Singapore University of Technology and Design (SUTD). Dataset Characteristics. Accessed: 13-01-2021. [Online]. Available: https://itrust.sutd.edu.sg/itrust-labs_datasets/dataset_info/

[218] D. Myers, S. Suriadi, K. Radke, and E. Foo, "Anomaly detection for industrial control systems using process mining," *Computers and Security*, vol. 78, pp. 103–125, 2018. [Online]. Available: https://doi.org/10.1016/j.cose.2018.06.002

[219] QUT_S7 Communication by Myers et al. dataset. Accessed: 19-12-2020. [Online]. Available: https://cloudstor.aarnet.edu.au/plus/index.php/s/9qFfeVmfX7K5IDH

[220] M. F. Abdelaty, R. Doriguzzi Corin, and D. Siracusa, "Daics: A deep learning solution for anomaly detection in industrial control systems," *IEEE Transactions on Emerging Topics in Computing*, pp. 1–1, 2021.

[221] M. Housh and Z. Ohar, "Model-based approach for cyber-physical attack detection in water distribution systems," *Water Research*, vol. 139, pp. 132–143, 2018.

[222] H.-K. Shin, W. Lee, J.-H. Yun, and H. Kim. IL-based Augmented ICS (HAI) Security Dataset. Accessed: 22-10-2020. [Online]. Available: https://github.com/icsdataset/hai

[223] Y. Kim and H. K. Kim, "Anomaly Detection using Clustered Deep One-Class Classification," in *2020 15th Asia Joint Conference on Information Security (AsiaJCIS)*. IEEE, aug 2020, pp. 151–157. [Online]. Available: https://ieeexplore.ieee.org/document/9194140/

[224] M. Kravchik and A. Shabtai, "Efficient cyber attack detection in industrial control systems using lightweight neural networks and PCA," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2021.

[225] S. Pan, T. Morris, and U. Adhikari, "A specification-based intrusion detection framework for cyber-physical environment in electric power system," *International Journal of Network Security*, vol. 17, no. 2, pp. 174–188, 2015.

[226] ——, "Classification of disturbances and cyber-attacks in power systems using heterogeneous time-synchronized data," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 3, pp. 650–662, 2015.

[227] ——, "Developing a Hybrid Intrusion Detection System Using Data Mining for Power Systems," *IEEE Transactions on Smart Grid*, vol. 6, no. 6, pp. 3104–3113, 2015.

[228] J. Fürnkranz and G. Widmer, "Incremental reduced error pruning," in *Machine Learning Proceedings 1994*. Elsevier, 1994, pp. 70–77.

[229] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[230] H.-K. Shin, W. Lee, J.-H. Yun, and H. Kim, "HAI 1.0: Hil-based augmented ICS security dataset," in *13th USENIX Workshop on Cyber Security Experimentation and Test (CSET 20)*. USENIX Association, Aug. 2020. [Online]. Available: https://www.usenix.org/conference/cset20/presentation/shin

[231] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, and S.-K. Ng, "Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks," in *International Conference on Artificial Neural Networks*. Springer, 2019, pp. 703–716.

[232] Ö. Yüksel, J. D. Hartog, and S. Etalle, "Reading between the fields: Practical, effective intrusion detection for industrial control systems," *Proceedings of the ACM Symposium on Applied Computing*, vol. 04-08-Apri, pp. 2063–2070, 2016.

[233] C. Feng, T. Li, and D. Chana, "Multi-level Anomaly Detection in Industrial Control Systems via Package Signatures and LSTM Networks," *Proceedings - 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2017*, pp. 261–272, 2017.

[234] K. Demertzis, L. Iliadis, and I. Bougoudis, "Gryphon: a semi-supervised anomaly detection system based on one-class evolving spiking neural network," *Neural Computing and Applications*, vol. 32, no. 9, pp. 4303–4314, 2020.

[235] A. Mansouri, B. Majidi, and A. Shamisa, "Metaheuristic neural networks for anomaly recognition in industrial sensor networks with packet latency and jitter for smart infrastructures," *International Journal of Computers and Applications*, vol. 0, no. 0, pp. 1–10, 2018. [Online]. Available: https://doi.org/1206212X.2018.1533613

[236] J. Jägersküpper, "How the (1+1) es using isotropic mutations minimizes positive definite quadratic forms," *Theoretical Computer Science*, vol. 361, no. 1, pp. 38–56, 2006.

[237] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in engineering software*, vol. 69, pp. 46–61, 2014.

[238] Á. L. Perales Gómez, L. Fernández Maimó, A. Huertas Celdrán, F. J. García Clemente, M. Gil Pérez, and G. Martínez Pérez, "Safeman: A unified framework to manage cybersecurity and safety in manufacturing industry," *Software: Practice and Experience*, vol. 51, no. 3, pp. 607–627, 2021.

[239] Y. Li, X. Ji, C. Li, X. Xu, W. Yan, X. Yan, Y. Chen, and W. Xu, "Cross-domain anomaly detection for power industrial control system," in *2020 IEEE 10th International Conference on Electronics Information and Emergency Communication (ICEIEC)*. IEEE, 2020, pp. 383–386.

[240] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," *IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009*, no. Cisda, pp. 1–6, 2009.

[241] D. L. Mills, "Internet time synchronization: the network time protocol," *IEEE Transactions on communications*, vol. 39, no. 10, pp. 1482–1493, 1991.

[242] P. Schneider and K. Böttinger, "High-performance unsupervised anomaly detection for cyber-physical system networks," *Proceedings of the ACM Conference on Computer and Communications Security*, pp. 1–12, 2018.

[243] S. D. Anton, L. Ahrens, D. Fraunholz, and H. D. Schotten, "Time is of the essence: Machine learning-based intrusion detection in industrial time series data," in *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 2018, pp. 1–6.

[244] J. Luswata, P. Zavarsky, B. Swar, and D. Zvabva, "Analysis of scada security using penetration testing: A case study on modbus tcp protocol," in *2018 29th Biennial Symposium on Communications (BSC)*. IEEE, 2018, pp. 1–5.

[245] A. H. Lashkari, G. Draper-Gil, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of tor traffic using time based features." in *ICISSp*, 2017, pp. 253–262.

[246] D. Bond. Quickdraw snort. Accessed: 09-10-2020. [Online]. Available: https://github.com/digitalbond/Quickdraw-Snort

[247] Nmap: the Network Mapper - Free Security Scanner. Accessed: 11-12-2020. [Online]. Available: hhttps://nmap.org/

[248] F. Turrin, A. Erba, N. O. Tippenhauer, and M. Conti, "A statistical analysis framework for ics process datasets," in *Proceedings of the 2020 Joint Workshop on CPS&IoT Security and Privacy*, 2020, pp. 25–30.

[249] C. M. Ahmed, J. Zhou, and A. P. Mathur, "Noise matters: Using sensor and process noise fingerprint to detect stealthy cyber attacks and authenticate sensors in cps," in *Proceedings of the 34th Annual Computer Security Applications Conference*, 2018, pp. 566–581.

[250] Q. Lin, S. Adepu, S. Verwer, and A. Mathur, "Tabor: A graphical model-based approach for anomaly detection in industrial control systems," in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, 2018, pp. 525–536.

[251] C. Feng, V. R. Palleti, A. Mathur, and D. Chana, "A systematic framework to generate invariants for anomaly detection in industrial control systems." in *Proceedings of the Network and Distributed System Security Symposium(NDSS)*, 2019.

[252] I. E. Commission, "Security for industrial automation and control systems. Security risk assessment for system design," International Electrotechnical Commission, Standard IEC 62443-3-2:2020, 2020.

[253] N. C. S. S. D. Team, "Cyber Security — Critical Cyber Asset Identification," International Electrotechnical Commission, Standard NERC CIP-002-3 through CIP-009-3, 2010.

[254] ENISA, "ICS Security Related Working Groups , Standards and Initiatives For the Report : Good practices for an EU ICS testing," ENISA, Tech. Rep., 2013. [Online]. Available: https://www.enisa.europa.eu/events/good-practices-for-an-eu-ics-testing-coordination-capability/ics-security-related-working-groups-standards-and-initiatives

[255] J. L. Torres, C. A. Catania, and E. Veas, "Active learning approach to label network traffic datasets," *Journal of Information Security and Applications*, vol. 49, p. 102388, 2019.

[256] M. Zolanvari, M. A. Teixeira, and R. Jain, "Effect of Imbalanced Datasets on Security of Industrial IoT Using Machine Learning," in *2018 IEEE International Conference on Intelligence and Security Informatics (ISI)*. IEEE, nov 2018, pp. 112–117. [Online]. Available: https://ieeexplore.ieee.org/document/8587389/

[257] D. Ramyachitra and P. Manikandan, "Imbalanced dataset classification and solutions: a review," *International Journal of Computing and Business Research (IJCBR)*, vol. 5, no. 4, 2014.

[258] S. K. Lim, Y. Loo, N. Tran, N. Cheung, G. Roig, and Y. Elovici, "Doping: Generative data augmentation for unsupervised anomaly detection with gan," in *2018 IEEE International Conference on Data Mining (ICDM)*, 2018, pp. 1122–1127.

[259] G. O. Diaz and V. Ng, "Modeling and prediction of online product review helpfulness: a survey," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 698–708.

[260] T. J. Williams, "The purdue enterprise reference architecture," *Computers in industry*, pp. 141–158, 1994.

[261] A. Mirian, Z. Ma, D. Adrian, M. Tischer, T. Chuenchujit, T. Yardley, R. Berthier, J. Mason *et al.*, "An internet-wide view of ics devices," in *Proceedings of the Annual Conference on Privacy, Security and Trust (PST)*, 2016, pp. 96–103.

[262] X. Feng, Q. Li, H. Wang, and L. Sun, "Characterizing industrial control system devices on the internet," in *2016 IEEE 24th International Conference on Network Protocols (ICNP)*, 2016, pp. 1–10.

[263] H. Al-Alami, A. Hadi, and H. Al-Bahadili, "Vulnerability scanning of IoT devices in jordan using shodan," in *2017 2nd International Conference on the Applications of Information Technology in Developing Renewable Energy Processes & Systems (IT-DREPS)*, 2017, pp. 1–6.

[264] J. M. Ceron, J. J. Chromik, J. Santanna, and A. Pras, "Online discoverability and vulnerabilities of ics/scada devices in the netherlands," *arXiv preprint arXiv:2011.02019*, 2020.

[265] Shodan, https://www.shodan.io/.

[266] Censys, https://censys.io/.

[267] M. Nawrocki, T. C. Schmidt, and M. Wählisch, "Uncovering vulnerable industrial control systems from the internet core," in *NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium*, 2020, pp. 1–9.

[268] VSIX Internet Exchange Point, https://www.vsix.it/.

[269] G. Bernieri, M. Conti, and F. Turrin, "Evaluation of machine learning algorithms for anomaly detection in industrial networks," in *Proceedings of the IEEE International Symposium on Measurements Networking (M N)*, July 2019, pp. 1–6.

[270] S. D. Anton, S. Kanoor, D. Fraunholz, and H. D. Schotten, "Evaluation of machine learning-based anomaly detection algorithms on an industrial modbus/tcp data set," in *Proceedings of the 13th International Conference on Availability, Reliability and Security*, 2018, pp. 1–9.

[271] A. V. Serbanescu, S. Obermeier, and D.-Y. Yu, "Ics threat analysis using a large-scale honeynet," in *Proceedings of the International Symposium for ICS & SCADA Cyber Security Research 2015 (ICS-CSR 2015) 3*, 2015, pp. 20–30.

[272] "sFlow Standard," https://sflow.org/sflow_version_5.txt.

[273] A. Lemay and J. M. Fernandez, "Providing {SCADA} network data sets for intrusion detection research," in *9th Workshop on Cyber Security Experimentation and Test ({CSET} 16)*, 2016.

[274] D. Formby, P. Srinivasan, A. M. Leonard, J. D. Rogers, and R. A. Beyah, "Who's in control of your control system? device fingerprinting for cyber-physical systems." in *NDSS*, 2016.

[275] "sFlow Sampling," https://sflow.org/packetSamplingBasics/index.htm.

[276] J. Jedwab, P. Phaal, and B. Pinna, *Traffic estimation for the largest sources on a network, using packet sampling with limited storage*. Hewlett-Packard Laboratories, Technical Publications Department, 1992.

[277] I. T. Institute, "Control systems ports," https://github.com/ITI/ICS-Security-Tools/blob/master/protocols/PORTS.md.

[278] "nDPI," https://www.ntop.org/products/deep-packet-inspection/ndpi/.

[279] GreyNoise Intelligence, https://greynoise.io/.

[280] A. Banks, E. Briggs, K. Borgendale, and R. Gupta, "MQTT version 5.0," *OASIS Standard*, 2019.

[281] S. Andy, B. Rahardjo, and B. Hanindhito, "Attack scenarios and security analysis of MQTT communication protocol in IoT system," in *Proceedings of the International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, 2017, pp. 1–6.

[282] A. C. Panchal, V. M. Khadse, and P. N. Mahalle, "Security issues in IIoT: A comprehensive survey of attacks on IIoT and its countermeasures," in *Proceedings of the IEEE Global Conference on Wireless Computing and Networking (GCWCN)*, 2018, pp. 124–130.

[283] NIST, "Vulnerability Metrics," https://nvd.nist.gov/vuln-metrics/cvss.

[284] Rapid7, "The internet of gas station tank gauges," https://blog.rapid7.com/2015/01/22/the-internet-of-gas-station-tank-gauges/.

[285] K. Wilhoit and S. Hilt, "The little pump gauge that could: Attacks against gas pump monitoring systems," *Blackhat USA*, 2015.

[286] speedguide.net, "Port 10001 Details," https://www.speedguide.net/port.php?port=10001.

[287] E. López-Morales, C. Rubio-Medrano, A. Doupé, Y. Shoshitaishvili, R. Wang, T. Bao, and G.-J. Ahn, "Honeyplc: A next-generation honeypot for industrial control systems," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 279–291.

[288] D. Antonioli and N. O. Tippenhauer, "Minicps: A toolkit for security research on cps networks," in *Proceedings of the First ACM workshop on cyber-physical systems-security and/or privacy*, 2015, pp. 91–100.

[289] D. I. Buza, F. Juhász, G. Miru, M. Félegyházi, and T. Holczer, "Cryplh: Protecting smart energy systems from targeted attacks with a plc honeypot," in *International Workshop on Smart Grid Security*. Springer, 2014, pp. 181–192.

[290] K. Kołtyś and R. Gajewski, "Shape: A honeypot for electric power substation," *Journal of Telecommunications and Information Technology*, no. 4, pp. 37–43, 2015.

[291] K. Wilhoit and S. Hilt, "The gaspot experiment: Unexamined perils in using," Trend Micro Incorporated, Tech. Rep., 2015.

[292] W. O. Redwood, "Cyber physical system vulnerability research," Ph.D. dissertation, The Florida State University, 2016.

[293] A. Jicha, M. Patton, and H. Chen, "Scada honeypots: An in-depth analysis of conpot," in *2016 IEEE conference on intelligence and security informatics (ISI)*. IEEE, 2016, pp. 196–198.

[294] D. Antonioli, A. Agrawal, and N. O. Tippenhauer, "Towards high-interaction virtual ics honeypots-in-a-box," in *Proceedings of the 2nd ACM Workshop on Cyber-Physical Systems Security and Privacy*, 2016, pp. 13–22.

[295] S. Litchfield, D. Formby, J. Rogers, S. Meliopoulos, and R. Beyah, "Rethinking the honeypot for cyber-physical systems," *IEEE Internet Computing*, vol. 20, no. 5, pp. 9–17, 2016.

[296] J. Cao, W. Li, J. Li, and B. Li, "Dipot: A distributed industrial honeypot system," in *International Conference on Smart Computing and Communication*. Springer, 2017, pp. 300–309.

[297] F. Xiao, E. Chen, and Q. Xu, "S7commtrace: A high interactive honeypot for industrial control system based on s7 protocol," in *International Conference on Information and Communications Security*. Springer, 2017, pp. 412–423.

[298] G. Bernieri, M. Conti, and F. Pascucci, "Mimepot: a model-based honeypot for industrial control networks," in *IEEE International Conference on Systems, Man and Cybernetics (SMC)*, 2019, pp. 433–438.

[299] V. Pothamsetty and M. Franz. (2005) SCADA HoneyNet Project: Building Honeypots for Industrial Networks. Accessed: 01-07-2021. [Online]. Available: http://scadahoneynet.sourceforge.net/

[300] S. Soderi, R. Colelli, F. Turrin, F. Pascucci, and M. Conti, "Senecan: Secure key distribution over can through watermarking and jamming," *IEEE Transactions on Dependable and Secure Computing*, 2022.

[301] R. Baker and I. Martinovic, "Losing the car keys: Wireless phy-layer insecurity in EV charging," in *28th USENIX Security Symposium (USENIX Security 19)*. Santa Clara, CA: USENIX Association, Aug. 2019, pp. 407–424.

[302] P. R. Babu, B. Palaniswamy, A. G. Reddy, V. Odelu, and H. S. Kim, "A survey on security challenges and protocols of electric vehicle dynamic charging system," *Security and Privacy*, vol. 5, no. 3, 2022.

[303] M. Conti, D. Donadel, R. Poovendran, and F. Turrin, "Evexchange: A relay attack on electric vehicle charging system," in *European Symposium on Research in Computer Security*. Springer, 2022.

[304] A. Brighente, M. Conti, D. Donadel, and F. Turrin, "Evscout2.0: Electric vehicle profiling through charging profile," *ACM Transactions Cyber-Physical Systems*, sep 2022.

[305] K. Iehira, H. Inoue, and K. Ishida, "Spoofing attack using bus-off attacks against a specific ecu of the can bus," in *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2018, pp. 1–4.

[306] K. Koscher, S. Savage, F. Roesner, S. Patel, T. Kohno, A. Czeskis, D. McCoy, B. Kantor, D. Anderson, H. Shacham *et al.*, "Experimental security analysis of a modern automobile," in *2010 IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2010, pp. 447–462.

[307] M. Enev, A. Takakuwa, K. Koscher, and T. Kohno, "Automobile driver fingerprinting," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 1, pp. 34–50, 2016.

[308] A. Hazem and H. Fahmy, "Lcap-a lightweight can authentication protocol for securing in-vehicle networks," in *10th escar Embedded Security in Cars Conference, Berlin, Germany*, vol. 6, 2012.

[309] S. Woo, H. J. Jo, and D. H. Lee, "A practical wireless attack on the connected car and security protocol for in-vehicle can," *IEEE Transactions on intelligent transportation systems*, vol. 16, no. 2, pp. 993–1006, 2014.

[310] C. Beek and R. Samani. DEFCON – Connected Car Security. Accessed: 25-02-2021. [Online]. Available: https://www.mcafee.com/blogs/other-blogs/mcafee-labs/defcon-connected-car-security/

[311] R. Ross, V. Pillitteri, R. Graubart, D. Bodeau, and R. McQuaid, "Developing cyber resilient systems: A systems security engineering approach," NIST, Tech. Rep., 2019.

[312] G. Bella, P. Biondi, G. Costantino, and I. Matteucci, "Toucan: A protocol to secure controller area network," in *Proceedings of the ACM Workshop on Automotive Cybersecurity*, 2019, pp. 3–8.

[313] A.-I. Radu and F. D. Garcia, "Leia: A lightweight authentication protocol for can," in *European Symposium on Research in Computer Security*. Springer, 2016, pp. 283–300.

[314] P.-S. Murvay and B. Groza, "Tidal-can: Differential timing based intrusion detection and localization for controller area network," *IEEE Access*, vol. 8, pp. 68 895–68 912, 2020.

[315] M. Foruhandeh, Y. Man, R. Gerdes, M. Li, and T. Chantem, "Simple: Single-frame based physical layer identification for intrusion detection and prevention on in-vehicle networks," in *Proceedings of the 35th Annual Computer Security Applications Conference*, 2019, pp. 229–244.

[316] M. Kneib, O. Schell, and C. Huth, "Easi: Edge-based sender identification on resource-constrained platforms for automotive networks," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2020, pp. 1–16.

[317] B. W. Lampson, "A note on the confinement problem," *Commun. ACM*, vol. 16, no. 10, pp. 613–615, Oct. 1973.

[318] Z. Wu, Z. Xu, and H. Wang, "Whispers in the hyper-space: high-bandwidth and reliable covert channel attacks inside the cloud," *IEEE/ACM Transactions on Networking*, pp. 603–615, 2014.

[319] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, A. Fogh, J. Horn, S. Mangard, P. Kocher, D. Genkin *et al.*, "Meltdown: Reading kernel memory from user space," in *27th {USENIX} Security Symposium ({USENIX} Security 18)*, 2018, pp. 973–990.

[320] X. Ying, G. Bernieri, M. Conti, and R. Poovendran, "Tacan: Transmitter authentication through covert channels in controller area networks," in *Proceedings of the 10th ACM/IEEE International Conference on Cyber-Physical Systems*, 2019, pp. 23–34.

[321] B. Groza, L. Popa, and P.-S. Murvay, "Incanta-intrusion detection in controller area networks with time-covert authentication," in *Security and Safety Interplay of Intelligent Software Systems*. Springer, 2018, pp. 94–110.

[322] ——, "Canto-covert authentication with timing channels over optimized traffic flows for can," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 601–616, 2020.

[323] B. Groza, L. Popa, and P. Murvay, "Tricks—time triggered covert key sharing for controller area networks," *IEEE Access*, vol. 7, pp. 104 294–104 307, 2019.

[324] A. Mueller and T. Lothspeich, "Plug-and-secure communication for can," *CAN Newsletter*, no. 4, pp. 10–14, 2015.

[325] S. Soderi, L. Mucchi, M. Hämäläinen, A. Piva, and J. H. Iinatti, "Physical layer security based on spread-spectrum watermarking and jamming receiver." *Trans. Emerging Telecommunications Technologies*, vol. 28, no. 7, 2017.

[326] D. A. Wagner and S. M. Bellovin, "A "bump in the stack" encryptor for ms-dos systems," in *Proceedings of the 1996 Symposium on Network and Distributed System Security (SNDSS '96)*, ser. SNDSS '96. USA: IEEE Computer Society, 1996, p. 155.

[327] S. Soderi, "Acoustic-Based Security: A Key Enabling Technology for Wireless Sensor Network," *International Journal of Wireless Information Networks*, vol. 27, no. 1, pp. 45–59, nov 2019.

[328] ——, "Enhancing Security in 6G Visible Light Communications," in *2020 2nd 6G Wireless Summit (6G SUMMIT)*, 2020, pp. 1–5.

[329] I. O. for Standardization, "Road vehicles — Interchange of digital information — Controller area network (CAN) for high-speed communication," International Organization for Standardization, Standard ISO 11898:1993, 1993.

[330] ——, "Road vehicles — Controller area network (CAN) — Part 2: High-speed medium access unit," International Organization for Standardization, Standard ISO 11898-2:2016, 2016.

[331] ——, "Road vehicles — Controller area network (CAN) — Part 3: Low-speed, fault-tolerant, medium-dependent interface," International Organization for Standardization, Standard ISO 11898-3:2006, 2006.

[332] ——, "Road vehicles — Controller area network (CAN) — Part 1: Data link layer and physical signalling," International Organization for Standardization, Standard ISO 11898-1:2015, 2015.

[333] M. Hanspach and M. Goetz, "On covert acoustical mesh networks in air," *CoRR*, vol. abs/1406.1213, 2014. [Online]. Available: http://arxiv.org/abs/1406.1213

[334] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding-a survey," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1062–1078, 1999.

[335] X. Li, C. Yu, M. Hizlan, W.-T. Kim, and S. Park, "Physical layer watermarking of direct sequence spread spectrum signals," in *MILCOM 2013 - 2013 IEEE Military Communications Conference*, 2013, pp. 476–481.

[336] I. J. Cox, J. Kilian, F. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Transactions on Image Processing*, vol. 6, no. 12, pp. 1673–1687, Dec 1997.

[337] C. Shannon, "Communication theory of secrecy systems," *The Bell System Technical Journal*, vol. 28, no. 4, pp. 656–715, Oct 1949.

[338] A. Van Herrewege, D. Singelee, and I. Verbauwhede, "Canauth-a simple, backward compatible broadcast authentication protocol for can bus," in *ECRYPT Workshop on Lightweight Cryptography*, 2011, p. 20.

[339] H. Schweppe, Y. Roudier, B. Weyl, L. Apvrille, and D. Scheuermann, "Car2x communication: securing the last meter-a cost-effective approach for ensuring trust in car2x applications using in-vehicle symmetric cryptography," in *2011 IEEE Vehicular Technology Conference (VTC Fall)*. IEEE, 2011, pp. 1–5.

[340] B. Groza, S. Murvay, A. van Herrewege, and I. Verbauwhede, "Libra-can: A lightweight broadcast authentication protocol for controller area networks," in *Cryptology and Network Security*, J. Pieprzyk, A.-R. Sadeghi, and M. Manulis, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 185–200.

[341] R. Kurachi, Y. Matsubara, H. Takada, N. Adachi, Y. Miyashita, and S. Horihata, "Cacan-centralized authentication system in can (controller area network)," in *14th Int. Conf. on Embedded Security in Cars (ESCAR 2014)*, 2014.

[342] Q. Wang and S. Sawhney, "Vecure: A practical security framework to protect the can bus of vehicles," in *2014 International Conference on the Internet of Things (IOT)*. IEEE, 2014, pp. 13–18.

[343] S. Nürnberger and C. Rossow, "– vatican – vetted, authenticated can bus," in *Cryptographic Hardware and Embedded Systems – CHES 2016*, B. Gierlichs and A. Y. Poschmann, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 106–124.

[344] J. Van Bulck, J. T. Mühlberg, and F. Piessens, "Vulcan: Efficient component authentication and software isolation for automotive control networks," in *Proceedings of the 33rd Annual Computer Security Applications Conference*, ser. ACSAC 2017. New York, NY, USA: Association for Computing Machinery, 2017, p. 225–237.

[345] S. Fassak, Y. E. H. El Idrissi, N. Zahid, and M. Jedra, "A secure protocol for session keys establishment between ecus in the can bus," in *2017 International Conference on Wireless Networks and Mobile Communications (WINCOM)*. IEEE, 2017, pp. 1–6.

[346] B. Palaniswamy, S. Camtepe, E. Foo, and J. Pieprzyk, "An efficient authentication scheme for intra-vehicular controller area network," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3107–3122, 2020.

[347] Y. Xiao, S. Shi, N. Zhang, W. Lou, and Y. T. Hou, "Session key distribution made practical for can and can-fd message authentication," in *Annual Computer Security Applications Conference*, 2020, pp. 681–693.

[348] H. Mun, K. Han, and D. H. Lee, "Ensuring safety and security in can-based automotive embedded systems: A combination of design optimization and secure communication," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 7, pp. 7078–7091, 2020.

[349] B. Groza, L. Popa, and P.-S. Murvay, "Highly efficient authentication for can by identifier reallocation with ordered cmacs," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 6129–6140, 2020.

[350] T.-Y. Youn, Y. Lee, and S. Woo, "Practical sender authentication scheme for in-vehicle can with efficient key management," *IEEE Access*, vol. 8, pp. 86 836–86 849, 2020.

[351] M. D. Pesé, J. W. Schauer, J. Li, and K. G. Shin, "S2-can: Sufficiently secure controller area network," in *Annual Computer Security Applications Conference*, 2021, pp. 425–438.

[352] T. Limbasiya, A. Ghosal, and M. Conti, "Autosec: Secure automotive data transmission scheme for in-vehicle networks," in *23rd International Conference on Distributed Computing and Networking*, 2022, pp. 208–216.

[353] H. J. Jo and W. Choi, "A survey of attacks on controller area networks and corresponding countermeasures," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[354] B. Groza, L. Popa, P.-S. Murvay, Y. Elovici, and A. Shabtai, "{CANARY}-a reactive defense mechanism for controller area networks based on active {RelaYs}," in *30th USENIX Security Symposium (USENIX Security 21)*, 2021, pp. 4259–4276.

[355] S. U. Sagong, X. Ying, A. Clark, L. Bushnell, and R. Poovendran, "Cloaking the clock: emulating clock skew in controller area networks," in *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 2018, pp. 32–42.

[356] J. Zhou, P. Joshi, H. Zeng, and R. Li, "Btmonitor: Bit-time-based intrusion detection and attacker identification in controller area network," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 18, no. 6, pp. 1–23, 2019.

[357] W. Choi, K. Joo, H. J. Jo, M. C. Park, and D. H. Lee, "Voltageids: Low-level communication characteristics for automotive intrusion detection system," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 8, pp. 2114–2129, 2018.

[358] X. Ying, G. Bernieri, M. Conti, L. Bushnell, and R. Poovendran, "Covert channel-based transmitter authentication in controller area networks," *IEEE Transactions on Dependable and Secure Computing*, 2021.

[359] S. Soderi and R. De Nicola, "6g networks physical layer security using rgb visible light communications," *IEEE Access*, vol. 10, pp. 5482–5496, 2022.

[360] J. G. Proakis, *Digital communications*, 4th ed. Boston: McGraw-Hill, 2000. [Online]. Available: http://www.loc.gov/catdir/description/mh021/00025305.html

[361] H. Malvar and D. Florencio, "Improved spread spectrum: a new modulation technique for robust watermarking," *IEEE Transactions on Signal Processing*, vol. 51, no. 4, pp. 898–905, Apr 2003.

[362] A. Palanca, E. Evenchick, F. Maggi, and S. Zanero, "A stealth, selective, link-layer denial-of-service attack against automotive networks," in *Detection of Intrusions and Malware, and Vulnerability Assessment*, M. Polychronakis and M. Meier, Eds. Cham: Springer International Publishing, 2017, pp. 185–206.

[363] S. Jain and J. Guajardo, "Physical layer group key agreement for automotive controller area networks," in *International Conference on Cryptographic Hardware and Embedded Systems*. Springer, 2016, pp. 85–105.

[364] S. Vanderhallen, J. Van Bulck, F. Piessens, and J. T. Mühlberg, "Robust authentication for automotive control networks through covert channels," *Computer Networks*, vol. 193, p. 108079, 2021.

[365] The Guardian. (2021) "Electric vehicles on world's roads expected to increase to 145m by 2030". [Accessed: 20-05-2021]. [Online]. Available: https://www.theguardian.com/environment/2021/apr/29/electric-vehicles-on-worlds-roads-expected-to-increase-to-145m-by-2030

[366] J. Ramey. (2021) "Honda Will Go Electric- and Fuel Cell-Only by 2040". [Accessed: 20-05-2021]. [Online]. Available: https://www.autoweek.com/news/green-cars/a36230978/honda-electric-and-fuel-cell-by-2040/

[367] U. S. E. P. Agency. (2016) "Sources of Greenhouse Gas Emissions". [Accessed: 20-05-2021]. [Online]. Available: https://www.epa.gov/ghgemissions/sources-greenhouse-gas-emissions

[368] E. Sortomme and M. A. El-Sharkawi, "Optimal charging strategies for unidirectional vehicle-to-grid," *IEEE Transactions on Smart Grid*, vol. 2, no. 1, pp. 131–138, 2011.

[369] J. Antoun, M. E. Kabir, B. Moussa, R. Atallah, and C. Assi, "A Detailed Security Assessment of the EV Charging Ecosystem," *IEEE Network*, vol. 34, no. 3, pp. 200–207, 2020.

[370] M. A. Mustafa, N. Zhang, G. Kalogridis, and Z. Fan, "Smart electric vehicle charging: Security analysis," *2013 IEEE PES Innovative Smart Grid Technologies Conference, ISGT 2013*, no. February, p. 7, 2013.

[371] C. Sun, T. Li, S. H. Low, and V. O. Li, "Classification of electric vehicle charging time series with selective clustering," *Electric Power Systems Research*, vol. 189, p. 106695, 2020.

[372] L. Attanasio, M. Conti, D. Donadel, and F. Turrin, "MiniV2G: An Electric Vehicle Charging Emulator," in *Proceedings of the 7th ACM Cyber-Physical System Security Workshop (CPSS '21), June 7, 2021, Virtual Event, Hong Kong*, vol. 1, no. 1. ACM, 2021.

[373] P. Van den Bossche, *Electric Vehicle Charging Infrastructure*. Elsevier B.V, 2010.

[374] K. Clement-Nyns, E. Haesen, and J. Driesen, "The impact of vehicle-to-grid on the distribution grid," *Electric Power Systems Research*, vol. 81, no. 1, pp. 185–192, 2011.

[375] L. Noel, G. Zarazua de Rubens, J. Kester, and B. K. Sovacool, "The Technical Challenges to V2G," in *Vehicle-to-Grid*. Cham: Springer International Publishing, 2019, pp. 65–89.

[376] Z. Garofalaki, D. Kosmanos, S. Moschoyiannis, D. Kallergis, and C. Douligeris, "Electric vehicle charging: A survey on the security issues and challenges of the open charge point protocol (ocpp)," *IEEE Communications Surveys & Tutorials*, 2022.

[377] C. Alcaraz, J. Lopez, and S. Wolthusen, "Ocpp protocol: Security threats and challenges," *IEEE Transactions on Smart Grid*, vol. 8, no. 5, pp. 2452–2459, 2017.

[378] J. E. Rubio, C. Alcaraz, and J. Lopez, "Addressing security in ocpp: Protection against man-in-the-middle attacks," in *2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*. IEEE, 2018, pp. 1–5.

[379] I. O. for Standardization, "Road vehicles — Vehicle-to-Grid Communication Interface — Part 1: General information and use-case definition," International Organization for Standardization, Geneva, CH, Standard, Mar. 2019.

[380] ——, "Road vehicles — Vehicle-to-Grid Communication Interface — Part 2: Network and application protocol requirements," International Organization for Standardization, Geneva, CH, Standard, Mar. 2014.

[381] L. Buschlinger, M. Springer, and M. Zhdanova, "Plug-and-patch: Secure value added services for electric vehicle charging," *ACM International Conference Proceeding Series*, 2019.

[382] I. E. Commission, "Plugs, socket-outlets, vehicle connectors and vehicle inlets - Conductive charging of electric vehicles - Part 1: General requirements," International Electrotechnical Commission, Geneva, CH, Standard, 2014.

[383] P. Klapwijk and L. Driessen-Mutters, "Exploring the public key infrastructure for ISO 15118 in the EV charging ecosystem," ElaadNL, Tech. Rep., 2018.

[384] G. P. Hancke, K. E. Mayes, and K. Markantonakis, "Confidence in smart token proximity: Relay attacks revisited," *Computers and Security*, vol. 28, no. 7, pp. 615–627, 2009.

[385] K. Bao, H. Valev, M. Wagner, and H. Schmeck, "A threat analysis of the vehicle-to-grid charging protocol ISO 15118," *Computer Science - Research and Development*, vol. 33, no. 1-2, pp. 3–12, 2018.

[386] S. Lee, Y. Park, H. Lim, and T. Shon, "Study on analysis of security vulnerabilities and countermeasures in iso/iec 15118 based electric vehicle charging technology," *2014 International Conference on IT Convergence and Security, ICITCS 2014*, pp. 6–9, 2014.

[387] M. ElKashlan, H. Aslan, M. Said Elsayed, A. D. Jurcut, and M. A. Azer, "Intrusion detection for electric vehicle charging systems (evcs)," *Algorithms*, vol. 16, no. 2, p. 75, 2023.

[388] J. Cumplido, C. Alcaraz, and J. Lopez, "Collaborative anomaly detection system for charging stations," in *European Symposium on Research in Computer Security*. Springer, 2022, pp. 716–736.

[389] C. Höfer, J. Petit, R. Schmidt, and F. Kargl, "POPCORN: Privacy-preserving charging for emobility," *Proceedings of the ACM Conference on Computer and Communications Security*, pp. 37–48, 2013.

[390] A. Fuchs, D. Kern, C. Krauß, and M. Zhdanova, "HIP: HSM-based Identities for Plug-and-Charge," in *Proceedings of the 15th International Conference on Availability, Reliability and Security*. New York, NY, USA: ACM, aug 2020, pp. 1–6.

[391] L. Francis, G. Hancke, K. Mayes, and K. Markantonakis, "Practical NFC Peer-to-Peer Relay Attack Using Mobile Phones," in *Radio Frequency Identification: Security and Privacy Issues*, S. B. Ors Yalcin, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 35–49.

[392] D. Cavdar and E. Tomur, "A practical NFC relay attack on mobile devices using card emulation mode," *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2015 - Proceedings*, no. May, pp. 1308–1312, 2015.

[393] M. Casagrande, M. Conti, and E. Losiouk, "Contact tracing made un-relay-able," in *Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy*, 2021, pp. 221–232.

[394] A. Francillon, B. Danev, and S. Capkun, "Relay attacks on passive keyless entry and start systems in modern cars," in *Proceedings of the Network and Distributed System Security Symposium (NDSS)*. Eidgenössische Technische Hochschule Zürich, Department of Computer Science, 2011.

[395] A. S. Sani, D. Yuan, E. Bertino, and Z. Y. Dong, "Crypto-chain: A relay resilience framework for smart vehicles," in *Annual Computer Security Applications Conference*, ser. ACSAC. New York, NY, USA: ACM, 2021, p. 439–454.

[396] Fastned. (2020) "Autocharge". [Accessed: 19-11-2020]. [Online]. Available: https://support.fastned.nl/hc/en-gb/articles/115012747127-Autocharge-

[397] C. Unal, E. Yirik, E. Ünal, M. Cuma, B. Onur, and M. Tümay, "A review of charging technologies for commercial electric vehicles," *International Journal Of Advances On Automotive And Technology*, pp. 61–70, 01 2018.

[398] S. Pelletier, O. Jabali, G. Laporte, and M. Veneroni, "Battery degradation and behaviour for electric vehicles: Review and numerical analyses of several models," *Transportation Research Part B: Methodological*, vol. 103, pp. 158–187, 2017.

[399] C. Wu, J. Sun, C. Zhu, Y. Ge, and Y. Zhao, "Research on overcharge and overdischarge effect on lithium-ion batteries," in *2015 IEEE Vehicle Power and Propulsion Conference (VPPC)*, 2015, pp. 1–6.

[400] R. R. Fontes, S. Afzal, S. H. Brito, M. A. Santos, and C. E. Rothenberg, "Mininet-wifi: Emulating software-defined wireless networks," in *2015 11th International Conference on Network and Service Management (CNSM)*. IEEE, 2015, pp. 384–389.

[401] V. Clarity. (2020) "Reference Implementation Supporting the Evolution of the Vehicle-2-Grid communication interface ISO 15118". [Accessed: 14-05-2021]. [Online]. Available: https://v2g-clarity.com/rise-v2g/

[402] S. Hemminger. (2012) "bridge - show / manipulate bridge addresses and devices". [Accessed: 16-07-2021]. [Online]. Available: https://man7.org/linux/man-pages/man8/bridge.8.html

[403] M. Mahalingam, D. G. Dutt, K. Duda, P. Agarwal, L. Kreeger, T. Sridhar, M. Bursell, and C. Wright, "Virtual extensible local area network (vxlan): A framework for overlaying virtualized layer 2 networks over layer 3 networks." *RFC*, vol. 7348, pp. 1–22, 2014.

[404] S. Brands and D. Chaum, "Distance-bounding protocols," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 765 LNCS, pp. 344–359, 1994.

[405] S. Drimer and S. J. Murdoch, "Keep your enemies close: Distance bounding against smartcard relay attacks," *16th USENIX Security Symposium*, pp. 87–102, 2007.

[406] M. Henzl, P. Hanacek, and M. Kacic, "Preventing real-world relay attacks on contactless devices," *Proceedings - International Carnahan Conference on Security Technology*, no. October, pp. 13–18, 2014.

[407] C. Thorpe, J. Tobin, and L. Murphy, "An ISO/IEC 7816-4 Application Layer Approach to Mitigate Relay Attacks on near Field Communication," *IEEE Access*, vol. 8, pp. 190 108–190 117, 2020.

[408] T. Yang, L. Kong, W. Xin, J. Hu, and Z. Chen, "Resisting relay attacks on vehicular Passive Keyless Entry and start systems," in *2012 9th International Conference on Fuzzy Systems and Knowledge Discovery*. IEEE, may 2012, pp. 2232–2236.

[409] N. O. Tippenhauer, H. Luecken, M. Kuhn, and S. Capkun, "UWB rapid-bit-exchange system for distance bounding," *Proceedings of the 8th ACM Conference on Security and Privacy in Wireless and Mobile Networks, WiSec 2015*, 2015.

[410] T. Korak and M. Hutter, "On the power of active relay attacks using custom-made proxies," *2014 IEEE International Conference on RFID*, pp. 126–133, 2014.

[411] M. Y. Chung, M. H. Jung, T. J. Lee, and Y. Lee, "Performance analysis of HomePlug 1.0 MAC with CSMA/CA," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 7, pp. 1411–1420, 2006.

[412] I. O. for Standardization, "Road vehicles — Vehicle to grid communication interface — Part 20: 2nd generation network layer and application layer requirements," International Organization for Standardization, Geneva, CH, Standard, 2021.

[413] A. Brighente, M. Conti, and I. Sadaf, "Tell me how you re-charge, i will tell you where you drove to: Electric vehicles profiling based on charging-current demand," in *European Symposium on Research in Computer Security*. Springer, 2021, pp. 651–667.

[414] J. Antoun, M. E. Kabir, B. Moussa, R. Atallah, and C. Assi, "A detailed security assessment of the ev charging ecosystem," *IEEE Network*, vol. 34, no. 3, pp. 200–207, 2020.

[415] H. Liu, R. Spolaor, F. Turrin, R. Bonafede, and M. Conti, "Usb powered devices: A survey of side-channel threats and countermeasures," *High-Confidence Computing*, vol. 1, no. 1, p. 100007, 2021.

[416] M. Conti, M. Nati, E. Rotundo, and R. Spolaor, "Mind the plug! laptop-user recognition through power consumption," in *Proceedings of the 2nd ACM International Workshop on IoT Privacy, Trust, and Security*, 2016, pp. 37–44.

[417] A. Molina-Markham, P. Shenoy, K. Fu, E. Cecchet, and D. Irwin, "Private memoirs of a smart meter," in *Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building*. New York, NY, USA: Association for Computing Machinery, 2010, p. 61–66.

[418] Q. Yang, P. Gasti, G. Zhou, A. Farajidavar, and K. S. Balagani, "On inferring browsing activity on smartphones via USB power analysis side-channel," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 5, pp. 1056–1066, 2016.

[419] P. Cronin, X. Gao, C. Yang, and H. Wang, "Charger-Surfing: Exploiting a power line Side-Channel for smartphone information leakage," in *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, Aug. 2021, pp.

681–698. [Online]. Available: https://www.usenix.org/conference/usenixsecurity21/presentation/cronin

[420] W. Kempton, J. Tomic, S. Letendre, A. Brooks, and T. Lipman, "Vehicle-to-grid power: battery, hybrid, and fuel cell vehicles as resources for distributed electric power in california," *UC Davis: Institute of Transportation Studies*, 2001.

[421] R. Gottumukkala, R. Merchant, A. Tauzin, K. Leon, A. Roche, and P. Darby, "Cyber-physical system security of vehicle charging stations," in *2019 IEEE Green Technologies Conference(GreenTech)*, 2019, pp. 1–5.

[422] L. Wang, Z. Qin, T. Slangen, P. Bauer, and T. van Wijk, "Grid impact of electric vehicle fast charging stations: Trends, standards, issues and mitigation measures-an overview," *IEEE Open Journal of Power Electronics*, vol. 2, pp. 56–74, 2021.

[423] C. Troepfer, "SAE electric vehicle conductive charge coupler, SAE J1772," Society of Automotive Engineers, Tech. Rep., 2009.

[424] Y.-s. Bai and C.-n. Zhang, "Experiments study on fast charge technology for lithium-ion electric vehicle batteries," in *2014 IEEE Conference and Expo Transportation Electrification Asia-Pacific (ITEC Asia-Pacific)*, 2014, pp. 1–6.

[425] H. Wu, G. K. H. Pang, K. L. Choy, and H. Y. Lam, "An optimization model for electric vehicle battery charging at a battery swapping station," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 2, pp. 881–895, 2017.

[426] F. Marra, G. Y. Yang, C. Træholt, E. Larsen, C. N. Rasmussen, and S. You, "Demand profile study of battery electric vehicle under different charging options," in *2012 IEEE Power and Energy Society General Meeting*, 2012, pp. 1–7.

[427] ThunderSky, "Instruction manual for LFP/LCP/LMP lithium power battery," Thunder Sky, Tech. Rep., 2007.

[428] Z. J. Lee, T. Li, and S. H. Low, "Acn-data: Analysis and applications of an open ev charging dataset," in *Proceedings of the Tenth ACM International Conference on Future Energy Systems*, ser. e-Energy '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 139–149.

[429] N.Rajesh Kumar and J.Uday Kumar, "A Spatial Mean and Median Filter For Noise Removal in Digital Images," *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, vol. 4, no. 1, pp. 246–253, 2015.

[430] H. Deng, G. Runger, E. Tuv, and M. Vladimir, "A time series forest for classification and feature extraction," *Information Sciences*, vol. 239, pp. 142–153, 2013.

[431] M. Christ, A. W. Kempa-Liehr, and M. Feindt, "Distributed and parallel time series feature extraction for industrial big data applications," *arXiv preprint arXiv:1610.07717*, May 2016.

[432] M. Barandas, D. Folgado, L. Fernandes, S. Santos, M. Abreu, P. Bota, H. Liu, T. Schultz, and H. Gamboa, "Tsfel: Time series feature extraction library," *SoftwareX*, vol. 11, p. 100456, 2020.

[433] F. Mörchen, "Time series feature extraction for data mining using dwt and dft," 2003.

[434] M. Christ, N. Braun, J. Neuffer, and A. W. Kempa-Liehr, "Time Series FeatuRe Extraction on basis of Scalable Hypothesis tests (tsfresh – A Python package)," *Neurocomputing*, vol. 307, pp. 72–77, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231218304843

[435] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[436] T. Schreiber and A. Schmitz, "Discrimination power of measures for nonlinearity in a time series," *Phys. Rev. E*, vol. 55, pp. 5443–5447, May 1997. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevE.55.5443

[437] C. Roth, N. Thanh Dinh, and D. Kesdoğan, "Crowdabout: Using vehicles as sensors to improve map data for its," in *2020 Seventh International Conference on Social Networks Analysis, Management and Security (SNAMS)*, 2020, pp. 1–8.

[438] P. K. Bera and C. Isik, "Identification of stable and unstable power swings using pattern recognition," in *2021 IEEE Green Technologies Conference (GreenTech)*, 2021, pp. 286–291.

[439] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters*, vol. 9, no. 3, pp. 293–300, 1999.

[440] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, "Knn model-based approach in classification," in *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*, R. Meersman, Z. Tari, and D. C. Schmidt, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 986–996.

[441] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and regression trees*. CRC press, 1984.

[442] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, *Applied logistic regression*. John Wiley & Sons, 2013, vol. 398.

[443] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[444] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[445] C. Ferri, J. Hernández-Orallo, and R. Modroiu, "An experimental comparison of performance measures for classification," *Pattern Recognition Letters*, vol. 30, no. 1, pp. 27–38, 2009.

[446] A. Barré, B. Deguilhem, S. Grolleau, M. Gérard, F. Suard, and D. Riu, "A review on lithium-ion battery ageing mechanisms and estimations for automotive applications," *Journal of Power Sources*, vol. 241, pp. 680–689, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.jpowsour.2013.05.040

[447] J. Vetter, P. Novák, M. R. Wagner, C. Veit, K.-C. Möller, J. O. Besenhard, M. Winter, M. Wohlfahrt-Mehrens, C. Vogler, and A. Hammouche, "Ageing mechanisms in lithium-ion batteries," *Journal of Power Sources*, vol. 147, no. 1, pp. 269–281, 2005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378775305000832

[448] G. Kalogridis, C. Efthymiou, S. Z. Denic, T. A. Lewis, and R. Cepeda, "Privacy for smart meters: Towards undetectable appliance load signatures," in *2010 First IEEE International Conference on Smart Grid Communications*, 2010, pp. 232–237.

[449] M. Shateri, F. Messina, P. Piantanida, and F. Labeau, "Real-time privacy-preserving data release for smart meters," *IEEE Transactions on Smart Grid*, vol. 11, no. 6, pp. 5174–5183, 2020.

[450] R. Spolaor, H. Liu, F. Turrin, M. Conti, and X. Cheng, "Plug and power: Fingerprinting usb powered peripherals via power side-channel," in *IEEE International Conference on Computer Communications*. IEEE, 2023.

[451] A. Brighente, M. Conti, D. Donadel, and F. Turrin, "Hyperloop: A cybersecurity perspective," *arXiv preprint arXiv:2209.03095*, 2022.

[452] Statista, "Smartphones - Statistics & Facts," accessed: 07-02-2021. [Online]. Available: https://www.statista.com/topics/840/smartphones/

[453] Tom's Guide, "Best phone battery life in 2021: The longest lasting smartphones," accessed: 07-02-2021. [Online]. Available: https://www.tomsguide.com/us/smartphones-best-battery-life,review-2857.html

[454] W. Meng, W. H. Lee, S. R. Murali, and S. P. Krishnan, "JuiceCaster: Towards automatic juice filming attacks on smartphones," *Journal of Network and Computer Applications*, vol. 68, pp. 201–212, 2016.

[455] W. Meng, W. H. Lee, Z. Liu, C. Su, and Y. Li, "Evaluating the Impact of Juice Filming Charging Attack in Practical Environments," in *Information Security and Cryptology – ICISC 2017*, H. Kim and D.-C. Kim, Eds. Cham: Springer International Publishing, 2018, pp. 327–338.

[456] M. A. Al Faruque, S. R. Chhetri, A. Canedo, and J. Wan, "Acoustic side-channel attacks on additive manufacturing systems," in *2016 ACM/IEEE 7th international conference on Cyber-Physical Systems (ICCPS)*. IEEE, 2016, pp. 1–10.

[457] S. Cecconello, A. Compagno, M. Conti, D. Lain, and G. Tsudik, "Skype & type: Keyboard eavesdropping in voice-over-ip," *ACM Trans. Priv. Secur.*, vol. 22, no. 4, Dec. 2019. [Online]. Available: https://doi.org/10.1145/3365366

[458] R. Spolaor, L. Abudahi, V. Moonsamy, M. Conti, and R. Poovendran, "No free charge theorem: A covert channel via USB charging cable on mobile devices," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10355 LNCS. Springer, Cham, 2017, pp. 83–102.

[459] South China Morning Post, "Police warn shared power banks could transmit malware, but the industry continues to thrive in China," accessed: 07-02-2021. [Online]. Available: https://www.scmp.com/tech/gear/article/3112824/police-warn-shared-power-banks-could-transmit-viruses-industry-continues

[460] D. V. Pham, A. Syed, and M. N. Halgamuge, "Universal serial bus based software attacks and protection solutions," *digital investigation*, vol. 7, no. 3-4, pp. 172–184, 2011.

[461] C. Paoli, "Microsoft Releases Security Update for Autorun Vulnerability," accessed: 07-02-2021. [Online]. Available: https://redmondmag.com/articles/2011/02/10/update-for-autorun-vulnerability.aspx

[462] N. Nissim, R. Yahalom, and Y. Elovici, "USB-based attacks," *Computers and Security*, vol. 70, pp. 675–688, sep 2017.

[463] J. Tian, N. Scaife, D. Kumar, M. Bailey, A. Bates, and K. Butler, "SoK: 'Plug & Pray' Today-Understanding USB Insecurity in Versions 1 Through C," in *Proceedings of the IEEE Symposium on Security and Privacy*, vol. 2018-May. Institute of Electrical and Electronics Engineers Inc., jul 2018, pp. 1032–1047.

[464] L. Cai, S. Machiraju, and H. Chen, "Defending against sensor-sniffing attacks on mobile phones," in *Proceedings of the 1st ACM workshop on Networking, systems, and applications for mobile handhelds*, 2009, pp. 31–36.

[465] R. Spreitzer, V. Moonsamy, T. Korak, and S. Mangard, "Systematic classification of side-channel attacks: A case study for mobile devices," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1, pp. 465–488, 2017.

[466] M. Conti, Q. Q. Li, A. Maragno, and R. Spolaor, "The dark side(-channel) of mobile devices: A survey on network traffic analysis," *IEEE Communications Surveys Tutorials*, vol. 20, no. 4, pp. 2658–2713, 2018.

[467] Compaq, Digital Equipment Corporation, IBM PC Company, Intel, Microsoft, NEC, and Northern Telecom, "Universal serial bus specification - revision 1.0," 1996.

[468] Compaq, Intel, Microsoft, and NEC, "Universal serial bus specification, revision 1.1," 1998.

[469] Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, and Philips, "Universal serial bus specification - revision 2.0," 2000.

[470] Hewlett-Packard, Intel and Microsoft, NEC, ST-NXP Wireless, and Texas Instruments, "Universal serial bus 3.0 specification - revision 2.0," 2008.

[471] Hewlett-Packard Inc., Intel Corporation, Microsoft Corporation, Renesas Corporation, ST-Ericsson, and Texas Instruments, "Universal serial bus 3.1 specification - revision 1.0," 2013.

[472] Apple Inc., Hewlett-Packard Inc., Intel Corporation, Microsoft Corporation, Renesas Corporation, STMicroelectronics, and Texas Instruments, "Universal serial bus 3.2 specification - revision 1.0," 2017.

[473] ——, "Universal Serial Bus 4 (USB4TM) Specification - Revision 1.0," 2019.

[474] B. Bencsáth, G. Pék, L. Buttyán, and M. Felegyhazi, "The cousins of stuxnet: Duqu, flame, and gauss," *Future Internet*, vol. 4, no. 4, pp. 971–1003, 2012.

[475] S. Shin, G. Gu, N. Reddy, and C. P. Lee, "A large-scale empirical study of conficker," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 676–690, 2011.

[476] M. Guri, M. Monitz, and Y. Elovici, "Usbee: Air-gap covert-channel via electromagnetic emission from usb," in *2016 14th Annual Conference on Privacy, Security and Trust (PST)*. IEEE, 2016, pp. 264–268.

[477] H. Peng and M. Payer, "Usbfuzz: A framework for fuzzing USB drivers by device emulation," in *29th USENIX Security Symposium (USENIX Security 20)*, 2020, pp. 2559–2575.

[478] D. Spill, "Usbproxy-an open and affordable usb man in the middle device," *2014 ShmooCon Proceedings*, 2014.

[479] Y. Su, D. Genkin, D. Ranasinghe, and Y. Yarom, "USB Snooping Made Easy : Crosstalk Leakage Attacks on USB Hubs," in *Proceedings of the USENIX Conference on Security Symposium*. USENIX Association, 2017, pp. 1145–1161. [Online]. Available: https://www.usenix.org/system/files/conference/usenixsecurity17/sec17-antonakakis.pdf

[480] O. Angelopoulou, S. Pourmoafi, A. Jones, and G. Sharma, "Killing your device via your usb port." in *HAISA*, 2019, pp. 61–72.

[481] USB 3.0 Promoter Group, "Universal serial bus type-c authentication specification - revision 1.0," 2016.

[482] ——, "Universal serial bus type-c authentication specification - revision 1.0 with ecn and errata," 2019.

[483] USB Implementers Forum, "Uon-the-go supplement to the usb 2.0 specification - revision 1.0a," 2003.

[484] V. F. Taylor, R. Spolaor, M. Conti, and I. Martinovic, "Robust smartphone app identification via encrypted network traffic analysis," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 1, pp. 63–78, 2018.

[485] R. Spolaor, B. Riccardo, M. Veelasha, and M. Conti, "No Free Charge Theorem 2.0: How to Steal Private Information from a Mobile Device Using a Powerbank, BlackHat Europe 2018 Briefings," 2018, accessed: 07-02-2021. [Online]. Available: https://www.blackhat.com/eu-18/briefings/schedule/index.html#no-free-charge-theorem--how-to-steal-private-information-from-a-mobile-device-using-a-powerbank-12630

[486] A. Acar, H. Fereidooni, T. Abera, A. K. Sikder, M. Miettinen, H. Aksu, M. Conti, A.-R. Sadeghi, and S. Uluagac, "Peek-a-boo: I see your smart home activities, even encrypted!" in *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2020, pp. 207–218.

[487] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-end encrypted traffic classification with one-dimensional convolution neural networks," in *2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*. IEEE, 2017, pp. 43–48.

[488] Veloxity team, "Phone battery statistics across major us cities," 2015, accessed: 07-02-2021. [Online]. Available: https://veloxity.us/phone-battery-statistics/

[489] W. Lianzhang, "Phone Chargers: China's Latest Sharing Economy Fad, Sixth Tone," 2017, accessed: 07-02-2021. [Online]. Available: http://www.sixthtone.com/news/2182/phone-chargers-chinas-latest-sharing-economy-fad

[490] L. Mack, "Power up: A guide to us airport charging stations, Cheap Flights," 2015, accessed: 07-02-2021. [Online]. Available: https://www.cheapflights.com/news/power-up-a-guide-to-us-airport-charging-stations

[491] S. Larson, "Please stop charging your phone in public ports, cnn business," 2017, accessed: 07-02-2021. [Online]. Available: https://money.cnn.com/2017/02/15/technology/public-ports-charging-bad-stop/index.html

[492] W. Meng, W. H. Lee, S. R. Murali, and S. P. T. Krishnan, "Charging Me and I Know Your Secrets!: Towards Juice Filming Attacks on Smartphones," in *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security*, ser. CPSS '15. New York, NY, USA: ACM, 2015, pp. 89–98. [Online]. Available: http://doi.acm.org/10.1145/2732198.2732205

[493] Y. Oren and A. Shamir, "How not to protect pcs from power analysis," *Rump Session, Crypto*, vol. 2006, 2006.

[494] M. Vuagnoux and S. Pasini, "Compromising electromagnetic emanations of wired and wireless keyboards." in *USENIX security symposium*, 2009, pp. 1–16.

[495] D. Genkin, I. Pipman, and E. Tromer, "Get your hands off my laptop: Physical side-channel key-extraction attacks on pcs," *Journal of Cryptographic Engineering*, vol. 5, no. 2, pp. 95–112, 2015.

[496] W. Meng, F. Fei, W. Li, and M. H. Au, "Harvesting Smartphone Privacy Through Enhanced Juice Filming Charging Attacks," in *Information Security*, P. Q. Nguyen and J. Zhou, Eds. Cham: Springer International Publishing, 2017, pp. 291–308.

[497] P. Belgarric, P.-A. Fouque, G. Macario-Rat, and M. Tibouchi, "Side-channel analysis of weierstrass and koblitz curve ecdsa on android smartphones," in *Cryptographers' Track at the RSA Conference*. Springer, 2016, pp. 236–252.

[498] D. Genkin, L. Pachmanov, I. Pipman, E. Tromer, and Y. Yarom, "Ecdsa key extraction from mobile devices via nonintrusive physical side channels," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 1626–1638.

[499] D.-j. Sim, H. S. Lee, J.-G. Yook, and K. Sim, "Measurement and analysis of the compromising electromagnetic emanations from usb keyboard," in *2016 Asia-Pacific International Symposium on Electromagnetic Compatibility (APEMC)*, vol. 1. IEEE, 2016, pp. 518–520.

[500] Q. Yang, P. Gasti, G. Zhou, A. Farajidavar, and K. S. Balagani, "On Inferring Browsing Activity on Smartphones via USB Power Analysis Side-Channel," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 5, pp. 1056–1066, 2017.

[501] Q. Yang, P. Gasti, K. Balagani, Y. Li, and G. Zhou, "USB side-channel attack on Tor," *Computer Networks*, vol. 141, pp. 57–66, 2018. [Online]. Available: https://doi.org/10.1016/j.comnet.2018.05.018

[502] P. Cronin, X. Gao, C. Yang, and H. Wang, "{Charger-Surfing}: Exploiting a power line {Side-Channel} for smartphone information leakage," in *30th USENIX Security Symposium (USENIX Security 21)*, 2021, pp. 681–698.

[503] M. A. Heald and J. B. Marion, *Classical electromagnetic radiation*. Courier Corporation, 2012.

[504] R. Paavilainen, "Method and device for signal protection," Apr. 8 2008, uS Patent 7,356,626.

[505] L. Jiang, W. Meng, Y. Wang, C. Su, and J. Li, "Exploring Energy Consumption of Juice Filming Charging Attack on Smartphones: A Pilot Study," in *Network and System Security*, Z. Yan, R. Molva, W. Mazurczyk, and R. Kantola, Eds. Cham: Springer International Publishing, 2017, pp. 199–213.

[506] W. Meng, L. Jiang, K. K. R. Choo, Y. Wang, and C. Jiang, "Towards detection of juice filming charging attacks via supervised CPU usage analysis on smartphones," *Computers and Electrical Engineering*, vol. 78, pp. 230–241, 2019.

[507] N. Farhi, N. Nissim, and Y. Elovici, "Malboard: A novel user keystroke impersonation attack and trusted detection framework based on side-channel analysis," *Computers and Security*, vol. 85, pp. 240–269, aug 2019.

[508] I. Barankova, U. Mikhailova, and G. Lukyanov, "Software development and hardware means of hidden usb-keylogger devices identification," *Journal of Physics: Conference Series*, vol. 1441, pp. 12–32, 01 2020.

[509] O. A. Ibrahim, S. Sciancalepore, G. Oligeri, and R. D. Pietro, "Magneto: Fingerprinting usb flash drives via unintentional magnetic emissions," *ACM Transactions on Embedded Computing Systems*, vol. 20, no. 1, Dec. 2020. [Online]. Available: https://doi.org/10.1145/3422308

[510] R. Matovu, A. Serwadda, A. V. Bilbao, and I. Griswold-Steiner, "Defensive charging: Mitigating power side-channel attacks on charging smartphones," in *Proceedings of the Tenth ACM Conference on Data and Application Security and Privacy*, ser. CODASPY '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 179–190. [Online]. Available: https://doi.org/10.1145/3374664.3375732

[511] W. Meng, L. Jiang, Y. Wang, J. Li, J. Zhang, and Y. Xiang, "JFCGuard: Detecting juice filming charging attack via processor usage analysis on smartphones," *Computers and Security*, vol. 76, pp. 252–264, 2018. [Online]. Available: https://doi.org/10.1016/j.cose.2017.11.012

[512] Q. Zhang, F. Li, and Y. Wang, "Mobile crowd wireless charging toward rechargeable sensors for internet of things," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 5337–5347, 2018.

[513] Wirelss Power Consortium, "Wirelss power consortium memberlist," accessed: 07-02-2021. [Online]. Available: https://www.wirelesspowerconsortium.com/members/

[514] Wireless Power Consortium and others, "System description wireless power transfer," *Volume I: Low Power, Part*, vol. 1, 2010.

[515] TechCrunch, "Apple reveals airpower wireless charging pad coming in 2018," accessed: 07-02-2021. [Online]. Available: https://techcrunch.com/2018/09/12/where-the-heck-is-apples-airpower-wireless-charging-mat/

[516] Xiaomi Blog, "Forget about cables and charging stands with revolutionary mi air charge technology," accessed: 07-02-2021. [Online]. Available: https://blog.mi.com/en/2021/01/29/forget-about-cables-and-charging-stands-with-revolutionary-mi-air-charge-technology/

[517] Larson, CNN Business. (2017) Please stop charging your phone in public ports. https://money.cnn.com/2017/02/15/technology/public-ports-charging-bad-stop/index.html. [Accessed: 07-06-2022].

[518] N. Nissim, R. Yahalom, and Y. Elovici, "USB-based attacks," *Computers & Security*, vol. 70, pp. 675–688, 2017.

[519] J. Tian, N. Scaife, D. Kumar, M. Bailey, A. Bates, and K. Butler, "Sok:" plug & pray" today–understanding USB insecurity in versions 1 through c," in *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2018, pp. 1032–1047.

[520] K. Suzaki, Y. Hori, K. Kobara, and M. Mannan, "Deviceveil: Robust authentication for individual USB devices using physical unclonable functions," in *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2019, pp. 302–314.

[521] P. Cronin, X. Gao, H. Wang, and C. Cotton, "Time-print: Authenticating USB flash drives with novel timing fingerprints," in *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2022.

[522] J. V. Monaco, "Device fingerprinting with peripheral timestamps," in *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2022.

[523] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," in *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ser. ICML'12. Madison, WI, USA: Omnipress, 2012, p. 1467–1474.

[524] O. A. Grigg, V. Farewell, and D. Spiegelhalter, "Use of risk-adjusted cusum and rsprtcharts for monitoring in medical contexts," *Statistical methods in medical research*, vol. 12, no. 2, pp. 147–170, 2003.

[525] A. S. La Cour, K. K. Afridi, and G. E. Suh, "Wireless charging power side-channel attacks," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 651–665.

[526] J. Liu, X. Zou, L. Zhao, Y. Tao, S. Hu, J. Han, and K. Ren, "Privacy leakage in wireless charging," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2022.

[527] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[528] Jason Brownlee. (2020) One-vs-Rest and One-vs-One for Multi-Class Classification. https://machinelearningmastery.com/one-vs-rest-and-one-vs-one-for-multi-class-classification/. [Accessed: 07-06-2022]. [Online]. Available: https://machinelearningmastery.com/one-vs-rest-and-one-vs-one-for-multi-class-classification/

[529] E. Musk, "Hyperloop alpha," *SpaceX: Hawthorne, CA, USA*, 2013.

[530] "Hyperloopitalia," [Accessed Sept. 2022]. [Online]. Available: https://hyperloopitalia.com/

[531] "Delft hyperloop," [Accessed Sept. 2022]. [Online]. Available: https://www.delfthyperloop.nl/

[532] "Hyperloop transportation technology," [Accessed Sept. 2022]. [Online]. Available: https://www.hyperlooptt.com/

[533] Reuters, "Hackers breach iran rail network, disrupt service," [Accessed Sept. 2022]. [Online]. Available: https://www.reuters.com/world/middle-east/hackers-breach-iran-rail-network-disrupt-service-2021-07-09/

[534] "Virgin hyperloop," [Accessed Sept. 2022]. [Online]. Available: https://virginhyperloop.com/

[535] M. Janić, "Estimation of direct energy consumption and co2 emission by high speed rail, transrapid maglev and hyperloop passenger transport systems," *International Journal of Sustainable Transportation*, vol. 15, no. 9, pp. 696–717, 2021.

[536] J. C. Chin and J. S. Gray, "Open-source conceptual sizing models for the hyperloop passenger pod," in *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2015, p. 1587.

[537] J. K. Nøland, "Prospects and challenges of the hyperloop transportation system: A systematic technology review," *IEEE Access*, vol. 9, pp. 28 439–28 458, 2021.

[538] A. Tavsanoglu, C. Briso, D. Carmena-Cabanillas, and R. B. Arancibia, "Concepts of hyperloop wireless communication at 1200 km/h: 5g, wi-fi, propagation, doppler and handover," *Energies*, vol. 14, no. 4, p. 983, 2021.

[539] W. Hedhly, O. Amin, B. Shihada, and M.-S. Alouini, "Hyperloop communications: Challenges, advances, and approaches," *IEEE Open Journal of the Communications Society*, vol. 2, pp. 2413–2435, 2021.

[540] N. Singh, J. Karhade, I. Bhattacharya, P. Saraf, P. Kattamuri, and A. M. Parimi, "Onboard electrical, electronics and pose estimation system for hyperloop pod design," in *2021 7th International Conference on Control, Automation and Robotics (ICCAR)*. IEEE, 2021, pp. 223–230.

[541] J. Zhang, L. Liu, B. Han, Z. Li, T. Zhou, K. Wang, D. Wang, and B. Ai, "Concepts on train-to-ground wireless communication system for hyperloop: Channel, network architecture, and resource management," *Energies*, vol. 13, no. 17, p. 4309, 2020.