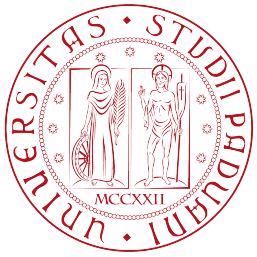


UNIVERSITY OF PADUA

PH.D. SCHOOL IN BRAIN, MIND AND COMPUTER SCIENCE
Curriculum in Computer Science and Innovation for Societal Challenges
XXXVII Cycle

Securing Modern Vehicles: Electric Charging and In-Vehicle Communication Protocols

---

Candidate:
Denis DONADEL

Supervisor:
Prof. Mauro CONTI
University of Padua

Co-Supervisor:
Prof. Anna SPAGNOLLI
University of Padua

Life is for living.

C. O.

Abstract

In the last decades, the massive size and price reduction of computationally capable devices have increased the spread of the so-called Cyber-Physical Systems (CPSs). These connected devices employ computing capacity together with sensing and actuating functions. Vehicles represent a widely spread and extremely complex example of CPS, where modern cars are equipped with more than 100 different microcomputers to control each vehicle's component and communicate with other devices. Although these technologies unquestionably improve the driving experience and its safety, they also expand the attack surface, which malicious entities may exploit to compromise the vehicle. In recent years, researchers have proved that several vehicle components are vulnerable to cyberattacks, and many novel technologies under development may introduce new security and privacy concerns. This dissertation offers an analysis of two different aspects of modern vehicles from a security point of view, discovering and mitigating some security vulnerabilities.

In the first part, we investigate the new security issues introduced with the spread of electric vehicles. We analyzed the new threats born from the connection of vehicles to the smart grid for charging purposes. We found a potential issue that allows an attacker to steal energy from another connected vehicle and develop *EVExchange*, the first relay attack on the Vehicle to Grid environment. Moreover, we developed a countermeasure to defend against it based on the distance bounding technique. Additionally, we show how a malicious charging operator can exploit the charging pattern of the vehicle's battery to profile and then track a particular car, mining the driver's privacy.

In the second part of this dissertation, we analyze security solutions connected to the internal bus of vehicles, which allows communication between internal devices. CAN bus, the de facto standard for these communications, is a legacy protocol not equipped with any security feature, allowing attackers to compromise the system easily. However, security improvements are complex to deploy due to the real-time requirements of the system, which often hold companies back from rolling them out. In this thesis, we applied novel methods to solve the problem. We employed a fast and reliable technique for feature extraction to identify attacks using Machine Learning, obtaining good results with a lightweight implementation. With the intent to reduce false positives as much as possible, we developed *CANTXSec*, the first deterministic solution to identify and stop certain kinds of attacks by comparing Electric Control Unit activations with bus traffic. Finally, we investigate authentication systems in vehicles employing data from the CAN bus. In particular, we analyzed issues in authentication systems based on the driver's behavior, developing two adversarial attacks against them and providing insights on how they should be efficiently and securely deployed.

Contents

List of Figures

List of Tables

1 Introduction

The technological revolution of recent decades has made computing capabilities increasingly cheap and widespread in every aspect of life. In particular, Cyber-Physical Systems (CPSs) represents the intricate interaction between the digital and physical worlds [143]. With respect to standard IT systems (e.g., personal computers, web servers) equipped with computational and networking capabilities, CPSs also integrate sensing and control functions. Their connection opened to a wide variety of applications, which were impossible some decades ago. In fact, nowadays, smart devices are pervasive in modern houses and employed to control many different aspects, from opening gates to regulating temperatures [326]. CPSs has also found huge applications in the industry where Industrial Control Systems (ICSs) are employed to automatize and remotely control production [89]. Healthcare has also benefited from this technological innovation, giving light to applications such as telemedicine and mobile health [201].

Aside from the undeniable benefits this revolution has brought, it is also important to consider the other side of the coin. The spread of computational capabilities and networking-capable devices has enhanced the number of possible targets for cybercriminals [175]. Moreover, for a long time, cybersecurity has been seen more as a cost than a benefit, and companies refrained from investing in it [128]. Over the last years, this combination has generated a huge amount of unsecured connected devices, sometimes even accessible from everywhere in the world [321]. In addition to that, many legacy devices designed to be run in a secure and gapped environment have been connected to networks to facilitate remote control and management, opening up several vulnerabilities [278].

While cyberattacks targeting IT systems usually aim at breaking the confidentiality of the system, for instance, stealing sensitive information, the target for CPSs may be different and depending on the application. For instance, in a nuclear plant, the most important security aspect is the availability of all communication since safety may depend on it: imagine the consequences of having a problem transmitting the temperature of the core during overheating. At the same time, the actual temperature of the core may not be attractive as information to be stolen since it will have value for a short period only. A similar scenario happens on vehicles. Transmitting the information about a crash to the airbags is the most important thing, while the confidentiality of the information can take a back seat. In both cases, the danger is tangible and may result in people getting hurt. Therefore, it is essential to address the security of CPSs.

Despite ongoing efforts by researchers, companies, and governments, the frequency of attacks on CPSs continues to rise. Some attacks in recent years have raised awareness on the topic. In 2010, an incredibly advanced malware called *Stuxnet* targeted the turbines of an Iranian nuclear plant, causing significant damage [219, 375]. Since then, other malicious attacks have targeted ICS, such as *TRITON* [109] and *BlackEnergy* [323]. The first demonstration to the general public of a cyberattack in the vehicle domain dates back to 2015, when two researchers, Charlie Miller and

Chris Valasek, showcased how it was possible to take remote control of an untouched car [255]. In the following year, a team of hackers compromised a Tesla Model S 12 miles away [196].

Nowadays, modern vehicles employ small and simple computers called Electronic Control Units (ECUs) to command each component, from the infotainment systems such as the car radio to safety measures like airbags and brakes. An advanced vehicle like a Tesla contains up to 100 million code lines [285], which inevitably may include bugs and possible vulnerabilities. Many entry points to interface with the in-vehicle network are available, ranging from the physical On-Boad Diagnostic (OBD)-II port, an interface to the internal bus, easily accessible and legitimately employed for diagnostic, to more sophisticated techniques exploiting wireless channels (e.g., Bluetooth, 802.11x, 4G, 5G) available in the majority of the modern automobiles [70]. Moreover, novel connections are added to vehicles whenever there is a need. An example is the connection between an Electric Vehicle (EV) and the grid to manage the charging process [25].

This latter topic of EVs is even more critical if we look at the fast growth of the selling of these products in the market. 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 [351]. Furthermore, the International Energy Agency estimates that if governments agreed to encourage the so-called "Green Transition", EVs could reach 230 million by 2030. Figure 1.1 illustrates the expected increase of EVs usage in the world in the following years from a Deloitte report [376]. Vehicle vendors such as Honda plan to convert to electric its entire car production by 2040 [289]. 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 [5]. In such an expanding scenario, cybersecurity needs to be considered to ensure the safety of all the parties involved.

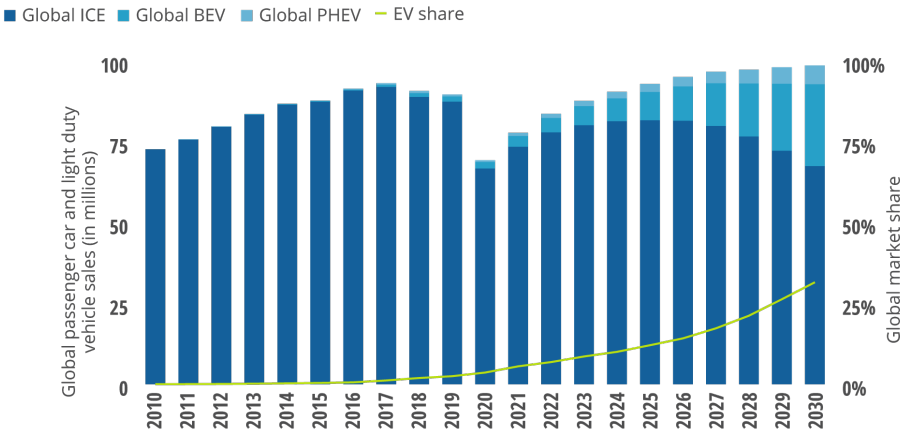


FIGURE 1.1: Expected global increase in EVs share in the next years [376].

In this dissertation, we investigated two promising targets for cybercriminals who want to compromise a vehicle. In the first part, we investigated EVs and the Vehicle-to-Grid (V2G) paradigm, proposing some new attacks and countermeasures. In the second part, we focused on the Controller Area Network (CAN) bus, which

connects the internal components of a vehicle. We design some systems to detect attacks without adding latency to such a real-time system. Moreover, we investigated the usage of the data transmitted for authentication purposes.

Vehicle cybersecurity is a broad field that cannot be fully covered in a PhD dissertation. This document focuses on addressing, at least in some parts, the following research questions.

RQ1 What are the peculiarities in terms of cybersecurity that an electric vehicle exhibits with respect to more investigated petrol-based vehicles?

RQ2 Are there possibilities to compromise the security of EVs by exploiting features that are unique to these types of vehicles?

RQ3 Are there privacy leakages that specifically affect EVs?

RQ4 Despite the growing need to improve vehicle cybersecurity, automotive companies still refrain from implementing security enhancements proposed by the research community. Are there more attractive solutions that can increase adoption in in-vehicle communications?

RQ5 Can novel usages of CAN bus data represent a threat to the security of the vehicle and its owner?

1.1 Thesis Contributions

The dissertation is organized into two parts, investigating two different attack vectors in the vehicle domain:

1. *Electric Vehicles Security*. In the first part of this dissertation (Part I), we investigated the differences in security and privacy between normal petrol-based cars with respect to EV, surveying solutions available in the literature and highlighting future challenges (Chapter 2). Starting from this overview, we discover and exploit an issue in a charging protocol to develop an attack that a malicious entity can use to steal energy from a victim. To prevent the attack, we developed an effective solution that is transparent to the user (Chapter 3). Moreover, based on the physical proportions of batteries, we demonstrated the feasibility of profiling vehicles using features extracted from the charging process, proposing applications both as authentication and as a tracking mechanism (Chapter 4).

2. *CAN Bus Security*. In this second part of the thesis (Part II), we addressed the security of the most employed protocol that allows ECUs to communicate with each other. We focus on the development of novel Intrusion Detection Systems (IDSs) that address the issues that prevent already developed systems from being widely deployed in real vehicles. In particular, we addressed the limited computation capabilities of ECU, proposing a lightweight IDS employing an innovative feature extraction technique (Chapter 5). Another in this field is related to false positives. To eradicate them, we designed a deterministic system that can detect and prevent certain kinds of attacks without misdetection (Chapter 6). The last chapter of this part investigated the application of the data transmitted in the bus for authenticating drivers based on their driving behavior. We designed two adversarial attacks and proposed several pieces of advice for a secure and practical implementation (Chapter 7).

Finally, Chapter 8 concludes the thesis by summarizing the findings achieved and identifying future work directions.

In this dissertation, some passages have been quoted verbatim, and some figures have been reused from the works [88, 53, 52, 115, 31], all coauthored by the author of the thesis. In the following, we summarized the main content of each chapter.

Electric Vehicles Security and Privacy: Challenges, Solutions, and Future Needs [52]

EVs share common technologies with classic fossil-fueled cars, but they also employ novel technologies and components (e.g., Charging System and Battery Management System) that create an unexplored attack surface for malicious users. Although multiple contributions in the literature explored cybersecurity aspects of particular components of the EV ecosystem (e.g., charging infrastructure), there is still no contribution to the holistic cybersecurity of EVs and their related technologies from a cyber-physical system perspective.

Contributions. Chapter 2 provides an in-depth study of the security and privacy threats associated with the EVs ecosystem. We analyze the threats associated with both the EV and the different charging solutions. Focusing on the CPSs paradigm, we provide a detailed analysis of all the processes that an attacker might exploit to affect the security and privacy of both drivers and the infrastructure. To address the highlighted threats, we present possible solutions that might be implemented. We also provide an overview of possible future directions to guarantee the security and privacy of the EVs ecosystem. Based on our analysis, we stress the need for EV-specific cybersecurity solutions.

EVExchange: A Relay Attack on Electric Vehicle Charging System [88]

To support the increasing spread of EVs, Charging Stations (CSs) are being installed worldwide. The new generation of CSs employs the V2Gs paradigm by implementing novel standards such as ISO 15118. This standard enables high-level communication between the vehicle and the charging column, helps manage the charge smartly, and simplifies the payment phase. This novel charging paradigm, which connects the Smart Grid to external networks (e.g., EVs and CSs), has not been thoroughly examined yet. Therefore, it may lead to dangerous vulnerability surfaces and new research challenges.

Contributions. In Chapter 3, we present *EVExchange*, the first attack to steal energy during a charging session in a V2G communication, i.e., charging the attacker's car while letting the victim pay for it. Furthermore, if reverse charging flow is enabled, the attacker can even sell the energy available on the victim's car! Thus, gaining the economic profit of this selling, while leaving the victim with a completely discharged battery. We developed a virtual and a physical testbed in which we validate the attack and prove its effectiveness in stealing the energy. To prevent the attack, we propose a lightweight modification of the ISO 15118 protocol to include a distance bounding algorithm. Finally, we validated the countermeasure on our testbeds. Our results show that the proposed countermeasure can identify all the relay attack attempts while being transparent to the user.

EVScout2.0: Electric Vehicle Profiling Through Charging Profile [53]

EVs represent a green alternative to traditional fuel-powered vehicles. To enforce their widespread use, both the technical development and the security of users shall be guaranteed. Users' privacy represents a possible threat that impairs the adoption of EVs. In particular, recent works showed the feasibility of identifying EVs based on the current exchanged during the charging phase. In fact, while the resource negotiation phase runs over secure communication protocols, the signal exchanged during the actual charging contains features peculiar to each EV. In what is commonly known as profiling, a suitable feature extractor can associate such features to each EV.

Contributions. In Chapter 4, we propose *EVScout2.0*, an extended and improved version of a previously proposed framework to profile EVs based on their charging behavior [55]. By exploiting the current and pilot signals exchanged during the charging phase, our scheme can extract features peculiar for each EV, hence allowing their profiling. We implemented and tested *EVScout2.0* over a set of real-world measurements considering over 7500 charging sessions from a total of 137 EVs. In particular, numerical results show the superiority of *EVScout2.0* with respect to the previous version. *EVScout2.0* can profile EVs, attaining a maximum of 0.88 for both recall and precision scores in the case of a balanced dataset. To the best of the authors' knowledge, these results set a new benchmark for upcoming privacy research for large datasets of EVs.

CANLP: Intrusion Detection for Controller Area Networks using Natural Language Processing and Embedded Machine Learning [31]

The CAN protocol is the most widely used standard in the automotive industry for in-vehicle networks. However, it lacks essential security features such as encryption and message authentication. The absence of such security features has been shown to make the vehicle network vulnerable to exploits by an adversary. Although multiple types of IDS have been developed for CAN, it can be difficult to deploy them in real-time with low latency. Further, many of these IDSs are unable to isolate specific CAN frames on which an attack has been mounted, which makes it challenging to design defense mechanisms.

Contributions. In Chapter 5, we develop *CANLP*, a Natural Language Processing (NLP)-based IDS to determine whether each transmitted message originated from a legitimate ECU or an adversary. *CANLP* uses Term Frequency-Inverse Document Frequency (TF-IDF), a NLP technique to discern complex features associated with CAN data and trains machine learning models to identify three types of attacks, namely fuzzing, spoofing, and masquerade. When an attack is detected, *CANLP* identifies the malicious CAN frame, which is important for developing resilient systems. Extensive experiments on 4 publicly available vehicle network datasets (which represent data collected from over three vehicle makes and four models) show that *CANLP* performs attack classification with high F1-scores of 0.9974. We also show that *CANLP* can be deployed for attack detection on resource-constrained hardware through experiments on a testbed with latency as low as $< 0.05ms$, making it suitable for real-world automotive applications.

CANTXSec: A Deterministic Intrusion Detection and Prevention System for CAN Bus Monitoring ECU Activations [115]

Despite being a legacy protocol with various known security issues, CAN still represents the de-facto standard for communications within vehicles, ships, and industrial control systems. Many research works designed IDSs to identify attacks by training machine learning classifiers on bus traffic or its properties. Actions to take after detection are, on the other hand, less investigated, and prevention mechanisms usually include protocol modification (e.g., adding authentication). An effective solution has yet to be implemented on a large scale in the wild. The reasons are related to the effort to handle sporadic false positives, the inevitable delay introduced by authentication, and the closed-source automobile environment that does not easily permit modifying ECUs software.

Contributions. In Chapter 6, we propose *CANTXSec*, the first deterministic Intrusion Detection and Prevention system based on physical ECU activations. It employs a new classification of attacks based on the attacker's need in terms of access level to the bus, distinguishing between Frame Injection Attacks (FIAs) (i.e., using frame-level access) and Single-Bit Attacks (SBAs) (i.e., employing bit-level access). *CANTXSec* detects and prevents classical attacks in the CAN bus while detecting advanced attacks that have been less investigated in the literature. We prove the effectiveness of our solution on a physical testbed, where we achieve 100% detection accuracy in both classes of attacks while preventing 100% of FIAs. Moreover, to encourage developers to employ *CANTXSec*, we discuss implementation details, providing an analysis based on each user's risk assessment.

When Authentication Is Not Enough: On the Security of Behavioral-Based Driver Authentication Systems [115]

Many research papers have recently focused on behavioral-based driver authentication systems in vehicles. Pushed by Artificial Intelligence (AI) advancements, these works propose powerful models to identify drivers through their unique biometric behavior. However, these models have never been scrutinized from a security point of view, rather focusing on the performance of the AI algorithms. Several limitations and oversights make implementing the state-of-the-art impractical, such as their secure connection to the vehicle's network and the management of security alerts. Furthermore, due to the extensive use of AI, these systems may be vulnerable to adversarial attacks. However, there is currently no discussion on the feasibility and impact of such attacks in this scenario.

Contributions. Driven by the significant gap between research and practical application, Chapter 7 seeks to connect these two domains. We propose the first security-aware system model for behavioral-based driver authentication. We develop two lightweight driver authentication systems based on Random Forest (RF) and Recursive Neural Network (RNN) architectures designed for our constrained environments. We formalize a realistic system and threat model reflecting a real-world vehicle's network for their implementation. When evaluated on real driving data, our models outclass the state-of-the-art with an accuracy of up to 0.999 in identification and authentication. Moreover, we are the first to propose attacks against these systems by developing two novel evasion attacks, SMARTCAN and GAN-CAN. We show how attackers can still exploit these systems with a perfect attack success rate (up to 100%). Finally, we discuss the requirements for deploying driver

authentication systems securely. Through our contributions, we aid practitioners in safely adopting these systems, help reduce car thefts, and enhance driver security.

1.2 Research Methodology

This dissertation explores the security of the vehicle environment by applying different research methods and approaches depending on the task.

Chapter 2 originated from a survey of the relevant literature collected through academic search engines (e.g., SCOPUS, Google Scholar) and by looking at proceedings of the top conferences in cybersecurity (e.g., Usenix Security, IEEE Security and Privacy). From the literature analysis, qualitative research was conducted to develop tables providing a complete overview of the main threats and possible countermeasures in the EV environment.

Chapter 3's idea originated from a literature analysis, together with a deep investigation of the standards involved. Once the possible vulnerability has been discovered, we verify its exploitability on a virtual testbed we developed as a previous work [21]. Once the attack was properly implemented on the virtualized system, we repeated a similar validation on a physical testbed with emulated components developed by ourselves. Finally, we employed the same systems to develop and deploy the countermeasure. We conducted a quantitative analysis to collect data regarding our solution's performance.

Chapter 4 tries to solve and improve some limitations of a related work [55]. We noticed some possible improvements coming from an incomplete usage of the data. We tested our hypotheses with a quantitative approach in order to choose the best solutions and compare our results with those of the previous work, measuring the improvements in each scenario. We employed data that the ACN Portal freely shares online [224], expanding the dataset employed by the previous work.

Both Chapter 5 and Chapter 6 intuitions originated from discussions with peers and colleagues from companies during conferences. The intuitions relate to ways for reducing the computational requirements in Machine Learning (ML) based IDS for vehicles (Chapter 5) and eradicate the false positive problem (Chapter 6). For both projects, we conducted our experiments in a physical testbed we developed to resemble a car without exposing the researchers to safety issues. Moreover, Chapter 5 also employed data from widely used datasets in related works to simulate traffic and attacks on the bus [193, 315, 161, 370].

Finally, all the analysis done in Chapter 7 originated from an analysis of advanced usage of CAN bus data in the literature. From such an analysis, we decide to employ a mixed strategy to investigate, compare, attack, and improve behavioral authentication systems. For this project, we employed a dataset that is widely used in the related works we analyzed (OCSLab dataset [217]).

1.3 Paper references

The findings proposed in this dissertation are based on the results of the following papers, ordered by acceptance or submission date.

- Conti, Mauro, Denis Donadel, Radha Poovendran, and Federico Turrin. "Evexchange: A relay attack on electric vehicle charging system." In *European Symposium on Research in Computer Security (ESORICS)*, pp. 488-508. Cham:

Springer International Publishing, 2022 [88]. (**CORE:A, LiveSHINE:A+, MA:A+, Italian GGS 1/A+**). This paper is discussed in Chapter 3.

- Brighente, Alessandro, Mauro Conti, Denis Donadel, and Federico Turrin. "EVScout2. 0: Electric vehicle profiling through charging profile." *ACM Transactions on Cyber-Physical Systems* 8, no. 2 (2024): 1-24, 2022 [53]. (**JCR IF 2023: 2.0**). This paper is discussed in Chapter 4.

- Brighente, Alessandro, Mauro Conti, Denis Donadel, Radha Poovendran, Federico Turrin, and Jianying Zhou. "Electric Vehicles Security and Privacy: Challenges, Solutions, and Future Needs." Submitted to *Computer and Security*, 2023 [52]. Preprint: arXiv:2301.04587. (**JCR IF 2023: 4.8**). This paper is discussed in Chapter 2.

- Balasubramanian, Kavya, Adithya Gowda Baragur, Denis Donadel, Dinuka Sahabandu, Alessandro Brighente, Bhaskar Ramasubramanian, Mauro Conti, and Radha Poovendran. "CANLP: NLP-Based Intrusion Detection System for CAN." In *Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing*, pp. 212-214, 2024 [31]. (**CORE:B, LiveSHINE:A, MA:A, Italian GSS 2/A-**). This paper is discussed in Chapter 5.

- Efatinasab, Emad, Francesco Marchiori, Denis Donadel, Alessandro Brighente, and Mauro Conti. "When Authentication Is Not Enough: On the Security of Behavioral-Based Driver Authentication Systems." Submitted to *IEEE Symposium On Security & Privacy*, 2025 [115]. Preprint: arXiv-2306.05923. (**CORE:A++, LiveSHINE:A++, MA:A++, Italian GSS 1/A++**). This paper is discussed in Chapter 7.

- Balasubramanian, Kavya, Adithya Gowda Baragur, Denis Donadel, Dinuka Sahabandu, Alessandro Brighente, Bhaskar Ramasubramanian, Mauro Conti, and Radha Poovendran. "Intrusion Detection for Controller Area Networks using Natural Language Processing and Embedded Machine Learning" Submitted to *IEEE Transactions on Dependable and Secure Computing*, 2024. (**JCR IF 2023: 7.0**). This paper is discussed in Chapter 5.

- Donadel, Denis, Kavya Balasubramanian, Alessandro Brighente, Bhaskar Ramasubramanian, Mauro Conti, and Radha Poovendran. "CANTXSec: A Deterministic Intrusion Detection and Prevention System for CAN Bus Monitoring ECU Activations". To be presented in *23rd International Conference on Applied Cryptography and Network Security (ACNS)*, 2025. (**CORE:B, LiveSHINE:A, MA:A-, Italian GSS 2/A-**). This paper is discussed in Chapter 6.

1.4 Other papers published during the Ph.D.

Aside from the paper previously mentioned, other papers have been produced during the Ph.D., ordered by publication or submission date.

- Conti, Mauro, Denis Donadel, and Federico Turrin. "A survey on industrial control system testbeds and datasets for security research." *IEEE Communications Surveys & Tutorials* 23, no. 4 (2021): 2248-2294, 2021 [89]. (**JCR IF 2023: 34.4**).

This paper focused on another area in the CPS security field related to the ICSs. In particular, we focused on the difficulties in performing research in such a critical field, where systems under test are expensive and difficult to manage. Therefore, we investigated and cataloged the available testbeds and datasets that can be employed to perform research in the field, highlighting the pros and cons of the many available solutions.

- Carboni, Federico, Mauro Conti, Denis Donadel, and Mariano Sciacco. "If You're Scanning This, It's Too Late! A QR Code-Based Fuzzing Methodology to Identify Input Vulnerabilities in Mobile Apps." In *International Workshop on Security in Mobile Technologies (in conjunction with ACNS)*, 2023 [60]. (**CORE:B, LiveSHINE:A, MA:A-, Italian GSS 2/A-**)

  In this paper, we developed a framework to test Android applications against bugs and vulnerabilities caused by scanning malicious QR codes. We found two bugs in popular applications and enabled others to test applications by making the code open source.

- Brighente, Alessandro, Mauro Conti, Denis Donadel, and Federico Turrin. "Hyperloop: A cybersecurity perspective". *Workshop on Automotive Cyber Security (in conjunction with IEEE EuroS&P)*, 2024 [54]. (**MA:C**).

  In this paper, we investigated the security and privacy aspects of one of the most promising future transportation technologies. Due to the lack of official technical details, we grasp from the publicly available specifications and research papers to understand the main communication inside Hyperloop. Then, we started from our knowledge in similar fields, such as ICS and automotive security, to identify potential security and privacy issues inside the Hyperloop system.

- Donadel, Denis, Francesco Marchiori, Luca Pajola, and Mauro Conti. "Can LLMs Understand Computer Networks? Towards a Virtual System Administrator". In *49th IEEE Conference on Local Computer Networks (LCN)*, 2024 [111]. (**CORE:B, LiveSHINE:A-, MA:B, Italian GSS 3/B**).

  This paper investigates the capabilities of general-purpose Large Language Model (LLM) in comprehending computer networks and answering questions related to network infrastructures. It introduces a framework to measure LLMs capabilities in this task, provides some numerical results related to both proprietary and open weights LLMs, and suggests some enhancements to the prompt to improve answers' accuracy.

Part I

Electric Vehicle Security

2 Electric Vehicles Security and Privacy: Challenges, Solutions, and Future Needs

The recent climate crisis demands green alternatives to replace technologies with high environmental impact. Among the others, fossil-fueled transportation is one of the significant causes of greenhouse gases. EVs have been proposed as a green alternative, where electric batteries are employed as a power source. During the last years, the number of people opting for the EV alternative increased up to the point where the market share of new EV sales reached more than 50% in countries such as Iceland (55.6%) and Norway (82.7%) [319]. The adoption of EVs is further expected to increase in the next years. In fact, governments are incentivizing the adoption of EVs thanks to the deployment of a large number of Electric Vehicle Supply Equipment (EVSE) in public charging infrastructures [239] and planning to ban sales of fossil-fueled vehicles [147]. Furthermore, technology advancements remove the current barriers against consumers' adoption of EVs, providing extended driving range and seamless charging [59].

The increasing number of EVs demands a thorough analysis of the security of both vehicles and infrastructure operations. Like traditional vehicles, EVs are equipped with many ECUs, sensors and actuators that measure, process, and control the different stimuli inside and outside the vehicle. However, EVs include additional components. Indeed, an EV integrates components to govern the hardware and software that is dedicated to smartly managing electric energy. These components are, for instance, the Battery Management System (BMS) and the charging system.

Different studies have already proven the impact of potential cybersecurity attacks on automotive systems. For instance, Miller and Valasek [255] proved the feasibility of hijacking a vehicle by remotely controlling it through the infotainment system. Furthermore, most existing vehicles exploit CAN as in-vehicle network architecture, which has already been proven as non-secure [229] and, therefore, may be vulnerable to potential cyberattacks. Lastly, privacy shall also be guaranteed to prevent malicious users from obtaining sensitive information on the driver, such as his/her location or habits. It is essential to include security and privacy features by design to prevent these and other attacks. Researchers investigated vehicle security, focusing on the different aspects of in-car communications [311, 308]. However, EVs are equipped with specific components that provide fundamentally different attack surfaces and exploitation points. For instance, EVs are equipped with electric batteries to power the vehicle components ranging from the infotainment system to the acceleration pedal, together with systems to manage the electrical power as shown in Figure 2.1.

Therefore, analyzing the in-vehicle threats associated with these components is essential. Furthermore, the power supply must be regulated by dedicated hardware, not classic vehicles. Researchers discussed how the EV charging infrastructure could be exploited by attackers [148], however, neglecting the in-vehicle threats.

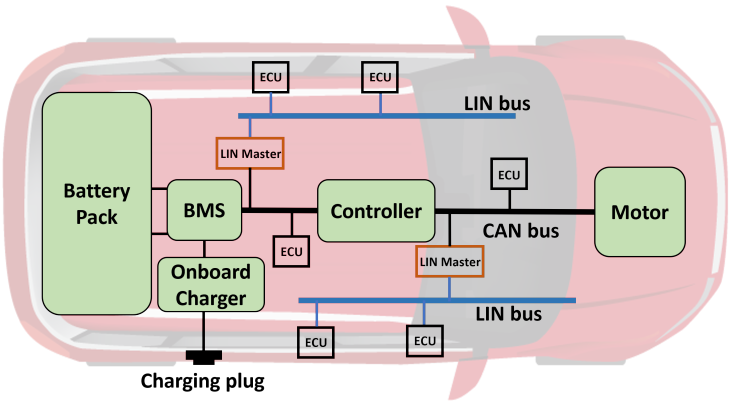


FIGURE 2.1: Main components of an EV. Green components are EV specific.

Contributions. In this chapter, we examine the security and privacy issues of EVs from a CPS point of view. Given the high demand for EVs and the increasing number of deployed charging facilities, it is fundamental to guarantee the security and privacy of both vehicles and users. Many literature contributions discuss solely technical aspects of the EV ecosystem without focusing on security issues. Other security-focused works study a single system component (e.g., the vehicle's internal bus, the smart grid, or the communication protocols) without comprehensively analyzing the whole environment. We provide a general overview of EV functioning, focusing on their core components to build the basic knowledge needed to analyze the possible threat vectors. We then discuss possible attacks and countermeasures specific for EV and underline the existing security solutions for fuel vehicles that are also effective in EVs. With the bird-eye on the CPS concept, we are not only able to discuss the issues related to the exchange of information between the different involved entities, but also the side channels that may leak sensitive information or that could lead to hazardous behavior impacting on users' safety. We hence shed light on the unresolved challenges of EVs ecosystems, providing interested researchers with possible directions worth investigating to guarantee the security and privacy of the overall EV ecosystem. We also consider future directions such as the Wireless Power Transfer (WPT) charging of EVs, which has only been developed on small-scale testbeds at the time of writing. We believe that delving into this emerging system's security and privacy issues will help future developers design and implement secure-by-design WPT solutions for EV.

We summarize the contribution of this chapter as follows.

- We examine the peculiar components that differentiate EVs from fossil-fueled vehicles and provide an overview of their role and how they exchange information.
- We provide an overview of the different technologies employed to charge electric vehicles, comprising both wired and wireless charging. We present the available standards for each of them and describe their basic functioning.
- We provide an in-depth discussion of the security and privacy issues of the EVs ecosystem. We analyze the threats related to the in-vehicle network and the threats related to the charging process. We particularly focus on their effects on the peculiar EV components and analyze them from a CPS point of view.

- We analyze and compare possible countermeasures proposed in the literature for each of the presented attacks, even grasping from other similar areas.
- We outline future directions for research in the EV cybersecurity domain.

Organization. The rest of the chapter is organized as follows. In Section 2.1, we review the related literature. In Section 2.2, we describe the EV components the charging infrastructure. In Section 2.3, we then discuss the in-vehicle security and privacy threats, and those related to the charging infrastructure in Section 2.4. Along with the threats, we also present possible countermeasures. Then, we discuss the possible future direction in Section 2.5, and lastly, we conclude the chapter in Section 2.6.

2.1 Related Literature on EV Security

Automotive cyber-security requires standardization to allow for security guarantees and interoperability. Schmittner et al. [311] reviewed the available standards, including designing and validation aspects. These standards, however, do not consider the peculiar features of EVs. Scalas et al. [308] provided an overview of the cybersecurity requirements for the future of the automotive industry, focusing on in-vehicle components. They discussed several technologies and attacks but were not specific for EVs. Furthermore, different works present technical reviews of the EV ecosystem [27, 101]. However, none of them consider the security aspects.

Some contributions in the literature focused on specific components of EVs. For instance, Khalid et al. [199] focus on the BMS, discussing the lack of a cybersecurity standard to guarantee its security and providing an overview of the possible standardization framework that could be adopted to achieve this goal. Chandwani et al. [68] presented an overview of the cybersecurity threats associated with the onboard charging system of EVs. Despite providing an accurate analysis of the security of this component, their contribution does not consider how these attacks can impact the other peculiar components of the EV. These contributions do not provide a general overview of the EV ecosystem. Furthermore, they do not discuss the threats associated with privacy.

Acharya et al. [3] provide the first discussion on how EVs can be considered as CPS. The authors discuss how different attacks can be conducted inside the car and during communication with the power supplier. However, they do not consider the specific components of the EVs such as the BMS. Jin et al. [387] focus on the CPS system represented by the power electronics in EVs. However, they do not consider how these attacks may impact the other components of the EV and did not discuss the issues related to WPT. Most of the literature related to EVs' cybersecurity focus on the charging infrastructure and process. Gottumukkala et al. [148] provide an overview of the CPS threats associated with a wired EV charging infrastructure. Antoun et al. [18] discuss the security threats associated with the negotiation and actuation of a charging session investigating the communications between the multiple involved entities. They presented different charging scenarios, neglecting the WPT option. Costantino et al. [95] briefly investigates the security of charging methods (wired, WPT, and battery swap). However, they only consider the connection between the vehicle and the charging column, neglecting all the other devices and connections.

Vehicles can be interconnected with one another to form the internet of vehicles. This is also feasible with EVs, which imposes additional security challenges. Fraiji

TABLE 2.1: Summarization of the related works and their limitations.

Reference	BMS	Onboard Charger	Battery Pack	Controller	Electric Motor	Wired Charging	Wireless Charging
Khalid et al. [199]	✓						
Chandwani et al. [68]		✓					
Acharya et al. [3]						✓	
Ye et al. [387]	✓			✓	✓	✓	
Sripad et al. [334]			✓				
Gottumukkala et al. [148]						✓	
Costantino et al. [95]						✓	✓
Antoun et al. [18]						✓	
Van Auben et al. [366]						✓	
Babu et al. [25]						✓	
This chapter	✓	✓	✓	✓	✓	✓	✓

et al. [125] discuss the cybersecurity threats associated with the internet of electric vehicles, discussing the threats associated with the communication with the multiple involved entities being part of the road infrastructure. The cybersecurity focus is on the communication links, therefore neglecting the impact of the peculiar EVs' components.

Some works in the literature focus on the security issues related to the charging columns and their communications with the backend [266]. Nash et al. [267] proposed ChargePrint, a framework for discovering and performing security analysis of charging management systems. Garofalaki et al. [139] present a detailed survey on Open Charge Point Protocol (OCPP) and the corresponding threat and vulnerabilities on the V2G ecosystem due to its adoption, similar to the work of Alcaraz et al. [12] which explicitly focuses on the OCPP protocol. However, in this chapter, we focus on the issues interesting the EV and its connection with the charging column without focusing on the security of the charging column itself or on the backend communication.

Table 2.1 compares the related works on EV security with our contribution. We can see that most of the contributions focus on vehicle-to-grid communications in the wired case. However, none of the available papers focus on intertwining the different cyber-physical aspects of EVs. Therefore, this chapter provides a more complete analysis of the security and privacy challenges for EVs.

2.2 Electric Vehicles from a Cyber-Physical System Perspective

In this section, we analyze EVs from a CPS perspective. We emphasize those components that differentiate EVs from gas-fueled vehicles. In particular, we first describe the traditional vehicle architecture in Section 2.2.1. Then, we present the main components of an EV in Section 2.2.2, showing how it differs from traditional vehicles. Lastly, we provide an overview of the EV charging infrastructure in Section 2.2.3.

2.2.1 Traditional Vehicle Architecture

Nowadays, vehicles contain dozens of different microcomputers, called Electronic Control Units (ECUs), running millions of lines of code [391]. Each ECU is responsible for controlling a mechanical (e.g., brakes) or electrical (e.g., light) component of a modern vehicle. Depending on the component it has to manage, an ECU generally employs a wide range of microcontrollers, from simple 8-bit RISC controllers to more complex 32-bit multicore processors. ECU are typically implemented with ad-hoc firmware, even if complex ECUs may run complete operating systems: the infotainment system, for instance, usually runs a Linux-based kernel. In order to provide more flexibility during updates, more advanced solutions envisage the implementation of multiple ECUs on a single FPGA board [76].

Communications among ECUs that reside in the vehicle pass through wires that connect multiple components. The two mostly implemented technologies are CAN and Local Interconnect Network (LIN). CAN represents the main network that allows for cost-effective wiring, self-diagnosis and error correction [47]. The CAN bus consists of two wires and implements a distributed architecture, where car modules (i.e., the ECUs) share messages upon winning a contention phase. However, CAN has been designed to be a reliable solution, neglecting possible security ad privacy shortcomings.

The LIN bus is a supplement to CAN [182]. In particular, it connects a smaller number of ECUs (one master and up to 16 slave nodes) and offers a drastically cheaper implementation at the cost of lower performance and reliability. A LIN master node is typically a gateway to CAN, and multiple LIN buses can communicate via the CAN bus. The LIN bus can be used to control, among the others, sensors and actuators for steering wheels, comfort, powertrain, engine, air conditioning, doors, and seats.

Besides CAN and LIN, also other technologies such as FlexRay [242] and Media Oriented Systems Transport (MOST) [122] are currently used for automotive networks. To overcome some of these technologies' limitations and ease their interoperability, automotive Ethernet has recently been introduced as a possible solution [93]. Given the CPS nature of our investigation, we do not prefer one technology over another, as these all represent communication means for the exchange of information inside the EV. We refer the interested reader to [93] for a discussion on automotive Ethernet security.

Modern vehicles also include mechanisms to update the internal software. This service is generally implemented with the aid of external device plug (e.g., USB flash drive) or Over-The-Air updates (OTA) software update [158] (e.g., via Internet connection). Furthermore, many vehicles nowadays include complex entertainment systems, which may expand the vulnerable surface, exposing new connections (e.g., Bluetooth) and operating systems (e.g., Android).

2.2.2 Electric Vehicle Specific Components

EVs share most of the architecture with fuel-based vehicles. However, they comprise a different set of hardware modules that manage how the vehicle generates power and how to generate motion. In particular, an EV comprises the following components [362], depicted in Figure 2.1.

Battery. The battery is where the charge is stored in the form of Direct Current (DC). It provides the power needed to operate the EV components. Batteries are usually combined in packs and connected in series or parallel to increase the voltage and

Ampere/hour they can deliver to the EV. Batteries suitably combined are enclosed into a metal casing to prevent damage. The case usually includes a cooling system to avoid damage due to batteries overheating.

Battery Management System. This module manages all operations regarding the battery. It manages the current output and the charging and discharging of the battery by keeping it in a safe operating area. Hence, it regulates the electricity flow through the battery. The BMS is unique for each EV model, and may be designed according to various topologies, i.e., modular, centralized or distributed [199]. The BMS monitors each battery in the pack and measures each cell's voltage, current, and temperature. It is instructed with a threshold limit for each of them and disconnects the load if values exceed the threshold value. Furthermore, the BMS measures the State of Charge (SoC) and state of health of the battery. The BMS communicates with the human-machine interface to report information on this information. All the information are exchanged via CAN or LIN bus.

Battery Charger/Onboard Charger. This component provides an interface between the charging system and the EV battery. As soon as an Alternate Current (AC) charging process begins, the charger converts the input voltage to DC and passes it to the battery for storage. For high power DC charging, the conversion phase is done on the charging column. Furthermore, it prevents possible damages to the battery or the supply system (e.g., overheating) by limiting the power flow [68].

Controller. The controller handles the flow of current from the battery to the EV associated with all operations, ranging from motors-related operations to powering the infotainment system. It receives the input from the driver to control the acceleration, brake pressure, and driving mode and converts the energy in the battery from DC to AC to control the EV accordingly. On the other hand, the EV may generate electricity due to, e.g., regenerative braking. In this case, the controller converts the generated AC to DC such that the energy can be stored in the battery.

Electric Motor. The motor is powered by the EV battery, which provides the electricity needed to turn it and move the EV. The electric motor communicates with sensors and actuators in the EV that control the amount of thrust required [154]. There exist many implementations of electric motors. The most commonly used for EVs are AC induction due to their lower cost implementation thanks to the absence of permanent magnets.

These components characterize an EV and differentiate it from other types of vehicles. In particular, the conventional motor is replaced with an electric one, and a battery pack replaces the fuel tank. Notice that all the components mentioned above need to share messages inside and outside the vehicle to guarantee the correct functioning. An attacker might exploit some of these messages to create inconsistencies on the EV status or to cause damages to both the vehicle and driver. We provide a detailed security analysis based on these components in Section 2.3.

2.2.3 Electric Vehicles Charging Infrastructure

The EV needs to charge its battery periodically to provide power to its components. To this aim, the EV shall be connected to a charging infrastructure with whom it negotiates a charging session. According to the negotiated session, the infrastructure then delivers the needed energy to the EV. Charging may happen either in public areas (e.g., shopping malls or offices) or at a private site (e.g., home). To prevent possible malfunctioning, the charging infrastructure must be carefully managed. This is particularly true when considering a scenario where handling a massive number of EVs may lead to blackout and other grid malfunctions [293].

Charging an EV differs from other devices, such as smartphones or laptops, as it requires dedicated hardware and a drastically larger energy supply. Indeed, if many EVs are concurrently charging, there can be grid overloading, leading to malfunctions and local blackouts. To avoid these issues, the grid must employ a communication channel with the EV to negotiate to charge parameters that respect the vehicle's battery requirements without overloading the grid. V2G refers to the technology enabling this communication type. There are two solutions to manage a charging session: wired and wireless. While the former is more diffused and widely implemented nowadays, the latter is still in the initial stage and under development. Unfortunately, there is no unique world standard to regulate this communication channel. Instead, different manufacturers implement different standards based on the technologies used for the charging process. For instance, CHAdeMO [43] (Japan) or GB/T [227] (China) can be used only with wired charging, while ISO 15118 (Europe, North America) also supports WPT [185, 184].

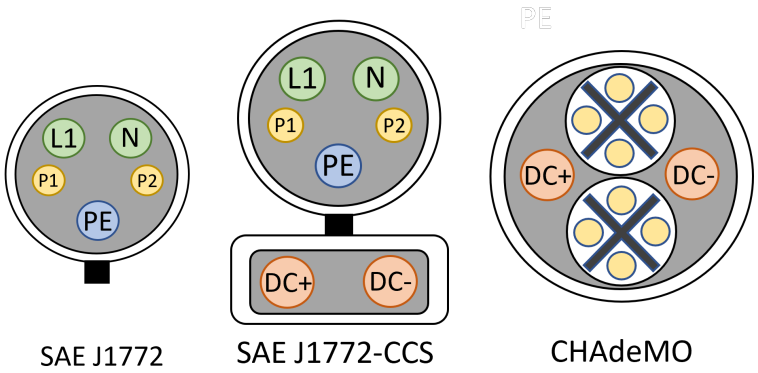
Wired Charging

With this setting, the EV is connected to an EVSE through a cable that transmits both the control signals ad the charging current. In turn, EVSEs negotiate with power grids for the energy needed to charge the vehicle, based on both EV and grid requirements. However, these basic functions are integrated by every charging standard, which employs different communication methods. Low current charging levels, such as AC Level 1 or AC Level 2, require a simple control channel which is generally provided by a Pulse-Width Modulation (PWM) communication. More advanced charging, such as DC charging, needs better management of the energy provided by a High-Level Communication (HLC) provided by protocols such as CAN or Programmable Logic Controller (PLC). These technologies enable the development of additional services, such as the automatizing of the billing process [120, 183], or the download of firmware updates [58]. In case of a lack of automated authentication solutions, EVSE may be equipped with RFID readers through which users can authenticate and pay for the service. EVSEs can be deployed at private or public premises: private charging columns are generally less advanced and support less charging level with respect to public EVSE.

There are mainly two protocols supporting the HLCs between EV and EVSE during DC charging sessions. The first one, employed by Combined Charging System (CCS), is the ISO 15118 [183], which modulates data over the control pilot pin using PLC. The second one, CHAdeMO [43] employs a CAN channel for the communication.

The physical connection between EV and EVSE may be implemented with different plugs according to different standards. In particular, we can classify EVSEs according to different levels. Figure 2.2 shows the different charger levels together with their lead characterization.

Level 1 and Level 2 EVSEs exploit a five-leads connector implementing the SAE J1772 protocol [328], as shown in Figure 2.2a. This connector exploits two leads to deliver the charging current, two leads for pilot signals, and one lead for ground (or protective earth). The two current leads plus the ground one are used by the EVSE for metering and computing the session cost. The two pilot lines have two different functions. The first one, the control pilot, is used to exchange information with the EV during the charging session. The signals exchanged through the control pilot either control the amount of current delivered to the EV [223] or are used to check the connection status and remove power from the adapter in case of disconnection



(A) Level 2 (B) Level 3

FIGURE 2.2: Different types of EV chargers. L1 = AC line 1; N = AC line neutral, P1 and P2 = proximity lines, PE = ground.

to prevent the user injuries [148]. The second pilot line is the proximity pilot, used by the EV to check whether a proper physical connection has been established with the EVSE.

Level 3 EVSEs, i.e., those allowing for fast charging, are based on different implementations and are showed in Figure 2.2b. The first is the CCS expansion of the SAE J1772, which allows for direct current exchange for fast charging. Furthermore, it implements PLC to exchange information between the EV, the EVSE, and the smart grid [328]. The second implementation is the Japanese CHAdeMO [177], which implements a fast charging protocol. Besides delivering power, this implementation allows for data exchange via the CAN bus protocol. Thanks to this type of connection, it is possible to avoid applying power to the connector in case of a non-safe connection or to exchange information related to the battery SoC. Furthermore, CHAdeMO allows V2G communication, where the EV battery is later used as energy storage to provide service to the grid [43]. Other protocols exist, such as the proprietary protocol employed by Tesla vehicles and the Chinese GB/T, which will probably be replaced by Chaoji, an evolution of CHAdeMO [27].

The main differences between Level 3 and Level 2 chargers lie in the higher number of leads in Level 3, and in the implemented circuitry which converts AC to DC, which is inside the charging columns for Level 3, while it is onboard in the EV for Level 2. Furthermore, Level 3 charging includes richer communication capabilities thanks to the support of HLC.

Wireless Power Transfer

Charging via WPT allows charging an EV's battery without physically connecting the vehicle to the charging infrastructure. In WPT, a source (powered by the grid) generates a time-varying electromagnetic field that triggers the generation of a current at the receiver's (EV's) side. This current is generated thanks to a coil mounted on the EV's side that receives the transmitted electromagnetic field and, due to Faraday's law of induction, generates an AC [397].

Via WPT, it is possible to create multiple charging scenarios depending on the mobility of the EV [372]. In fact, thanks to the absence of a physical connection, EVs can be either charged while parked or while driving in a dynamic scenario [25]. The static scenario is similar to the one previously described in Section 2.2.3, where a user books a charging session and receives the power from the grid while parked

at a charging facility. Instead, the dynamic case requires a suitably designed infrastructure composed of multiple sequential WPT transmitters. Figure 2.3 shows a pictorial representation of a dynamic WPT system for EVs. The street is equipped with multiple WPT transmitters deployed underneath the street. These transmitters are connected to the grid that provides the power needed to charge the EV.

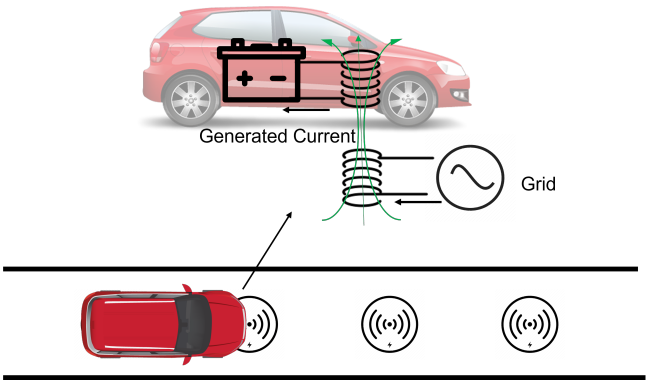


FIGURE 2.3: Representation of a WPT system for EVs.

Dynamic WPT can be further divided into two categories: quasi-dynamic and fully-dynamic [238]. In the former case, charging is limited to the cases where the EV is not moving, e.g., while waiting at stops or traffic lights. In fully-dynamic WPT, charging is continuously delivered to the EV as long as it drives near transmitting coils. In both dynamic scenarios, the challenge is to guarantee that transmitters are activated only when needed to avoid energy waste and that only legitimate users access the emitted power. In fact, due to the absence of a medium, users could steal power by driving close to an EV that paid for the charging session. We discuss all the security problems related to WPT in Section 2.3.

As specified in the ISO 15118 standard [185], the connection between the vehicle and the charging column during a static WPT scenario uses WiFi (IEEE 802.11). The vehicle can connect before being correctly parked or when it is already over the coil. If needed, the EVSE provides the EV with fine positioning messages to help the driver correctly place the vehicle to reduce energy dispersion. After establishing the connection, the two entities communicate similarly to wired cases. Some modifications are introduced to adapt to the wireless scenario, including the WPT charging mode and the fine positioning messages.

Due to their novelty, dynamic and quasi-dynamic charging are not yet covered by approved and widely adopted standards. Some research works [296] adopt Dedicated Short-Range Communications (DSRC) to create a channel between vehicles and the Road Side Units (RSUs), which are in charge of controlling a portion of the road coils.

Another possible solution can be to extend WPT to deliver also information along with power. In fact, Wireless Information and Power Transfer (WIPT) represents a technology that might be exploited for electric vehicle [94]. WIPT can be adopted to implement a system similar to that exploited in wired EV charging, where control signals are exchanged through the pilot line and the charging current. In WIPT, control signals can be coded into the time-varying electromagnetic field to deliver power and check the connection's status simultaneously. Furthermore, this solution can be exploited to authenticate EVs and solve part of the security challenges in

TABLE 2.2: Summary of In-Vehicle Challenges. C. = Components.

C.	Attack Type	Effect	Impact	Possible Solutions
Battery and BMS	DoS	Prevent energy delivery Prevent information reception Increase energy consumption Physically damage the battery	Medium	Flow control Time-lock puzzles Rate limiting Intrusion detection
	Tampering	Short circuit Prevent energy delivery	High	Anomaly detection Tamper-proof hardware
	Malicious Code Injection	Modify BMS response to command Collect sensitive information	Medium	Authentication Remote attestation Intrusion detection
	Spoofing, Replaying, and MiTM	Report false information to the driver Report false information to the other ECUs Physically damage the battery Disrupt charging process Excessive discharging Overcharging	High	Identity management Authentication Intrusion detection Redundancy Timestamps Integrity protection
Controller and Charger	MiTM	Report false information Isolate charger components Modify control signals Increase energy consumption	High	Anomaly detection Intrusion detection Intrusion prevention Integrity Protection
	DoS	Prevent the exchange of energy	Medium	Cookies Time-lock puzzles Rate limiting Intrusion detection
	Spoofing and Replaying	Report false information Physical damage Increase energy consumption	High	Intrusion detection Identity management Timestamps
	Malicious Code Injection	Modify EV response to commands Collect sensitive information Remote control/hijack	High	Authentication Remote attestation Intrusion detection
	Tampering	Impair the charging process Power loss and overvoltage	High	Anomaly detection Tamper-proof hardware
	Eavesdropping and Side Channels	Track the user Profile users' preferences	Low	Differential privacy Encryption

WPT. Although not yet discussed in the literature, we believe that WIPT represents a suitable line of research for EV charging technologies.

2.3 In-Vehicle Security and Privacy Challenges

This section discusses the security and privacy challenges related to the components and protocols used inside EVs. First discuss in Section 2.3.1 the challenges related to the battery and the BMS. Then, we discuss the challenges related to the controller and charger in Section 2.3.2. The security of CAN bus has been extensively studied in the literature, as it does not envision secure by design solutions [190]. However, how these attacks may impact EVs has never been studied. Since all in-vehicle messages are exchanged through CAN and LIN buses, we discuss how their vulnerabilities can be exploited to impact those components specific to the EVs. Table 2.2 summarizes the in-vehicle security and privacy challenges, their effects, impact severeness, and possible countermeasures.

2.3.1 Battery and BMS

The battery pack is a sensitive component of an EV. In case of malfunctions, it may catch fire and even explode [342, 45]. Such situations can severely harm the passengers and create financial damage to the owner and a reputation loss to the manufacturer. Less severe cyberattacks can, however, create financial damage, for instance, by reducing the battery's lifespan, forcing the owner to a premature battery replacement [334].

The battery pack is managed by the BMS, which handles communication with the other ECUs via the vehicle bus. Again, this channel has been proven to be vulnerable to many cyberattacks [190, 47]. In the following, we discuss how cyberattacks impact EVs, extend their effects to the CPS domain, and highlight their effect on the battery and BMS.

Denial of Service. The BMS is responsible for reporting information on the battery status and managing the energy delivery. An attacker might flood the BMS controller by forging and sending a vast number of requests, in similar ways to what may happen with Denial of Service (DoS) attacks against websites [152]. An overload of the BMS may slow responses to legitimate requests or even prevent the BMS from sending response messages completely. This may lead to multiple effects depending on the information requested to the BMS and how the requester device reacts to the absence of a response. In fact, this might cause damage to the battery if power is not properly removed in case of abnormal behavior or physical tampering by the attacker. A DoS may target sensor measurements, such as temperature, and it may prevent the activation of cooling mechanisms, forcing the battery into critical temperatures, which may be irreversible [98]. Furthermore, this attack may also prevent the user from obtaining information on the amount of charge left, causing range anxiety and possibly jeopardizing the drivers' safety in case of a sudden EV stop.

Flow control might prevent the BMS from handling many fake requests. In this context, source authentication may provide information regarding the legitimacy of the sender [150]. A solution for flow control may be given by an adapted version of time-lock puzzles [294]. Furthermore, rate limiting can help mitigate against DoS attacks [215], while intrusion detection strategies can help in identifying the attack before it creates damage [389]. Redundancy on the controllers can also help in mitigating severe DoS attacks against the BMS [46].

Tampering. An attacker might physically tamper with the battery and the BMS. Depending on the specific tampered component, an attacker may be able to cause a short circuit that may lead to catastrophic events such as the start of a fire that might harm both the vehicle and the passenger. This consideration holds for battery and BMS, as they both manage high voltages. Tampering may also lead to less severe consequences, such as the BMS being unable to communicate with the battery or to deliver the full power to the battery during charging. These attacks may also include detaching or cutting cables.

As a possible countermeasure, the battery and BMS shall include an anomaly detection system to prevent applying a voltage to tampered components and causing the aforementioned damages [340]. Another solution may be the physical protection of these components with tamper-proof hardware. For instance, in case of

physical tampering, the battery should be designed so that it cannot receive or deliver power [140]. The SAE J2464 standard contains safety measures that can also be effective against tampering [329].

Malicious Code Injection. The battery pack is managed by the BMS, which is a piece of hardware with firmware onboard. Attackers may try to reverse engineer the software to discover vulnerabilities and build exploit against them [204]. To patch bugs, EVs' software may be updated over the air or via the charging cable [58], thus easing the update process for manufacturers and users. However, this represents a security challenge, as software updates need to access the overall EV network [158]. A malicious user may inject malware via software update to gain control of the BMS [158]. By having partial or complete control over it, the attacker may thus impact the normal functioning of the vehicle. For instance, the malware may prevent the BMS from requesting energy from the battery, causing a blackout in the EV. Contrarily, the BMS may be forced into requesting more energy than needed to speed up the discharging process. Furthermore, thanks to the malware, the attacker may measure other sensitive information of the driver, which may lead to privacy leakages.

To prevent code injection and its effects, access to the EV's internal network shall be strictly regulated. Possible solutions include the use of external source authentication. In case of a successful injection, it is fundamental to identify and mitigate its effect. To this aim, remote attestation and its collective extension may be used to validate the in-vehicle components [15]. Furthermore, anomaly and intrusion detection techniques may help identify attacks to the in-vehicle network [389]. Injection of malicious updates can be detected by integrity verification on the new software, possibly employing a blockchain [40].

Spoofing, Replaying, and Man-in-the-Middle. An attacker may spoof or modify messages to report to the driver false information on the battery SoC, thus impairing a safe drive. The attacker may also report incorrect information to the charging infrastructure by impersonating the BMS or modifying information in the middle of the communication. This may cause the charging process to provoke damage to the battery or the EV circuitry. Furthermore, an attacker may report false information to prevent the correct exchange of energy from the battery to the BMS, for instance, by lowering the current demand and preventing the exchange of a sufficient amount of power from the battery to the EV. By requiring excessive power, an attacker can discharge the battery faster than expected in a battery exhaustion attack [265] or may force the battery to overcharge, leading to a massive shortening of the battery lifetime [334]. Finally, through Man-in-the-Middle (MiTM), an attacker may modify the voltage values of the battery pack, leading to over-discharging and consequent battery degradation [156].

To prevent these attacks, the battery and BMS should be given an identity, and all messages shall provide source authentication and integrity protection. The cryptographic material shall be embedded in these devices, with examples in trusted platform modules [207] or physical unclonable functions [137], and shall not be disclosed during communications to prevent MiTM attacks. An intrusion detection system can help in identifying ongoing attack [265]. Redundant controllers can be employed to enhance the resilience of the BMS against adversarial attacks during charging [46]. Kim et al. [205] proposed to employ blockchain to provide authentication and access control in the communication between the BMS and the other

devices inside the EV. Strategies to mitigate the effect of an attacker who has gained direct access to the vehicle's bus have been proposed [190]. Some works have considered peculiar features of EV to detect spoofing attacks: Guo et al. [155] proposed a physically-guided machine learning method to detect replay and false data injection attacks on the bus. Their system, tested in a Hardware-In-the-Loop (HIL) simulation testbed, could identify the attacks with an accuracy of more than 98%. Finally, to prevent replaying attacks, designers can consider the addition of timestamps to packets and signals transmitted [300].

2.3.2 Controller, Charger, and Electric Motors

The controller and charger are fundamental elements that communicate with the BMS to exchange power to recharge the battery and to feed the EV components with energy. The charger communicates with the EVSE to negotiate the parameters of the charge. Moreover, it manages the energy received and forward it to the battery pack according to the BMS requirements. If bidirectional charging is available to EV, the charger may also deliver energy from the EV battery to the charging column upon request [283]. The controller manages the energy delivered from the battery to the other components. Some of these components are powered by the battery also in petrol-based vehicles, such as the infotainment system or the lights. Others, such as the electric motors, are instead specific to EV. The controller sends energy to them following the driver's input, such as the torque pedal pressure. This section discusses how an attacker may impair their correct functioning.

Man-in-the-Middle. Modifying the data in the bus may disrupt the regular operation of the charger since the control signals are usually transmitted through this channel. An attacker may isolate certain charger components (e.g., the load relay), leading to a surge in the DC voltage. These attacks can damage the battery causing degradation in the performance and shortening the lifetime of the battery [108]. An attacker may also modify the signals managing the electric motors by adding noise or other mutations to the original signal. This attack can damage the correct functionality of the motors and put the driver in dangerous situations [383].

Mitigation techniques can be applied using algorithms that can detect the attack in almost real-time by monitoring the physical properties of the vehicle, such as sensor data [68, 108]. To make the receiver aware of possible MiTM attacks targeting certain packets, integrity protection mechanisms must be in place. Furthermore, intrusion detection and prevention systems that monitor the data exchanged on the bus can also be implemented to strengthen the defense mechanism [265, 365].

Denial of Service. The operations handled by the charger and controller heavily rely on the sensors reporting information on the charging status. A malicious user can generate a large number of requests to the sensors reporting data or overload the charger and control modules by flooding them with packets, thus preventing the receipt of legitimate messages. If not properly handled, this attack may cause the controller to stop receiving correct state information, impairing the overall state control.

Flow control may prevent controllers and motors from handling a large number of fake requests similarly to the BMS, for instance, by employing an adapted version of time-lock puzzles [294]. Source authentication might be employed to verify

the sender's lawfulness [150], while rate limiting can help mitigate against DoS attacks [215]. Furthermore, intrusion detection can be adopted to identify ongoing DoS attacks [389].

Spoofing and Replaying. An attacker may spoof sensor identities to create multiple packets with legitimate identifications. By exploiting the same concept, an attacker may also report false information to the charger and controller. Therefore, the controller may take actions based on false data. This may cause damage to the hardware, possibly impairing the whole charging system [68]. False data, if correctly crafted, may also impact the electric motors' functionality. For instance, they could force a stop of the motors by sending false control signals. Encryption can be a countermeasure to spoofing, preventing a malicious user from freely creating new packets. However, replay attacks can be employed to send correct sensor measurements or actuator updates previously recorded from the bus. An attacker may also spoof the information from the infotainment system, acceleration pedal, or other energy-hungry devices in the EV. The malicious entity may demand a power amount higher than the truly needed one, thus causing higher energy consumption and shortening the battery's lifespan causing the driver to charge the EV frequently. The controller also handles the information regarding acceleration and breaking. An attacker may spoof the related sensors to report false state changes to the electricity supplier. For instance, an attacker may spoof the gas pedal and prevent the receipt of the amount of power needed by the driver to speed up. This may cause safety issues, for instance, when the driver needs to surpass another vehicle.

As already depicted, encryption can only prevent certain kinds of spoofing attacks, but it is insufficient to mitigate replay attacks. To prevent the latter, a combination of unique identifiers and timestamps can be adopted [300]. Identity management may be another fundamental countermeasure against these threats [150]. In fact, the controller and charger need to have, by design, access to the identities of all legitimate components. The identification of attacks is possible using intrusion and anomaly detection techniques [389].

Malicious Code Injection. Similarly to the BMS case, controllers, chargers, and motors also contain software, which may often require updates. However, software updates represent a security challenge since it needs access to the overall EV network [158]. A malicious user may force the installation of a malicious software update to gain control of some components of the EV's internal network [158]. The attacker may thus impact the safety of the driver. For instance, the malware may cause the EV to respond to the driver commands oppositely (e.g., decelerating while pushing on the gas pedal) and may also propagate to all the EV's components. Furthermore, thanks to the malware, the attacker may measure sensitive information on the driver (e.g., location) or profile the driver. A further threat is due to the implementation of controllers and ECUs via Field Programmable Gate Array (FPGA). In this case, an attacker may be able to inject malicious software into the central management system via communication lines by manipulating the FPGA controller [68].

Countermeasures are similar to the BMS case. The access to the EV's internal network shall be strictly regulated, for instance, using strong authentication mechanisms. Remote attestation and its collective extension may be used to validate the in-vehicle components [15], while blockchain can have a role in the verification of new software [40]. Finally, intrusion detection techniques may help the identification of ongoing attacks [389].

Tampering. An attacker might physically tamper one of the controllers, charger, or motor components. In this case, for instance, the attacker may prevent the charger from correctly detecting the presence of a power source (either wired or wireless), thus impairing the possibility of charging the vehicle. This is the case for proximity sensors. For instance, the attacker may attach a shield to the pin on the EV side such that it cannot correctly communicate with the proximity pilot line. Furthermore, an attacker may tamper with the power converter to degrade its quality, causing power losses or overvoltage.

To prevent physical tampering, controllers, chargers, and motors may implement anomaly detection frameworks to detect the application of a voltage in non-safe situations and react to the attack [340]. Furthermore, these devices can be designed as tamper-proof, so they will stop functioning in case of tampering [140]. Although this may impair the vehicle's functioning, it allows for safeguarding the user's safety.

Eavesdropping and Side Channels. The current exchanged during the charging process leaks features that can be exploited for user tracking and profiling [55, 53]. An attacker may attach a module to the charger and controller to collect the current exchanged during the charging process and extract those features, thanks to the absence of encryption methods. For the same reason, an attacker may also eavesdrop on the information exchanged between the controller, the charger, the motors, and the BMS to launch the attacks mentioned above. An adversary may also analyze the power exchanged between the controller and the infotainment system to obtain users' sensitive information, such as preferences, habits, and passwords. This attack has been shown in other scenarios [226], such as smartphone charging, where users' activities can also be detected in case of encrypted traffic [90]. Therefore, it is fundamental to include methods to prevent malicious users from accessing the communication among these entities.

To guarantee the user's privacy, the current exchanged may be altered via a noisy signal that hides the original signal's features similar to differential privacy in other contexts [114]. On the receiver side, the components shall be able to guarantee that the input current does not cause any damage to the circuitry. These solutions may hold for all the involved sources of current. Cryptographic methods may not always represent a viable solution, as they would add computational overhead to a possibly safety-critical system. Furthermore, they do not represent a solution to side channels, which are challenging to mitigate [200].

2.4 EV Charging Security and Privacy Challenges

This section discusses the security and privacy issues related to the EV charging process. In particular, we discuss the challenges associated with wired charging in Section 2.4.1. Then, we discuss the challenges associated with WPT and WIPT in Section 2.4.2. For both technologies, we also discuss possible solutions and countermeasures. In Table 2.3, we summarize all the security and privacy challenges associated with the charging process, their effects, impact severeness, and possible countermeasures.

2.4.1 Wired Charging Challenges

In the following, we focus on attacks targeting a wired charging scenario, which is the most common way at the moment of writing. Some of the attacks are specific

TABLE 2.3: Summary of EV Charging Challenges. S. = System.

S.	Attack	Effect	Impact	Possible Solutions
Wired Charging	Tampering	Prevent charging Cause a shock to the driver Get sensitive information	High	Tamper-proof hardware Inconsistencies handler
	Energy repudiation	Cheat on billing Steal energy from the system	Low	Aggregate signature schemes Blockchain for energy transactions
	Denial of Charging	Prevent EV charging Disruption of the charging service	Medium	Identity verification Authentication Intrusion detection
	MiTM	Prevent proper charging Modify charging parameters	High	Integrity protection Encryption
	Spoofing and Replaying	Create charging state inconsistencies Steal energy from another EV	Medium	Identity management Authentication Encryption Timestamps
	Relaying	Steal energy from another EV	Medium	Distance bounding Fingerprinting
	Eavesdropping	Steal sensitive EV information	Medium	Encryption
	Side Channels and Information Leakages	Track user Profile user's preferences	Low	Differential privacy Secondary batteries
Wireless Power Transfer (WPT)	Overpower	Damage to EV battery	High	Overvoltage protection Anomaly detection
	Jamming and DoS	Prevent EV charging	Medium	Channel hopping Identity verification Authentication Intrusion detection
	Freeriding attack	Steal energy from the system	Low	Authentication Blockchain
	Energy repudiation	Cheat on billing Steal energy from the system	Low	Aggregate signature schemes Blockchain for energy transactions
	Spoofing and Replaying	Create charging state inconsistencies Steal energy from another EV	Medium	Authentication Encryption Physical layer authentication Timestamps
	Relaying	Steal energy from another EV	High	Distance bounding
	MiTM	Prevent proper charging Modify charging parameters	Medium	Integrity protection Encryption Physical layer authentication
	Eavesdropping	Steal sensitive EV information	Medium	Encryption
	Side Channels and Information Leaking	Track user Profile users' preferences	Medium	Differential privacy Secondary batteries

to cases where HLC is available (e.g., MiTM, spoofing), while others are suitable for every type of wired charging, such as tampering or side channel analysis.

Tampering. In this attack, a malicious user physically tampers with the devices involved in the charging process. In particular, an attacker might manipulate the pilot lines and tamper with the proximity sensor to prevent an EV from deeming a secure connection and hence prevent charging. Furthermore, this can also impact users' safety, as it might be possible to detach the cable before removing the current. By observing electromagnetic leaks or operations in the chip components both in the EVSE and EV, an attacker might infer sensitive information on the user, such as private keys used for billing purposes [148]. Furthermore, by tampering with the charging cable, an attacker might prevent the proper charging of the victim EV or steal energy from an EV in charge by connecting additional cables [30, 88].

As possible countermeasures, tamper-proof hardware may represent a viable solution [140, 7]. Thanks to these devices, the attack may be limited to the car functioning without impacting the users' safety. Lastly, proper inconsistency handling mechanisms may be implemented to check that all involved components report the same physical status.

Energy Repudiation. A malicious user may report to the EVSE that the EV's battery did not receive any power by exploiting the behavior of the pilot line and the feedback associated with it. In this situation, the attacker may be able to charge a smaller amount compared to the amount of energy effectively used. If bidirectional charging is available, an attacker may pretend to have sold more energy than it has actually sold, thus stealing money from the energy provider.

A possible countermeasure to energy repudiation is the use of blockchain technology to handle transactions and guarantee traceability and non-repudiation [191, 174]. Aggregate signature schemes from different physical components can represent another possible mitigation to the problem [392].

Denial of Charging. A malicious actor may try to prevent a vehicle from charging. It may be done at the data level by modifying values on the packets exchanged during the handshake between the EV and EVSE [21]. In some cases, DoS can also be performed remotely, exploiting unshielded cables, which are often used for the recharge [209]. DoS may also be launched against more than one vehicle, trying to compromise a portion of the grid. A greedy attacker may falsify the information on the battery's SoC, such that s/he can demand an energy amount higher than needed, thus preventing other users from benefiting from the service. The number of users that can simultaneously charge their EVs and the energy effectively delivered each moment depends on the grid's capacity. If the grid capacity is limited, the attacker can successfully launch this attack and prevent other users from charging.

Possible countermeasures to DoS attacks include low-complexity authentication services in all the packets exchanged such that the EVSE can rapidly decide whether to accept or discard a request. Identity-based traffic filtering may be combined with a physical state update related to the charge level of a certain user to prevent multiple malicious requests. Intrusion detection can be employed to detect ongoing DoS attacks which may generate strange communication patterns [389]. Furthermore, enforcing physical security by adopting shielded cables can prevent some kinds of DoS and eavesdropping attacks [209].

Man-in-the-Middle. When operating charging modes employing HLCs, such as CHAdeMo or ISO 15118, the EV and EVSE exchange data through network packets. A MiTM attack can be employed to modify the content of this communication. It may be a consequence of tampering if the malicious actor can insert a device on the pilot line between the vehicle and the charging column. In some cases, MiTM can be performed from other charging columns attacking the SECC Discovery Protocol [113]. A malicious actor may exploit this channel to manipulate the exchanged information and create inconsistencies in the recharging process. For instance, an attacker who can modify packets on the fly may prevent proper charging by modifying request and response parameters. Further attacks can be launched starting from MiTM, such as malware injections or DoS [21].

To identify modified data, integrity protection can be added to packets [9]. Another possible countermeasure is encryption. Novel versions or ISO 15118 mandates

the usage of Transport Layer Security (TLS) for all the communications between the vehicle and charging column, even if in real life, data are often exchanged in plaintext [30].

Spoofing and Replaying. An attacker might interact in the communication link between the vehicle and the charging column by injecting packets spoofing other devices' identities. For instance, a malicious user can spoof the identity of an ECU and report false information on the battery SoC. Furthermore, an attacker may inject false information by spoofing the identity of an EVSE and stealing sensitive information from an EV. For example, in the case of automatic billing based on the EV features, a malicious user can extrapolate those features from an EV and store them for later use to bill the victim. The same concept can also be applied to other types of connectors, as long as billing is based on automatic feature recognition [120].

Possible countermeasures to these attacks include using a proper identity management scheme, authentication, and data encryption [67]. Authentication systems shall include information related to the charging status of the EV or the energy delivered by the EVSE to help guarantee the consistency between the reported information and the actual physical state. It is important to consider that encryption cannot prevent the replaying of packets, which may instead be enforced with unique identifiers and timestamps [300].

Relaying. A relay attack is possible if an attacker has access to the network traffic and can relay it to a nearby charging column. By relaying information, a malicious user can manipulate the billing system. For instance, a malicious user can relay the data between two neighboring EVSEs to bill a closely-located victim user for a charging session [88]. If bidirectional charging is available, a malicious user can sell the energy of a victim's vehicle and get paid for it.

The location information of EV and EVSE may be exploited to prevent relay attacks, e.g., employing distance bounding protocols [65, 88]. Furthermore, the physical features of the EV may be exploited to design dedicated authentication protocols [66, 172].

Eavesdropping. An attacker may be able to read the information exchanged between the vehicle and charging columns in different ways, similarly to what was presented before for MiTM attacks. With access to all the network traffic, a malicious entity can steal sensitive information from the user, from simple charging parameters to credit card numbers.

To protect against eavesdropping, encryption can be applied. As already explained, novel versions of ISO 15118 mandate the usage of TLS for all the communications between the vehicle and the charging column, even if real-life data are still often exchanged in plaintext [30]. It is important to recall that even if the exchanged data are encrypted, side channel analysis is possible to extract some users' preferences, as presented in the following section.

Side Channels and Information Leakages. An attacker in control of an EVSE may be able to track and profile users who authenticate to the EVSE even if data are encrypted. It may rely on different information, such as the MAC address of the EV or the certificate employed by Plug and Charge [183]. However, in Level 1 and Level 2 charging, these kinds of data are unavailable since no HLC is generated between the two entities. In that case, an attacker may rely on other features, such

as the exact voltage of the control pilot pin or the duration of the handshake at the beginning of the charging process [172]. Another side channel that may transfer information is the effective current exchange. This does not convey information in a network sense, i.e., it does not involve the creation of packets with the sender's and receiver's information. Therefore, no encryption method is applied to this signal, which is transmitted in plaintext. However, it has been shown that it is possible to profile users by extracting features from the charging current [55, 53]. In particular, the charging current contains features peculiar to each EV, allowing for EV tracking and user profiling based on the current demand. Therefore, it is fundamental to manipulate the current signal to prevent these attacks.

Countermeasures to privacy threats shall not undermine the efficiency of the charging process. Therefore, possible solutions must allow the involved parties to retrieve sufficient information, e.g., to the SoC. Differential privacy methods may represent a viable solution [165]. An alternative is represented by the use of secondary batteries to create a connection between the EV and EVSE, similarly to what was discussed in [55, 53]. When HLC is available, MAC address randomization may represent a good mitigation technique to reduce the profiling power of an attacker.

2.4.2 WPT Challenges

Due to the exposure of the wireless medium, WPT incurs in a large number of safety, security, and privacy issues. In fact, it is likely that WPT signals to impact more vehicles and that an attacker gets access to the signals or information wirelessly exchanged [232]. In this section, we review and extend the taxonomy of the possible attacks to WPT presented in [232] and we adapt it to the EV case.

Overpower attack. The wireless medium's intrinsic vulnerability makes it possible that a single EV receives both its signal and the signal intended for another vehicle. For instance, if two cars are closely located, and both are charging their batteries via WPT, they will receive more power than expected. This is even more likely when considering fully-dynamic WPT, where vehicles move and cannot hence guarantee that a reasonable safety space is kept between them. The excessive received power might harm some components of the BMS or the battery if a proper overvoltage regulator is not deployed. Furthermore, an attacker might exploit this concept to launch an overvoltage attack to damage the EV's components.

Possible countermeasures include implementing overvoltage protection mechanisms at the EV's side. Such mechanisms shall, however, guarantee the efficiency of the charging process to avoid requiring excessive charging times. Anomaly detection methods can also be applied to detect the reception of abnormal power values or other anomalies in the charging process. Design choices can help mitigate overpower attacks. For instance, the distance between coils must be designed to make overpower attack unfeasible or, at least, more complicated.

Jamming and Denial of Service In the case of WIPT, the reception of multiple signals might cause excessive interference at the receiver's side, thus preventing the correct reception of messages. Due to the openness of the WPT medium, an attacker might be able to simultaneously jam multiple EVs by sending random WIPT messages and degrading the channel quality up to the point where messages are not correctly received. Furthermore, this concept can be exploited to prevent a successful charging negotiation phase, thus preventing a connection between the EV and the charging system. This represents a DoS attack. Similarly, an attacker may launch

a jamming attack against the charging column's WiFi access point, preventing legitimate users from connecting and using the service. An attacker may also target a portion of the energy grid by continuously sending charging requests. If many users engage in this session, they might prevent other users from benefiting from the service availability. Although feedback mechanisms to report on the SoC of the receiver might be implemented to automatically detach an EV when fully charged, an expert attacker might be able to craft feedback packets to avoid showing full battery's SoC.

Possible solutions include frequency hopping mechanisms, where channels are selected according to different strategies to avoid using a channel under jamming attack [149]. Low-complexity authentication services in all the packets exchanged such that the EVSE can rapidly decide whether to accept or discard a request can help in preventing DoS attacks. To detect a DoS attack, intrusion detection systems can be deployed [389]. Identity-based traffic filtering may be combined with a physical state update related to the charge level of a specific user to prevent multiple malicious requests.

Freeriding attack. As previously mentioned, a user might connect to public infrastructure and pay for charging via WPT. Due to the openness of the WPT medium, a malicious user could exploit the proximity to a vehicle in charge to steal energy and charge his/her EV. A similar scenario envisions the collusion of multiple EV owners when a single one registers for the service and multiple users share the bill and benefit from the charging process. These attacks are feasible in all types of dynamic WPT models; the only requirement is a short inter-EV distance. This attack is challenging to detect, as it does not impact the legitimate channel. In fact, although a second EV might be connected to the charging channel, the main channel will not face any performance degradation, thus making it unfeasible to detect the attack.

WPT sessions need to be authenticated to prevent other users from benefiting from a charging session they are not paying for. Furthermore, authentication procedures might include the physical features of the involved devices and the amount of power transferred. The blockchain solutions proposed by Jiang et al. [188] may be adapted to the EV case to guarantee security against this attack.

Energy repudiation. WPT is less efficient compared to its wired counterpart, as the wireless medium is characterized by losses due to both attenuation and the relative position of the transmitter and receiver devices. Therefore, part of the transmitted energy may be lost during the charging process. A fair system requires that users pay for the energy they actually receive. Therefore the billing system needs to compare the transmitted power with the received one. However, this might create security issues. In fact, a malicious user might continuously report a received power value smaller than the true one or report zero received energy. This is commonly known as a repudiation attack, where the user denies benefiting from a service.

To guarantee the correctness of the reported power usage information, possible solutions might include the use of aggregate signature schemes from different physical components [392] or the blockchain technology [174, 188].

Spoofing and Replaying. A malicious user who knows the standard employed or who is able to eavesdrop on the communication can easily craft malicious packets. Based on the crafted information, this class of attacks may have different impacts on the system. For instance, an attacker may use the identifier of another vehicle to negotiate a charging session for which the victim will pay. Furthermore, a malicious

actor can craft packets declaring weird SoC and spoof other vehicles' identifiers to create inconsistencies in the charging process.

The use of authentication and integrity protection mechanisms can be effective countermeasures against spoofing. In this context, using physical layer authentication may help in designing suitable protocols [379]. In the context of WPT, the transmission frequency can be regulated to encrypt information and guarantee that only the legitimate party can receive power [396]. This also represents a possible solution to the attacks aforementioned in this section. Finally, timestamps can be added to identify multiple sending of the same packet in a replaying attack [300].

Possible solutions include the use of cryptographic techniques to hide information. The newest release of ISO-15118 [184] mandates TLS on every communication.

Relaying. An attacker may relay information from a victim vehicle to the access point of the attacker's charging column to steal energy [88]. This kind of attack works even if the traffic is encrypted since the data is only relayed and not modified. With respect to the wired counterpart, where the attacker has to tamper in some way with the charging column, a wireless relay attack does not need any hardware modification.

To protect against relay attacks, a distance bounding protocol can be employed to assess if a malicious entity is relaying the network flow [88].

Man-in-the-Middle. With respect to wireless charging, when dealing with WPT, the interception and forwarding of communication flow are easier due to the openness of the medium [176]. At the same time, directional jamming can be employed in some cases to prevent the receiver from getting both the original and the modified data. If the communication flow is unencrypted or the cryptography is weak, an attacker can launch a MiTM attack to modify on-the-fly packets. For instance, a malicious user might modify the information sent by the victim (i.e., report full SoC) after establishing a connection with the service provider. To perform such an attack, a malicious entity may set up a fake access point and use it to relay the communication to the legitimate one, gaining the ability to modify packets at will.

Partial solutions include the previously mentioned solutions, such as encryption, authentication, and integrity protection mechanisms. In the context of WPT, the transmission frequency can be regulated to encrypt information and guarantee that only the legitimate party can receive power [396]. Furthermore, physical layer authentication may enhance the security of the authentication process [276].

Eavesdropping. Due to the exposure of the wireless medium, an attacker may easily intercept WPT packets. These packets may contain different types of information, such as the vehicle identifier, SoC information, or billing information.

Side Channels and Information Leaking. Although WPT signals might be encrypted or avoid sensitive reporting information on the user, the power signal can be exploited for profiling purposes. This attack has been proven feasible for smartphones, where the WPT signal analysis reveals information on the user's activity [218]. This might also be the case for EVs, where an attacker can infer different types of information. This attack is similar to the profiling performed in the wired case, where tracking a user and obtaining information on his/her habits and power demands might be possible. Preventing this attack represents a challenging task, as it cannot be detected, and data encryption is not sufficient [304].

TABLE 2.4: Summary table with attacks and countermeasures for each asset. The first row indicates the assets interested in each attack, while the following rows point out which countermeasure is effective against each attack. : BMS and battery; : Controllers, charger and motors; : wired charging; and : WPT.

Attack / Countermeasure	DoS	Tampering	MitM	Replaying	Spoofing	Malware	Overpower	Freeride	Jamming	Repudiation	Eavesdropping	Side-Channels	Relaying
Affected assets	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Aggregate signature										✓			
Anomaly detection		✓					✓						
Authentication	✓		✓		✓	✓		✓					
Blockchain								✓		✓			
Channel hopping									✓				
Cookies	✓												
Differential privacy											✓	✓	
Distance bounding													✓
Encryption			✓		✓						✓		
Fingerprinting													✓
Flow control	✓												
Intrusion detection and prevention	✓		✓	✓	✓	✓							
Identity management			✓		✓								
Identity verification	✓		✓		✓								
Inconsistencies handler		✓											
Integrity protection			✓										
Overvoltage protection							✓						
Physical layer security			✓	✓	✓								
Rate limiting	✓												
Redundancy	✓		✓	✓	✓								
Remote Attestation						✓							
Secondary battery												✓	
Tamper-proof hardware		✓											
Time-lock puzzles	✓												
Timestamps				✓									

A possible solution is represented by differential privacy, where data is corrupted with a noisy pattern that might prevent inferring sensitive users' data [114]. Furthermore, as for the wired counterpart, secondary batteries may prevent the attacker from inferring sensitive users' information.

2.5 Future Directions

Looking at the impact of the different attacks in Tables 2.2 and 2.3, we can conclude that many security issues related to the cyber-physical nature of EVs may impact the safety of the driver. We notice that some of the attacks and countermeasures discussed can also be applied to other EV assets. However, to avoid repetitions, we only discussed those we considered to be the most relevant. Nevertheless, we summarize all the attacks and countermeasures in the comprehensive Table 2.4. Although many threats concern the charging infrastructure, the most severe in terms of safety are related to the in-vehicle network. In fact, the electric component of EVs may be tampered with or impaired to electroshock the user. Furthermore, the increasing cyber nature of the EVs' components leads to challenges regarding the coherency of the information coming from the cyber and the physical worlds. Lastly, the increasing interest in the application of the WPT technology to EVs impose significant challenges that still need to be properly addressed from a CPS point of view.

Based on our analysis, we foresee the following future directions and needs in the field of EV CPS security.

- Denial of Service (DoS) represents one of the most challenging threats in EVs security. It is known as difficult to prevent, and almost every component of the EV can suffer from it. Compromised internal components can attack other ECU to compromise the in-vehicle network, but DoS attacks can be launched from charging columns to EV during charging, or vice-versa. In certain cases, DoS can have an impact not only on a single vehicle but can compromise EVSEs in a certain geographical area. To mitigate this risk, not only do all the vehicle entities need to be associated with an identity, but their allowed flow of information (and hence generated traffic) should depend on the vehicle's physical situation. In fact, it might be that a specific ECU needs to send messages at a higher rate when the vehicle is experiencing certain physical stimuli. At the same time, all ECUs shall be guaranteed a sufficient amount of resources. Therefore, future protections against DoS for in-vehicle networks should account for the physical factors and the possible impact on the whole electric grid.

- The potential tampering with the EV components might represent a significant threat to the user's safety and may also have repercussions on other elements of the vehicle system. All vehicle components shall be equipped with anomaly detection capabilities or should prevent the application of a voltage or current flow in case of tampering. Possible future solutions might include the collective verification of multiple components to make tampering with a single unit ineffective.

- The increasing attackers' capabilities impose additional challenges in guaranteeing the cyber-security of EVs. In fact, an attacker might be able to combine multiple attacks to impair the EV functioning. To strengthen the defense mechanisms, it is essential to implement in EVs frameworks collecting information from multiple sources, combining the cyber and the physical world. For instance, verifying the message integrity might employ data from different sensors and actuators to increase the difficulty of information manipulation. Similarly, intrusion detection techniques might combine network data exchanged through the bus with physical signals from sensors to model better the state of the EV. This will also help prevent attacks related to malicious ECUs controlling actuators for mechanical operations (e.g., steering). Future work should consider the EV-specific components such as the battery or the charger as data sources regarding the vehicle's state. For instance, the charge and discharge curves of batteries can be modeled by computers with discrete confidence [212, 303, 73]. A simple application of these simulations is a reference to identify packets declaring modified SoC.

- One of the strengths of EVs compared to previous generations of vehicles relies on the software managing all their operations. However, this implies that vehicles are more subject to cybersecurity attacks. Some of these attacks may include malware injection into some of the EV's components. To this aim, collective remote attestation can be used to verify the integrity of all the EV's components and prevent possible safety threats. Remote attestation measures should, however, account for the resource-limited nature of EVs' components and the time-critical nature of the exchanged information.

- WPT is one of the promising technological solutions to alleviate the range anxiety of drivers fearing not reaching their destination with the available charge.

Thanks to the charging while driving paradigm, EVs can be charged during their operation. However, deploying the required public infrastructure poses many security challenges both to the operators and the users. Some examples include the billing process and the openness of the wireless medium. WPT related challenges heavily rely on the cyber-physical nature of the overall infrastructure. Therefore, security solutions in this area should account for the coherency of information from the cyber and the physical domains.

2.6 Conclusion

The increasing market for EVs demands an in-depth analysis of EV technology's security and privacy challenges. In this chapter, we provided an overview of the components of an EV, focusing on their characteristic components. We provided the basic information needed to understand how in-vehicle communication networks work and which devices need to communicate with one another. We then discussed how an EV battery could be charged via wire and WPT. We provided the information needed to understand both technologies and discussed the different implementations. We also provided the security and privacy issues of in-vehicle communications and those related to the charging infrastructure. Focusing on a CPS perspective, we discussed how different attacks might impact both the user and the system's security and privacy. We then discussed possible countermeasures and proposed some future direction to improve the overall EV ecosystem security and privacy. We conclude that the EV technology currently presents a large attack surface that users with malicious intents can exploit. Therefore, it is fundamental to develop technologies considering the CPS nature of EVs to provide full security.

3 EVExchange: A Relay Attack On lectric Vehicle 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.

With such a forecast on the increase of EVs [351, 376, 289], the energy request from the electric grid will grow greatly in the next years. 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 [333]. 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 [17, 263]. 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 [30]. 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 delivering of the charge service to the users. Other exploits which have been proved to be effective in the V2G scenario include the profilation of the battery behavior [339] and the profilation of the vehicle charging process [53] based on the electric traces generated from the charging process.

Contributions. In this chapter, 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 by the attacker. We implemented *EVExchange* in both an emulated scenario employing MiniV2G [21] and in a physical testbed composed of different Raspberry Pi [291], proving its functioning and effectiveness. Finally, we propose an extension of the International Organization for Standardization (ISO) 15118 protocol (i.e., the standard protocol in V2G communication) [183] 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. The contributions of the paper are summarized as follows:

- We propose *EVExchange*, the first relay attack conceived for V2G communication.
- We implemented *EVExchange* in simulated and emulated scenarios based on the ISO 15118 charging protocol standard.

- We prove the effectiveness of *EVExchange* in stealing the power intended for the victim's car.
- We propose a countermeasure that allows early identification of relay attacks such as *EVExchange*. We tested such countermeasure under different scenarios and conditions, proving its effectiveness.

Organization. The remainder of the chapter is organized as follows. Section 3.1 briefly recalls the main concepts useful for the goal of the chapter, while Section 3.2 provides an overview of the related works. Section 3.3 outlines the system model and the adversary model considered. Then, Section 3.4 presents the *EVExchange* attack and its implementation, while Section 3.5 describes the proposed countermeasure. Finally, Section 3.6 concludes the chapter with some final remarks.

3.1 Background

This section overviews the basic concepts related to the electric vehicle charging system from a communication perspective. In Section 3.1.1, we introduce the V2G paradigm, while in Section 3.1.2 we analyze the most advanced standard in this field. Then, in Section 3.1.3 we recall the concept of relay attacks.

3.1.1 Vehicle-To-Grid (V2G)

The Vehicle-to-Grid (V2G) concept refers to how an EV 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 [367]. 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 the 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 [333].

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 [84].

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 EVSE are ISO 15118, SAE J2847, and CHAdeMO. In the back-end communication between EVSEs and control centers, ISO 61850 and OCPP are the most used [269].

In this chapter, we uniquely focus on front-end communication. Nowadays, CHAdeMO can be considered the defacto standard. It enables communication through a CAN 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 PLC 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 the ISO 15118 [183, 180]. It supports both AC and DC charging and shares the same communication means of SAE J2847, making it possible to employ the same infrastructure partially. Since ISO 15118 can support a vast number of services, ranging from authentication to vehicle's firmware update [58], it aims to be implemented globally and become the standard for the future of electric mobility.

3.1.2 ISO 15118

Firstly released in 2013, ISO 15118 is a modern standard for the regulation of the communications between the Electric Vehicle Communication Controller (EVCC) and the Supply Equipment Communication Controller (SECC). EVCC and SECC are, respectively, the endpoints that manages the transmission on the EV and EVSE [183, 180]. It defines a communication channel via PLC on the Control Pilot (CP) of the IEC 62196 connectors [86].

At the beginning of the connection, the Signal Level Attenuation Characterization (SLAC) 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 (HCP) starts using a TCP communication, generally ciphered using TLS. More information on the packets exchanged can be found in [21].

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 that supports External Identification Means such as credit cards, RFID cards, or QR codes. Furthermore, ISO 15118 provides a highly comfortable service called Plug and Charge (PnC). This mechanism allows the user to be automatically accounted for the energy requested without using a card or other payment means at the moment of the recharge. In this way, the user only needs to insert the plug in his vehicle socket to start the charging process. The PnC authentication mechanism employs the TLS certificate installed in the vehicle and used by the charging system to identify the car [208]. The owner can obtain their personal certificate by registering with a charging service provider, as defined in the ISO 15118 standard [183]. However, as we will see in this chapter, PnC can expose the user to some security threats.

3.1.3 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 [162]. It differs from aMiTM 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 the case of transmission of 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 the receiver perform mutual authentication, and then all the 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 account the victim for the payment.

3.2 Related Works

Although electrical vehicle charging systems are a novel topic, various research papers have examined various aspects of their security. Mustafa et al. [263] 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. [17] showing possible countermeasures for ISO 15118 and OCPP. Other works addressed specifically the ISO 15118 standard [32, 222] proposing threats analysis and security mitigations. 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 related to 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 exploiting the electromagnetic emissions of the PLC on an unshielded cable [30]. Hofer et al. [169] focused on privacy aspects 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. [130] designed HIP, a backward-compatible protocol extension for ISO 15118, which enables the generation and storing of keys in a Trusted Platform Module (TPM) 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.

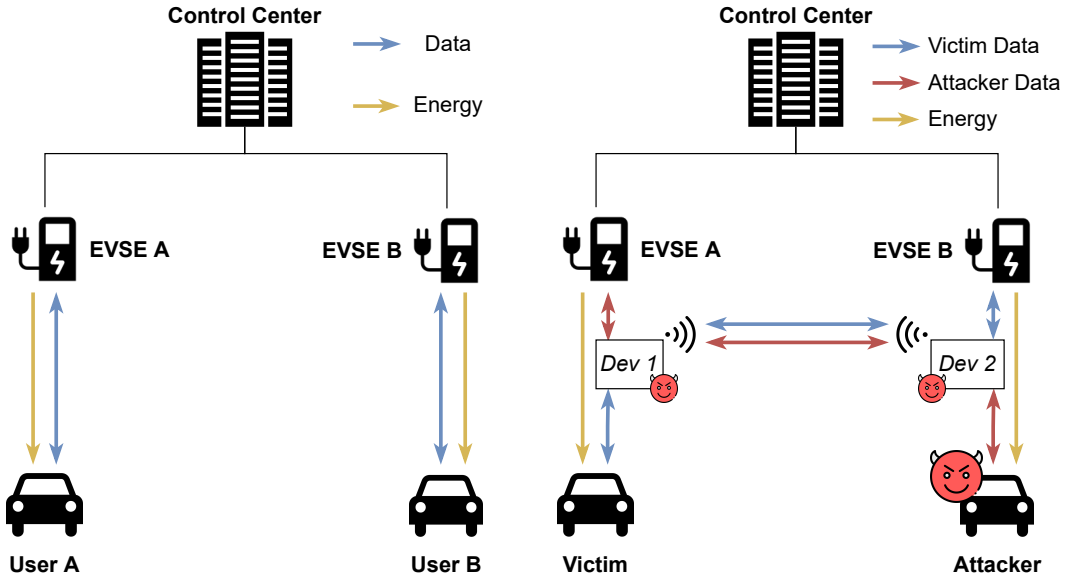
There are many scenarios in which relay attacks are used. Its application on Near Field Communication (NFC), for instance, is analyzed in different works in literature [127, 63]. Recently, researchers have successfully proved the effectiveness of a relay attack on the SARS-CoV-2 contact tracking application, proposing a hashing-based countermeasure to secure the environment without losing privacy [62]. 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 [126]. In [305], the authors propose a solution to enforce the relay resilience of cryptographic protocols in such applications based on a crypto-chain framework. While there are numerous studies focused 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.

3.3 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 we detail the assumption an attacker must respect to implement *EVExchange*.

System Model. Figure 3.1a 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. If more than two EVSEs are available, the attack can be easily extended. However, in this work, we focus 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 [120] which employs the MAC address of the EV and is commonly used in North Europe.

EV and EVSE are connected via wired cables, which is the most common setting for power and data, which 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 [363]. There are no substantial differences between them for the purpose of this chapter, as soon as the communication is established and billing data are transmitted through the cable in the CP pin. It can also be possible to 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 currently rarely used in the real world.



(A) Without attack. (B) With *EVExchange* attack.

FIGURE 3.1: 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.

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 3.1b. The two devices can be two simple microcomputers (e.g., Raspberry Pi) with two interfaces to demodulate the PLC in the CP pin and WiFi connection capabilities. A highly skilled attacker could design an ad-hoc device to minimize the device's size so that it remains 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

two PLC interfaces of the device. The best solution depends on the type of charging column.

Furthermore, the two devices must be connected with 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 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 [180]. However, as reported in [30], this security measure is often not implemented in practice, exposing the users to many security issues [21]. 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, as required by modern charging standards (ISO 15118-20 [184] and OCPP v2.0.1 [12]). 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 3.1b). In this way, the attacker can control the energy supplied by the victim's EVSE and vice versa.

3.4 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 3.1b as 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 are required to set the charging options they need (e.g., time of departure, 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, they 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. As an example, the forecast duration of the charging process is variable based on the state of charge, the charger type, and the time of the charging. 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 the 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, they 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 of their 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), they 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 3.2.

0. The attacker places the two devices as depicted in Figure 3.1b;
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, they stop the attacker's charging process.

3.4.1 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 [84]). In this case, since the attacker controls the victim's communication with the EVSE, they can decide to sell to the grid the power contained in the victim's battery, leading to a complete discharge. Moreover, by doing so, the revenue from the sale will be credited to 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 [279]. In [53] 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 [377]. Since the attacker controls the victim's charging parameters, he or she can overcharge the

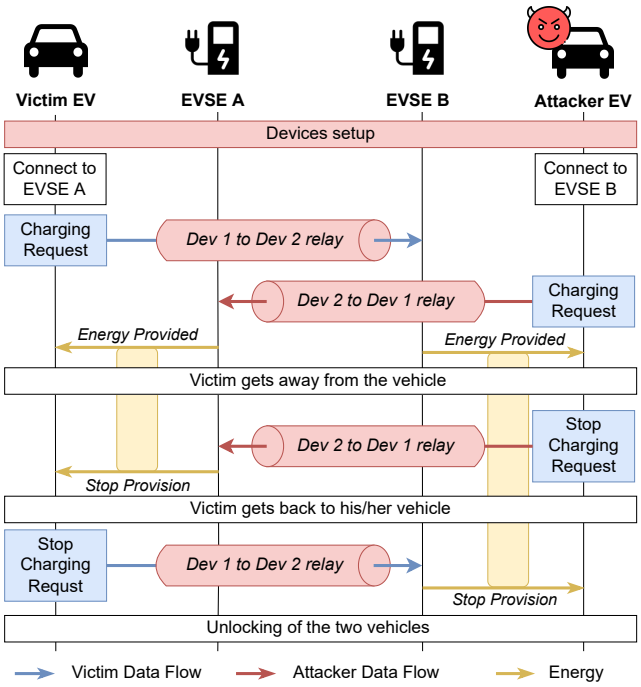


FIGURE 3.2: The different phases of the *EVExchange* attack. We represent the Unidirectional V2G scenario for simplicity.

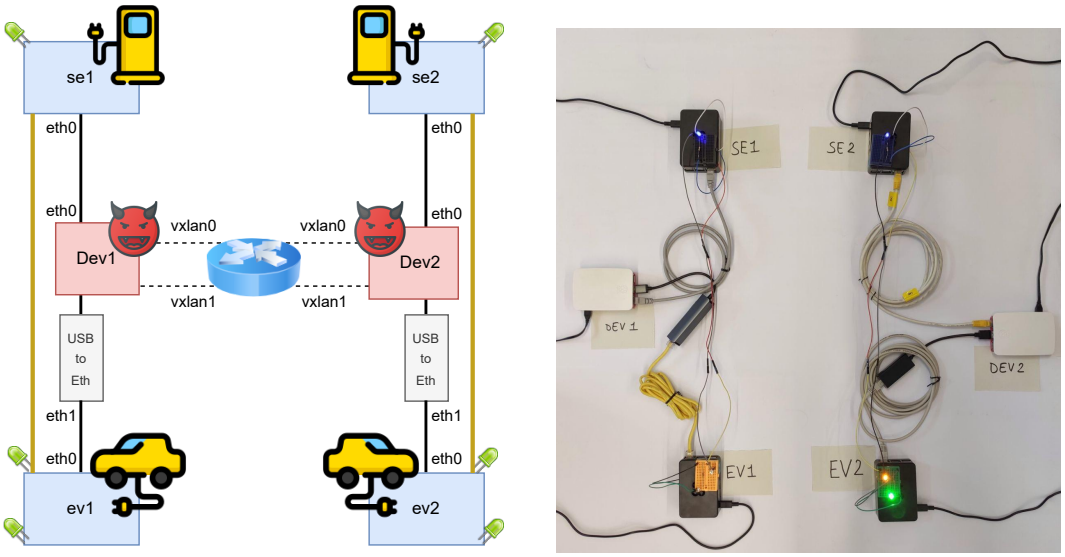
battery by requiring energy even if the battery is full. If the bidirectional charge is available, full discharge can be performed as well. Furthermore, an advanced attacker could modify the EVCC or, more simply, modify packets with battery status on the fly to send abnormal charging parameters to the victim's charging column requiring an amount of energy that may damage the battery.

3.4.2 Attack Validation

The EV charging infrastructure is complex to reproduce and manage since it involves different technical aspects, from the energy to the 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 [21], an open-source emulator able to simulate networks of EVs and EVSEs. MiniV2G is built on top of Mininet-WiFi [123], a popular software to create realistic virtual networks, running real kernel, switch, and application code. Furthermore, MiniV2G includes RiseV2G [83], 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 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 PLC communication while we employed GPIO pins to emulate the energy exchange. We install LEDs to monitor the different stages (i.e., battery charging,

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



(A) High-level architecture of the testbed.

(B) A picture of the testbed.

FIGURE 3.3: The testbed employed to test *EVExchange* attack and countermeasure.

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 3.3a represents a high-level schema of the testbed, while Figure 3.3b illustrates a picture of the testbed developed.

To connect the malicious devices and allow the packets forwarding, we employ Linux `bridge` [166] 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) [241]. 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 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.

3.5 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 3.5.1, we design the distance bounding protocol, while in Section 3.5.2, we discuss the security and the limitation of the proposed algorithm. Then, in Section 3.5.3, we describe an implementation of the protocol, providing some numerical results.

3.5.1 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* [49]. 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 [112], NFC devices [167, 353], and Passive Keyless Entry [386]. This protocol is well suited to work at the application layer in preventing relay attacks since these threats inevitably introduce a measurable delay in the 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*) [357]. 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. [353], 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. Figure 3.4 graphically summarizes the protocol's steps.

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 using TLS between the two entities can start as depicted in ISO 15118. Before actually exchanging charging parameters and setting, SE sends to EV the string $S_{SE} = \{\tilde{\alpha_1}, \beta_1, \ldots \tilde{\alpha_k}, \beta_k\}$.

6. EV computes $S_{EV} = \{\alpha_1, \tilde{\beta_1}, \ldots, \alpha_k, \tilde{\beta_k}\}$ and compares S_{EV} with S_{SE}. If the two strings differ, an alert is raised since an attacker might have forged some packets.

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

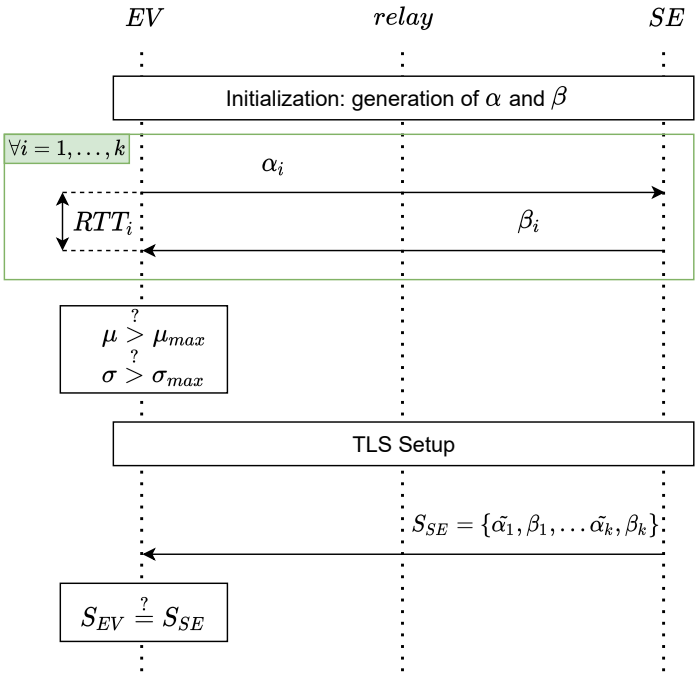


FIGURE 3.4: The different steps of the distance bounding protocol.

3.5.2 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 3.4, 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 *RTT*s. 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 *RTT*s, 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 [367], making even more identifiable a wired relay. An alternative is to employ faster wireless communication modes with respect to the IEEE 802.11 standard, 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 [210]. It is worth noticing that the PLC employs HomePlug Green PHY, which has almost

no delay at the MAC layer when applied between two entities only [82], 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.

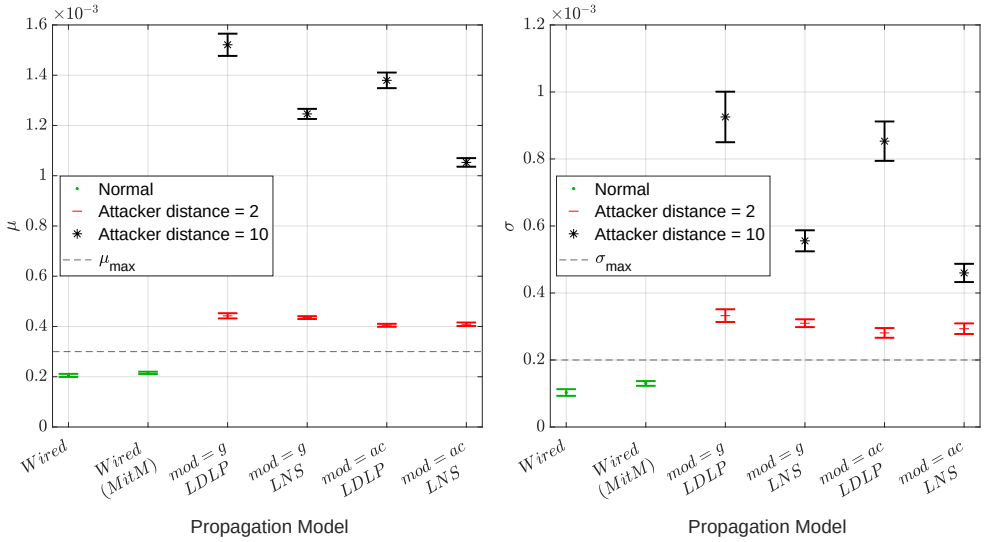
Note that the proposed protocol does not try to prevent *relay* from knowing both α and β. Instead, it imposes bounds on the *maximum time* by which the information must be received. In other words, when *relay* read the packet containing β_i, it introduces a delay that makes it too late for the forwarding 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 the legitimate entities check the validity of their counterpart's certificates before starting the charging process.

3.5.3 Evaluation

To implement the distance bounding algorithms, we wrote two Python scripts to be executed in the EV and the 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 the 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. The validation was done on different scenarios and performed in a virtual machine with Ubuntu 20.04.2 LTS x64 and 2GB of RAM. The most important parameter that governs the attack's success or failure is the distance between the two malicious devices. We consider two scenarios: two EVSEs at the opposite ends of a parking lot ($10m$) and two adjacent parking spots ($2m$). To emulate a wireless connection in the emulator, we employ different propagation models included in Mininet-WiFi [123].

We chose as possible models Log Distance Path Loss (LDPL) and Log Normal Shadowing (LNS), both with $exp = 2$. As presented in [11], these two models are suited to simulate a connection in free space and urban area. Furthermore, we test with two different WiFi versions (i.e., IEEE 802.11g and IEEE 802.11ac).



(A) Mean *RTT* (μ) (B) Standard deviation of *RTT* (σ)

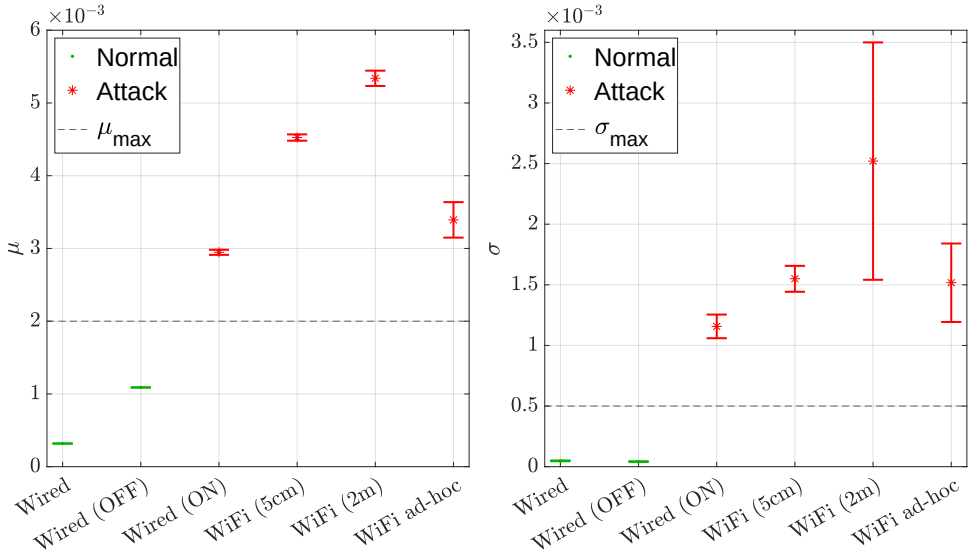
FIGURE 3.5: 99% confidence interval for the mean (a) and standard deviation (b) in each different scenario emulated using MiniV2G.

We represent the mean *RTT* in Figure 3.5a and the standard deviation of the *RTT* in Figure 3.5b. As in the data presented in Section 3.5.3, the error bar represents the 99% percentile, and there is a clear separation between the wired data and all the other malicious cases.

After the emulation on MiniV2G, we moved to a physical testbed with different distances between the devices. 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 3.6a and the standard deviation of the *RTT* in Figure 3.6b. 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 3.6: 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 0.06*s* using 100 fast exchanges, with tops of about 0.3*s* 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 chapter.

3.6 Conclusions

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 0.3*s* without affecting the 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, integrating countermeasures against relay attacks. In future works, the development of novel technology like WPT could enable a possible extension of *EVExchange* to wireless communication between EV and EVSE.

4 EVScout2.0: Electric vehicle profiling through charging profile

EVs represent one of the technologies enabling a solution for mitigating petroleum consumption and the consequent reduced emissions amount. Multiple countries have already started providing financial incentives to facilitate the purchasing and widespread of EVs ownership [374]. The EV global forecast expects a compound annual growth rate of 29% over the next ten years, with a total EV sales reaching up to 31.1 million by 2030 [105]. Among the factors decreasing the adoption of EVs is consumers' concerns regarding the availability of charging infrastructures [105]. In fact, while gas stations are widely deployed in cities and rural areas, EVSEs are not deployed in a sufficient number or, in some areas, are not present. Although charging stations may be deployed at house premises, the absence of publicly available EVSEs represents a limit, as a user is forced to limit the traveling distance to stay close to a charging point. To mitigate this issue and increase the interest towards EVs, current recovery plans envisioned by countries such as Germany and China designate part of their funds to the development of EV charging infrastructures [105]. Furthermore, companies are equipping their parking spots with EVSEs to serve employees EVs. For instance, the United States is incentivizing the adoption of EVSEs at companies' parking spots via the workplace charging challenge [364]. Therefore, in the next years, we expect a significant increase in the number of publicly available EVSEs, allowing users to charge their EV at any time, removing availability concerns.

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 towards 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 [55]. 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 [18]. Therefore, an attacker can easily install devices to collect data regarding the charging process.

Contributions. In this chapter, we propose *EVScout2.0*, an extension of EVScout [55], that 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 on 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 [55], 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 [55], both on the old and new datasets.

The contribution of this chapter and its improvements with respect to [55] can hence be summarized as follows:

- We propose an enhanced feature extraction framework, which allows for better classification performance compared to [55].
- We propose the use of Delta TS, a novel TS given by the linear combination of the current and pilot TSs. Delta TS allows for the extraction of more significant features compared to those in [55].
- We analyze the performance of *EVScout2.0* on a large real-world dataset, comprising more than 7500 employable current and pilot time series from 137 EVs. Compared to the dataset in [55], we consider both a larger number of TSs, as well as a larger number of EVs.
- We perform a thorough evaluation of *EVScout2.0*, showing its performance over different training set sizes, as well as its robustness towards different unbalancing amounts in the training-testing dataset sizes.
- Compared to [55], we analyze the performance of a higher number of classifiers and provide an in-depth discussion on the choice of their hyper-parameters. We show that *EVScout2.0* is able to profile EVs with precision and recall up to 0.88.
- We investigate the battery degradation over time and show that *EVScout2.0* can extract features that allow for high classification scores (on average 0.8 $F1$ score) in time.
- We compare *EVScout2.0* with the framework in [55], showing its superiority both in the old and in the new datasets.

Organizations. This chapter is organized as follows. In Section 4.1 we review the related works. Then, in Section 4.2 we present the considered system and threat model in an EV charging infrastructure. In Section 4.3 we describe the features and the steps of *EVScout2.0*. In Section 4.4 we describe the evaluation framework and the new dataset. Then, in Section 4.5 we show the results achieved by *EVScout2.0*, while in Section 4.6 we proposed some insight on additional analysis related to training set size and battery degradation. In Section 4.7 we provide a comparison between *EVScout2.0* and its previous version EVScout [55], clearly presenting the advantages of the new approach. We discuss some countermeasures to avoid profiling in Section 4.8, and, lastly, we summarize the results and draw the conclusions in Section 4.9.

4.1 Related Works

Power consumption can be exploited as a side-channel for different purposes [231]. For instance, an attacker may implement a laptop user recognition by exploiting the current drawn by a smart wall socket during users' activity given [91]. 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 [260]. 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 [385]. 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 [97].

Regarding EVSEs, security and privacy research and contributions focus on the negotiation phase, as it allows the negotiation of the charging service by sharing personal users' data. This includes both communications from the EVSE to the power distributor and from the EV to the EVSE. In the former communication scenario, the standards J1772 [87] and CHAdeMO [16] are exploited to regulate the physical standards needed for EV to EVSE connections, together with the signaling required for the charging process. These standards do not employ encryption or privacy measures on the exchanged information. In the latter case, the standard ISO 15118 [264] is exploited to create a secure communication link, implying that both EV and EVSE must be able to encrypt messages. In [30], the authors proved that the charging cable was not shielded and data was not encrypted, leading to dangerous privacy threats. This weakness was exploited in [209] to inject a signal into the cable and stop the communication. In [88], the authors present a relay attack on the ISO 15118 charging system. The attack enables an attacker to charge a vehicle making a victim pay for it. The overall V2G system has been analyzed in literature from a CPS security point of view, and several threats have been identified [18, 148]. However, security and privacy analysis should also focus on the charging phase. In fact, the signals exchanged during the charging process create a time series that can be analyzed to extract features that lead to the profiling of vehicles [55]. However, an in-depth analysis of the robustness of these types of attacks is still missing. In particular, the feasibility of the attacks has been shown for a limited dataset, and the attacker's requirements in terms of the number of samples needed for EV profiling have not been discussed yet.

4.2 System and Threat Model

In this section we introduce the scenario in which we conceived our experiments. In particular, in Section 4.2.1 we recall the EV charging system, which represents our system model, then in Section 4.2.2 we present the threat model designed for *EVScout2.0*.

4.2.1 EV Charging System

According to the Vehicle-to-Grid (V2G) paradigm [197], 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 4.1 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 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 [18]. 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.

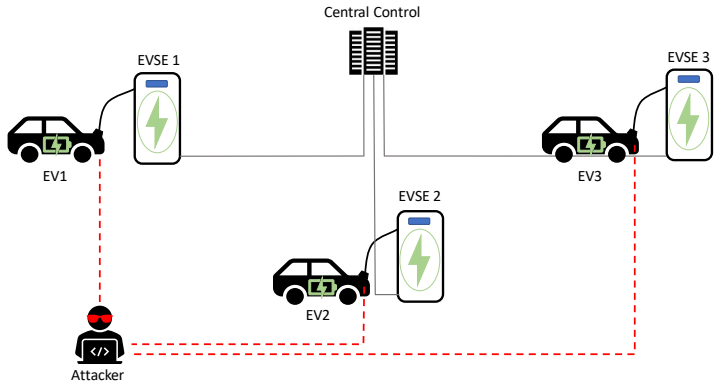


FIGURE 4.1: System and threat model. Multiple EVSEs are connected to the central control, which provides coordination and power distribution among them. A single EV is connected to each EVSE. The attacker has access to the physical quantities exchanged by multiple EVs during the charging phase.

Current implementations of EVSEs are organized in three levels [148, 373]. Level 1 and 2 use a 5 lead connector based on SAE J1772 standard [358], 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) [28, 378]. Current and voltage values exchanged during the charging process depend on the SoC of the EV battery and can be divided into two classes: constant current/constant voltage and constant power/constant voltage [247]. 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 4.2. We here remark that constant current and constant voltage phases are mutually exclusive in time, as this will be exploited by *EVScout2.0*.

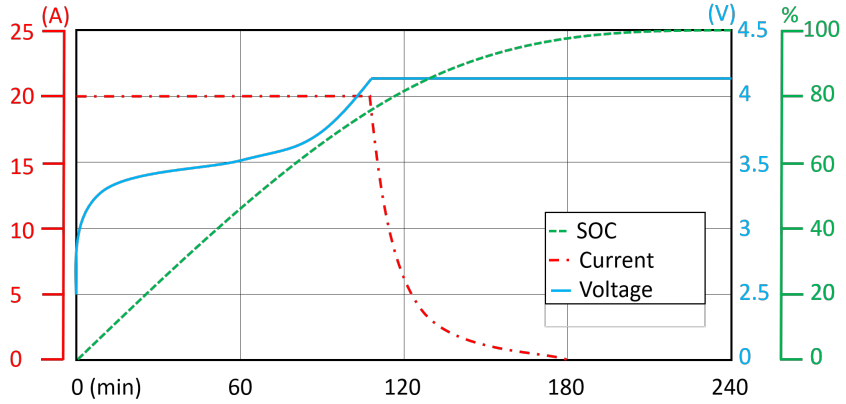


FIGURE 4.2: Charging profile of a Li-ion battery [354]: we see that, as the SoC increases, the charging mode switches from constant current to constant voltage. We further notice that the two phases are mutually exclusive.

4.2.2 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 [19] and a standard power consumption monitoring module to sample both current absorbed and pilot[1]. 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 [55, 97, 385, 91].

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 chapter, 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, 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 [18]. 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

[1]Example of power consumption monitoring module: http://www.sparkfun.com/datasheets/BreakoutBoards/0712.pdf

hand, we notice that the EVSE devices can be routinely checked by the staff of the running company.

Figure 4.1 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 they consume 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 collective use of the EVSE charging stations by EVs to EVSE companies. Although profiling can be implemented using cameras, this would not allow for 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 their 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.

4.3 *EVScout2.0*

In this section we describe the *EVScout2.0* analysis methodology. We propose a high-level description of the attack configuration in Section 4.3.1. Then we describe the preprocessing we apply to the dataset. Starting from Section 4.3.2, we present the concept of tails and outline the method we designed to extract them automatically. To improve the performance with respect to the solution in [55], we propose to exploit Delta TS, i.e., the TS given by the combination of the current and pilot TSs. In Section 4.3.3 we present Delta TS, providing both the motivation behind our choice and the means to compute it. Lastly, in Section 4.3.4, we describe the novel and automatic feature extraction technique we employ in *EVScout2.0*.

4.3.1 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 4.3 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 4.6.1, 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 [55], 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 [339], we name the current TS during the constant voltage phase as *tail*. In Section 4.3.2 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 under-representative 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.

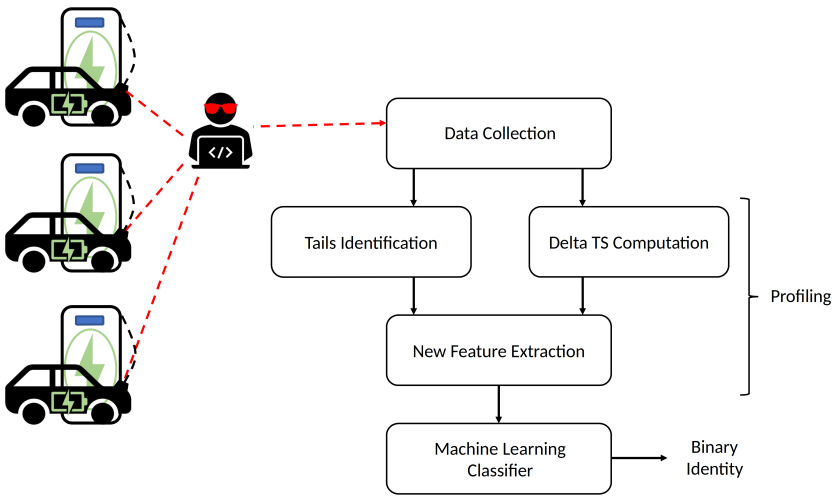


FIGURE 4.3: Block diagram of *EVScout2.0*'s steps.

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 4.3.3.

4.3.2 Tail Identification

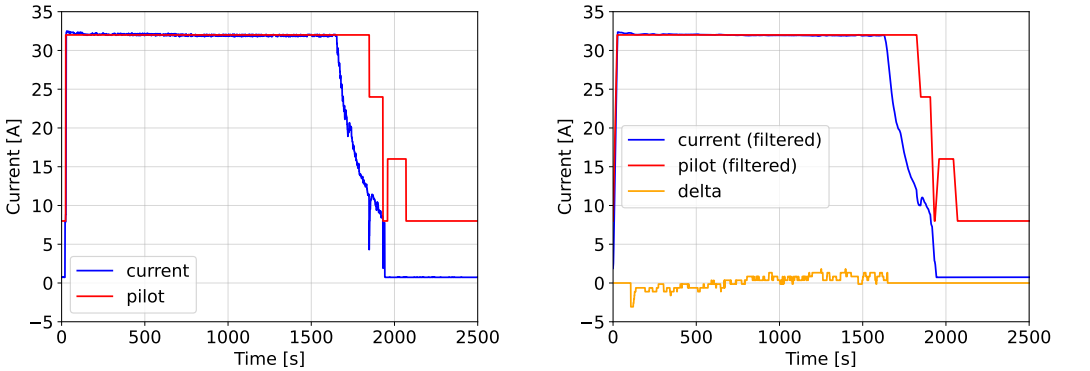
Charging sessions are not necessarily comprehensive of the constant voltage phase, as a user may need to leave before the full charge is reached. In [339], the authors presented a framework to cluster similar charging behavior based on the charging tail. We exploit this portion of the TS to perform more detailed profiling. Since *EVScout2.0* exploits tails during the constant voltage phase, we adopt the algorithm we proposed in [55] 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

$$y(t) = \frac{1}{N_{\text{avg}}} \sum_{m=0}^{N_{\text{avg}}-1} c(t-m). \tag{4.1}$$

The effects of the moving average filter are shown in Figure 4.4. We see that, with respect to the non-filtered current TS in Figure 4.4a, the current TS in Figure 4.4b 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 4.2 and verified in Figure 4.4. 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 Figure 4.4b). A perfectly backward-ascending trend is given by a negative difference between the values at time t and $t-1$, i.e., $y(t) - y(t-1) < 0$.



(A) Raw current and pilot profile.

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

FIGURE 4.4: 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.

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 1. 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 (4.1). We then search for the time instant t_{start} starting 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 hence proceed to identify the number of samples of the tail. Given our definition of backward ascending trend, we compute the number of samples for which the filtered time series is such 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, if an ascending segment is found after a descending samples series, the counter is reset. 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.

Algorithm 1: Tail extraction algorithm.

Data: $\mathcal{W}$, $\mathcal{I}$, $C_{\max}$, $T_{\max}$, N_{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 (4.1);
    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

4.3.3 Delta Time Series computation

Aside from current tails, we compute another TS to be used for extracting 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 [55]. Furthermore, pushed by advanced charging algorithms [224], during some periods, the EV can be forced into charging at a lower current, e.g., to deal with peak leveling during peaks hours. However, as visible in Figure 4.4, 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. In order 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 4.4a. 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 [270], we use it instead of the moving average filter used in Section 4.3.2.

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:

$$z(t) = p(t) - median\left(c\left[t - \left\lfloor \frac{N_{\text{avg}}}{2} \right\rfloor, t + \left\lceil \frac{N_{\text{avg}}}{2} \right\rceil\right]\right). \tag{4.2}$$

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).

4.3.4 Improved Feature Extraction

Segmentation represents a classical approach for feature extraction in TSs [106, 81, 91]. 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 a previous work [55], the authors 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, the authors 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 [34, 261, 106]. We use Time Series Feature Extraction based on Scalable Hypothesis tests (`tsfresh`) [80], which is available as a Python package easily integrable with other tools such as scikit-learn [277]. `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 [55], 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 [224] 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 [277] 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 c3 statistics [313] (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 [80].

4.4 Evaluation Framework Description

In the following, we present the experiments we use to test *EVScout2.0*. In particular, in Section 4.4.1 and Section 4.4.2 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 4.4.3 we outline the machine learning classifiers we use to profile EVs.

4.4.1 The ACN Infrastructure and Dataset

In order to test *EVScout2.0*, we exploit the Adaptive Charging Network (ACN) proposed in [224]. 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.

4.4.2 New Dataset

Like in the previous work [55], 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 [224]. 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 [55].

4.4.3 Classification Algorithms Comparison

The first step of *EVScout2.0* is to build a suitable dataset to be exploited for profiling, as explained in Section 4.4.2. 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 4.2 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 Boost (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.

4.5 *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 4.5.1 the classifiers, and we provide a discussion on their hyper-parameters setting. Then, in Section 4.5.2, we present the results obtained via *EVScout2.0* with the different classifiers.

4.5.1 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 [298, 39], namely:

- Support Vector Machine (SVM) classifier [343];
- k-Nearest Neighbors (kNN) classifier [153];
- Decision Tree (DT) classifier [51];
- Logistic Regression (LR) classifier [170]
- RF classifier [50];
- ADA classifier [129].

Hyper-parameters optimization is obtained via grid search with cross-validation to fine-tune our models and extract the best results. Table 4.1 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 [55], it is not big enough to provide interesting results with deep models or a multi-class scenario.

4.5.2 Profiling Results

We exploit the implementation of the classification algorithms employing the scikit-learn library [277]. Results are assessed in terms of precision P, recall R, 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

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	$[10, 100, None]$
ADA	n_estimators	$[10, 100, 500, 1000, 5000]$

TABLE 4.1: Grid search cross validation parameters for each model.

suitable performance metric [121]. 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 recall as $R = \frac{TP}{TP+FN}$, and specificity as $\alpha = \frac{TN}{FP+TN}$. G-Mean 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 a previous work [55], 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 [277]. 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 4.5 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,

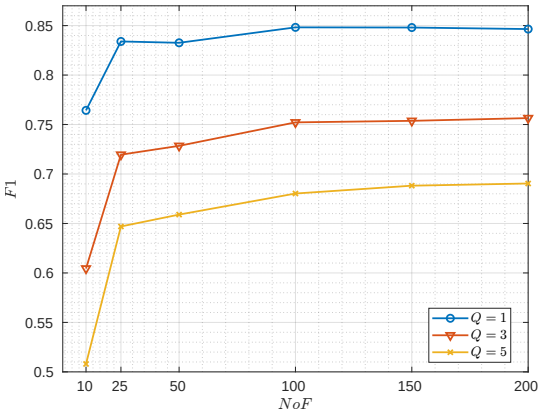


FIGURE 4.5: $F1$ score of kNN classifier with different numbers of features NoFs.

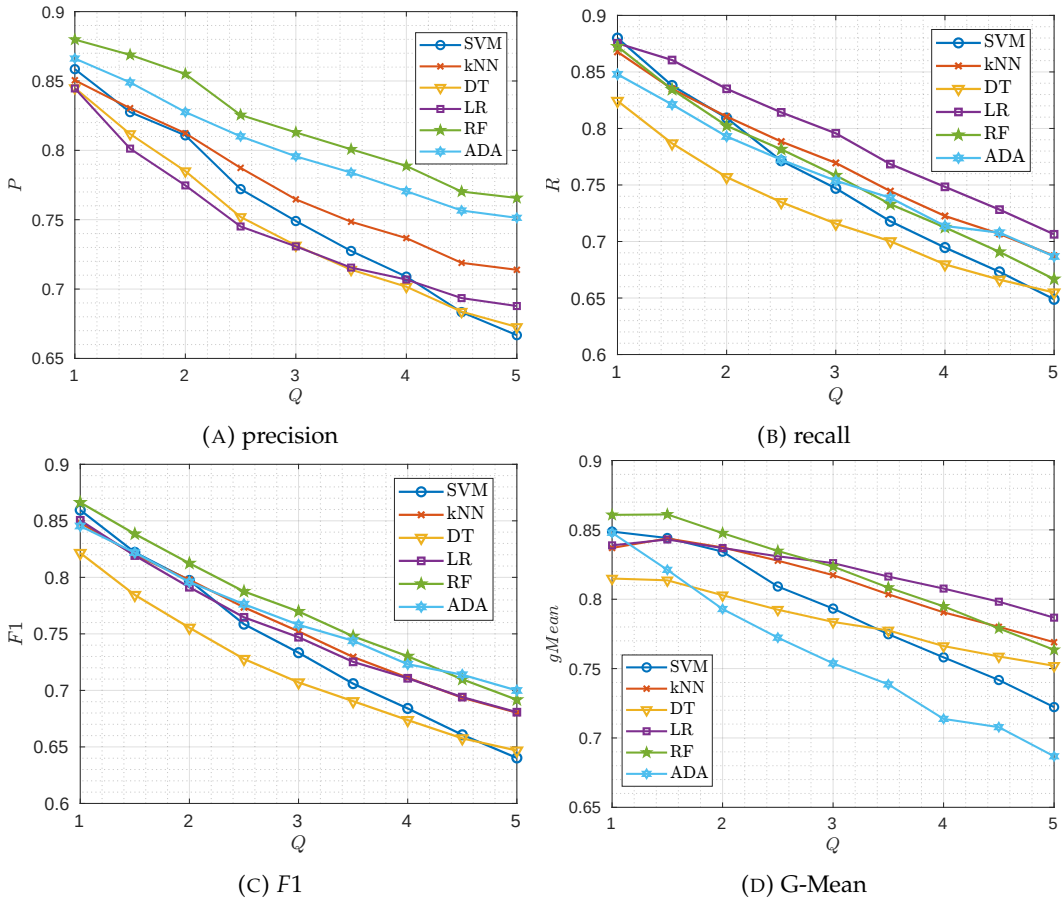
especially for higher ratios Q. From 100 features up, the increase is instead negligible, meaning that the features already capture almost all the entropy available. For this reason, we selected $NoF = 100$ for the other experiments in this chapter unless otherwise specified.

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 4.6 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,



(A) precision (B) recall

(C) $F1$ (D) G-Mean

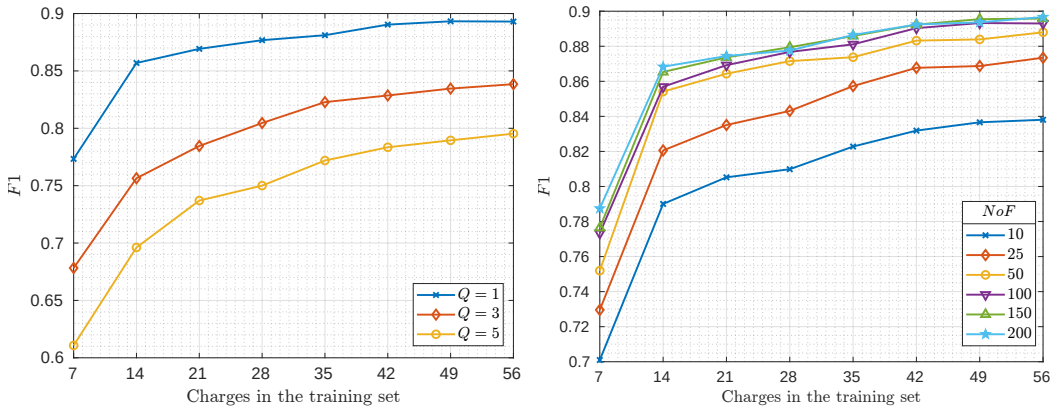
FIGURE 4.6: Performance of *EVScout2.0* for a different amount of unbalancing in the dataset. Results are shown for the different classifier algorithms and $NoF = 100$. We see that good classification performance are obtained for all classifier. As Q increases, some classifiers are more robust than others to increasing unbalancing. Results on G-Mean show that profiling can also be achieved in largely populated networks.

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.

4.6 *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



(A) Training set size changing. (B) Changing NoF and training set size for $Q = 1$.

FIGURE 4.7: $F1$ scores of kNN classifier while reducing the training set size.

in the following. In particular, in Section 4.6.1, 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 4.6.2, 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.

4.6.1 Training Size Variation

In a real-world scenario, an attacker may be limited by the number of labeled charges they 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 for each EV, with steps of 7 charges.

In Figure 4.7a, 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 4.7b, 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 4.5.2 and Figure 4.5.

4.6.2 Battery Degradation Performance

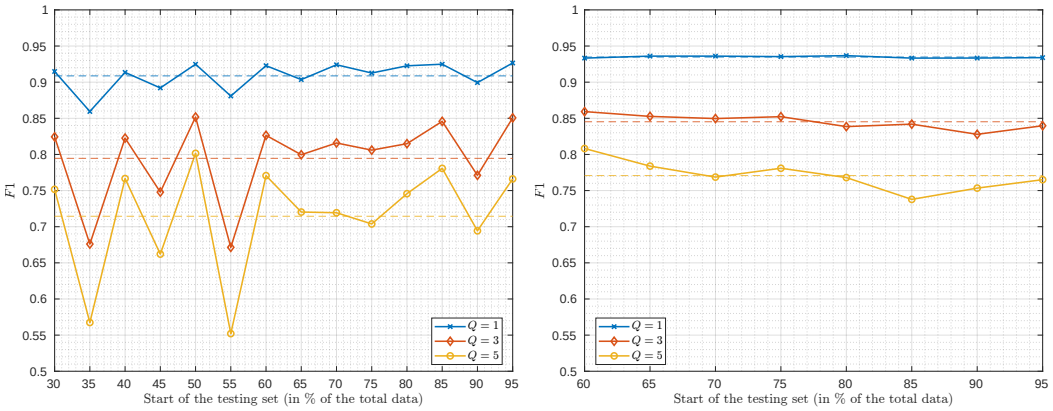
The degradation of a Li-ion battery used in an EV is widely discussed in literature [279]. During a battery life, many aspects can adversely affect its performance depending on the number of charge cycles, aging, operating, and storing conditions. Degradation on EV batteries can lead to a shorter traveling distance and a reduced battery's available power output [36]. 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 [371, 36].

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 4.8a, 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 shown in Figure 4.8b. 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



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

(B) Train set size 60% of the data.

FIGURE 4.8: $F1$ score for different ratios while varying the start of the testing set. Each values 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.

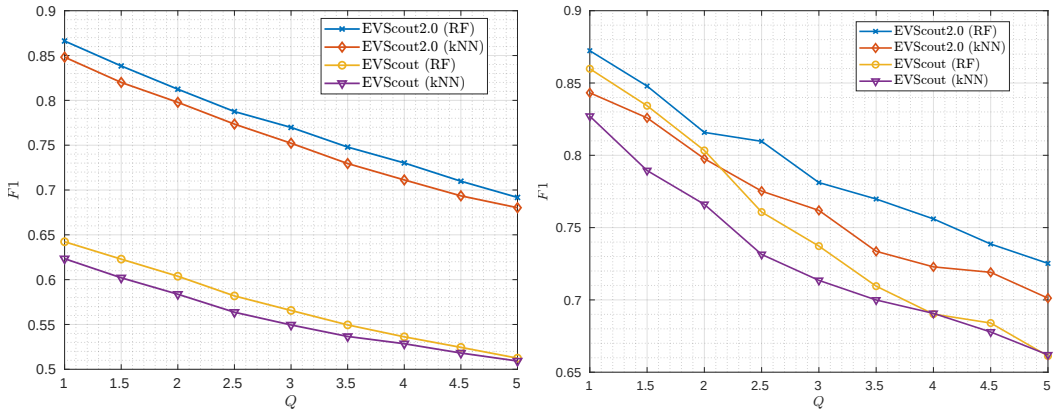
or by the ACN scheduling algorithm, especially when considering highly unbalanced datasets and a small training set. Furthermore, this experiment can be seen as a remark on the need for a big training set as presented in Section 4.6.1. We will provide more analysis on the degradation in future work, considering datasets that span more than two years.

4.7 EVScout2.0 Performance: Comparison with EVScout

In a previous paper [55], the authors 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 [55] 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 [55]. It is worth mentioning that we employed the exact dataset used in [55] and not simply the same EVs updated with the new charges.

4.7.1 EVScout on the new dataset

We tested EVScout [55] in our new dataset. To generate results comparable with those presented in this chapter 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 4.9a 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 performance improvement is probably due to the combination of a feature extraction technique that better represents the time series of the charging process, together with the addition of the delta time series, which enriches the number of features generated. The difference in the classification performance ranges from about 0.15 for $Q = 5$ to more than 0.20 for $Q = 1$. We remark



(A) Comparison on the new dataset.

(B) Comparison on the original dataset.

FIGURE 4.9: $F1$ values of the comparison with the previous work [55].

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.

4.7.2 *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 4.9b. 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.

4.8 Possible Countermeasures

To deal with information leakage from smart meters, [192] 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 hence still be able to perform profiling.

A similar concept is exploited in [320], where noise is added to smart meters data via an adversarial learning 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 manage 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 [97], 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. A possible future work could focus on identifying ad-hoc solutions to prevent the inference of sensitive information via power-side channels, specifically in communications between EVs and EVSEs. Solutions coming from the smart grid sector could also be considered to be adapted for the EV charging scenario, such as obfuscation or perturbation [299]

4.9 Conclusions

EVs security is a novel topic that raises many novel challenges in protecting such systems. As we demonstrated, introducing a charging system that utilizes personal information can lead to privacy leakage and profiling attacks. In this chapter, we proposed *EVScout2.0*, an expanded version of a previous work [55] by employing a bigger dataset, 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 [55], 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.

Part II

CAN Bus Security

5 *CANLP* : Intrusion Detection for Controller Area Networks using Natural Language Processing and Embedded Machine Learning

In-vehicle security has been a rising concern as a consequence of modern vehicles using software-driven ECUs and wireless connectivity. The CAN protocol is the most popular and extensively used in-vehicle messaging standard in the automotive industry [211, 171, 48]. However, the design of the CAN protocol does not include essential security features such as encryption or message authentication, which makes the network vulnerable to attacks by an adversary [47]. An adversary has been shown to be able to exploit vulnerabilities in remote endpoints to compromise an ECU, access the in-vehicle network, and cause abnormal vehicle behavior, thus jeopardizing driver and passenger safety [70, 211, 254].

IDSs are a commonly proposed solution to detect possible adversarial behavior on the CAN bus. IDSs for CAN are typically categorized as rule-based [360], signature-based [213], or ML based [256] depending on the methodology used. While the former two have been implemented in a wide range of settings, they have been shown to fail if an adversary takes control of an ECU before the rules of the IDS have been determined [235], or might incorrectly identify benign signals on the CAN bus as malicious [173]. In comparison, ML-based IDSs [256] provide a scalable way to detect the presence of an adversary while handling large amounts of data transmitted on the CAN bus. ML-based IDSs also cause minimal disruption to the CAN protocol and ECUs and are hence widely utilized for in-vehicle security to detect a variety of attacks [256, 202, 35].

ML-based IDSs use contents of CAN data or derive features from a window of messages and develop models to detect attacks. Some IDSs [78] also consider periodicity of ECU transmissions to predict the next set of CAN messages and detect attacks based on changes observed between actual and predicted values. However, we believe that deploying these IDSs in real-time can be challenging because: (i) Stealthy attacks are performed at the expected periodic frequency of legitimate CAN messages and timing-based IDSs are unable to detect these attacks; CAN frame on which the attack was mounted, thus IDSs using a window of messages [202, 35, 6] cannot perform this distinction; and (iii) CAN protocol allows message transmissions at speeds of up to 1 Mbps, which makes it difficult for IDSs with large ML models [384, 202, 398] to be deployed in resource-constrained hardware with low latency.

In order to overcome these challenges, we use the insight that the structure and formatting of messages on the CAN bus can be exploited to construct a human-readable format according to a set of well-defined rules in the CAN Database (DBC) file. Such transformation of the CAN messages enables interpreting this data using

techniques from NLP. The data in a CAN frame can be segmented into blocks of 8 one-byte values, 64 one-bit values, 1 sixty-four bits value, or any combination of these [179]. The Term Frequency-Inverse Document Frequency (TF-IDF) [290] encoding technique provides a way to group binary data into bits of different lengths.

Contributions. In this chapter, we develop *CANLP*, an intrusion detection system for CAN using NLP and embedded ML. *CANLP* uses TF-IDF [10] to extract fine-grained features from CAN data. We propose to extract character-level (1,2)-gram TF-IDF features from CAN data bytes to enable capturing frame-specific fine-grained bit patterns [302]. *CANLP* identifies frequency patterns in CAN data with respect to each frame as well as a set of frames observed on the bus, and subsequently uses this information for real-time attack detection. *CANLP* analyzes individual CAN frames and identifies the specific type of attack, overcoming a problem faced by ML-based IDSs [202, 35, 6]. Isolation of the transmitting ECU following such detection can be carried out by ECU fingerprinting methods.

CANLP uses the extracted features to train a Deep Neural Network (DNN) to distinguish between legitimate transmissions and adversarial behavior to perform *multi-class attack classification*. Our feature extraction technique demonstrates variability in the features among different attacks, thus allowing us to use a small DNN model which is further reduced in size using quantization techniques [141]. This enables *CANLP* to perform attack detection and additionally enable real-time deployment with low overhead.

Through extensive experiments on four publicly available vehicle network datasets —(i) HCRL Car Hacking: Attack and Defense Challenge (AD) [193], (ii) HCRL Car-Hacking (CH) [315], (iii) HCRL Survival Analysis Dataset for Automobile IDS (SA) [161], and (iv) Real ORNL (Oak Ridge National Laboratory) Automotive Dynamometer (ROAD) CAN Intrusion Dataset [370]— we evaluate the effectiveness of *CANLP* in detecting attacks. Our experimental evaluations on the CAN testbed show that *CANLP* detects fuzzing, spoofing, or masquerade attacks mounted on the CAN bus with a high F1-Score (F1-Score) of 0.9974 even after model compression. We implement *CANLP* on a Raspberry Pi 4 Model B running at $1.8GHz$ with $1GB$ of RAM and deploy it on a CAN testbed. The latency of *CANLP* to classify each CAN frame is as low as $0.049ms$.

Our contributions are:

- We develop a scheme for feature extraction of CAN messages by exploiting patterns within CAN data frame and ID correlations using TF-IDF.
- We propose *CANLP*, an ML-based IDS, which gives an end-to-end framework enabling real-time detection of fuzzing, spoofing, and masquerade attacks.
- We deploy *CANLP* on resource-constrained hardware using a CAN Bus testbed with transmission speed up to 1Mbps and demonstrate low latency.

Organization. The rest of the chapter is organized as follows. Section 5.1 provides background of CAN, NLP, and quantization. Section 5.2 describes our threat model; Section 5.3 explains defense capabilities. Section 5.4 presents datasets and preprocessing performed to evaluate *CANLP*. Section 5.5 details the design of *CANLP* and the experimental setup and evaluation results are presented in Section 5.6. Section 5.7 summarizes related works, and Section 5.8 concludes the chapter.

5.1 Preliminaries

In this section, we provide necessary background on CAN attacks, DNNs, and quantization techniques.

5.1.1 Controller Area Network

The CAN protocol is one of the most widely used in-vehicle networking standards [181, 211]. The broadcast nature of CAN Bus enables the transmission and reception of messages by any ECUs on the bus to any other ECU and provides every ECU the capability to observe all ongoing transmissions. The CAN messages follow a specified frame structure, which is illustrated in Figure 5.1. It contains ID-, control-, data-, CRC- and ACK-bit(s) but does not include encryption, authentication, or timestamps. The message identifier (ID) describes the data content. The lower the binary ID, the higher is its priority. An ID consisting entirely of zeros is the highest priority message on a network because it holds the bus dominant the longest. The CAN protocol is specified such that the ECUs only transmit a message when the bus is idle [181]. The CAN protocol includes an arbitration mechanism to resolve potential conflicts when more than one ECU attempts to occupy the bus at the same time. In such a scenario, the winner at the end of the arbitration phase is the ECU with the highest priority ID.

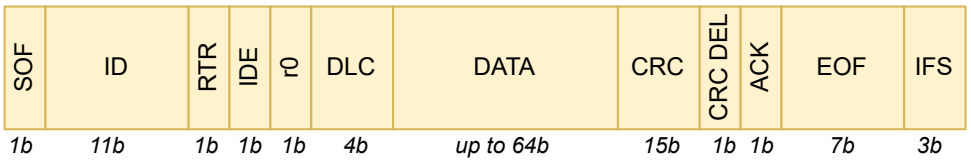


FIGURE 5.1: A standard CAN frame showing the bit size of each field.

5.1.2 TF-IDF

N-gram Term Frequency (TF) and Inverse Document Frequency (IDF) are widely used computational methods in NLP tasks such as text mining [290, 10, 359, 26]. An N-gram is a contiguous sequence of N terms (characters, words) in the content of a document. Typically, N-grams are extracted by moving a window of length N forward, one term at a time, along the content of document. The TF [290] of a term τ in a document is:

$$TF(\tau) = \frac{\text{No. of times } \tau \text{ appeared in the document}}{\text{No. of terms in the document}}. \tag{5.1}$$

The IDF measures informativeness of terms in a collection of documents (corpus) [290]. It assigns lower values to terms that commonly appear in the corpus as they do not contribute to distinguishing contents of documents. Conversely, higher values are assigned to less frequent terms in the corpus as they may constitute patterns inherent to document contents. The IDF of a term τ is:

$$IDF(\tau) = \log\left(\frac{\text{No. of documents in corpus} + 1}{\text{No. of documents with } \tau + 1}\right) + 1. \tag{5.2}$$

The TF-IDF is a statistic that quantifies the importance of a term to a document in a corpus [290], and is obtained as:

$$TF\text{-}IDF(\tau) = TF(\tau) \times IDF(\tau). \tag{5.3}$$

This value of TD-IDF is termed *features*.

5.1.3 Deep Neural Network (DNN)

DNN [310] classifiers take in an input X and return a vector S of scores for each class, where $s \in S$ (given by $s = [s_1, s_2, \ldots, s_C]$ called logits, where C is the number of classes) is a class score indicating whether X belongs to class C. The softmax function is used to translate all these scores to probabilities, where:

$$softmax(S) = \frac{1}{\sum_{k=1}^{C} e^{s_k}} [e^{s_1}, e^{s_2}, \ldots, e^{s_C}]. \tag{5.4}$$

The categorical cross-entropy (CE) loss function is commonly used in multi-class classification tasks due to its effectiveness in measuring differences between predicted probabilities and true class labels [146], and given as [233]:

$$CE = -\log\left(\frac{e^{s_p}}{\sum_{k=1}^{C} e^{s_k}}\right), \tag{5.5}$$

where s_p is the class score for true label/positive class for an input X. To learn complex patterns and relationships between features in DNN for X, the dense layer [146] is used which is a layer of neurons (each neuron is connected to every neuron within both the preceding and subsequent layers). A DNN classifier is considered overfitted [322] when it classifies members (samples in the training set) correctly but exhibits lower F1 when classifying nonmembers. To prevent overfitting and improve the generalization of DNN, the dropout layer randomly deactivates a fraction of neurons during training, wherein the dropout rate represents the proportion of neurons being deactivated [335].

5.1.4 Quantization

Quantization [141] is a process by which the numerical value of a signal is represented using fewer bits or discrete levels. In *CANLP*, we quantize weights and activations of DNN layers to achieve model compression before deployment on resource-constrained hardware. We examine two types of quantization: (i) *Float32*, in which each number is represented as a fixed decimal with 32 bits of precision, and (ii) *Dynamic range*, in which quantization levels are determined based on the range of values that the quantity can take.

5.2 Threat Model

We introduce the threat model we consider for *CANLP* and discuss our assumptions about adversarial capabilities.

TABLE 5.1: Overview of the types of attacks that an adversary can carry out on CAN Bus and the importance of each of the attributes (ID, Data, and Timestamp) of CAN malicious packets for attack identification. Here, symbols ●, ◐, and ○ represent that the attack has full, partial, and no dependency, respectively, on the particular attribute. *CANLP* aims to detect Timing Opaque attacks using feature extraction techniques on ID and Data.

Attack Type	Attack	ID	Data	Timestamp
Timing Transparent	Replay	◐	◐	●
	DoS	◐	○	●
Timing Opaque	Spoofing	●	◐	○
	Fuzzing	●	●	○
	Masquerade	●	◐	○

5.2.1 Adversary Assumptions

We assume that the adversary can intercept all CAN traffic and has the ability to inject messages of their own into the bus. We assume that the adversary performs attacks through physical access by introducing a new ECU to the bus or by taking control of a compromised ECU [48]. We consider different levels of compromised ECUs based on the control that the adversary exerts [370]. A weakly compromised ECU is a node that an adversary can disrupt from transmitting on the bus, while a strongly compromised ECU is a node over which the adversary has complete control, with the ability to send fabricated messages and access the node's memory. The adversary can strongly compromise ECUs by gaining control of the vehicle's OBD-II port and exploit vulnerabilities on ECUs.

The adversary performs attack injection in a sporadic manner, where there is no observed pattern in attack frequency to reduce possibility of or avoid being detected. We also assume the adversary carries out multiple attacks in no particular order and can target any component of the vehicle to cause abnormal behavior.

5.2.2 Attack Scenarios

The objective of the adversary is to compromise the safety of the vehicle by injecting anomalous messages into the CAN bus. This goal is accomplished by carrying out one of the attacks described below. We define different classes of CAN attacks shown in Table 5.1 and highlight the attacks which *CANLP* aims to detect, consistent with attack definitions in [78] and [370].

Based on the frequency of message injection, attacks can broadly be classified into two types:

1. Timing Transparent (TT) Attack: A TT attack is any attack that is hypothetically detectable using a frequency-based method [370]. Primarily, Denial of Service (DoS) and replay attacks fall under this category as they are characterized by abnormally fast message timing [370] and can easily be detected using timing-based IDSs.

2. Timing Opaque (TO) Attack [370]: A TO attack, on the other hand, is any attack that cannot be detected using a frequency-based IDS because it does not disrupt normal timing or ID distributions. Attacks that may alter the overall state of the vehicle to cause out-of-control behavior fall under this category as defined below:

Fuzzing Attack [398] — For a fuzzing attack, messages with random IDs and arbitrary payloads are injected into the bus at the expected periodicity of CAN messages

on the bus with intent to alter vehicle behavior. Fuzzing attacks can also be implemented by injecting messages at high frequencies which causes an impact similar to DoS attacks, where the bus is occupied with injected messages instead of legitimate ECU transmissions. However, our model considers fuzzing attacks where the adversary aims to create abnormal behavior rather than a straightforward DoS. Additionally, we assume that the adversary only injects messages with arbitrary IDs that do not conflict with legitimate ECU IDs on the bus to implement a fuzzing attack.

Spoofing Attack [398] — To implement a spoofing attack, an adversary injects messages using the target ID of a legitimate ECU (collected by observing traffic on the bus) and a manipulated data field. The adversary uses a specific target ID but modifies selected bits of the payload randomly. This attack is also referred to as targeting a signal and aims to cause unexpected behavior of vehicle components since the data appears as it if were transmitted by a legitimate ECU. On the other hand, a targeted ID attack which aims to manipulate specific functionality is implemented by injecting data frames that are designed to have a particular effect on the vehicle based on the selected target ID.

Masquerade Attack [370] — To implement masquerade attacks, the adversary first suspends messages of a weakly compromised target ECU, for instance, using bus-off attack [44, 77], and then injects spoofed messages with this target ID using a strongly compromised ECU, thus masquerading as the target ECU. Using this strategy, a targeted ID attack can be carried out without message conflict, thus allowing for a more stealthy attack than simple message injection. Using masquerade attacks, the adversary can not only inject attack messages from the compromised/impersonating ECU but also change the expected periodicity of messages, significantly degrading the in-vehicle network performance and increasing complexity for detection. Masquerade attacks have been shown to cause critical issues such as non-abortable transmission requests, deadline violation, and significant priority inversion [370].

Due to the nature of TO attacks, specific methods such as a payload-based detector that uses data field, or a side-channel method monitoring the physical layer must be deployed [370]. In theory, binary search algorithms can be used to detect fuzzing attacks. However, our defense model is designed for multi-attack classification on the bus, necessitating the use of scalable models capable of distinguishing between various attack types. *CANLP* is primarily designed to detect TO attacks using feature extraction techniques on the CAN payload and work against adversaries implementing stealthy attacks. Here, we develop *CANLP* to detect fuzzing, spoofing, and masquerade attacks.

5.3 Defense Model

In this section, we present capabilities of the defense (Section 5.3.1) and define metrics to evaluate *CANLP* (Section 5.3.2).

5.3.1 Defense Capabilities

The defense has limited real-time computational resources and uses ML to train a model or re-train publicly available models to achieve high F1-Score for attack classification. We further assume that the defense is aware that the adversary performs different types of attacks on the CAN Bus and has knowledge of these attacks. However, the defense has no knowledge of the order or frequency in which these attacks

are performed. We assume that the defense has access to a set of CAN packets whose labels are known for model training. Our objective is to develop an IDS that detects fuzzing, spoofing and masquerade attacks on the CAN Bus in real-time and can be deployed inside a vehicle.

5.3.2 Performance Metrics

We assume that *CANLP* returns a 'Normal' label for legitimate messages and identifies the type of attack for other messages. Each type of attack is considered as a separate class and each of the chosen datasets for our experiments have different combinations of the attacks described above.
True Positive Rate (TPR): The fraction of attack packets that are classified correctly [324].
False Positive Rate (FPR): The fraction of legitimate packets that received an incorrect prediction, hence raising a false alarm [324].
F1-Score: An effective IDS should have a high TPR and a low FPR (minimizing false alarms). Defining $TNR = 1 - FPR$, the F1-Score [324] combines TPR and TNR as:

$$F1 = \frac{2 \times TPR \times TNR}{TPR + TNR}. \tag{5.6}$$

We employ this metric to equally take into account false positives and false negatives, since both are dangerous in our scenario.
Latency: The time taken by the hardware to perform data pre-processing, run the prediction model, and provide the output using received CAN data frame.

5.4 Datasets

This section provides details about datasets we use to evaluate *CANLP* (Section 5.4.1). We analyze these datasets to highlight certain shortcomings (Section 5.4.2).

5.4.1 Dataset Description

To evaluate *CANLP*, we use four datasets that are popular in literature and each of them is described below:
Car Hacking: Attack and Defense Challenge (AD) Dataset [193]: The AD dataset consists of data collected from a Hyundai Avante CN7 under both dynamic driving and stationary conditions, which allows our model to analyze adversarial behavior irrespective of the vehicle state. The data contains four primary features: Timestamp, CAN Arbitration_ID, DLC, and Data_Field. Two distinct labels are used to categorize the data. The primary label, 'class', indicates whether the captured frame is benign or malicious; labeled as 'Normal' or 'Attack' respectively. For the CAN frames identified as an 'Attack', a secondary label termed 'subclass' is used to represent the type of attack. The dataset classifies attacks into four predominant categories: DoS/ flooding, spoofing, replay, and fuzzing attacks.
HCRL Car Hacking (CH) Dataset for Intrusion Detection [315]: The CH dataset contains CAN data collected from an unspecified real vehicle and has four features similar to the AD dataset: Timestamp, CAN Arbitration_ID, DLC, and Data_Field. The dataset consists of three different types of attacks: DoS/ flooding, fuzzing, and spoofing attacks which are available as individual CSV files. The spoofing attacks

TABLE 5.2: Number of normal, attack and total CAN messages in the AD (Driving), AD (Stationary), CH, SA (all modified as per Section 5.5.1) and ROAD datasets.

Dataset	#Normal	#Fuzzing	#Spoofing	#Masquerade	#Total
AD (Driving)	500078	45474	164409	N/A	709961
AD (Stationary)	73955	44405	156337	N/A	274697
CH	81306	491847	475422	N/A	1048575
SA	236787	237923	231824	N/A	706534
ROAD	218756	N/A	N/A	18080	236836

are further divided into Revolutions Per Minute (RPM) gauge and gear spoofing based on the vehicle component that the attacker aims to compromise. The dataset contains two distinct labels, 'T' and 'R', where 'T' represents injected message while 'R' represents a normal message.

HCRL Survival Analysis Dataset for Automobile IDS (SA) [161]: The SA dataset captures in-vehicle network traffic from three distinct car models: the Hyundai Sonata, KIA Soul, and Chevrolet Spark collected during stationary state of each vehicle. The datasets include Timestamp, CAN Arbitration_ID, DLC, and Data_Field features. Each vehicle dataset is split into 'Normal' data and four attack categories, namely: DoS/ flooding, fuzzing, malfunction, and replay attacks available as individual CSV files for each car. This dataset uses the same labelling as the CH dataset.

Real ORNL Automotive Dynamometer (ROAD) CAN Intrusion Dataset [370]: The ROAD dataset is a compilation of real-world vehicle attacks where each injected attack is physically tested for its impact on the vehicle. The vehicle's make and model are undisclosed. All attacks were performed on a dynamometer, simulating real driving conditions. The ambient data was collected both from the dynamometer and actual road conditions in various driving scenarios. The dataset includes fuzzing and fabrication attacks, and simulated stealthy masquerade attacks. Due to the complexity of implementing masquerade attacks, no real CAN data with these attacks has been made publicly available. The dataset also contains accelerator attacks which cause abnormal accelerator behavior out of the driver's control. The accelerator attack captures have no injected messages or disclose how the attacks were performed, but simply record the CAN data when the vehicle is in this state.

5.4.2 Dataset Analysis

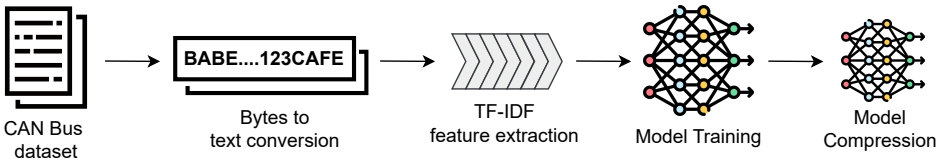
During our evaluation of the datasets, we observed a limitation in the spoofing attack data points for AD, CH and SA datasets. The datasets contain only n (< 10) unique spoofing attack messages, which have been repetitively injected into the vehicular system at different intervals to perform the attack. While this is a legitimate spoofing attack, it limits the dataset to very specific cases of spoofing and restricts variability in data frames on the bus. This redundancy inherently biases ML models, leading to overfitting on recurrent data points and patterns. The identical attack data points are also present in testing data, which should ideally contain only data that is distinct from training data to correctly evaluate model performance.

We noticed that certain State of the Art (SOTA) ML-Based IDS using these datasets without modification produce skewed experimental results with F1-Scores reaching 100.0% due to extreme overfitting. To confirm our observations, we ran experiments on these datasets and achieved F1-Scores of 100% for spoofing attacks for different ML models we tested upon. In the real-world, where attack patterns

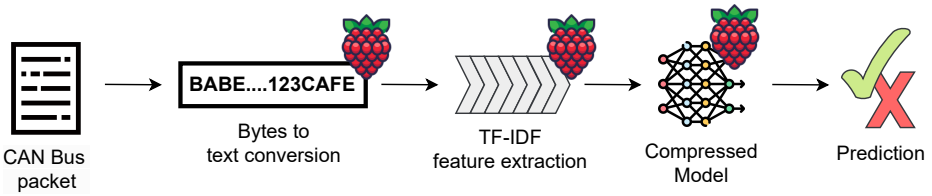
may be injected in different ways, redundancy in these datasets compromises generalizability of IDS models and provides a misleading interpretation of the model's F1-Score in predicting attack types.

5.5 *CANLP* Implementation

This section details design and implementation of *CANLP*. *CANLP* processes CAN bus data through a series of methodical steps to detect and classify adversarial attacks. The *CANLP* framework processes CAN bus data by first converting raw CAN frames into a human-readable format for NLP. It employs the TF-IDF method to extract features, which are then fed into a DNN model to classify frames as legitimate or malicious, identifying specific attack types. The optimized DNN model is compressed using quantization and deployed on a Raspberry Pi [291] equipped with a CAN bus extender [314], which can be integrated into a vehicle's CAN Bus by simply plugging it into the OBD-II port. It can also be integrated inside the vehicular CAN system if the maker chooses a more seamless build. Figure 5.2a summarized the training steps, while Figure 5.2b illustrates the overall testing process done in the Raspberry Pi [291].



(A) Overview of the *CANLP* training steps.



(B) Overview of the *CANLP* testing steps.

We explain how we carry out dataset preprocessing in Section 5.5.1, discuss feature extraction and learning models used in Section 5.5.2 and 5.5.3, and hardware deployment (Section 5.5.4).

5.5.1 Dataset Pre-Processing

Data Synthesis: To overcome the limitations observed and expand the dataset for better evaluation of our model, we synthesized spoofing attack data through a series of steps:

1. *Target CAN ID Identification*: We identified a specific set of legitimate CAN IDs for our spoofing attack. These target IDs are representative of ECUs selected by an adversary to launch the attack after observing CAN data for some time.

2. *Data Frame Randomization*: Once the target CAN IDs have been identified, we selected all legitimate messages already transmitted by each target ECU. From each message, three bytes of the data frame were chosen and randomized. While each modified payload might not directly impact the vehicle due to randomization, we generate synthetic attack data as a targeting signal attack (as

described in Section 5.2.2) or spoofing attack with the aim of preventing data transmission from legitimate ECUs to vehicle components.

We generate the synthetic data to mimic targeting signal attacks as closely as possible and ensure reduced redundancy in the dataset. This randomization technique was chosen after recursively testing the synthetically generated data on different models to ensure that it does not skew model training or performance. We inject the synthesized data points into the datasets in a random manner since our model is time agnostic and focuses on the CAN IDs and data frames.

Data Processing for Feature Extraction: In our pre-processing phase, we consider the four datasets separately. From each dataset, we only extract the CAN ID, the data frame (as *Data_Field*) and the associated *Attack Type*, which is the provided label. We rename each label to represent the type of attack. Within each dataset, we combine all the subsets containing individual attacks (by consolidating the CSV files) to evaluate a true multi-class attack classification model. Additionally, we traverse the dataset to remove all flooding and replay attacks since our focus is to detect TO attacks. We also remove all duplicate frames to prevent any form of redundancy in the datasets and ensure that our model evaluation is not skewed. We define a frame as a duplicate if another frame in the dataset, belonging to the same class (both normal and attack classes), and have identical CAN ID and data frame values.

Since the number of attack messages is typically a small fraction of legitimate messages, we perform data balancing to mitigate any bias towards the majority class (here legitimate messages) during DNN training [341]. This process of data balancing is achieved through a series of steps: First, we identify distinct labels in the dataset representing different attack categories. Then, we balance the number of data points in each class by randomly removing data points from classes having higher number of data points. We split the balanced dataset into training and testing subsets with 80% - 20% split, ensuring sufficient data for training. The training subset is used to perform extraction of TF-IDF features which are then fed to the model for learning.

5.5.2 Feature Extraction

As noted in Section 5.1.1, a raw CAN data frame is transmitted in a hexadecimal format and is a collection of consecutive bytes representing signal information for in-vehicle components. CAN DBC is used to decode each raw frame data in a bit-wise manner to interpret signal definition as physical parameters [179]. Therefore, we choose a character as a term while using N-gram TF-IDF for extracting features from a CAN data frame, specifically using the 'Message Identifier' and 'Data Field' bits. Our proposed character-level N-gram TF-IDF feature extraction interprets signal definition as physical parameters and identifies bit patterns corresponding to CAN data within a single frame as well as an overall set of frames observed on the bus.

Our intuition is that adversary-manipulated CAN data related to TO attacks might exhibit different bit orderings or consecutive bit patterns compared to the patterns found in normal CAN messages. We observe that such consecutive bit patterns of length N can be captured through N-gram units [302]. The total number of required TF-IDF features to represent each CAN frame using this technique and the number of trainable DNN parameters are shown in Table 5.3.

In *CANLP*, we employ the following methods to extract features from CAN ID and Data_Field for multi-class classification. The CAN ID is converted from hexadecimal to decimal format, followed by normalization from the range of 0 to 2048 to

TABLE 5.3: Numbers of features and trainable DNN parameters for 1-gram, (1,2)-gram and (1,2,3)-gram Character (Char) level TF-IDF features.

Granularity of Features	No of TF-IDF Features	Trainable Parameters
Char 1-gram	16	10755
Char (1,2)-gram	272	43523
Char (1,2,3)-gram	4368	567811

a new range of 0 to 1, and then used as a feature. For the Data_Field, we consider encoded Char (1,2)-gram TF-IDF for feature extraction followed by L2-normalization [38] for multi-class classification. Our approach involves the following steps: (i) identify the smallest unit that has meaning (the NLP equivalent of a word) within the CAN data frame; (ii) identify the fixed-length patterns that need to be extracted from the data frame; (iii) form a frequency vector of all possible lengths that can be used as the set of features for a CAN frame. The frequency vector must meet the following criteria [302]:

1. Length of data frame should not affect frequency values.
2. Frequency values corresponding to common patterns found in data frames (considered as noise) should be reduced, as they do not contribute to distinguishing between legitimate and attack messages.
3. Values corresponding to frequent patterns appearing only in a small subset of the dataset should be boosted since such patterns will have a higher probability of being attack-specific patterns.

Remark: Although the total number of features for Char(1,2,3)-gram are significantly higher than Char (1,2)-gram, F1-Scores obtained using each method are comparable. On the other hand, the Char 1-gram yields lower F1-Scores since it does not contain sufficient features to train the DNN model. We opt to use the Char (1,2)-gram in our experiments to maximize the efficiency while optimizing the allocated compute resource and latency values.

5.5.3 Learning Models

The features extracted from CAN frames are subsequently fed into learning models to perform classification of each frame as a legitimate transmission or an attack, while also predicting the type of attack mounted on the frame.

We train five ML models for classification, including SVM, LR, Gaussian Naive Bayes (GNB), DT, RF [163]. We also train a DNN to effectively extract features at different levels of abstraction which allows them to learn more complex patterns when compared to traditional ML algorithms [234] and provide techniques for easy deployment on resource-constrained hardware [347]. Our DNN model uses three dense layers with Rectified Linear Unit (ReLu) activation [4] and one dropout layer. The first two dense layers consist of 128 and 64 units, while the final layer is adjusted to match the number of attacks in dataset. A dropout layer with a rate of 0.15 is incorporated to prevent overfitting and to improve learning efficiency during training. The final dense layer employs softmax activation using Eq. (5.4) to predict the type of attack.

To train the model, Adam optimizer [206] with a constant learning rate of 0.001 is employed with the categorical cross-entropy loss function from Eq. (5.5). The model is trained for 30 epochs with a batch size of 128. The training data is further split into training and validation in an 80:20 ratio for fine-tuning the DNN. The model is implemented using Keras [198], leveraging standard Keras functions for defining architecture, training, and evaluating the DNN.

5.5.4 Deployment on Hardware

To evaluate and validate our experiments in real-time, we use post-training quantization techniques to further compress our learning model before deployment on the CAN testbed. In order to achieve an optimal balance between F1-Score and latency, the model is compressed by quantizing the model's weights using float32 and dynamic quantization employing Tensorflow Lite [347].

To deploy the complete feature extraction process and quantized DNN model on hardware for real-time inference, the following approach is used: (i) CAN ID is normalized from the range of 0 to 2048 into 0 to 1. (ii) For Data_Field, the feature extraction process involves utilizing IDF values obtained from the training data. TF-IDF values are derived using Eq. (5.3). These calculated TF-IDF features are then normalized using L2-normalization. (iii) Finally, both normalized CAN ID and normalized TF-IDF values are fed as inputs to the quantized DNN model. We use Tensorflow Lite C API [348] to integrate the quantized model on hardware.

5.6 Experiments and Results

In this section, we present results of our experiments evaluating *CANLP* on the datasets and models described in Section 5.4. We evaluated *CANLP* on four datasets using five different ML algorithms- SVM, LR, GNB, DT, and RF. We also used the DNN described in Section 5.5.3 and evaluated the classification F1-Score.

We use character-level (1,2)-gram features for multi-class classification. Table 5.4 shows F1-Scores achieved by the different algorithms on the four datasets. The DNN achieves the best performance compared to other models: it has the highest F1-Score for three datasets and close to the best F1-Score for the remaining datasets. Hence, we choose DNN for deploying *CANLP* on hardware.

5.6.1 Selection of Best Model Weights for DNN

We train the DNN model for 30 epochs across all the datasets. Figure 5.3 plots training (blue) and validation (orange) F1-Scores and training (green) and validation (red) loss function values. The proximity between training and validation F1-Score curves shows that the DNN has very minimal overfitting [336]. Furthermore, the reduction in values of both training and validation losses over 30 epochs indicates that the model has been trained effectively. In order to determine the optimal model for deployment of *CANLP*, the model weights corresponding to the epoch with the highest validation F1-Score are chosen ($\approx$ 5 epochs in most cases).

5.6.2 Evaluation of DNN after Quantization

To assess effectiveness of *CANLP*, F1-Scores of quantized model are compared with the original model for the same test dataset. The quantization process results in a

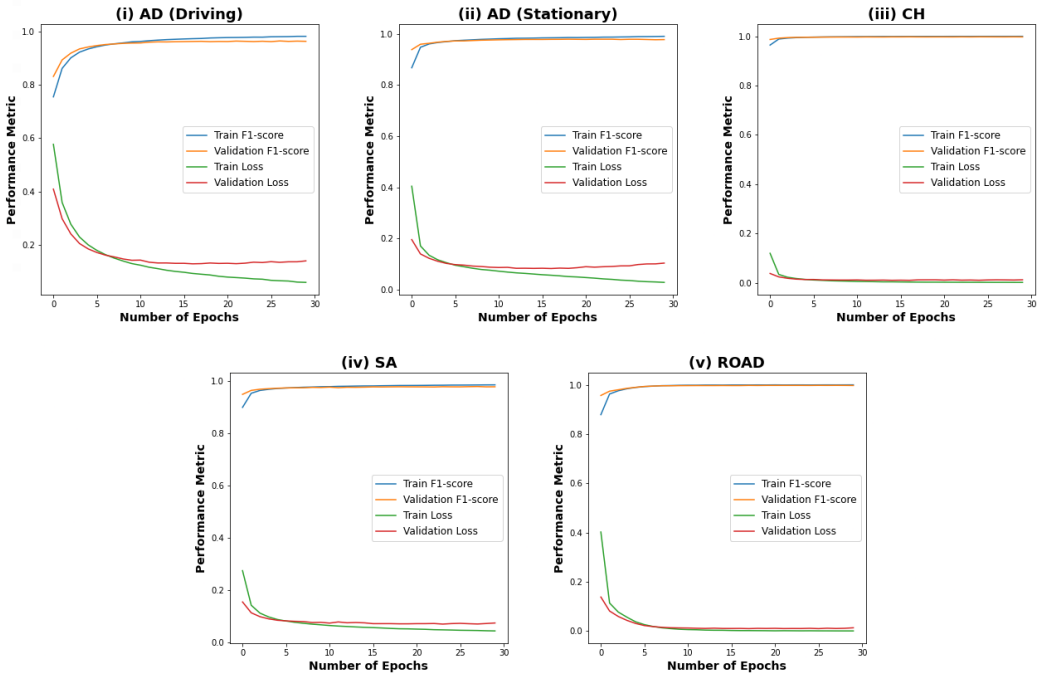


FIGURE 5.3: Four performance metrics: (a) training F1-Score (blue); (b) validation F1-Score (orange); (c) training loss (green); (d) validation loss (red) vs. number of epochs for the **DNN** when trained on: (i) AD (Driving); (ii) AD (Stationary); (iii) CH; (iv) SA and (v) ROAD datasets. Proximity between training and validation F1-Scores indicates that the DNN has very minimal overfitting. Additionally, the observed reduction in training and validation losses over the course of the epochs indicates that the selected learning rate is appropriate for the model training process. To determine the optimal model for deployment of *CANLP*, model weights corresponding to the epoch with the highest validation F1-Score are selected.

reduction of the original model size from 540.57KB to 171.75KB when float32 quantization is employed, and a further reduction to 45.58KB when dynamic quantization is utilized. We define a metric called *Post-Quant Model Reduction (PQMR)* to quantify the reduction in the size of the model as:

$$\text{PQMR} = \frac{\text{OMS} - \text{PQMS}}{\text{OMS}} \times 100\%, \tag{5.7}$$

where Post-Quant Model Size (PQMS) and Original Model Size (OMS) are the model size after and before quantization.

Table 5.5 shows F1-Score before (F1 Pre-Quant.) and after (F1 Post-Quant.) quantization, PQMS, and PQMR for float32 and dynamic quantization for all datasets. We observe close to zero reduction in F1-Scores for both float32 and dynamic quantization, indicating that the compressed model retains its predictive power despite significant reduction in size. We choose dynamic quantization for hardware deployment of *CANLP* due to its better PQMR value.

5.6.3 CAN Testbed Evaluation

To prove the feasibility of deploying *CANLP* in a real scenario, we construct a CAN testbed, shown in Figure 5.4. To simulate bus traffic from different ECUs, we used a

TABLE 5.4: F1-Score for four different datasets for five simple ML models (SVM, LR, GNB, DT, RF), and a small DNN. We observe that the DNN showcases the best performance compared to other models, exhibiting a higher average F1-Score for three out of five datasets and close to the best F1-Score for the remaining datasets.

Model	F1 AD (Driving)	F1 AD (Stationary)	F1 CH	F1 SA	F1 ROAD
SVM	0.815	0.921	0.971	0.873	0.982
LR	0.816	0.925	0.971	0.884	0.976
GNB	0.737	0.836	0.865	0.787	0.934
DT	0.967	0.961	**0.998**	0.974	0.991
RF	**0.968**	0.979	0.995	0.973	0.995
DNN	0.964	**0.979**	0.997	**0.976**	**0.997**

TABLE 5.5: Classification F1-Scores before (**F1 Pre-Q.**) and after **(F1 Post-Q.)** quantization, model size after quantization (**Post-Q. Size**), and model reduction percentage after quantization (**Post-Q. Reduction**) for both float32 and dynamic quantization for the AD (Driving), AD (Stationary), CH, SA and ROAD datasets. Since quantization has a negligible impact on F1-Score, deploying a quantized model on hardware enables achieving a high F1-Score in real-time. We use dynamic quantization for *CANLP* due to its improved PQMR value.

Dataset	F1 (Pre-Q.)	F1 (Post-Quant.)		Post-Q. Size		Post-Q. Reduction	
		Float32	Dynamic	Float32	Dynamic	Float32	Dynamic
AD (Driving)	0.964	0.964	0.964				
AD (Stationary)	0.979	0.979	0.979				
CH	0.997	0.997	0.997	171.75KB	45.58KB	68.23%	91.57%
SA	0.976	0.976	0.976				
ROAD	0.997	0.997	0.997				

Raspberry Pi (RPi) 3 Model B [291] equipped with a PiCAN2 Duo hat [92]. We program it to replay one dataset at a time, thus generating traffic in the CAN bus. As per specifications [181], our bus is terminated with 120Ω resistors and can support different bandwidths up to 1Mbps. The IDS is implemented and deployed on a RPi equipped with PiCAN2 Duo hat [92] as an interface with the bus since it is a low-cost (starting from $35) and popular microcomputer with sufficient resources. For real world implementation, this component can simply be connected to the OBD-II interface inside an automotive vehicle and function as an IDS with minimal modification to internal systems. For fair comparison with SOTA, we tested on two distinct versions individually: a RPi 4 Model B to compare with [398], and a RPi 3 Model B for comparison with [384]. The RPi 4 (Pi 3) Model B has a CPU running at 1.8GHz (1.2GHz) and 1GB (1GB) of RAM.

When the quantized model is deployed on the RPi 4 Model B on the testbed, we observe that the latency of attack prediction has a mean of 0.0492ms with a standard deviation of 0.0011ms for a sample of 27000 packets sent consecutively on the 1Mbps bus. Moreover, when the quantized model is deployed on the Raspberry Pi 3 Model B in the same setup, latency is observed to have a mean of 0.2133ms and a standard deviation of 0.0030ms.

5.6.4 Comparison with SOTA Models

A comprehensive analysis of *CANLP* and for other SOTA hardware-deployed models is presented in Table 5.6 when tested on the CH dataset for the performance

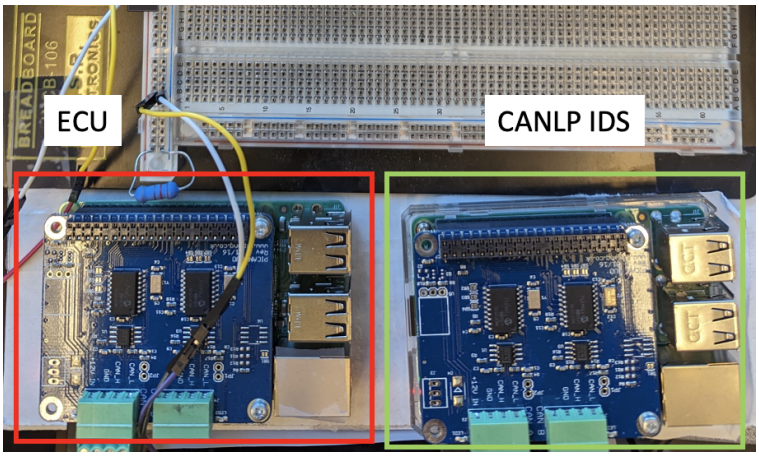


FIGURE 5.4: A picture of our CAN bus testbed. It has a Raspberry Pi 3 Model B equipped with a PiCAN2 Duo hat (Red) to send real-world traffic to the bus. A Raspberry Pi 4 Model B with PiCAN2 Duo hat (Green) reads data sent through CAN Bus and uses *CANLP* to make attack predictions.

metrics: F1-Score and Latency. Although direct comparison of F1-Scores is not possible due to the addition of synthesized data into our dataset, the attained F1-Score still aligns closely with those accomplished by other SOTA models. The results of our study emphasize that *CANLP* demonstrates superior performance compared to the SOTA models in latency, while also maintaining a comparable F1-Score across all attack scenarios. Hence, these findings strongly advocate *CANLP* as the optimal choice for real-time deployment.

TABLE 5.6: F1-Score and mean and standard deviation (Std) of latency for *CANLP* and other SOTA hardware-deployed models when tested on the CH dataset. NR indicates Not Reported. F1-Score and latency of models which perform binary classification (MTH-IDS and MA-QCNN) have been averaged and summed up respectively across all attacks for fair comparison against multi-class classification models presented in ACGAN and *CANLP* (Our Work).

Method	**Platform**	**CPU**	**F1**	**Latency [ms]**	
				Mean	**Std**
MTH-IDS [384]	Raspberry Pi 3B	1.2 GHz Broadcom BCM2837	0.999	1.722	NR
MA-QCNN [202]	Zynq FPGA	1.3 GHz Arm Cortex-A53	0.996	1.290	NR
ACGAN [398]	Raspberry Pi 4B	1.8 GHz ARM Cortex-A72	0.992	0.538	0.030
CANLP	**Raspberry Pi 3B**	**1.2 GHz Broadcom BCM2837**	**0.997**	**0.213**	**0.030**
CANLP	**Raspberry Pi 4B**	**1.8 GHz ARM Cortex-A72**	**0.997**	**0.049**	**0.011**

5.6.5 Quality of Features using t-SNE

t-Distributed Stochastic Neighbor Embedding (t-SNE) [369] is a dimensionality reduction technique to visualize high-dimensional datasets. t-SNE can be implemented via Barnes-Hut approximations [368]. Figure 5.5 shows 2-D t-SNE representations of $(1, 2)$-gram TF-IDF features extracted from training data across the datasets. The scarcity of individual data points illustrating Flooding and Spoofing messages in the plots is due to their overlapping nature, which makes them appear as a few distinct data points. Our t-SNE analysis reveals high level of separation

between legitimate and adversarial frames (while represented in the feature space), which validates the effectiveness of using character-level (1,2)-gram TF-IDF features to achieve high F1-Score.

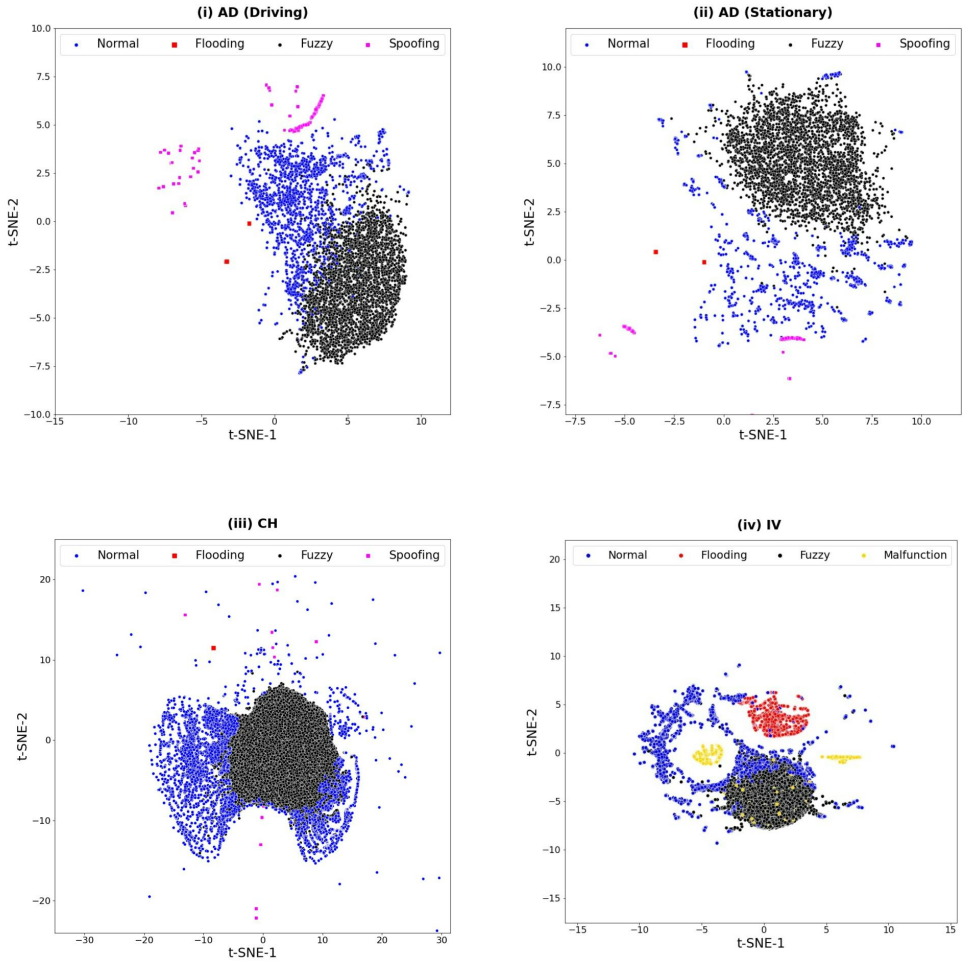


FIGURE 5.5: 2D t-SNE Representations of (1,2) gram character level TF-IDF features for (i) AD (Driving), (ii) AD (Stationary), (iii) CH and (iv) SA datasets. The visualization in each plot indicates a clear separability among features corresponding to the following classes: Normal message, Flooding message, Fuzzy message, Spoofing message, and Malfunction message. The limited number of data points representing Flooding and Spoofing messages in the plots is due to their overlapping nature, causing them to appear as few distinct data points.

5.7 Related Work

Many works in literature focus on intrusion detection for attacks on the CAN bus [256, 202, 35, 6, 20]. The authors of [256] converted the CAN ID and data to a decimal format and used them as input features for models such as DT, RF, Stochastic Gradient Descent (SGD) with hinge loss, and Naive Bayes (NB). An alternate approach in [202] segments CAN data into sliding windows and provides them as input to a quantized Convolution Neural Network (CNN) deployed on a FPGA.

However, FPGA-based CAN models only evaluated two types of attacks (DoS and Fuzzing) and showed marginal improvement in performance as indicated in [202]. A one-hot encoding technique for both CAN ID and Data and providing encoded data to a Self-Organizing Map (SOM) Network was proposed in [35]. However, the SOM network is computationally large and did not consider CAN data sequentially, making it impractical to deploy in real-time. In comparison, we evaluate *CANLP* on three different attacks: Fuzzing, Spoofing, and Masquerade.

Agrawal et al. [6] combine multiple CAN frames and provide it to a CNN followed by a Long Short-Term Memory (LSTM) network. Although LSTM enables capturing temporal dependencies in data, it incurs long processing delays and provides only incremental increase in classification scores over SOTA models [6]. In [20], the total packets from unique CAN ID and size of outbound messages from unique IDs are extracted as features and then given as input to an LSTM autoencoder-based model for attack prediction. A limitation, highlighted in [20], is that the model cannot be used for multi-class classification, which is used in *CANLP*.

The application of the N-gram TF-IDF feature model has found practical implementation in the domain of cybersecurity, encompassing diverse applications such as software vulnerability assessments [37, 390], and cyber threat detection [393, 14]. The researchers in [37] employed TF-IDF features derived from bug reports to create a tool specifically designed to identify software bugs. The authors of [390] used TF-IDF features of package manifest files to assess the security level of Android applications.

We use the insight that information transmitted on the CAN bus can be interpreted as 'words', which allows us to leverage techniques from NLP. Specifically, *CANLP* uses TF-IDF features extracted from data on the CAN bus to effectively distinguish normal messages from three different types of attack messages. We also show that our solutions can be deployed on resource-constrained hardware to detect attacks in real-time with high F1-Score and low latency.

5.8 Conclusion

In this chapter, we designed and developed *CANLP*, an intrusion detection system for CAN. To the best of our knowledge, *CANLP* is the first framework to use natural language processing to extract features from individual CAN frames that correspond to attack messages. Using the term frequency-inverse document frequency encoding technique to identify complex features associated with CAN data, we trained machine learning models and a deep neural network to detect and classify fuzzing, spoofing, and masquerade attacks. Compressing the models through quantization enabled deployment on resource-constrained hardware via a testbed.

Our experiments evaluating *CANLP* on four publicly available vehicle datasets yielded significantly high F1-Scores while maintaining low latency ($< 0.05ms$), indicating *CANLP*'s effectiveness as an intrusion detection system for timing opaque attacks. We demonstrate that *CANLP* is suitable for different makes and models of vehicles based on evaluation on our CAN testbed and the ease of implementation with minimal modification to in-built software or hardware.

6 CANTXSec: A Deterministic Intrusion Detection and Prevention System for CAN Bus Monitoring ECU Activations

Nowadays, automobiles are equipped with hundreds of ECUs controlling vehicle components and functionalities. The de facto standard to allow these devices to communicate is the CAN bus protocol [181]. Officially released in 1986 by Robert Bosch GmbH, it is still employed in almost every vehicle in the world. Moreover, it finds applications in other domains such as ICSs [352] and maritime vehicles such as ships [282]. Even though the protocol has some clear advantages, such as high resilience to faults and the requirements of two wires only [181], it suffers from security issues intrinsic in the protocol specifications [70, 255, 318, 216, 240]. For instance, the absence of authentication allows effortless spoofing [178], while the fault-tolerant error handling protocol allows attackers to launch DoS attacks against specific vehicle components [77]. Tackling these issues with an easily deployable solution is tough due to the sensitivity of networks in safety-critical CPS, where availability is the main concern. Moreover, the diffusion of the CAN bus technology in the automotive industry and beyond makes it difficult for manufacturers to accept radical changes to the standard.

Over the years, researchers have proposed different solutions to fix security issues, although none are widely implemented at the current time. A class of mitigations employs cryptographic primitives to authenticate frames and hide transmitted information through encryption [157, 151, 164, 271, 110, 325, 236]. However, these approaches suffer from several drawbacks. While it is difficult to deploy authentication mechanisms supporting retro-compatibility with already employed devices [110], it is even more complicated to imagine imposing a standard that mandates legacy ECUs to manage cryptographic functions, effectively making all ECUs already on the market unusable. Moreover, the close source automotive environment makes it cumbersome to reverse proprietary frame formats and reduces the number of people that may get involved [280]. Furthermore, as the CAN bus serves as a safety-critical communication channel, it is imperative to control delays tightly, and cryptographic solutions typically have a noticeable influence on performance [395].

To deal with the limitations of this environment, researchers started looking at it from different angles, implementing solutions working on top of the untouched bus. A classic approach is to employ an additional node to monitor the network and detect attacks [235]. This approach results in IDSs that monitor the format and content of frames to detect anomalies [346, 315, 107, 332] or match signatures of

known attacks [189, 213]. Other features of the communication have also been investigated. Researchers proposed solutions employing voltage-based security systems that monitor the precise voltage each ECU imposes on the bus to identify the transmitter [309, 381]. These solutions require no additional computational costs for ECUs but have been proven vulnerable to attacks [41, 301].

The inability to autonomously stop attacks is a considerable limitation of IDSs, which need to rely on external systems or human action. Countermeasures to attacks are often on a case-by-case basis, presenting mitigations to newly discovered attacks [317, 216, 345, 103]. However, applying several defense mechanisms could be expensive for a manufacturer, who may need to deal with incompatibilities. Moreover, it does not guarantee that the vehicle will be completely secure from attacks. Another approach is represented by Intrusion Prevention Systems (IPSs), offering the capabilities to detect and react to attacks, preventing or stopping entire categories of threats. Unfortunately, there has been limited research investigating IPSs on the CAN bus [250, 102, 316]. Although attacks can be effectively prevented [273], this could be explained by the dangers of blocking actions incorrectly identified as attacks (false positives), which may endanger the safety of the vehicle and its passengers. Moreover, in the automotive scenario, false positives represent also a problem in a detection-only paradigm. In fact, even a low number of false positives may result in alerts being ignored by drivers or several stops to have the vehicle unnecessarily inspected, deteriorating the reputation and trustworthiness of the automotive manufacturer. Therefore, developing a comprehensive solution that is unaffected by detection error is essential in this field.

Additionally, most of the attacks researchers proposed are based on CAN protocol-compliant attacks, exploiting the bus structure and the error handling mechanisms [77, 178, 317]. Less attention has been directed to attacks exploiting actions which do not conform to the CAN specification. Connecting a malicious device to a vehicle's bus [356] gives the attacker complete freedom in their actions, even injecting single bits instead of entire frames [42]. Despite ad-hoc devices, even standard compromised ECUs may suffer from design flaws that make launching these kinds of attacks possible [216]. If the offensive side of these attacks has been little investigated, to the best of our knowledge, the detection of such attacks has seldom been discussed.

There is, hence, the need for a solution that overcomes current limitations. It needs to offer retro-compatibility without increasing the computational burden on legacy ECUs. Moreover, no false positives should be allowed in the automotive scenario. This makes it difficult to employ ML models or other probabilistic approaches that are inherently open to rare but nonetheless present false alarms. A solution to develop a deterministic security solution resistant to false positives is to employ physical properties of the CPS. While an attacker can easily craft frames [355, 178], mimicking physical properties is at least harder than imitating network behaviors, if not impossible at all.

Contributions. In this chapter, we bridge these gaps by proposing *CANTXSec*, the first deterministic Intrusion Detection and Prevention System (IDPS) in the CAN bus based on the co-presence of activity on the transmitting line of an ECU and one of its frame on the bus. Specifically, after building a list of message IDs in the bootstrap phase, they are linked to the ECU activated when each ID is transmitted in the bus. Then, we employ this list to detect malicious traffic. Our approach relies on deterministic comparisons, representing a unique and essential improvement over probabilistic IDS and IPS in the literature (e.g., ML-based [257, 315, 102]), virtually

getting rid of any false positives. Moreover, *CANTXSec* does not require any modification to the standard, resulting in a solution compatible with legacy devices. It achieves 100% detection and prevention of common attacks compliant with the CAN standard discussed in the literature while being able to precisely detect advanced attacks based on deviations from the protocol specifications.

Our contributions can be summarized as follows:

- We propose a new classification system to separate known and future CAN bus attacks between classical FIAs (i.e., using frame-level access) and advanced SBAs (i.e., employing bit-level access). The two classes represent different access levels by the attacker and can be employed to classify future IDSs in the field.
- We introduce *CANTXSec*, a deterministic Intrusion Detection and Prevention System (IDPS) in the CAN bus based on the co-presence of frame IDs on the bus and corresponding ECU activations. Our solution only requires cheap additional hardware, without any software modifications, cryptographic protocol, or time-consuming training processes, while preventing FIA attacks and detecting SBA, zeroing out false positives.
- Through the development of a real-world testbed, we demonstrate that *CANTXSec* achieves 100% accuracy in detecting both categories of attacks. Moreover, we prove that our system can prevent FIA with a success rate of 100% without disrupting other communications in the bus. The deterministic detection ensures that no legitimate frames are stopped, thus making *CANTXSec* applicable in commercial devices without risks.
- We provide an analysis to facilitate the implementation of *CANTXSec* based on each user's risk assessment. In particular, we show that monitoring the activation of 30% of the overall number of ECUs in a car ensures attack detection on the majority of safety-critical functions.

Organization. This chapter starts by providing background insights in Section 6.1. Section 6.2 describes and categorizes attacks in the CAN bus, while Section 6.3 depicts the threat model we consider in this chapter. Then, Section 6.4 describes *CANTXSec*, our IDPS, and discusses its implementation. Section 6.5 illustrates the testbed we employed to obtain the results we discuss in Section 6.6. Relevant related works are presented in Section 6.7, while in Section 6.8, we analyze *CANTXSec* with respect to other papers and discuss implementation details. Finally, Section 6.9 concludes the chapter with some final insights.

6.1 Background

In this section, we discuss some background topics needed to understand the rest of the chapter. We overview the CAN bus protocol in Section 6.1.1, we introduce the architecture of ECUs in Section 6.1.2, and we explain different systems to detect and prevent intrusions in Section 6.1.3.

6.1.1 CAN bus

CAN is a broadcast-based bus employed in CPSs and, in particular, in the automotive environment [181]. Differential signaling between the two cables is employed

to provide resistance to noise and interference. The bus works as a logical *AND*: dominant values (zeros) win over recessive values (ones). After transmitting a bit, each node senses the bus to ensure the transmitted value is actually on the bus. If not, a collision is identified.

CAN frames start with a Start-Of-Frame (SOF) bit followed by the ID, which is used to identify message content and as an arbitration mechanism to decide which node is allowed to transmit. Lower IDs correspond to higher priority. During the ID transmission, each node sends one bit at a time and senses the bus immediately afterward. If a node sending a one senses a zero, it loses the arbitration and thus stops transmitting. Through this mechanism, CAN guarantees that, beyond the ID, only one node is allowed to transmit.

Whenever a collision is identified after the ID, an error is raised. Every node keeps two error counters. A Trasmitter Error Counter (TEC) counts errors during transmission, while a Receiver Error Counter (REC) monitors reception errors. Every transmitting error increases TEC by 8, while receiving error increases REC by 1. Every correctly sent or received frame decreases the respective counter by 1. Upon encountering an error, the node signals it broadcasting an *error frame*.

Error frames have different aspects based on the *error state* of the node. Error states are defined by TEC and REC values. Nodes start in the *error-active state*. Once their REC or TEC exceeds 127, they enter the *error-passive state*. If the TEC exceeds 255, they enter the *bus-off state*, where they stop communicating with the bus.

When in active error state, error frames are called *active* and are composed of 6 dominant bits. On the other hand, during the passive error state, *passive* error frames are composed of 6 recessive bits. This represents the only occasion when a stream of 6 identical bits could be transmitted in the bus. During the transmission of other frames, this is not possible because of *bit stuffing* mechanism. It states that whenever a node sends five bits of the same logic level (dominant or recessive), it must send one bit of the opposite level. This extra bit is automatically removed by receivers. This process helps ensure continuous synchronization of the network.

Figure 5.1 illustrates the different sections of a CAN frame. After SOF and ID, two control bits are sent: the Remote Transmission Request (RTR) indicates remote frame requests, while the Identifier Extension Bit (IDE) signals extended IDs. Then, a dominant reserved bit (r0) is sent. Follows the Data Length Code (DLC), a 4-bit value indicating the length of the data in bytes. Then, the actual content is sent, followed by a Cyclic Redundancy Check (CRC) and a recessive bit used as a delimiter (CRC DEL). Then, the transmitter sends a recessive Acknowledge (ACK) bit, during which receivers can confirm the reception of the packet by imposing a dominant value. After the ID, this is the only bit where a node that lost the arbitration is supposed to send data to the bus. Finally, seven recessive bits called End-of-frame (EOF) conclude the frame. Before the start of a new frame, three recessive bits are sent and are called Inter-Frame Spacing (IFS).

6.1.2 Electronic Control Unit

An ECU is an embedded system employed in automotive electronics to control one or more of the electrical systems in a vehicle. It is usually connected to a variety of sensors and actuators. The process running on an ECU varies from simple and well-defined tasks (e.g., a brake system) to more complex and power-consuming operations (e.g., the infotainment ECU). The majority of ECUs need to communicate with each other to exchange information about the vehicle's state through the CAN bus. ECUs are therefore required to comply with the CAN standard [181] to allow

this communication channel to work properly. A controller is usually employed as an interface between the ECU's microcontroller (MCU) and the bus, as shown in Figure 6.1. It implements the standard and is in charge of all the CAN related operations, such as buffering and queuing CAN frames that need to be transmitted, packet filtering, arbitration management, and error handling, including the management of error counters [253]. The controller can be included in the MCU Printed Circuit Board (PCB) or added as an external device [273]. In this latter case, the MCU employs general-purpose communication protocols (e.g., Serial Peripheral Interface (SPI)) to communicate with the controller. This approach allows developers to increase the range of MCUs they can employ, including boards without CAN bus support.

The logical output of the controller needs to be converted to the differential signaling employed by the CAN protocol. A transceiver converts the controller output (i.e., CANTX) to a signal compliant to the CAN standard [181]. Moreover, it reads bits in the bus and transmits values to the controller through the CANRX wire [252]. While it is possible to implement the controller as software [42], the transceiver is essential to convert digital values to signals that can be understood by the bus.

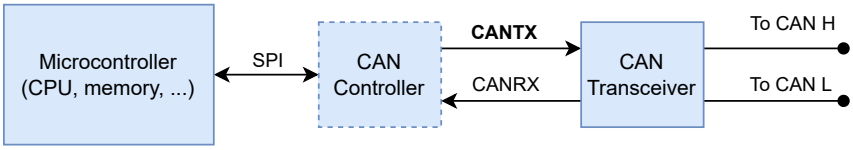


FIGURE 6.1: The architecture of a standard ECU. The MCU sends packets to the controller, which is in charge of queuing messages and transmitting them to the bus whenever possible. The transceiver is instead employed to convert the controller output to a CAN compliant signal, and vice-versa. While the transceiver is fundamental to allow communication with the bus, the controller may be implemented in the ECU software.

6.1.3 Intrusion Detection and Prevention Systems

To mitigate the possible presence of attacks in a system, an Intrusion Detection and Prevention System (IDPS) could be deployed. IDSs focus on detecting malicious behaviors, leaving it up to the user to act accordingly. On the other hand, IPSs try to prevent attacks by automatically acting on the system [64]. IDPS could be deployed to monitor single hosts' behavior or to get data by sensing the network traffic [228]. Since we want to avoid modifying ECUs' software, in this chapter, we focus on a network IDPS, similar to other papers in the literature [315, 332, 189, 235].

Two main strategies are usually employed to identify attacks [135, 349]. Signature-based IDPS looks for known malicious patterns to identify attacks. On the other hand, anomaly-based IDPS tries to detect abnormal features with respect to the usual state of the system, usually employing ML techniques. The former reduces the number of false positives to the minimum but cannot identify novel (e.g., zero-days) vulnerabilities as the latter. In our work, we will develop an anomaly-based IDPS with the peculiarity of not using ML. Instead, we employ a deterministic approach relying on simple comparisons between physical values sampled in different vehicle parts to assess if a malicious ECU is launching an attack.

TABLE 6.1: Attacks available in the CAN bus in the literature and effectiveness of detection (Det.) and prevention (Prev.) of *CANTXSec*. [1]: detectable for packets with spoofed IDs; [2]: starting from the first spoofed packet.

Attacks	**Type**	**Det.**	**Prev.**
Flooding/DoS [275]	FIA	✓	✓
Frame Spoofing [355]	FIA	✓	✓
Adaptive Spoofing [355]	FIA	✓	✓
Replay Attack	FIA	✓[1]	✓[2]
Original Bus-off Attack (BOA) [77]	FIA	✓	✓
Stealthy BOA "single-frame" [317]	FIA	✓	✓
Error Passive Spoofing [117]	SBA	✓	✗
Double Receiving Attack [355]	SBA	✓	✗
Stealthy BOA "one-packet" [316]	SBA	✓	✗
Freeze Doom Loop Attack [355]	SBA	✓	✗
Shutdown via Clock Gating [216]	SBA	✓	✗
Selective DoS [273]	SBA	✓	✗

6.2 Attacks on CAN bus

Being employed in safety-critical systems, the CAN bus has been the target of numerous attacks, both from researchers and from malicious actors [77, 316, 317, 356, 355, 216, 255]. Table 6.1 summarizes most of the attacks documented in the literature targeting CAN, indicating the type and the ability of *CANTXSec* to detect and prevent them.

In usual threat models in the literature [41, 70, 124, 211, 316], the attacker gains access to an ECU, which is employed to attack the bus. However, ECUs are not all the same. They can be deployed in different subnetworks while being connected to different sensors and actuators. Moreover, different ECUs are equipped with different capabilities that could allow attackers to compromise different parts of the system. The ECU's firmware and its hardware architecture are also part of the game: experienced attackers may find and exploit bugs in ECUs through reverse engineering that results in sophisticated access to the CAN bus, while unskilled attackers may rely only on high-level access to the bus provided by the ECU frontend. All of these factors are critical in assessing the types of attacks that can be carried out and, consequently, the effectiveness of identification and prevention methods. In this chapter, we introduce a distinction between two fundamental attack classes in the CAN bus: *Frame Injection Attack (FIA)* and *Single-Bit Attack (SBA)*.

Frame Injection Attacks. FIAs include the most common attacks in the literature, where requirements on the compromised ECU are relaxed since the attack does not require any particular capability other than the ability to ask the controller to send frames. A malicious entity can obtain such access by compromising an ECU with malware [186], exploiting software bugs [255], or installing a malicious device directly on the bus [356].

Many attacks can be adapted from the IT scenario within this threat model. Flooding attacks with low IDs (i.e., high priority) slow down periodic frames and may lead to DoS [275]. Through frame spoofing, an attacker can send tampered information on the bus, possibly taking care of avoiding collisions with other frames

with the same ID using an adapting spoofing attack [355]. Network captures from the bus can also be replayed later and several times to hide malicious activities or masquerade other attacks performing a replay attack. Moreover, the original Bus-Off Attack (BOA) does not require any special access to the bus [77]. BOA exploits the error handling mechanisms to increase error counters in a victim ECU, pushing it to a bus-off state and effectively stopping the device from sending any frames. The main challenge is related to synchronization because the attacker's message must overlap with the victim's one. Researchers found different ways to synchronize the attacker device with the bus without any special access to the bus, such as exploiting the periodicity of CAN bus messages [77] or preceding IDs [317].

Single Bit Attacks. If these kinds of ECU are the most spread, there exist cases where the attacker needs more *fine-grained access* to the bus, monitoring it not frame-by-frame but bit-by-bit. This can be the case when the adversary needs precise synchronization in the bus and the ability to inject single bits during other ECU transmissions, possibly disregarding the CAN bus specification [181]. Different ways exist to achieve this: 1) installing a malicious device developed ad-hoc without a controller, 2) bypassing or hacking the controller of an ECU [216], or 3) obtaining access to an ECU which does not employ a controller and which, therefore, allows fine-grained access to the bus. Apart from the installation of a malicious device, it is worth noticing that the effort needed to obtain this level of access is considerable, and it is not always possible. While it is theoretically feasible with some MCUs [337] that allow multiplexing the CAN controller output in standard GPIO pins, it is more complex if the output pins are not reconfigurable or the controller is connected via serial connections (e.g., SPI). Other strategies may be possible by exploiting wrong design choices or by interfering with advanced ECU functions such as the clock [216]. We call attacks requiring such an access SBAs since, usually, this control on the bus allows attackers to generate errors by injecting a *single bit* in specific instants during the transmission of frames by other ECUs [316, 317, 216].

With such precise access, many other advanced attacks are possible. For instance, Error Passive Spoofing [117] is a two-stage attack that requires 1) forcing an error passive mode victim into generating an error frame and 2) replacing the recessive data and CRC bits with the spoofed payload. A Double Receiving Attack forces the retransmission of a frame by imposing a dominant value in the last EOF bit [355]. Moreover, direct bus access enables stealthier versions of already discussed FIAs, such as the one-packet BOA [316] or the Selective DoS [273]. A legacy feature of the CAN bus offers a technique to decrease the bandwidth of the bus and gain time to finish computations for slow ECUs. It works by imposing a dominant value in the first IFS bit [355]. An adversary may exploit it with a Freeze Doom Loop Attack, forcing the dominant bit in a frame and then again to the consequent overflow error frame that will be generated. A peculiarity of these kinds of errors is that they do not increase error counters, making the attack stealthier while keeping the bus busy as long as the attack is ongoing. Kulandaivel et al. [216] proposed an attack to shutdown ECUs exploiting design errors of certain ECUs in the wild.

Since our approach is tied to the physical aspects of the attack, we decided to separate attacks into FIAs and SBAs. As extensively described in Section 6.4, *CANTXSec* can detect both kinds of attacks, but in two different ways strictly depending on the category. On the other side, using our approach, prevention is possible for the former category only. Moreover, this categorization is important for future work

on risk assessment. Even though it may happen that SBAs have more severe effects, they are usually less likely to happen because of the challenging environment needed to carry them out.

From our classification, we intentionally left out *modification attacks*, where an attacker compromises an ECU and uses it to replace the content of legitimate frames, i.e., without spoofing another ECU's ID. Being an attack on the data level and not on the network level, it is independent of the communication protocol employed and, therefore, not in the scope of this chapter. In addition, it is worth noting that attackers usually exploit vehicles through over-exposed ECUs, such as the infotainment system, which in normal scenarios is not required to send critical messages on the bus. Therefore, to create effective damage, attackers usually need to forge message IDs to spoof the identity of sensitive ECUs. To defend the system against these attacks, various techniques exist in the literature, even directly targeting CPS [258] or vehicle [75, 31] environments.

6.3 Threat Model

In this chapter, as common in the CAN bus security literature [41, 70, 124, 211, 316], we consider a malicious attacker with the capabilities to compromise ECUs software remotely (e.g., via unsecured over-the-air firmware update mechanisms [287], via Internet [255]) or physically.

We increase the strength of the attacker, including physical attackers able to install a new malicious device in the bus [356]. Furthermore, with respect to similar recent works [316], we equip the attacker with the capability to hack or bypass the ECU's controller to act on the transceiver directly and thus on the bus. This is trivial for a physical attacker but also possible for remote attackers [216]. However, this advanced threat model has some consequences on which attacks can be detected and prevented, as we will discuss in Section 6.8.2.

6.4 *CANTXSec*

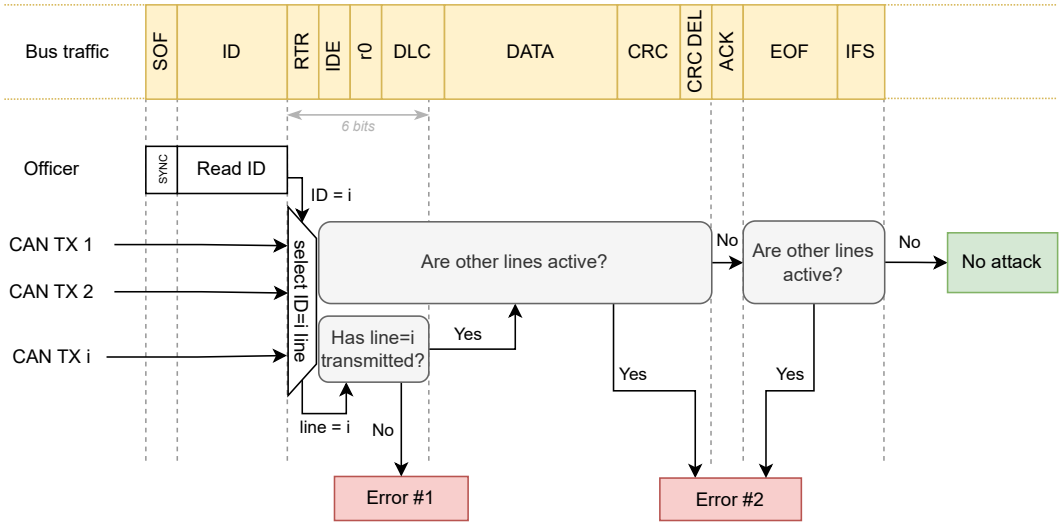


FIGURE 6.2: The flow chart explains the different checks *CANTXSec* performs during each bit of every frame sent through the bus.

In this chapter, we propose *CANTXSec*, the first IDPS on CAN bus, which employs fine-grained measurements from ECUs activities to detect and prevent network attacks in the bus. *CANTXSec* collects data on the CANTX pin of each ECU and correlates these values with a bit-by-bit reading of the bus. When a malicious communication is detected originating from a compromised ECU and stopping the frame is safe, *CANTXSec* blocks the transmission of the message by generating errors, eventually forcing the compromised ECU to a bus-off state. The architecture of a system employing *CANTXSec* is depicted in Figure 6.2. A new device, the so-called *officer*, is connected to the bus to detect and react in case of attacks.

6.4.1 Officer

An officer with bit-level access to the bus is the core component of *CANTXSec*. It should be an MCU connected to a CAN transceiver. For the officer, a CAN controller is unnecessary since the device should be able to take action on frames almost in real-time without waiting for the end of the packet. To this aim, the officer MCU demodulates frames on the fly at the software level. A transceiver is, hence, everything required to enable the MCU to interface with the bus if not already provided by the board.

Moreover, the officer is connected via a single dedicated wire to the CANTX lines of all the monitored ECUs to detect their activities. This wire is only intended to measure digital voltage levels and signal the activation of the CANTX line to the officer. As depicted in Figure 6.1, the CANTX line represents the connection between the controller and the transceiver or, if the controller is not present, between the transceiver and the MCU. The CANTX line shows a high value when either a recessive bit needs to be sent on the bus or when the transmitter is idle. It instead shows a low value when a dominant bit should be imposed on the bus.

6.4.2 Setup

The first part of the *CANTXSec* setup phase requires physical access to the vehicle and its components. In fact, we need to connect a wire from the officer to the CANTX of all the ECUs to be protected. A perfect time to perform the setup is during automobile manufacturing while assembling the vehicle, but a technician can also do it a posteriori. All the ECUs must be wired to protect against the highest possible number of attacks. However, a lot of attacks can be prevented by connected ECUs even without having all the devices connected to the officer, as we discuss in Section 6.8.2. Based on the risk assessment conducted on each vehicle, the developer and/or the owner can decide how many and what ECUs should be protected [268].

Each CANTX pin connected to the officer represents a unique ECU. For each ECU, the officer maintains a list indicating which IDs are allowed to be transmitted from that particular ECU. The second step of *CANTXSec* setup is hence the creation of this list. The manufacturer can populate the list through manual inspection of the specifications. Otherwise, it can be achieved by running the automobile in a controlled environment for a sufficient time to ensure that all relevant ECUs activate at least one time. Table 6.2 illustrates an example of the information the officer needs to maintain. For instance, the ECU connected to $Pin = 1$ is allowed to transmit three different IDs, namely $0x123$, $0x234$, and $0x345$.

As discussed in Section 6.1.1, each ID is strictly linked to a particular ECU, and there should never be two ECUs with the same ID in the bus, as per standard [181].

TABLE 6.2: The officer maintains a table connecting to each pin (which is wired to the CANTX of an ECU) the IDs that ECU is allowed to transmit.

Pin #	**ID(s)**
1	$0x123$, $0x234$, $0x345$
2	$0x222$
...	...
n	$0x333$, $0x444$

With respect to other IPSs [102], *CANTXSec* only requires this setup step to start working properly without any time-consuming data collection and training of ML models. Updates on the vehicle, such as the replacement or installation of a new ECU, require connecting the appropriate cable to the CANTX line and updating the list on the officer. These kinds of actions are straightforward for a technician, and no additional expertise is needed, making *CANTXSec* very easy and fast to upgrade.

6.4.3 Attack Detection

The intuition on how *CANTXSec* works is as follows. Imagine that a frame with arbitration ID = i is being transmitted on the bus. For each bit transmitted, we want to assess that 1) the correct ECU (i.e., the one associated with ID i) is transmitting, and 2) all the other ECUs are quiet. Implementing this approach using loops is inconvenient due to the real-time nature of the problem. Instead, a better solution involves the employment of interrupts. They are available in modern MCUs and can be triggered by a certain voltage change in a pin [338]. It is worth recalling that the CANTX lines are set to 1 while the connected ECU is in an idle state. This makes it more difficult to assess if a certain ECU is transmitting a 1 (recessive value) or if it is waiting. However, because of bit stuffing [181], a 0 should be transmitted periodically, even if not included in the original message. This makes it possible to detect attacks using edge interrupts with reasonable delays, as discussed in Section 6.8.3. These interrupts fire when a voltage variation is registered in the line, both from a high value to a low value and vice versa.

The SOF is used by the officer for synchronization, as shown in Figure 6.2. During the arbitration, *CANTXSec* is silent, waiting to know which ID wins the arbitration. After arbitration, only the CANTX of the winner ECU is authorized to transmit dominant values (except for the Acknowledge bit, as discussed later). The officer performs the first check after 6 bits following the end of the arbitration ID. In that period, the CANTX line i should have transmitted at least a 0 (because of bit-stuffing, as detailed in Section 6.1.1). Otherwise, *CANTXSec* raises an Error #1.

The second check starts again after the arbitration ID but continues up to the end of the packet, checking if some of the other CANTX lines (i.e., all except for CANTX line i) are transmitting a dominant value. If so, *CANTXSec* raises an Error #2. This ensures that no ECU except the one who wins the arbitration sends bits in the bus.

As depicted in Figure 6.2, there is one bit that is not considered in this check. In fact, the ACK bit is meant to be set dominant by other ECUs to acknowledge the reception of the frame. Therefore, the check for Error #2 is paused during that bit since other ECUs are allowed to transmit. It is also worth mentioning the existence of a legitimate behavior where the first IFS bit is set to zero to signal an overflow. This was used in legacy systems to allow a receiver to delay the next message while

executing computation on the previous message [181]. However, it is no longer employed today and can be exploited to launch a Doom Freeze Attack on the bus [355]. Therefore, we do not consider this possibility in our system. However, it can be easily implemented by excluding the check for Error #2 during that bit.

Compared to other IDS and IPS in the literature, *CANTXSec* does not rely on statistical or ML models to detect and prevent attacks. Instead, it is based on *physical measurements* and *co-presence* between ID values and corresponding ECU *activation* in real-time. This ensures a deterministic solution that leads to perfect attack detection.

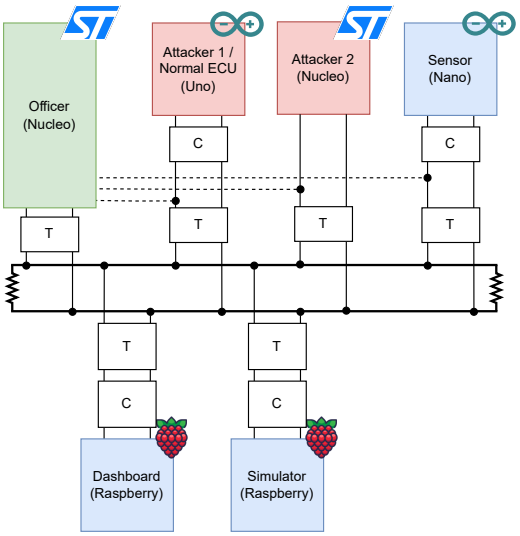
6.4.4 Attack Prevention

When the officer detects an attack, it can act in two different ways. One option is just to detect it and alert the driver, who can then decide what action to take. Another option is instead to stop the frame and, therefore, the attack. If detection cannot prevent attacks in real time, it allows alert verification and avoids blocking legitimate packets when false positives are detected. This is one reason why such systems have not found huge implementation in the literature: even if the false positive rate is usually low [315, 331, 309], it may have an impact on the driving experience. *CANTXSec*, instead, reaches 100% accuracy without false positives: this allows us to safely implement a prevention mechanism, being sure not to compromise the reliability of the bus.

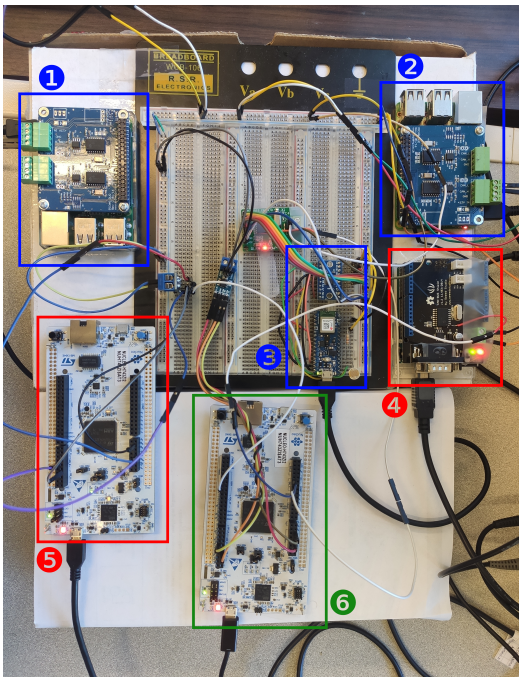
To stop a frame on the bus, the officer *injects a dominant value until a recessive bit is overwritten*. This generates an error on the transmitter, which stops the frame transmission and starts sending an error frame. The error frame depends on the ECU state. In error active mode, it is composed of 6 dominant bits, while in passive mode, it comprises 6 recessive bits. However, even if this is enough to stop the frame [102], it may not be the smartest strategy to block the attack completely. By default, controllers encountering a transmission error will reschedule the same packet again in the next frame time, which will be stopped by the officer again [253]. Such a process increases the busload, and the attacker may exploit this behavior to slow down the bus, performing a DoS.

A more efficient strategy aims to force the compromised ECU into a bus-off state. While there exist different ways to perform the so-called *bus-off attack* [77, 317], the most efficient is the *Instant Bus-Off* proposed by Serag et al. [316]. It forces an ECU into a bus-off state, targeting a single packet in $510\mu s$ on a $500kbps$ bus. The idea behind this attack is to target one packet with an error and then target the error frames generated by the first error injection. By iterating this process, error counters on the compromised ECU will increase, eventually resulting in a bus-off state. More details are available in the paper [316].

While FIAs can be easily stopped with this strategy, SBAs cannot. SBAs generate errors while transmitting of another frame by a legitimate ECU, violating the CAN protocol [181]. Since, when the attack is detected, the frame being transmitted is the legitimate one, the same strategy cannot be applied. Otherwise, it will target the legitimate ECU. For this reason, the safest solution is only to detect the attack and notify the user, who should decide the most suited action based on the context. Discriminating between attacks to be stopped and attacks to be detected is easy since Error #1 is related to FIAs that can be prevented, while Error #2 indicates that a user action is required to avoid jeopardizing the vehicle's safety.



(A) The officer is connected to the CANTX of the monitored ECUs, which is the wire connecting the transceiver (T) and the controller (C), when available.



(B) ❶: Dashboard; ❷: Simulator; ❸: Sensor; ❹: Attacker 1 / Normal ECU; ❺: Attacker 2; ❻: Officer.

FIGURE 6.3: The architecture (Figure 6.3a) and picture (Figure 6.3b) of the employed testbed.

6.4.5 Development

A suitable officer to implement *CANTXSec* should reproduce the logic summarized in Figure 6.2. Since no ML or other computationally expensive calculation is involved, a cheap MCU is enough as long as it provides edge interrupts and enough GPIO pins to connect all the monitored ECUs. Precise analog voltage measurement capability by GPIO pins is not needed since the officer only needs to obtain the digital level of CANTX lines. Moreover, the officer should include a connection to the bus, which allows reading each singular bit in real-time (i.e., without a controller). Because of the controller's absence, the software is responsible for decoding each frame ID and deciding which CANTX lines should be monitored. To identify errors, a computationally lightweight approach includes the employment of interrupts to detect changes in the CANTX lines connected to the officer's GPIO pins. Errors and prevented attacks should be logged and reported to the user so that further analysis can be performed.

The code of our prototype and all the devices employed in the testbed are available in our Github repository [1].

6.5 Testbed

To test *CANTXSec*, we developed a testbed employing different components to simulate a real CAN bus as closely as possible. We employed different microcontrollers

[1] https://github.com/donadelden/CANTXSec

and microprocessors to create a more realistic environment. The testbed schema is depicted in Figure 6.3a, and the use of each component is described in the following.

- **Officer**: an STM32 Nucleo H743ZI2 board [337] employed to run *CANTXSec*. It is connected to the bus via a transceiver without any CAN controller and to the ECUs under control through GPIO pins.
- **Attacker 1/Normal ECU**: an Arduino Uno [19] with Seeed Studio shield [314] for the connection to the bus employed as a data transmitter. During the frame spoofing attack, this ECU is considered compromised. Otherwise, it behaves as a normal ECU broadcasting a random value every 100 milliseconds. The Seeed Studio shield included both a transceiver and a controller.
- **Attacker 2**: an STM32 Nucleo H743ZI2 board [337] emulating an ECU with direct access to the bus (i.e., not mediated by a controller) through a transceiver. Even though it is not an ideal design solution, it may be an optimal target for an attacker aiming to perform a SBA.
- **Sensor**: developed with an Arduino Nano 33 BLE [19] acts as a sensor broadcasting a physical parameter (the light received by a photoresist) every 100 milliseconds. It represents one of the many sensors a modern vehicle contains to monitor different vehicle and environmental parameters (e.g., the temperature of the engine, external temperature, tire pressure).
- **Dashboard**: running in a Raspberry Pi 3b [291] with a PICan Duo hat [327], it emulates a dashboard, essentially receiving and logging the traffic on the bus. The shield includes both a controller and a transceiver.
- **Simulator**: running in a Raspberry Pi 3b [291] with a PICan Duo hat [327], it created background noise on the bus by continuously sending traffic from a dataset. The dataset has been created by collecting 10 minutes of traffic while using an instrumentation cluster simulator [281].

All the devices are connected to a simulated bus with terminal 120Ω resistors as mandated by the standard [181]. Figure 6.3b illustrates the final testbed with all its components.

6.6 Results

Even if the attacks presented in Table 6.1 have different goals, we can summarize them into two categories based on the physical mechanisms needed to perform the attack successfully, as described in Section 6.2. Since *CANTXSec* is focused on the bit level, the end goal of the attack is not relevant to the detection. In other words, independently of the final target of the attack, our system is able to identify malicious injections that compose the attack. Therefore, we implemented some representative attacks using different technical methodologies, which we believe are comprehensive for the majority of the attacks in the literature.

Table 6.3 summarizes the attack's scores and *CANTXSec*'s accuracy in detecting and preventing attacks. The Attack Success Rate (ASR) is measured depending on the attack's goal, and it is explained in the following sections. The dashboard collects frames successfully sent through the bus while all the relevant ECUs log the packets transmitted for further analysis and correlation. We ran each experiment for at least

10 minutes, generating several attacks during that timeframe. Furthermore, we propose a simple example showing how *CANTXSec* can effectively block an attack and restore the system's normal behavior.

TABLE 6.3: Effectiveness of *CANTXSec* in detecting and preventing the implemented attacks, which are presented with their ASR. Preventing Selective DoS is not possible with this approach.

Attack	Type	ASR	Detection	Prevention
Frame Spoofing without traffic	FIA	100%	100%	100%
Frame Spoofing with traffic	FIA	100%	100%	100%
Selective DoS	SBA	100%	100%	NA

6.6.1 Frame injection

As a representation of FIAs, we implemented two different frame spoofing attacks, considering that the spoofed ID is also transmitted from another ECU or is only sent by the attacker's device.

Frame spoofing without legitimate traffic. In this scenario, the attacker is able to stop a monitored ECU from sending data. For instance, they can exploit a BOA [77] or use malware to compromise the ECU [186]. Moreover, the attack has the control of an ECU through which they can send frames on the bus with arbitrary ID. We measure the ASR as the percentage of malicious frames transmitted by the attacker that are correctly received by the dashboard. An attack of this kind has a 100% ASR since there are no countermeasures applied by default against spoofing attacks.

Without detection or defense mechanisms, the recipient of the ID would not notice a difference in the packets since the ID is spoofed. If the officer is activated in detection mode, instead, the driver could be notified of every spoofed packet sent on the bus. In the testbed, this happens for 100% of the attacks, matching with theory since attack detection is deterministic, as explained in Section 6.4.

Furthermore, frame spoofing attacks can be prevented by setting the officer into prevention mode. In this case, when a malicious packet is sensed in the bus, the officer will launch a BOA against that ID and, therefore, against the compromised ECU. Figure 6.4 shows on an oscilloscope the prevention mechanisms firing against a malicious frame. Even in this case, the success rate is 100%, and no malicious packets are ever completely sent in the bus. Thus, the receiver does not receive any spoofed data.

Frame spoofing with legitimate traffic. Similarly to the previous attack, the malicious actor has compromised an ECU through which they can send spoofed frames with any arbitration ID. Compared to the previous scenario, this time, the attacker did not stop the legitimate ECU from sending frames on the bus. Therefore, if the attacker sends packets continuously without caring about what is happening on the bus, there may be collisions with the legitimate packets. However, a smart attacker can easily sense the bus and send packets just after legitimate ones [355]. With these techniques, they can ensure a 100% of ASR, which is measured as the percentage of malicious frames correctly received by the dashboard.

Detecting this attack may seem more challenging since the officer needs to differentiate between malicious and legitimate packets with the same ID. However,

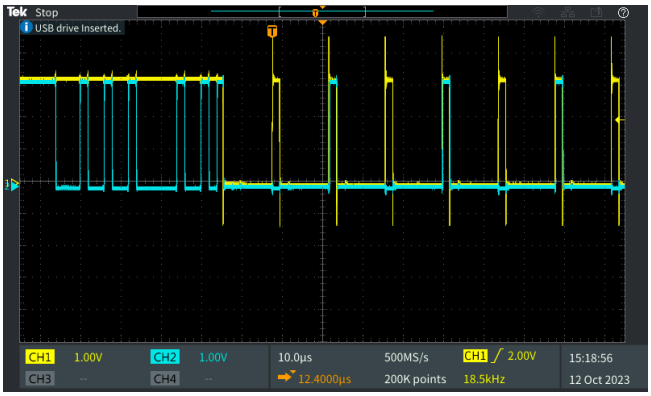


FIGURE 6.4: Start of a BOA as seen in an oscilloscope. The blue signal is the target frame (i.e., the malicious frame). The yellow signal represents the injection made by the officer. The first injection of a dominant value stops the malicious frame. Then, the following injections target error frames automatically generated by the attacker's ECU. This increases the attacker's ECU error counters, eventually reaching the bus-off state.

since the officer knows the state of the CANTX of the monitored ECUs, it can tell the source of a frame. With this strategy, during detection mode, 100% of the malicious packets are detected, while 0% of the legitimate packets are wrongly classified as malicious.

In this scenario, prevention shows the same difficulties as detection. When the officer is in prevention mode, it can block all the malicious frames, pushing the attacker's ECU into the bus-off state without modifying the behavior of legitimate frames that are received by other ECUs correctly. In our experiments, *CANTXSec* got a success rate of 100% in stopping malicious frames.

6.6.2 Single bit access attacks

If detecting and partially preventing frame spoofing has already been investigated by other works, SBAs are intrinsically more challenging to detect. These kinds of attacks bypass the controller with different techniques, for instance, using a modified ECU, hacking the controller, or exploiting design errors (e.g., bus clock gating [216]).

As a representative example of these attacks, we implemented the *Selective DoS*, first proposed by Palanca et al. [273]. Usually, these kinds of attacks exploit the generation of errors in the bus to maliciously stop a packet or increase error counters in ECUs. Since these bits are injected during the transmission of legitimate packets, trying to stop them by launching BOAs is counterproductive for two reasons. First, this will likely stop the legitimate frame and possibly help the attacker by increasing even more victims' error counters. Second, to control single bits in the bus, the attacker has probably bypassed the controller in some way and, therefore, they no longer need to behave to the standard. For these reasons, we just investigate the detection of these kinds of attacks since prevention is not possible with our strategy. After detection, the victim should stop the automobile and ask for assistance to identify and fix the problem.

The Selective DoS [273] works by injecting a dominant value during the transmission of a recessive value, thus generating an error that stops the packet transmission. The transmitting ECU reacts by sending an error frame and rescheduling the transmission of the packet as soon as possible. Then, the attacker uses the same strategy

to stop the retransmitted frame, and the cycle begins again. In some cases, the victim ECU may go into the bus-off state, completely stopping packet transmissions.

We implemented the attack on a Nucleo H743ZI2 board [337] (Attacker 2 in our testbed) by monitoring the bus for frames with the target arbitration ID and then injecting a dominant value during the first transmission of a recessive bit. To check the effectiveness of the attack, we monitor the reception of packets from the dashboard while launching the attack. We measure the ASR as the percentage of packets correctly stopped by the attacker. We assessed that no packets are received when the attack is active, indicating an ASR of 100%.

The detection of Selective DoS goes through the identification of injections of single bits after the transmission of the arbitration ID. Since, by design, the attack will generate a lot of consecutive traffic (i.e., the retransmissions), the officer MCU is not fast enough to print an alert for each message. Therefore, we employed a different strategy. We program the attacker to target exactly 200 frames (regardless of whether they are first sends or retransmissions), and we program *CANTXSec* to alert every 200 detection. By repeating this process several times, we were able to ensure that all the attacks were identified by *CANTXSec*, thus reaching a 100% detection rate for SBA. In a production environment, there are different strategies to mitigate this issue. First, a faster MCU may be employed, possibly implementing the logic on a Field Programmable Gate Array (FPGA) to maximize performances. Second, another MCU may be delegated to collect alerts and queue the printing of errors to the driver. Third, since getting the precise number of attacks is usually not essential, the driver may be alerted only for the first attack received.

6.6.3 A toy example of attack prevention

To demonstrate the capabilities of *CANTXSec*, we conducted an experiment in a scenario that could happen in a real vehicle. In particular, we imagine an attacker launching a spoofing attack against a sensor in order to tamper with the real value. We targeted a light sensor that is available in our testbed. Similar sensors are everywhere in vehicles, for instance, to measure engine temperatures or tire pressure. As explained in Section 6.5, the environment light value is broadcasted every 100ms. The value is almost constant and exhibits small changes through time since the environment brightness does not usually vary with high frequency.

We started with a bus without any security measures. At the time $t = t_0$, we imagine that a compromised ECU monitored by *CANTXSec* starts sending frames at a very high frequency containing a tampered value and spoofing the light sensor ID. The effect is visible in Figure 6.5, where it is easy to spot the malicious high value being imposed. In particular, from t_0, the flooding of tampered messages with an abnormally high value tries to push the reading out of the normal scenario. The consecutive drops are caused by the legitimate reading that is still periodically sent from the sensor and, therefore, received from the other ECUs. An advanced adversary could exploit a BOA against the legitimate sensor to avoid the drops [77, 178].

To prevent the attack, we activated *CANTXSec* in prevention mode at time $t = t_1$. Blocking malicious frames restores the normal value, and the attack is stopped. This highlights our system's capabilities to stop spoofing attacks, which are the starting point for all FIAs. Of course, with respect to this example, in a real vehicle, the system will be activated when the vehicle is turned on, so the attack will be stopped from the beginning.

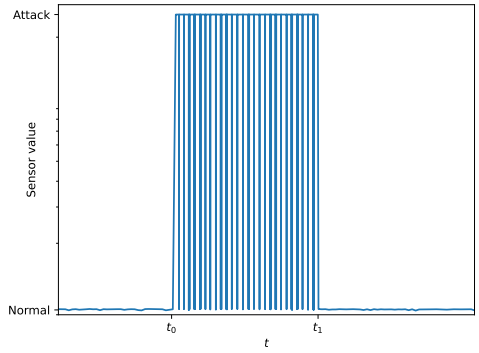


FIGURE 6.5: Values of a light sensor over time received by the dashboard. At $t = t_0$, a compromised ECU initiates an adaptive spoofing attack, forcing the sensor value to be abnormally high. At $t = t_1$, *CANTXSec* is activated in prevention mode, and the attack is defeated.

6.7 Related Works

IDS on CAN bus. Plenty of works in the literature propose to detect attacks in the CAN bus by analyzing frame content employing ML and Deep Learning (DL) models [315, 332]. Different models have been employed, such as Random Forest [257], CNN [107], GAN [315] and LSTM [332]. GIDS [315] is an IDS proposed by Seo et al. [315] together with a dataset including DoS, fuzzing (i.e., spoofing of packets with random content and ID), and modification. A GAN is trained with benign data and used to detect attacks with 98% of accuracy. The same dataset is also used by Song et al. [332] to test an IDS based on different DL models achieving higher accuracy.

Other approaches have been discussed in the literature as well. Five different features have been used by Xiu et al. [189] to detect spoofing attacks, including the ID, time interval between packets, and three features related to the frame content (correlation, changing amplitude, and value range). Other works [381, 309] try to fingerprint ECUs based on the voltage level imposed on the bus. However, these mechanisms exploit an average over different measurements during the transmission of the same packet, making them unreliable when dealing with SBAs. Moreover, they have been proven vulnerable to attacks [41, 301].

Time intervals are another feature employed in the literature to detect attacks in the CAN bus [331, 189, 346]. However, these techniques have some limitations and cannot provide protection against the injection of single bits during the transmission of legitimate frames (i.e., SBAs).

IPS on CAN bus. Because of the bus nature of the CAN protocol, IPSs are not frequent in the literature. The first to theorize the utilization of the error handling mechanism to prevent the reception of malicious messages have been Matsumoto et al. [250]. They theorize a modification of ECUs to include a flag to be set when transmitting so as to enable a security device to send an error if someone else is transmitting in the same period. However, a software and hardware modification of each ECU is requested, together with the wiring cost of flags and the extra device. It is suitable for new networks and devices but not for legacy systems. Moreover, the authors did not tackle the implementation challenge.

De Araujo-Filho et al. [102] propose an IPS based on an unsupervised Isolation Forest, which requires some unlabelled training data without attacks in it. Moreover, through the injection of error frames, malicious packets can be stopped. However, only fuzzing and modification attacks are discussed without including any SBA.

Another approach has been proposed with ZBCan [316], a security solution able to authenticate messages exploiting the intra-packet time and stop attacks using a SBA. However, the implementation requires a new device and a modification of ECUs' software to include authentication management. Moreover, SBAs are discussed only as a countermeasure, and the system cannot detect or prevent them.

6.8 Discussion

In this section, we discuss some aspects of our work. We compare *CANTXSec* with other approaches in the literature in Section 6.8.1. Then, we investigate the ECUs coverage requirement in Section 6.8.2 and delays in detecting attacks in Section 6.8.3. Finally, Section 6.8.4 discusses some limitations of the current work.

6.8.1 Comparison with other works

Despite the huge amount of IDS in the literature, only a few papers describe an effective prevention system that can block attacks in the CAN bus [102, 316], as described in Section 6.7. Table 6.4 summarizes the capabilities of our work with respect to other relevant papers in the literature. Detection of FIAs is usually one of the main targets of papers and is thus always satisfied. Detection of SBAs is, instead, a completely new topic, and, to the best of our knowledge, *CANTXSec* is the first work in the literature addressing this issue.

Moreover, our solution is also able to stop attacks, which is instead a topic discussed in just a couple of other works. Compared to our solution, ZBCan [316] requires software modification of each ECU, which may be cumbersome when dealing with proprietary software in the automotive scenario. On the other side, the hardware modification required by *CANTXSec* can be applied on every ECU by inspecting the PCB for the CANTX line, which is easier than dealing with integrity protection and digital signatures applied in all modern ECU firmware [195, 214]. Moreover, ZBCan [316] can only detect FIA, while SBAs will pass undetected. While both ZBCan and *CANTXSec* are based on almost deterministic solutions, the approach discussed by De Araujo-Filho et al. [102] includes training of ML models, resulting in increased setup time and need for computation resources. Finally, it is worth mentioning the development of secure CAN transceivers [272] providing certain security measures such as spoofing and flooding protection. However, they cannot protect against threats that are transparent for the transceiver, such as SBAs, which are still possible with such devices.

6.8.2 ECUs coverage

The most significant requirement of *CANTXSec* is the need to wire the ECUs to the officer, which is somehow a limiting feature in a CPS like an automobile, while it can be more easily applicable in ICSs or ships. However, it is worth noting that not all the ECUs in a vehicle have the same risk model, and *CANTXSec* can be easily implemented and deployed to cover only the most critical ECUs. The number of ECUs in a vehicle is quite variable. Cheap automobiles are equipped with only a bunch of them, while luxury cars may have up to 150 ECUs [160]. Accessing information

TABLE 6.4: Our solution compared to other IDSs and IPSs in the literature. ∼ indicates a lightweight (not ML-based) training process.

	Detection		**Prevention**		**Training**
	FIA	**SBA**	**FIA**	**SBA**	
GIDS [315]	✓	✗	✗	✗	✓
Song et al. [332]	✓	✗	✗	✗	✓
Xu et al. [381]	✓	✗	✗	✗	∼
VALID [309]	✓	✗	✗	✗	∼
Jin et al. [189]	✓	✗	✗	✗	✓
De Araujo-Filho et al. [102]	✓	✗	✓	✗	✓
ZBCan [316]	✓	✗	✓	✗	✗
CANTXSec	✔	✔	✔	✘	✘

about the number of ECUs in a vehicle and the related messages is tough because of the closed source of the automotive environment. According to openDBC [85], a collection of reverse-engineered database of CAN messages for different vehicles, they have a mean of 35 ECUs, even though the number is quite variable, and databases are not always complete. Usually, each ECU sends various messages through different frame IDs, indicating the content of the message and its priority. Since, in the worst-case scenario, attacks target the entire ECU (e.g., bus-off attacks will stop the ECU from sending any message), in our analysis, we considered only the lowest ID for each ECU.

Even though the repository [85] does not offer a complete collection of vehicle messages and ECUs, we analyzed the most complete dataset to estimate the number of ECUs *CANTXSec* should be connected to in different scenarios. Since the ID indicates the priority of the frame, we suppose that lower IDs correspond to safety-critical messages, while high IDs are reserved for infotainment and other non-critical applications. Therefore, to roughly estimate the number of safety-critical ECU, we sort them by IDs and threshold them with the first clearly not-critical ECU. A utility automobile such as a Hyundai Kia contains 43 ECUs. Out of them, we identify the 12 ECUs with lower IDs to be considered safety-critical, using as a threshold the ID of the parking assistance ECU, which can be considered non-critical. More expensive vehicles, such as BMW E9X, contain slightly more ECUs (50). We identify 14 of them as safety-critical, using as a threshold the IDs adaptive front lighting system.

Covering more or less ECUs with *CANTXSec* impacts the attacks that can detected and, possibly, prevented. The detection of FIAs is quite resilient to narrow coverage of ECUs by *CANTXSec*, as shown in Table 6.5a. In fact, spoofing is detected and prevented even if the compromised ECU is not connected to the security system. This happens because, during spoofing, the legitimate ECU's CANTX will be idle while the officer detects its ID on the bus. It is worth noticing that the malicious ECU could be a not monitored ECU or a completely new device connected from the attacker to the bus [356]. Therefore, for normal use cases, covering safety-critical ECUs is enough to secure the system against common FIAs.

When dealing with SBAs, the situation is more complicated, as shown in Table 6.5b. In this class of attacks, identifying the transmitter of the malicious bits is essential to detect attacks. All the attacks originated from ECUs connected to *CANTXSec* are identified since the officer will notice a dominant value during the transmission of a frame not linked to the transmitting ECU. However, this does not apply to unmonitored ECUs since they are outside the control of the officer. This implies that to have complete detection of SBAs, all the ECUs must be connected to

TABLE 6.5: Detectability of a spoofing attack (Table 6.5a) and a SBA (Table 6.5b) for different configurations of monitored or unmonitored ECUs. $\bar{X}$ means the ECU sending ID=X is monitored by *CANTXSec*.

		Spoofed ID			
		A	*B*	*C*	*D*
Origin ECU	$\bar{A}$	–	✓	✓	✓
	$\bar{B}$	✓	–	✓	✓
	C	✓	✓	–	✗
	D	✓	✓	✗	–

(A) Spoofing attack.

		Victim frame ID			
		A	*B*	*C*	*D*
Attacker ECU	$\bar{A}$	–	✓	✓	✓
	$\bar{B}$	✓	–	✓	✓
	C	✗	✗	–	✗
	D	✗	✗	✗	–

(B) SBA.

CANTXSec.

Based on the simple risk assessment we propose, or others in the literature [268], each vehicle owner can decide how many ECUs should be covered by *CANTXSec*. Just by connecting less than 30% of ECUs to the officer, it is possible to protect safety-critical ECUs from FIAs from both remote and physical attackers. While this could be the most widely adopted scenario, high-risk usages may require that all the ECUs should be covered with *CANTXSec* to ensure full detection of both FIAs and SBAs.

6.8.3 Attack Detection Delay

The fastest way for a MCU to monitor several lines is through the usage of interrupts. Therefore, we employed changing edge interrupts to check every CANTX line our officer is monitoring. We set the interrupts to fire every time a rising or falling edge is observed on the line. This is perfect for detecting Error #2, which is generated by an ECU passing from idle (recessive value) to a dominant value on the bus. However, it can also be used to identify if the correct ECU is transmitting some data. It can be done by monitoring the first bits after the arbitration ID, looking for a bit of change. Theoretically, in the space of binary messages, there exist cases where all the bits in the frames are the same. Practically, this is not possible in a CAN bus because of bit stuffing that forces a different bit after five consecutive equal bits.

Even if we understand that Error #1 will eventually trigger, it is interesting to know the mean delay for having such an error raised. The theoretical limit is given by the definition of bit stuffing: $5 + 1$ bit times. However, we investigated the probability of waiting that time before detecting Error #1 by computing the time to wait for an edge after the completion of the arbitration ID. We extracted data from a dataset of 10 minutes of CAN traffic to compute the number of bits to wait for an Error #1 to be raised. Figure 6.6 shows the probability of waiting up to a certain amount of bit times to have an alert. As shown, after 4 bits time, almost all the frames have transmitted a dominant value. However, 6 bits is the maximum number of bits to wait to have a 0, as mandated by the bit stuffing mechanism. This guarantees an upper bound on the time to wait for the detection, which is anyway negligible, and ensures detection and prevention of attacks before the end of the frame.

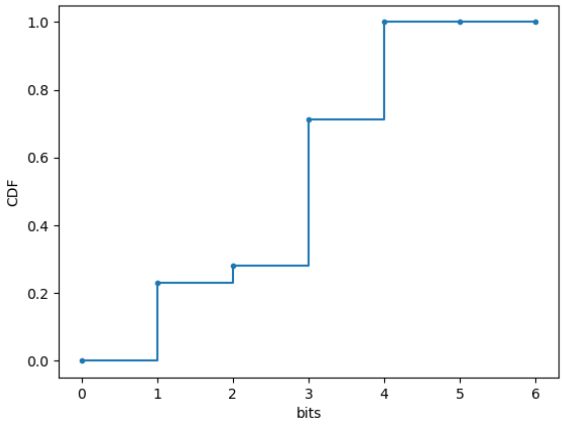


FIGURE 6.6: CDF of maximum bit times to wait for an Error #1 to be raised.

6.8.4 Limitations

This work introduced for the first time a novel way to reduce the long-discussed issue of false positives in the field of attack detection for CPSs and successfully develop the first strategy to eradicate the problem. However, some limitations should be addressed in future works.

First, the requirement to cover all the ECUs with *CANTXSec* to comprehensively detect all the discussed attacks may restrain some producers who could see production prices and vehicle weight rise. However, we offer the reader an analysis to consciously decide how to deploy the system based on the personal risk assessment. Moreover, the most common and straightforward attacks (i.e., FIAs) are detected and prevented with 100% accuracy just by connecting to *CANTXSec* the most safety-critical ECUs, relaxing the full wired requirement for the standard user while providing the possibility to increase security for high-risk cases.

Another weakness is represented by the partial capability of stopping attacks, which is limited to FIAs. SBAs represent a huge challenge regarding the prevention since they may be effective after a single injected bit, and therefore, the time frame to act and stop the attack is virtually zero. Mitigation strategies could include firmware and architecture hardening to prevent single-bit access to the bus. Moreover, forensics tools may be deployed to secure the system after the attack has happened.

Finally, this research focused on network-level threats, leaving aside data-level attacks such as modification attacks. *CANTXSec* correlates IDs on the bus with ECUs activations to spot attacks without looking at the actual data transmitted. Due to this architecture, modification attacks are not possible to be detected. Further research should be conducted to port the proposed deterministic approach to the data-level.

6.9 Conclusions

In this work, we bridged the gap in the detection and prevention of attacks in the CAN bus by proposing *CANTXSec*. Our solution is 1) deterministic, 2) covers advanced attacks never discussed in the literature before, and 3) does not require any software modification of ECUs and is thus easier to implement. To test and analyze its characteristics, we introduced a novel categorization of attacks in the CAN

bus based on the access an attacker needs to carry them out. In fact, we introduced FIAs requiring typical frame-level access and SBAs employing more sophisticated bit-level access. Through the usage of a physical testbed we developed, we demonstrated that *CANTXSec* achieves 100% accuracy in detecting both FIAs and SBAs. Moreover, we demonstrate how our solution is even able to prevent FIAs with 100% accuracy.

Future works include further tests through the deployment of *CANTXSec* in a real vehicle and augmenting the number of connected devices. Moreover, this work opens a new research direction aiming at the prevention of the newly introduced category of threats we called SBAs, which is a topic not addressed in the literature up to now.

7 When Authentication Is Not Enough: On the Security of Behavioral-Based Driver Authentication Systems

The traditional means of accessing a vehicle, whether through a physical or wireless car key, has long been the primary authentication method, linking the driver to the vehicle. However, evolving technology and security challenges have exposed vulnerabilities in relying solely on car keys [138, 126]. Despite advancements like stronger cryptographic protocols [138], recent incidents, including direct connections to the CAN bus, have demonstrated exploitable weaknesses [255]. Alternative methods, such as smart keys, remain susceptible to relay attacks [126] and distance reduction attacks [225]. Biometric-based authentication, an emerging trend [344], faces risks from the rapid progress of AI and generative models, capable of generating realistic data that can deceive even sophisticated systems [203].

In response to these enduring threats, researchers propose behavior-based driver authentication [194, 292, 361, 382, 8, 2] and identification [1, 23, 72, 133, 119, 159, 243] systems augmenting traditional methods. These systems collect and scrutinize data (e.g., collected from the CAN bus) by employing simple ML algorithms, such as SVM and RF, and more complex DL employing neural networks to verify the legitimacy of the driver by obtaining the driver's identity (i.e., identification) or detecting whether the driver is among those authorized (i.e., authentication). While a vehicle is in motion, it can distinguish drivers based on their driving behavior by analyzing unique driving patterns and characteristics. These authenticators constitute a paradigm shift that promises enhanced security and opens doors to applications beyond theft prevention. Indeed, they find applications in emerging fields like Usage-Based Insurance (UBI) policies, where individuals are rated and charged based on their driving behavior, and in identifying impaired drivers for safety reasons [168, 99, 251].

However, despite the growing interest in the research community, current proposals do not consider security-related issues in practical implementations. Furthermore, the current literature has neglected possible adversarial attacks on these systems, which can jeopardize their functionality. Given the lack of a formalized system and threat model, the effects of ML vulnerabilities in this scenario are unclear. Such attacks present challenges related to when and how to inject adversarial samples, which have never been considered in this setting at the time of writing. In particular, we identified the following gaps limiting practical implementations, constituting our research questions.

RQ7.1 How should the authenticator be deployed so as not to be easily bypassed by an attacker?

RQ7.2 How can we implement privacy-preserving model training and deployment?

RQ7.3 How can we eradicate the problem of false positives and thus make the system usable in commercial devices?

RQ7.4 Which solutions can help in securing these systems from adversaries?

RQ7.5 Could these systems replace traditional authentication mechanisms?

These practical considerations significantly impact the authentication system's security and safety, as different configurations may allow the attacker to disconnect the device and bypass authentication or cause high false positive rates, making the driving experience unpleasant, if not impossible.

Contributions. This chapter aims to reduce the gap between research and practice in behavioral-based driver authentication and identification systems. To this aim, we address our five identified implementation gaps and propose the first security-aware system model for behavioral-based driver authentication and identification systems. We are also the first to develop attacks against behavioral-based driver authentication systems and show their applicability in real-world scenarios. By evaluating our models and attacks on a state-of-the-art dataset of real-world driving data, we show how the attacker can impersonate the legitimate driver by modifying a reduced set of non-safety-critical CAN bus messages. We compare our system's contributions with the state-of-the-art in Table 7.1.

Our contributions can be summarized as follows.

- We propose the first security-aware system model for behavioral-based driver identification and authentication. Based on our design, we develop two new lightweight behavior-based driver authentication and identification systems whose requirements are compatible with commercial vehicle networks. Our two proposed architectures, i.e., a RF and a single-layer Gated Recurrent Unit (GRU), are designed to be efficient in a realistic system model and outclass the state-of-the-art, achieving an accuracy of up to 0.999.
- We introduce **SMARTCAN** and **GANCAN**, the first attacks against behavioral-based driver authentication and identification systems. Our attacks use evasion techniques to avoid detection and only inject minimal amounts of data to preserve driving functions while achieving success rates of up to 1.000.
- We evaluate our system and our attacks on real driving data. Furthermore, we test the accuracy and resilience of other state-of-the-art models and compare the results. Our evaluation highlights specific vulnerabilities in the systems' implementations, and we provide a list of security requirements for safe and secure implementation of behavioral-based driver authentication systems, aiding practitioners in real-world deployment.
- We make the code of our systems, attacks, and the dataset open source[1].

Organization. The rest of the chapter is organized as follows. We give necessary background information in Section 7.1 and mention challenges and limitations of related works in Section 7.2. We propose the system and threat models in Section 7.3 and Section 7.4. We describe our suite of evasion attacks in Section 7.5. Then, we

[1] https://anonymous.4open.science/r/WAINE-1518

TABLE 7.1: Comparison of our systems' contributions with the state-of-the-art. The difference between authentication and identification is detailed in Section 7.6.

Model	**Auth.**[1]	**Ident.**[2]	**System Model**[3]	**Threat Model**[4]	**Security Study**[5]
[382] - [2]	●				
[1] - [243], [24] - [288]		●			
[194], [292], [56]	●	●			
[361]	●		◐		
[104] - [274]		●	◐		
Ours	●	●	●	●	●

[1] Authentication. [2] Identification.

[3] If the paper considers issues of practical implementation.

[4] If the paper considers external threats.

[5] If the paper considers issues of possible attacks.

evaluate our attacks, authentication, and identification systems in Section 7.6. Finally, we report our takeaways in Section 7.7, while concluding our work in Section 7.8.

7.1 Background

The CAN bus has become a fundamental component of modern vehicles, serving as the communication backbone between a vehicle's ECUs. The CAN bus is a robust and widely adopted communication protocol for real-time, high-integrity data transmission [181]. At its core, the CAN bus employs a message-based communication model, facilitating data exchange among ECUs distributed throughout a vehicle. Unlike traditional point-to-point communication, the CAN bus employs a multi-master, multi-listener architecture. This means that multiple ECUs can send and receive messages simultaneously on the bus, providing a highly efficient and fault-tolerant communication medium. Each CAN message contains an identifier (ID), data, and a CRC for error detection. Messages are prioritized based on their identifier, allowing for the establishment of message priority levels within the network. Being a bus, the CAN protocol utilizes a "broadcast" communication approach, where all connected ECUs receive every message. Each ECU filters messages based on identifiers to process relevant data selectively, simplifying network design but requiring unique identifiers to avoid collisions. Messages on the CAN bus follow a time-triggered schedule, ensuring precise delivery intervals for critical information. This deterministic behavior is vital for real-time vehicle applications like engine control, braking systems, and behavior-based driver authentication systems. However, it must be noted that the CAN bus lacks native security measures. The broadcast nature and absence of encryption make it relatively simple for a malicious ECU to intercept and read all transmitted messages [47]. This security gap allows unauthorized vehicle control and creates potential entry points for attackers to compromise ECUs or the entire vehicle. Due to the broadcast structure and the high amount of information transmitted through it, the CAN bus is often employed to investigate novel functions, such as to study the driver's behavior.

7.2 Related Works

A vast amount of literature considers the problem of identifying the driver behind the wheel based on behavioral data. This could be useful in identifying and blocking vehicle thefts but can also be used to offer more personalized experiences to drivers. For example, it can be used to set some vehicle's parameters based on user preferences [116] or to collect data to optimize and enforce driver-based insurance by calculating risk factors on the fly [99, 168]. However, except for some of the works that try to extract the user's driving style (e.g., aggressive, mild, or gentle) [99, 132, 131], the main target is identifying the driver or their legitimacy.

Data. Most approaches extract CAN bus data via the OBD-II port, which can provide various sensors and actuator readings to characterize the vehicle state. Even if many researchers do not provide access to the dataset they used, making it complicated for the community to reproduce and improve the results, other works employ publicly available datasets. The most used one comes from the *OCSLab* [249, 217] and comprises 54 sensors reading from the vehicle bus, including ten drivers. Other less considered datasets contain different data types, such as stability [8] and Global Positioning System (GPS) data [288]. Few works also employ physiological data [312], even if its applicability in a real scenario is challenging. Our work will use the OCSLab dataset to make our results easily comparable with others in the literature.

Models. Almost all the works attempting to identify the driver employ classic ML or advanced DL algorithms, as summarized in Table 7.2.The paper by Erzin et al. [118] is one of the first to discuss how features such as pressure on pedals, vehicle speed, engine speed, and steering angle can be combined to identify a driver. They state that authentication should be done before the vehicle moves. However, these data can be solely employed to verify the driver's state (sleepy, active, etc.). In [8], the authorized driver has a specifically-trained ML model containing their profile information. They perform binary classification, showing that they can authenticate drivers and extract features such as the driver's gender. The authors of [243] use CAN bus data of an EV, mainly focusing on pedal operation patterns and GPS traces of different drivers driving on the same route. The authors use such data to implement an SVM and Universal Background Model (UBM)-based ML model to authenticate users. In [382], Xun et al. propose driver fingerprinting to authenticate a user in real-time using CAN bus data. The authors use DL methods such as CNN combined with Support Vector Domain Description (SVDD) to detect illegal drivers. Other ML models have been employed with discrete success, such as kNN [217, 230, 159, 119, 288], RF [8, 159, 119, 288], and Gaussian Mixture Model (GMM) [259]. A better successes have been observed with DL models, especially employing LSTM [23, 292, 2, 116] or RNN [116, 133, 394, 144]. Other models have been tried as well, such as AdaBoost [187], Autoencoder [72], and a Generative Adversarial Network (GAN) [274].

Parameters. Together with the models, the number of features to be employed has also been analyzed in the literature. Marchegiani et al. [243] conducted some tests to assess the feasibility of identifying a driver with one feature only (e.g., acceleration or brake) using SVM. Rahim et al. [288] employ GPS data only, training ML models with solely three features (orientation change, stable speed, total acceleration), but still reaching up to 90% accuracy in detecting drivers. Thanks to their complexity, DL models are usually more suited to manage a higher number of features. For example, Chen et al. [72] employed 51 features available in the OSCLab

TABLE 7.2: Summary on the different models employed by papers in the literature. The paper's target can be to profile the driving *style* of the driver or to identify the *driver*.

Goal	Type	Model	References
Style	Clustering	Clustering	[131]
	DL	CNN, LSTM	[99]
	Statistical	Time-domain	[132]
Driver	DL	AdaBoost	[187]
		Autoencoder	[72]
		CNN	[382, 394]
		DeepRCN	[1]
		GAN	[274, 361]
		LSTM	[23, 292, 2, 116]
		MLP	[104]
		RNN	[116, 133, 394, 144]
		SVDD	[382]
	ML	GMM	[259]
		K-means	[194]
		KNN	[217, 230, 159, 119, 288]
		RF	[8, 159, 119, 288]
		SVM	[159, 119, 288, 243, 56, 24]

dataset to train an Autoencoder, while Ravi et al. [292] employed 40 features in a LSTM.

Preprocessing. Several preprocessing techniques have been employed to use the data to the fullest. Miyajima et al. [259] employed cepstral spectral features, generally used for speech and speaker recognition, associated with a GMM model. Fugiglando et al. [132] employed a time-domain analysis of the features to extract the maximum entropy to characterize the drivers. DL and LSTM usually require less preprocessing effort since the model can automatically capture the peculiar features of the data. Most works in the literature divide the data with a sliding window approach to be better analyzed by the models. The window size may vary from 12 seconds [187] to 300 [382]. Moreover, to enhance the number of samples, some papers maintain an overlap between adjacent windows [1, 194, 217].

Implementation. Another factor impacting these systems is their implementation in real-world scenarios. Most of these works do not specify which system component is responsible for driver authentication. Although simple ML algorithms can be executed on cars, DL networks based on complex and deep structures might require prohibitive costs for the execution on a vehicle and are more likely to be implemented on a dedicated and resourceful server, at least for the training part. El Mekki et al. [116] implemented a proof-of-concept authenticator with Automotive Grade Linux [286], showing its easy applicability to UI personalization. However, no one ever implemented or discussed in detail how a physical anti-theft device based on driver behavior can be developed. Since running and, especially, training ML and DL models can be expensive, some literature works delegated at least one of these tasks to a central server in the cloud. For instance, Kwak et al. [217] propose implementing the anti-theft module in a remote server accessible via the internet. The car sends via wireless communication driver behavior CAN bus data

to the server, extracting specific features and feeding them to a previously trained ML model. Based on the same system model, Ezzini et al. [119] proposed a driver authentication scheme implemented on an internet-connected dedicated server that extracts predefined features from CAN bus data to authenticate the driver. In other works [116, 23, 134, 194], a remote server has a part in the authentication process. All these works, however, do not mention security on the channel between the vehicle and the server, ignoring the threat imposed by sharing users' sensitive plaintext data over a wireless channel, which may expose them to possible thefts by malicious users. Moreover, privacy concerns arising from sharing driving data with third parties should be considered.

Adversarial Attacks. There is growing attention in the literature towards adversarial attacks targeting AI-enabled authentication systems. Tan et al. [246] outlined several strategies a potential attacker could use to create synthetic adversarial samples to compromise Remote User Authentication systems. These strategies are classified into imitation-based, surrogate-based, or statistics-based methods. Marrone et al. [248] conducted a study to assess the impact of adversarial perturbations on Fingerprint-based Authentication Systems (FAS), aiming to pave the way for developing an adversarial presentation attack. The findings reveal the potential for adversarial perturbations to deceive both the FAS liveness detector and the authentication system. Solano et al. [330] investigated an intuitive attack method for mouse-based behavioral biometrics and evaluated its efficacy against adversarial machine learning attacks. The research demonstrated that attacks utilizing domain knowledge exhibit greater transferability across different ML techniques and pose more significant challenges for defense mechanisms.

7.3 Practical System Model

In this section, we detail the system model and its implications. We first discuss the physical implementation of the authenticator device[2] in Section 7.3.1, We then explain the data collection technique in Section 7.3.2. Finally, we define training and testing procedures in Section 7.3.3.

7.3.1 Physical Implementation

A common approach in the literature is to collect CAN bus data from a device connected to the OBD-II port [274, 2, 194]. By sending OBD requests, we can retrieve the data by simply decoding responses. However, this setting is problematic in a real-world driver authentication system. Firstly, being connected to an easily accessible port, the authenticator would be an external device that can be easily removed by a malicious party intending to steal the vehicle. For an attacker inside the car, this would be as easy as disconnecting a USB stick from the infotainment system, as the proposals in the literature do not discuss the use of anti-tampering solutions. Secondly, the models in the literature and the basic connection to the OBD port envision messages exchanged between the gateway and the authenticator to be unencrypted and without integrity checks. This allows attackers to tamper with the channel and inject malicious messages easily. For these reasons, we argue that *the best and most secure (by default) implementation of the authentication system is on the CAN bus, with*

[2]From here on, we will refer to "authentication system" and "identification system" interchangeably. From Section 7.6 we differentiate the tasks and treat them separately.

the authenticator acting as an ECU, as shown in Figure 7.1. By implementing the authenticator directly on the CAN bus, it can access every message exchanged in the network and thus sensor data from any ECU. This strategic placement strengthens the authentication process and makes tampering attempts challenging and inherently risky. Indeed, interference with the CAN bus could potentially jeopardize the car's functionality and put drivers at risk. Furthermore, by positioning the authenticator ECU in less accessible areas of the vehicle, we ensure no tampering can occur since compromising it would require a considerable amount of time and substantial expertise on the vehicle network.

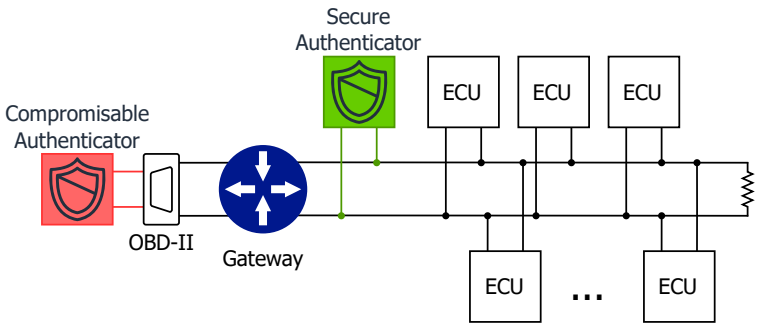


FIGURE 7.1: Examples of compromisable and secure implementations of the authenticator.

7.3.2 Data Collection

While the CAN bus is a well-known standard in the industry, each ECU and vehicle Original Equipment Manufacturer (OEM) encodes data differently. Moreover, manufacturers do not share the decoding procedure openly, so researchers have to reverse engineer each CAN message manually to create DBC files, which encode and decode those messages [85]. While message decoding can be challenging for researchers or third parties tampering with the car [79], the OEM can access all the decoding necessary to access the data. In our system model implementation, we assume the authenticator already has the information to decode the messages exchanged in the CAN bus and knows which IDs are associated to which ECU. We believe this is a fair assumption since in real-world implementation of the authenticator, the system should be manufactured in collaboration with the vehicle brand and the ECU OEM. Thus, data is collected by retrieving messages directly from the CAN bus. The authenticator filters only the messages with IDs containing relevant information and decodes them to extract the features needed for training and testing. Since its primary purpose is to provide continuous authentication while driving, the system periodically collects each of the features it uses. In the case of DL models leveraging causality between samples, data is aggregated in time windows and then in batches. An overview of this process is shown in Figure 7.2.

In our implementation, we extract 46 features each second. For our DL architecture, we aggregate data in time windows of 16 seconds with a step size of 8. Time windows are then aggregated in batches with a size of 4. Thus, an identification or authentication prediction is generated every 40 seconds. For our classical ML architecture, instead, a prediction is generated for each collected sample (i.e., each second). The advantages and disadvantages of each architecture are discussed in Section 7.6.3.

7.3.3 Authentication

Following a new vehicle purchase or commence of a new driver training procedure, the authenticator is in training mode and collects batches of data to feed to the model. To avoid sharing sensitive users' data with external entities, we assume that data collection and model training are performed in the driver's car. For the models we developed, these tasks can be accomplished in a reasonable time by leveraging low-resource devices, as we show in Section 7.6.3, avoiding the need for powerful external computing facilities. Once trained, the model is saved and deployed within the vehicle for real-time classification.

During the operational phase (testing), the authenticator monitors the driver's behavior and periodically produces an authentication response. If the response is positive, the driver appears legitimate, and no action is taken. Instead, if the response is negative, the system detects different driving behaviors, which can alert the vehicle owner. The manufacturer can choose the action performed in case of unauthorized driving. However, since ML models rarely produce perfect results (i.e., an accuracy value of precisely 1 in the test set), remotely stopping the vehicle may incur road safety issues. Thus, we argue *the best action would be to notify the vehicle owner, who can track the vehicle remotely and contact the authorities*.

To further decrease false positive alerts, instead of immediately sending an alert to the vehicle owner once a batch of unauthorized data has been found, we propose sending alerts based on multiple consecutive decisions. The rationale behind this is that, given a high enough authentication accuracy rate, it is improbable to have multiple consecutive false positive events (i.e., detecting multiple batches of unauthorized driving behavior while the driver is legitimate). We provide an analysis of this rate, contextualized to our system's accuracy, in Section 7.6.3.

7.4 Realistic Threat Model

Potential attackers can physically access the vehicle's CAN bus network leveraging its lack of security measures, thus gaining persistent access to the car's network. This is effective in injecting packets into a vehicle's bus and has already been used for stealing cars in real-world settings [356]. However, in our proposed implementation, the attacker cannot physically tamper with the authenticator device since it is positioned in more internal parts of the network, such as within the transmission tunnel or in a sealed compartment behind the engine bay firewall. Within this context, the adversary deploys a covert malicious device directly into the vehicle's CAN bus network in a place easily reachable such as external headlights [356]. This specialized device features GPS-enabled functionality, sufficient memory, and computational power to execute the perturbation function. This function receives data from the CAN bus, applies perturbations, and reintroduces the modified data into the CAN bus. We also assume the attacker knows the DBC file of the vehicle they

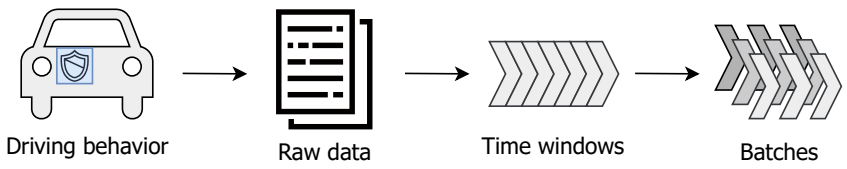


FIGURE 7.2: Overview of the data collection process for DL models.

are attacking and thus can encode and decode data at will. Indeed, the reconstruction of the DBC file can be performed manually or with automatic methods [57, 79]. Moreover, many reversed DBC files are available online [85].

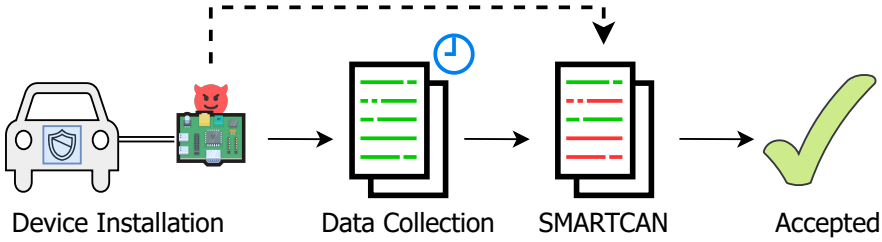
The attacker also knows the features the model uses for authentication since they have been widely used in the literature. We provide an overview of such features in Section 7.6.1. However, it is worth noting that the attacker must exercise caution, as injecting data with all features may not be feasible. The injection of data into the CAN bus is associated with inherent safety risks, given the potential impact on vehicle operation and driver safety. To address this concern, we consciously selected specific features to enable controlled data injection while minimizing safety hazards. Of the 46 features provided in our dataset, we identify 22 of them that can be manipulated without affecting the vehicle behavior (e.g., engine coolant temperature, intake air pressure), while altering the remaining 24 could potentially pose a risk for the driver (e.g., current gear, throttle position signal). However, since the interaction between these injected features might still create minor discomforts when driving a vehicle, it is in the attacker's best interest to try and keep the number of modified features at a minimum. Table 7.3 shows examples of modifiable, borderline, and non-modifiable features.

TABLE 7.3: Distribution of features into safety classes.

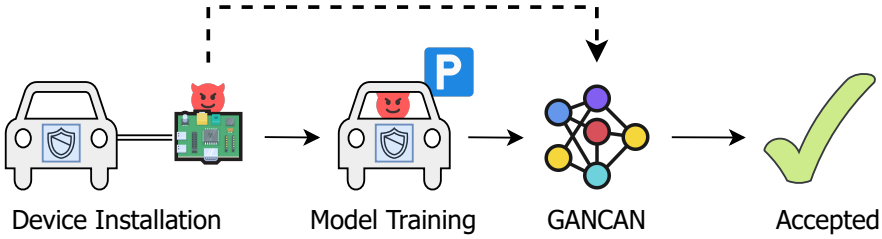
Importance	**#**	**Examples**
Modifiable	22	Engine coolant temperature Intake air pressure Calculated road gradient
Borderline	15	Engine speed Acceleration speed - Lateral Torque converter speed
Not-modifiable	9	Throttle position signal Current gear Master cylinder pressure

7.5 Attacks

We now discuss the implementation of our attacks. Based on the attacker's knowledge, we design two different methodologies. Section 7.5.1 describes the situation where the attacker has access to data originated from a legitimate user's driving (SMARTCAN). Section 7.5.2 details the attacker's strategy with access to the model (GANCAN). While not adhering to the traditional GAN training framework, our generator models (when used) are optimized to bypass a fixed discriminator (i.e., the authentication system) rather than incrementally training both components. Thus, the deployed generative networks detailed in this section leverage the discriminator feedback (i.e., the authentication system output) for optimization and loss computation. The attacks are summarized in Figure 7.3.



(A) Overview of our SMARTCAN attack. The second step is skipped if the attacker already has access to the data.



(B) Overview of our GANCAN attack. The second step is performed offline and in advance if the attacker already has access to the model.

FIGURE 7.3: Schema of our SMARTCAN and GANCAN attacks.

7.5.1 SMARTCAN

For this attack, summarized in Figure 7.3a, we assume the attacker can access the victim's data (i.e., legitimate driving behavior) but does not know the authenticator model implementation. A way to access it involves eavesdropping on an insecure channel where data are transmitted to third parties or possible leakages from companies collecting the behavioral data [74]. This might be the case for insurance companies when using UBI or rental companies monitoring and storing the driving behavior of their clients [168]. However, if the attacker cannot access the data, they can physically deploy a malicious device to be connected to the CAN bus of the target vehicle. The device sniffs the traffic and extracts the driving behavior, which is then sent to the attacker. Once enough data is collected, the attacker can launch the attack employing the same device.

Due to the feature safety reasons detailed in Section 7.4, the attacker cannot simply replay the legitimate traffic they have access to. Thus, a *smart*-replay attack must be deployed. With this attack, the attacker can replay the legitimate traffic using only the features labeled as *modifiable*. Instead, the data for all the other features (i.e., *borderline* and *non-modifiable*) is taken from the attacker's driving behavior while stealing the car. In this way, the attacker can drive the car without any interference in their driving safety and still be authenticated by the anti-theft system.

7.5.2 GANCAN

In this attack, summarized in Figure 7.3b, we assume the attacker can access the authenticator model response but does not know the legitimate driving behavior. An attacker may get the trained model by dumping ECUs memory remotely [284] or physically [96] (e.g., a colluded workshop personnel dumping models to smuggle). If trained on the cloud, it may be leaked when transmitted back to the vehicle [145, 100] or, if used as a system-as-a-service, when asked to evaluate a vehicle's

state. Other options include obtaining data on a similar car by colluding with a car-sharing platform and employing such data to train a model mimicking the vehicle's authenticator [71].

Instead, if the attacker has no access to a copy of the model, they can employ the malicious device connected to the victim's vehicle to access the model during the idle state of a parked vehicle. Indeed, while the car is stationary, the authenticator is still operative but does not alert the owner in case of unauthorized access. This happens because CAN traffic would inherently be different for a parked vehicle with respect to a vehicle moving on the road. It may be possible in cases where the authenticator is developed by a third party and integrated into the car by the manufacturer, utilizing a separate ECU to generate actions in response to alerts. In such a case, the authenticator must avoid taking actions when data is invalid, such as when the vehicle is parked. With the drawback of being more time-consuming, the attacker can perform the attack in just one stage. Indeed, as we will see in the evaluation section (Section 7.6), the authentication results are broadcasted at fixed intervals.

The attacker can preemptively use the legitimate model to train the generator of a GAN. Indeed, it is possible to use the authenticator model as an oracle and train a neural network with its feedback, obtaining a model that can craft legitimate fake packets starting from noise. Our attack aims to create authentic features resembling genuine drivers without risking safety by overwriting only *modifiable* features, as detailed in Section 7.5.1. This allows the GAN to generate complete batches of data, of which the validity is required only for the *modifiable* features that are injected in the CAN bus. *Non-modifiable* and *borderline* features are instead naturally extracted from the attacker's driving behavior. Given the vast search space and potential challenges in convergence and local minima encountered, we employ Reinforcement Learning (RL) to optimize the learning procedure for our generator. Doing so enables the generator to explore the search space more efficiently and adapt its strategy based on feedback and rewards received during the learning process.

GANCAN packet generation and injection complies with CAN bus format specification [181]. Our generator model works on the feature space of CAN bus packets data field to avoid overwriting data that does not need to be modified in our attack (e.g., arbitration IDs). In particular, let us define as $\boldsymbol{x}$ the frame/batch of frames to be used as input of the model. Denoting as $f(\boldsymbol{x})$ the function with image $\mathcal{F}$ mapping $\boldsymbol{x}$ to its feature space, our generation process g is such that $g(f(\boldsymbol{x})) \rightarrow \mathcal{F}$. In other words, our generation process ensures that the data we generate complies with the data expected from that specific feature. In the context of the CAN bus, the function f represents the decoding procedure that converts the raw bits from the message data field to the numeric values of a feature used for authentication. Since CAN message encoding is linear and detailed in each vehicle DBC file, it is represented as f^{-1}. The original data field of the packet (i.e., the one generated by the attacker's driving) is overwritten with the data generated via our model. Then, the fields dependent on the frame content (e.g., CRC field) are updated. This ensures that our generated packets are *(i)* compliant with the CAN bus standard and *(ii)* in a reasonable range for the specific generated feature. An overview of the generation process is shown in Figure 7.4.

The RL parameters include the maximum episode length, the number of episodes, the learning rate α, and the discount factor γ. These parameters govern how the generator's latent input will be updated using RL. The training loop consists of episodes, where each episode starts with initializing the latent input and the episode reward. The generator generates a sample within each episode based on the

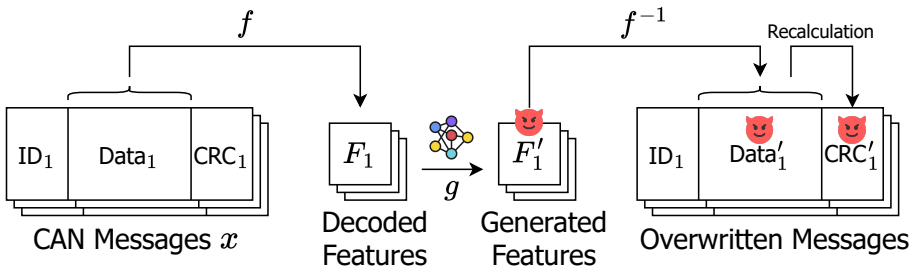


FIGURE 7.4: Malicious CAN message generation.

latent input. This generated sample is then evaluated by the surrogate model (i.e., the authentication system). The surrogate model's output makes predictions, and random target labels are created for comparison. The reward is calculated as the mean accuracy of the predictions matching the targets. The latent input is updated using RL by computing the temporal difference error (i.e., td_{error}) as the difference between the reward and the cumulative episode reward. The latent input is then updated by adding a scaled noise term to introduce randomness and exploration. The scaling factor is determined by α, td_{error}, and the γ factor raised to the power of the current step:

$$\alpha \cdot td_{error} \cdot \gamma^{step} \cdot latent\_input. \tag{7.1}$$

This update process helps the generator adjust its latent input based on the reward signal and explore different regions of the latent space. After the episode, the generator is updated using the final latent input. The surrogate model evaluates the generator's output, and a target label tensor is created for the loss calculation. The generator's loss is computed using the cross-entropy loss function, and backward propagation is performed to update the generator's parameters. The trained generator is returned once the specified number of episodes is reached. We use the Leaky ReLU activation function [380] to prevent the dying ReLU problem and to improve gradient flow, which in turn helps stabilize the training of our generator. The generator is composed of 5 feed-forward layers of 128, 256, 512, 64, and 46 units respectively.

7.6 Evaluation

We now delve into the evaluation of our driver authentication and identification systems. Our evaluation comprehends all scenarios detailed in the previous sections. For the evaluation, we will use the following.

- *True Positive (TP)*: legitimate driver correctly authenticated.
- *False Positive (FP)*: illicit driver mistakenly authenticated.
- *False Negative (FN)*: legitimate driver mistakenly repudiated.
- *True Negative (TN)*: illicit driver correctly repudiated.

As metrics, we use accuracy and F1-Score to evaluate the models. We found these two metrics an excellent solution to summarize the results on multiple drivers. Moreover, they allow an easy comparison with similar works in the literature. To evaluate attacks, instead, we choose the simple yet effective ASR.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN'}, \tag{7.2}$$

$$F1 = \frac{2TP}{2TP + FP + FN'}, \tag{7.3}$$

$$ASR = \frac{\text{\# malicious batches fooling the authenticator}}{\text{\# malicious batches sent}}. \tag{7.4}$$

In this section, we include results for the identification and authentication tasks for each evaluation. While the data and features utilized for both tasks remain consistent, fundamental conceptual distinctions set them apart.

- *Identification*: determining which specific driver is currently operating the vehicle. Thus, this task assigns each driver a unique identity within a set of known drivers. As a ML classification problem, this translates to a multiclass classification task where each label constitutes a driver.
- *Authentication*: verifying whether the current driver is legitimate. Thus, this task confirms the driver(s)' legitimacy based on behavioral characteristics. As a ML classification problem, this translated to a binary classification task.

We first give details on the dataset used and its processing (Section 7.6.1). We also provide details on our authentication and identification models and the ones reproduced from the literature and their architectures (Section 7.6.2). After that, we set the baseline for our model and attacks by testing our systems in unaltered settings where no attack has occurred (Section 7.6.3). We then focus on the specific attack scenarios and evaluate their effectiveness on our and other state-of-the-art systems (Section 7.6.4).

7.6.1 Dataset

To maintain consistency with prior research efforts, we utilize the widely adopted OCSLab dataset [217]. This is the most used dataset in the literature discussed in Section 7.2. Therefore, its usage allows us to assess the real-world feasibility of those works. This dataset includes 94,380 data points, equivalent to approximately 26 hours of driving data. This dataset consists of driving sessions from 10 different drivers, from which 54 distinct features have been extracted. These features are extracted from messages in the CAN bus, which are highly correlated with the driving behavior, e.g., acceleration speed, indication of brake switch, and fuel consumption. We employed Explainable Artificial Intelligence (XAI) techniques (i.e., SHapley Additive exPlanations (SHAP)) to extract the most important features used by our model. This analysis was done after the training and baseline evaluation of our system. These features, in decreasing order of importance, are the following.

1. `Long_Term_Fuel_Trim_Bank1` (modifiable)
2. `Torque_of_friction` (modifiable)
3. `Engine_coolant_temperature.1` (modifiable)
4. `Intake_air_pressure` (modifiable)
5. `Engine_soacking_time` (modifiable)
6. `Maximum_indicated_engine_torque` (modifiable)

7. `Activation_of_Air_compressor` (modifiable)
8. `Accelerator_Pedal_value` (borderline)
9. `Engine_coolant_temperature` (modifiable)
10. `Master_cylinder_pressure` (non-modifiable)
11. `Calculated_LOAD_value` (modifiable)
12. `Indication_of_brake_switch_ON/OFF` (non-modifiable)
13. `Acceleration_speed_-_Longitudinal` (non-modifiable)
14. `Short_Term_Fuel_trim_Bank1` (modifiable)
15. `Fuel_consumption` (modifiable)
16. `Engine_torque_after_correction` (modifiable)
17. `Engine_torque` (modifiable)
18. `Current_spark_timing` (modifiable)
19. `Torque_converted_turbine_speed_-_Unfiltered` (borderline)
20. `Torque_converter_speed` (borderline)

However, eight of those features consistently exhibited constant or zero values throughout the dataset. Consequently, we filtered out these eight features, producing a refined set of 46 features for our authentication system. Furthermore, we incorporate a windowing technique for our time series data. This entails partitioning the dataset into segments of a predefined size. Creating these windows involves iterating through the data with a step size equal to half of the window size. In our case, we opt for a window size of 16 seconds, slightly shorter than the average duration often used in existing literature [1, 2]. A smaller window size allows for more frequent classification, giving us finer insights into driving behavior patterns.

To ensure the quality and suitability of our dataset for authentication models, we perform essential preprocessing steps, including normalization. This process ensures uniform data quality and eliminates potential biases stemming from variations in feature magnitudes. We also identified a notable imbalance in the database, causing biased model behavior favoring majority classes over minority ones. To remedy this, we employed undersampling to rebalance the class distribution, concentrating on the majority classes. After this procedure, our dataset contains 74,200 data points. This adjustment enhances the model's ability to capture patterns in both majority and minority classes.

7.6.2 Implemented Models

We now delve into the details of implementing our models and the ones we reproduced from the literature. Unfortunately, most related works do not share their code openly, and only a fraction of them include enough details in the manuscript to replicate the results accurately. As such, we focused on the ones yielding the best claimed results. For our original implementations, we report only the best models among the ones tested in our experimentations. Indeed, we investigated other architectures, such as Decision Trees, SVMs, XGBoost, and Hidden Markov models,

which did not yield significant results in either baseline or attack evaluation. As such, we focus our evaluation on our best-performing models. The code for these models and our system is available in our GitHub repository. We summarize the parameters of the model we implemented in Table 7.4.

TABLE 7.4: Parameters of our models and the other systems we implemented for comparison.

Model	**Architecture**	**Layers x Neurons**
Girma et al. [144]	LSTM	1x160 + 1x200
Ravi et al. [292]	LSTM	1x460 + 1x495
Zhang et al. [394]	CNN+RNN+Attention	3x64 + 2x128 + 2x256
RNN (ours)	**GRU**	**1x220 (440 Bi-directional)**
RF (ours)	**RF**	**-**

Our RNN. To ensure practicality and efficiency, we designed our RNN driver authentication and identification model to be as straightforward as possible, minimizing computational demands. This approach aligns with our goal of creating a practical yet resource-efficient authentication system. The behavior-based authenticator employs a one-layer GRU with 220 neurons, allowing it to capture dependencies in both forward and backward directions in the time sequence. Following this, we have incorporated a dropout layer with a rate of 0.4. This dropout layer contributes to regularization, preventing overfitting. After dropout, the output of the GRU layer is passed through a Linear layer with 440 neurons, followed by an element-wise sigmoid activation function producing an output vector with a size equal to the number of considered drivers. The network takes sequential driving data as input, where each input represents a time window of observations. The input data format is three-dimensional, with batch size, window size, and number of features as dimensions.

We use the Cross-Entropy loss function, a common choice for multi-class classification tasks, to measure the dissimilarity between predicted and actual labels. We employ the Adam optimizer with a 1×10^{-3} learning rate. The training process is divided into a fixed number of epochs, which allows the model to learn from the training data iteratively. In this case, we have set the number of epochs to 10. We regularly evaluate our model's performance on a separate validation dataset as part of our training process. This validation step occurs at the end of each training epoch, ensuring that our GRU model learns from the training data and demonstrates effective generalization to previously unseen data. This practice helps us guard against overfitting.

Our RF. To implement our ML model, we employ a RF architecture. Indeed, this type of model has been used in several other works treating identification and authentication and has also been used in automotive-related tasks [245, 221]. After hyper-parameter tuning, our model is composed of 100 estimators using Gini impurity as criteria. Being a RF model, it is particularly lightweight in training and testing. Furthermore, as anticipated in Section 7.5.1, this model can generate identification and authentication predictions for each collected data sample. As such, continuous authentication is provided each second of driving.

Girma et al. This model, reproduced based on [144], features a detailed architecture tailored for behavior-based driver identification. The initial LSTM layer is configured with a batch-first format, processing input data with 160 hidden units and employing batch normalization. The subsequent LSTM layer, with 200 hidden

units, refines the temporal representations without batch normalization. The architecture culminates in a fully connected layer, mapping the output to 10 units for target classes. Furthermore, consistent with Section 7.6.2, the model is trained using the same hyper-parameters and the number of training epochs outlined in the referenced section for comprehensive comparability and benchmarking.

Ravi et al. This model has been reproduced following the architectural specifications outlined in [292]. This model consists of two LSTM layers with 460 and 495 hidden units, respectively, integrating batch normalization and dropout regularization for optimal performance. The design culminates with a fully connected layer mapping output to 10 units, accompanied by a sigmoid activation function capturing non-linear dependencies. Importantly, in alignment with Section 7.6.2, the model will be trained using identical hyper-parameters and the same number of training epochs, ensuring a consistent benchmark for performance assessment and facilitating meaningful comparisons across experiments.

Zhang et al. This model represents a hybrid architecture amalgamating convolutional and recurrent neural network components. Crafted with a convolutional layer stack, batch normalization, and max-pooling, the model systematically extracts hierarchical features from sequential data. Subsequently, fully connected, LSTM and GRU layers are strategically interwoven to capture intricate temporal patterns. The model dynamically weights the importance of different temporal features by incorporating an Attention mechanism. This reproduction is based on insights from [394], although specific details and code were not disclosed. While we endeavored to reproduce the model faithfully, it is crucial to acknowledge the inherent challenges stemming from the limited information provided in the paper. Certain design aspects were subject to interpretative choices during the reproduction process.

7.6.3 Baseline

We now evaluate the baseline performance of our behavior-based driver identification and authentication systems without any occurring attacks. As previously mentioned, we divide our dataset into three subsets: training (85% of the dataset), validation (5% of the dataset), and test (10% of the dataset). We train, validate, and test all the considered models (i.e., our implementations and the ones from the literature) on the same datasets. We differentiate only our RF model, which uses a different pre-processing as detailed in Section 7.3.2, since its predictions do not require time windows. We show the results of our baseline evaluation in Table 7.5.

Results. Our models can outclass the state-of-the-art in most tasks regardless of their architecture. The results achieved by the RF model are particularly significant, as they remark on the ability of the model to grasp patterns in the dataset without relying on the causality between the samples. As such, the interaction between features becomes as essential as the temporal dependency between each data collection instance. Therefore, while understanding parameter evolution can still aid driver identification (as seen by the RNN model performance), our RF excels due to feature correlations unique to each driver (e.g., acceleration during turns). We can also notice a general improvement from identification to authentication, in line with the results shown in related tasks in the literature [245]. In the case of authentication, our dataset was balanced again to maintain consistency in the data processing (i.e., the number of samples for the positive and negative labels was the same).

Combinatorial Accuracy. As mentioned in Section 7.3, our authentication systems' functionality is heavily influenced by various parameters. While our systems

TABLE 7.5: Comparison of our model's performance with the state-of-the-art.

Model	Identification		Authentication	
	Accuracy	**F1**	**Accuracy**	**F1**
Girma et al. [144]	0.993	0.994	**0.999**	**0.999**
Ravi et al. [292]	0.996	0.996	0.996	0.996
Zhang et al. [394]	0.882	0.879	0.984	0.984
RNN (ours)	**0.999**	**0.998**	**0.998**	**0.998**
RF (ours)	**0.998**	**0.998**	**0.999**	**0.999**

demonstrate high accuracy in evaluation, addressing misclassification events is crucial in practical implementation. Notifying the vehicle owner of each unauthorized batch detection could lead to multiple false alarms. To mitigate this, we wait for the identification of multiple consecutive unauthorized data batches before triggering a notification. We introduce *combinatorial accuracy*, considering the number of consecutive unauthorized batches needed to trigger an alert. This concept quantifies the likelihood of successfully detecting unlawful drivers while ensuring system robustness. The formula for combinatorial accuracy reflects a geometric distribution as follows.

$$combinatorial\_acc = \left(1 - (1 - accuracy)^{batches}\right). \tag{7.5}$$

By doing so, we ensure that a higher accuracy value supports each notification. An analysis of the false alarm probability with respect to the number of unauthorized consecutive batches is shown in Table 7.6. As it is possible to see, just by waiting for another unauthorized batch of data, the probability of sending a false alarm notification to the vehicle owner drops by three orders of magnitude, resulting in $1e-06$ and $4e-06$, respectively, for our RNN and RF model. Hence, we select two batches of unauthorized data as a parameter for notification delay since it strikes a fair trade-off between false alarm rate and time of collection. It is also worth noting that, in the case of our RF model, we do not use batches, and thus, we consider individual data samples.

TABLE 7.6: False alarm probability for identification when delaying notification by the number of consecutive unauthorized batches.

Model	False Alarm Probability		
	Batch #1	**Batch #2**	**Batch #3**
Girma et al. [144]	7.000e-03	4.900e-05	3.430e-07
Ravi et al. [292]	4.000e-03	1.600e-05	6.400e-08
Zhang et al. [394]	1.180e-01	1.392e-02	1.643e-03
RNN (ours)	**1.000e-03**	**1.000e-06**	**1.000e-09**
RF (ours)	**2.000e-03**	**4.000e-06**	**8.000e-09**

Local Training Feasibility. We also evaluate the feasibility of training and running our models in a contained environment to simulate the situation of performing the operations in a vehicle's ECU. We emulate it using a Raspberry Pi 4B [291], a small and cheap single-board computer, without any optimization in the hardware or on the code that may speed up the computations (e.g., rewriting in a compiled language). It took an average of 103 seconds to train the RF identification models,

while about 67 minutes to train our RNN identification model, as shown in Table 7.7. Regarding the actual usage of the models, the RF model needs 0.029 seconds only to evaluate one sample, while 0.010 seconds are needed to test one batch with the RNN model. Since the RF model needs to run for every sample (i.e., every second), while our RNN evaluates each batch every 40 seconds, we are well within limits for real-time use.

TABLE 7.7: Time results (in seconds) for training and testing of our models in the constrained environment of a Raspberry Pi.

Model	**Identification**		**Authentication**	
	Train	**Test**	**Train**	**Test**
RNN (ours)	4003.517	0.040	4969.321	0.065
RF (ours)	103.027	0.029	78.333	0.030

7.6.4 Attacks Evaluation

We now perform the attacks shown in Section 7.5 and evaluate their effectiveness on our authentication systems. To assess the effectiveness of our attacks, we evaluate separately the case where the attackers have access to the data (Section 7.6.4) or to the model (Section 7.6.4). We also include an analysis of our attacks working with different numbers of modifiable features in Section 7.6.4. Finally, we summarize our attacks in Section 7.6.4 and show their effectiveness in terms of success rate and time required for stealing the vehicle.

SMARTCAN Evaluation

To evaluate this attack, we consider each driver both as an attacker and a victim. We overwrite the *modifiable* features for each attacker batch of data with the ones contained in the victim driver's batches (target). We then feed these combined batches of data to all our models. The purpose of this evaluation is to determine if the models can classify the combined data as the same class as the target driver (i.e., *modifiable* features from the target driver data, *borderline* and *non-modifiable* features from the attacker's driving behavior data).

We show our experiments' results in Table 7.8. As we can see from the results of the identification tasks, our attack obtains an ASR high enough to disrupt the functionality of the systems. Moreover, in this task, we notice a correlation between the ASR and the models' baseline accuracies and F1 scores (i.e., the higher the system's accuracy, the more susceptible to the attack). Instead, our RF model appears to be one of the most resilient among the other models. This behavior can be explained by its reliance on feature interaction rather than temporal dependency. Indeed, RNN models rely on the causality of the samples and, as such, might focus on the time evolution of specific features that might be maliciously injected. Our RF model instead used only the interaction between features in each data sample to generate its prediction, and thus "mixed" samples (with *modifiable* features from one driver and *borderline* and *non-modifiable* features from another) can be detected more easily.

Nevertheless, our attack results show that identification is vulnerable regardless of whether RNN or RF models are used. We show the ASR for each combination of thief and target drivers for our ML model in Figure 7.5. It can be noticed that some drivers are more vulnerable than others (e.g., drivers 0, 4, and 5). This can be due to several factors, such as peculiar driving patterns in the training data or biases in

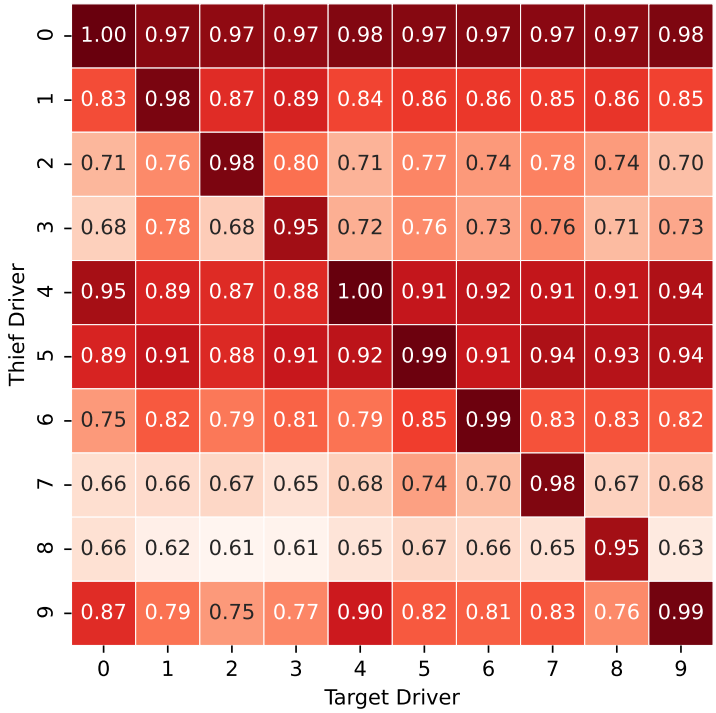


FIGURE 7.5: SMARTCAN ASR for each target and attacker drivers. The identification is based on the RF model.

the dataset creation (e.g., different routes and times of collection). Conversely, some drivers (e.g., driver 8) appear more resilient to the attack.

In the case of authentication, our attack shows perfect ASR (i.e., 1) against the RNN models. This highlights the heightened vulnerability of these models to these types of adversarial attacks. Instead, the ASR on our RF model decreases by ∼10%. This behavior strengthens our hypothesis on the importance of the interaction between features for the RF model, which is further manifested by the decrease in the number of labels in the dataset.

Getting knowledge of the legitimate data. If the attacker needs to obtain data samples through the same device, it is important to know how many batches are needed for the smart-replay attack to work. To evaluate it, we consider different percentages of the original test set (from 10% to 90%, with a step size of 10%) and repeat the evaluation. The results show that the test set size does not affect the attack significantly, as the ASR drops of at most 0.002 and the standard deviation of the obtained ASRs is at most 0.001 for the DL models. Therefore, the attacker can stop the first part of the attack after collecting just one batch of legitimate data and use that for all the following SMARTCAN attacks. In the case of our RF, we noticed that picking a sample of the test set randomly to use it for our attack is not the best strategy since we obtain a mean ASR of 0.820 and a standard deviation of 0.237 (performed multiple times). As such, the attack to be successful needs at least ten samples of data, which still constitutes an improvement over the 40 seconds needed for collecting a batch of data for a DL model.

TABLE 7.8: SMARTCAN ASR for identification and authentication evaluated on all models.

Model	ASR	
	Identification	**Authentication**
Girma et al. [144]	0.992	1.000
Ravi et al. [292]	0.994	1.000
Zhang et al. [394]	0.787	1.000
RNN (ours)	**1.000**	**1.000**
RF (ours)	**0.824**	**0.739**

GANCAN Evaluation

To evaluate this attack, we employ the generator model optimized through RL, which utilizes the output of our systems as the surrogate model for each driver in the dataset. We aim to generate data packages that could be classified as the authenticated driver. For this evaluation, we will consider each driver in the dataset as the legitimate driver once in the case of identification. For authentication, instead, we specifically aim to have the negative class labeled as positive. We trained each generator to forge data tailored to each dataset driver. Furthermore, we use the generated data only for the *modifiable* features, while the others are extracted from the attacker's driving behavior. We show the results of our evaluation in Table 7.9. From these ASRs, we notice a similar behavior to the case where the attacker knows the data (i.e., SMARTCAN). Namely, models performing best in the baseline evaluation are also more susceptible to the attacks. Furthermore, also in this case, authentication is the most vulnerable task, as ASRs reach values up to 1. Again, our RF appears to be the most resilient against our attacks, but its accuracy still drops at a level too low to allow the system's normal functioning.

TABLE 7.9: GANCAN ASR for identification and authentication evaluated on all models.

Model	ASR	
	Identification	**Authentication**
Girma et al. [144]	0.698	0.980
Ravi et al. [292]	0.599	0.998
Zhang et al. [394]	0.577	1.000
RNN (ours)	**0.809**	**1.000**
RF (ours)	**0.561**	**0.793**

Getting knowledge of the model. If the attacker does not have offline access to the model, they can use the authenticator as an oracle for training the GAN generator. Thus, the number of episodes the generator model needs to converge becomes tightly related to the time required to steal the car. Indeed, for all the tested DL models, we can generate only one batch of data collected by the authentication system over 40 seconds for each episode. We test the number of episodes needed for convergence for each target driver, obtaining a mean value of 30. The time needed for each episode to be processed, even on the CPU, is negligible (around 0.1 seconds). Therefore, the attacker will need, on average, 20 minutes (40 seconds for each of the 30 episodes) to train the generator and then be able to inject malicious packets to fool the authenticator model.

The attack targeting our RF model requires the same amount of time. Indeed, while the architecture that the attack is trying to fool is different, the surrogate model needed for the generation of the attack is the same. As such, even though authentication is performed each second, the generator needs 20 minutes to converge. After its deployment, each generated batch can be decomposed into single data samples to be injected.

Features

As discussed in Section 7.4, while overwriting the 22 modifiable features would not affect the vehicle's driving experience, the best course of action for an attacker would be modifying these features as little as possible. Thus, to identify the most critical features for compromising the authentication model, we use XAI techniques [297]. In particular, we use SHAP, which is a model-agnostic XAI technique [237]. 14 of the top 20 features identified as important based on Shapley values on our RNN model are modifiable features. As such, we study the ASR of the two attack strategies when modifying a smaller number of features.

We show the result of this study in Figure 7.6, wherein for each position x we perform the attack with the x most important modifiable features. The results show that the SMARTCAN attack can obtain an ASR of 0.992 with only 7 features. The optimal number of features to modify for the GANCAN attack seems to be 8, as it reaches an ASR of 0.809. Thus, for the attacker, injecting the eight most important modifiable features is the best tradeoff between ASR and ease of driving.

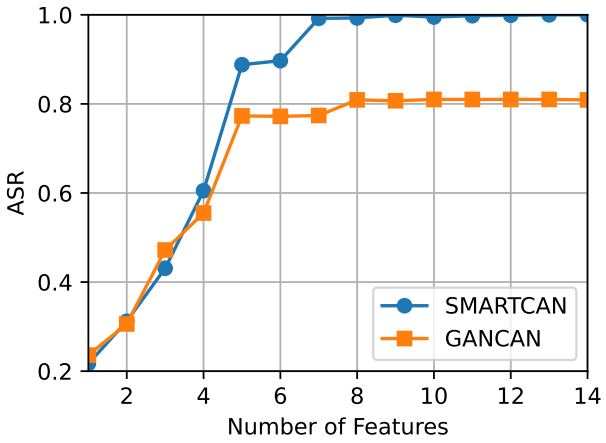


FIGURE 7.6: ASR of our attacks when considering different numbers of injectable features.

Summary

As explained throughout the chapter, in our threat model, the main object of an attacker is to steal a vehicle. However, similar strategies could be employed to modify user behavior, such as committing fraud against insurance companies. In this section, we presented many strategies adversaries may employ based on their knowledge, summarized in Table 7.10. The best solution for an attacker without knowledge of the victim's system depends on the attacker's access to the victim's vehicle. Getting access to the data is the stealthiest way to steal the car if they can access it more than once. They can also insert a GPS tracker in the malicious device to find

the vehicle for the second stage easily. In that scenario, the data collection time must also be considered. However, since we show that the attack is successful even with only one batch of legitimate data, this process should take only a few minutes.

When other security measures are in place, such as Anomaly Detection Systems (ADSs) [235], more data might be required to emulate the driver's behavior. Instead, when the attacker already has some knowledge of the victim, this procedure is not necessary since, in the former, data knowledge is part of the assumptions, and in the latter, data is generated from randomness through the generator model. Otherwise, an attacker can access the model at night or while the victim is away from the vehicle for some time (only 22 minutes are needed). However, independently of the scenario, an attacker can successfully steal the car in a reasonable time.

TABLE 7.10: Summary of the attacks against our models. Values in parentheses are needed only if the attacker has no prior knowledge of the model or data.

Attack	**Model**	**ASR***	**Time to steal the car**			
			Setup [356]	**Data**	**Train**	**Total**
SMARTCAN	RF	0.824_i	2′	(10″)	-	**2′ 10″**
		0.739_a				
	RNN	1.000_i		(40″)		**2′ 40″**
		1.000_a				
GANCAN	RF	0.561_i	2′	-	(20′)**	**22′**
		0.793_a				
	RNN	0.809_i				
		1.000_a				

*Subscripts i and a indicate, respectively, identification and authentication.

**If the attacker has previous access to the model, training can be done offline.

7.7 Takeaways

Behavior-based driver authentication systems are a well-researched topic in the literature, but they are under-appreciated by practitioners due to several limitations and oversights. Thus, we now summarize the main takeaway messages we identify to facilitate implementation in real-world scenarios. We do so by answering the research question formulated in the introduction to this chapter. In this way, we aid practitioners in the applications of these systems and suggest best practices for researchers to employ in their studies.

As detailed in Section 7.2, most works in the literature do not consider the deployment issues of the proposed driver authentication systems. However, real-world implementations of these devices must factor several aspects for their correct functioning: access to the CAN bus, proprietary message encoding and decoding, and computational capability to perform continuous authentication. Furthermore, they must be resilient against tampering since they provide security in the vehicle and deal with potentially sensitive data. For these reasons, it is necessary to implement these systems as ECUs connected directly to the CAN bus rather than external devices attached to the OBD-II port. These devices should also be installed in places

not easily accessible by third parties to increase the complexity of tampering with them.

RQ7.1 Takeaway – Behavioral-based driver authentication systems should be implemented as ECUs in the CAN bus to reduce the probability of tampering from malicious parties.

Given the potential usage of driver behavioral data beyond authentication, ensuring its security and acknowledging possible privacy issues is essential. However, given the widespread usage of complex models to obtain the highest accuracy scores, some solutions might require external resources such as CPUs and GPUs to provide continuous authentication. In this context, using cloud resources to compute the authentication response is problematic since external parties must handle the behavioral data. Despite the limited computational capabilities of in-vehicle ECUs, we demonstrated the feasibility of both training and testing for classical ML models and lightweight DL architectures. Therefore, local training and execution should be preferable to ensure that behavioral data never leave the vehicle where they were generated.

RQ7.2 Takeaway – Local training and testing are essential to preserve users' privacy and are feasible with lightweight models retaining a high accuracy.

One of the biggest challenges limiting the spread of these authentication systems is the presence of false positives. They are inherent in AI-based systems, and how to handle them is a challenge in many sectors [350]. In this chapter, we proposed the concept of combinational accuracy, which significantly reduces the probability of false positives by combining different model results evaluated on consecutive batches. This method reduces the risk of false positives of some order of magnitude at the cost of just a couple of seconds of delay, which is reasonable in the automotive context.

RQ7.3 Takeaway – Combinatorial accuracy should be used as a reference for balancing the system security with the false positive rate.

As shown in the descriptions of our attacks, all methods for fooling behavioral-based driver authentication models in our current system model involve injecting packets in the CAN bus. This technique leverages the vulnerabilities of the CAN bus implementation, where encryption, message authentication, or content integrity is not employed [181]. As such, several attacks on vehicles leverage this lack of security [47, 255, 77]. In response to these threats, researchers have proposed modifications of the CAN network to add confidentiality and integrity guarantees to the protocol [236]. With the implementation of these measures, packet injection would not be possible since an attacker would not be able to craft a legitimate signature without the private key. While other solutions, such as the ADS [244, 235], might help in the detection of attack attempts with high accuracy, we argue that the most effective way to prevent them is the implementation of CAN message authentication protocols, which are actively being implemented [22, 236].

RQ7.4 Takeaway – CAN message authentication of the frames is the most effective way to secure behavioral-based driver authentication systems.

Finally, we want to address the issue of the perceived benefit of these systems over traditional methods. While the usage of behavioral-based driver authentication systems has already been discussed in the previous sections, it can be argued that physical keys will still be the primary means of accessing a vehicle, as the security of a system is only as strong as its weakest link. However, it has been shown how keys are vulnerable to several attacks [138, 126, 225] and how attackers can employ different techniques to circumvent them [356]. Instead, driver authentication systems can detect, report, and stop attackers from stealing a vehicle even after their initial breach. However, these systems act best as a second-factor continuous authentication mechanism, incorporating the car key's convenience without compromising the system's security. Moreover, they enable other usages in different fields, such as the UBI policies for insurance [168, 99, 251].

RQ7.5 Takeaway – Behavioral-based driver authentication systems should be employed as a second-factor continuous authentication mechanism rather than a primary means for accessing a vehicle.

7.8 Conclusions

Behavioral-based driver authentication systems are promising tools that can enhance the security of vehicles by detecting unauthorized drivers. Furthermore, with other proposed usages such as UBI, their application can be beneficial for the driver's safety by detecting altered driving states. However, despite the interest gained in the literature, practitioners have not yet implemented any of these systems. This is due to several oversights in their design and the possibility of attacks that can hinder the functionality of the whole vehicle.

Contribution. In this chapter, we reduced the gap between research and practice in behavioral-based driver authentication systems. First, we defined a practical system model considering the restrictions of the CAN bus network and a realistic threat model reflecting the real-world capabilities of possible attacks. We then proposed two lightweight driver authentication and identification systems based on a RF architecture and a single GRU layer architecture, respectively. Through an extensive evaluation of real-world data, we show how our models outclass or match more complex state-of-the-art systems, achieving accuracy up to 0.999. Based on the different assumptions of the attacker's knowledge, we defined **SMARTCAN** and **GANCAN**, the first attacks fooling behavioral-based driver authentication systems. Unlike many other works, we make the code of our system and our attacks open-source. In this way, we help the community test their systems with state-of-the-art attacks and provide details on implementing safe systems. Finally, we provide researchers and practitioners with a list of takeaways for bridging the gap between theory and practice.

Limitations and Future Works. Although ambitious, testing our solutions on a real vehicle would materialize the proof of concept provided in this chapter and thus strengthen its scientific contribution. However, this proposal is challenging for several reasons. As disclosed in Section 7.5.1, the implementation of these systems requires full collaboration with the ECU manufacturers. The authenticator cannot access the data without the rules for encoding and decoding the messages. However, the closure of source codes and information in the automotive industry significantly hinders this requirement.

The collection and analysis of driver behavior data in behavior-based driver authentication systems raise serious concerns about data security and personal privacy [295, 220]. This area has received little attention in the literature. Strong privacy safeguards must be built into the system architecture to address these concerns.

In the literature, the shift from identification tasks to authentication tasks (and, thus, from multi-class classification to binary classification) has been performed by considering only one (or few) labels as the positive class and conglomerating many other labels as the negative class [29, 245]. However, the negative class should include large amounts of different labels. This can be problematic due to the privacy reasons listed before and the unrivaled amount of data that an automotive company can gather. Thus, despite the data collection efforts that researchers can perform, these systems would present lower accuracies with respect to the final product that a company might manufacture.

Finally, this chapter focused on attacks exploiting the CAN bus to inject malicious data. However, other attacks targeting sensors or the driver's driving behavior should be investigated.

8 Conclusions and Future Works

The increasing connection between vehicles and computer systems is enhancing the capabilities of these devices, providing solutions to big challenges of our times. However, as we saw in this dissertation, it also increases the space for an attacker to put the first foot inside the vehicle and launch more attacks.

We investigated issues in modern electric vehicles (Part I), proposing an overview of some security and privacy issues of this new field (Chapter 2), that highlighted different approaches to secure both EVs and their connection with the grid. Together with some similarities, this chapter highlights several differences of EV with respect to combustion vehicles, emphasizing the need to consider the peculiarities of the former from a security perspective. Therefore, by looking at vulnerabilities in the charging infrastructure, we introduced *EVExchange*, the first relay attack to the V2G system, showing how a malicious attacker could exploit a weakness in the standard to account another user for the energy consumed. To mitigate this vulnerability, we developed a distance bounding countermeasure, detailed in Chapter 3, that prevents the attack while being virtually transparent to the user. Privacy issues have been investigated in Chapter 4, proposing a methodology to track vehicles based on battery charging patterns. This approach shows how much information a simple energy delivery can bring, even when no data are transmitted. Moreover, it allowed us to reason about how, in the CPS field, even processes that may initially seem harmless could instead turn out to be sources of information leakage.

Part II, instead, investigated legacy but still widely employed communication systems in vehicles, both with electric and combustion engines. In particular, it investigates the most spread protocol for in-vehicle communication, the CAN bus, developed when developers did not usually consider security in the design of communication protocols. However, due to its efficiency and to maintain compatibility with legacy devices, it is still the de-facto standard nowadays. While several works in the literature focused on developing security solutions for the CAN bus, no one is widely spread in commercial vehicles. One possible reason is related to the economic impact of adding such systems to every vehicle. To answer this need, Chapter 5 proposed a lightweight IDS that is able to reach high performances while running on cheap microcomputers. Attempting to move in the same direction of increasing corporate adoption of security systems, we tackle the false positive and false negative issue that causes great trouble in the CPS field. In Chapter 6, we introduced *CANTXSec*, the first deterministic IDPS for the CAN bus. At the cost of some extra wiring, we showed how it can eradicate false alarms in detecting attacks by comparing ECU activations with the traffic on the bus. Moreover, it allows efficient prevention of the most common attacks in the CAN bus. Since much information related to the vehicle state is transmitted through the CAN bus, we investigated the application of these data to novel behavioral-based authentication systems in Chapter 7. In particular, we focused on the security weaknesses of state-of-the-art systems, developing two attacks against different models that were successful in breaking the system by mimicking the legitimate user driving style. To mitigate the vulnerabilities, we discuss

and provide takeaways to help practitioners in deploying secure solutions.

Our analysis provided an answer to the research questions formulated in the introduction, as follows.

RQ1 The EV environment exhibited a vast space for attacks, more significant than the one of petrol-based vehicles. This is mainly due to the increased number of electronic components (e.g., BMS, batteries, charger) and, in particular, the connection to the grid. Through the charging process, in addition to energy, data is exchanged, and a full Internet connection can be established, greatly increasing the space for attackers. A complete overview of the main EV-specific assets and their attack and countermeasures are summarized in Table 2.4. As discussed thoroughly in Chapter 2, the literature provides some security solutions that could be adapted to these scenarios, while others are still unexplored.

RQ2 From our analysis, we identify a security issue due to a vulnerability in a protocol that regulates the charging of EVs, as discussed in Chapter 3. We demonstrated the attack in an emulated scenario, proving how relay attacks in the V2G scenarios are a reality. Moreover, we also implemented a solution that is transparent to the user and helps enhance the security of the environment. However, this should not be taken as a definitive solution since cybersecurity is a process, and a solution that is working today may not be enough in the future with new standards and procedures.

RQ3 We prove how the energy delivered by a charging column to a vehicle battery contains enough information related to the vehicle to enable its profiling, mining the driver's privacy. What is detailed in Chapter 4 is a demonstration of how many side channels are widespread in the CPS environment. This should encourage companies and developers to analyze and design solutions that minimize leaked information, investigating even the most unsuspected channels.

RQ4 We developed two solutions to reduce the computational cost (Chapter 5) and to eradicate false positives in the CAN bus (Chapter 6). The former has been achieved by employing a fast feature extraction technique and a small quantized model. The latter, instead, employed the relationship between the traffic on the bus and the activation states of the devices connected to it. Both solutions represent a first step towards reducing two of the main issues that limit the diffusion of security interruptions in vehicles, possibly allowing a wider spread of security enhancements in the future.

RQ5 We analyzed the particular usage of CAN bus data to extract behavioral information of the driver to be employed as an authentication system in Chapter 7. We showed how these systems are subject to several security issues, such as the two attacks we developed. Moreover, we designed a secure implementation and provided several takeaways to help practitioners in deploying such systems. While the CAN bus is a massive source of data regarding the vehicle's state and its driver, researchers and companies should handle it with care to prevent privacy leakage or exposing of vulnerabilities.

Future Works. As already mentioned, this work presented some analysis to reason about the security of the vehicle environment, providing some solutions for EVs and their charging ecosystem, and the CAN bus communication. Despite the advancements introduced, the attack surface of the vehicular ecosystem goes beyond

this dissertation. Many issues remain open, not only regarding the attack vector addressed in this dissertation but also regarding the whole vehicle's environment and the new advancements that the future will bring. Some topics that represent interesting future works are presented below.

- *V2G Backend Security.* In Part I of this dissertation, we investigated the issues related to the ISO 15118 protocol and the connection between a car and the charging column. This *frontend* communication is the most accessible to an attacker and, therefore, the most exposed. However, the communication between charging columns and control infrastructures may open several issues in different directions. The protocol generally employed, the OCPP, is new, and security issues have been discovered in the last years [13, 12]. Moreover, attacks on the backend can impact the stability of the electric grid, which is an issue that goes beyond the EVs domain [307].

- *V2G Applications Security.* Most of the charging services offered to the users employ some mobile applications to enable user payments and enhance the charging experience. For instance, they allow the owner to monitor the progress of the charging and to start/stop it remotely. The security of this application requires important investigation to minimize the risk of fraud against the charging companies or the users. Moreover, charging apps may also be exploited as a means to attack the whole charging grid [306].

- *Effective CAN bus security solutions.* In Part II of this dissertation, we addressed the development of IDS and IPS for the CAN bus, focusing on the effective deployability of the proposed solutions. However, there are still some challenges that need to be accomplished, such as the reduction of false positives of the *CANLP* solution or the reduction of the wired requirement in the *CANTXSec* proposal. Designing security solutions for the CAN bus that i) do not increase communication delay, and ii) are cheap and easy to deploy is a fundamental challenge to allow companies to take into consideration a massive spread of these systems, enhancing the overall cybersecurity in the transportation means.

- *Car mobile applications security.* Many modern vehicles are equipped with mobile applications that the owner can use to control functions of the vehicle remotely, e.g., turning on the air conditioning, opening/closing the vehicle remotely, check the security cameras. These have increased the attack space including the users's smartphones, which can be compromised with completely other methods. Preliminary analysis has discovered security issues in some applications [69]. The security of these applications should be investigated in depth and with new technologies and methodologies [60].

- *Novel vehicles.* While cars are still the most used vehicles in the world, many other systems exist for public and private transportation. An example is the ultra-speed train under development in these years, Hyperloop, which suffers from peculiar security and privacy issues [54]. On the other side, personal devices such as e-scooters have started to spread and have been proven vulnerable to cyberattacks [61]. Similarly, wireless gearboxes for racing bikes are subject to jamming and replay attacks [262]. The investigation from a security point of view of novel vehicles is something that should be taken care of, focusing on the peculiar features of each device.

- *Vehicle-to-Everything (V2X).* Enabling vehicles to communicate with each other opens to many novel applications. Through the usage of 5G, for instance, emergency vehicles may interact with traffic lights to get right of way at intersections. Communication between nearby vehicles allows the formation of platoons, a series of vehicles proceeding in a line very close to each other, reducing air resistance and thus saving in fuel consumption and pollution [142]. Moreover, as seen in Chapter 3, V2G communications can generate a bidirectional flow of energy, helping the stability of the grid during peak hours. Despite the advancements in interconnections, security issues may arise. In particular, ensuring the trustability of the communications and the entities involved is challenging, especially limiting the privacy leakages of the various users involved [388]. This research field is fundamental nowadays and may increase its centrality in the following few years due to the spread of communication-capable vehicles.

- *Autonomous Driving*. Thanks to recent advancements in AI, autonomous driving is gaining a lot of interest both from academia and industry. This discipline interests a lot of different research fields [33]: computer vision is essential for analyzing the surrounding scenario through cameras; AI helps in identifying objects and deciding on the action to take; microelectronics is constantly challenged to develop smaller and more efficient devices with massive computational capabilities; regulators and ethics committees are also interested in the field due to the need for regulations and ethics guidelines. In this multidisciplinary field, privacy and cybersecurity research seeks to find a space to assess the state of the proposed systems and design secure-by-design systems for the future [136]. Being a field in constant development, security researchers need to stay up to date and participate in the innovation process.

Acknowledgements

I want to express my heartfelt thanks to Prof. Mauro Conti for giving me the opportunity to pursue this incredible journey, for believing in me, and for offering his guidance and support over the years. I enjoyed every discussion with him and truly appreciate the freedom he allowed me to pursue my projects.

If I could tell the young high school student from a small village in the Dolomites that one day he would be doing research in the United States, I think he would laugh. Instead, during this journey, I got this opportunity, and I would like to thank Prof. Radha Poovendran for his support during this adventure.

A special thanks also to Omitech S.R.L. for their financial support over these years.

Moreover, I would like to thank Prof. Alessandro Brighente, Federico Turrin, and my fellow colleague Gabriele Orazi for their patience while listening and discussing my (usually not-so-smart) ideas.

A big thanks is due also to all my co-authors: Kavya Balasubramanian, Adithya Gowda Baragur, Alessandro Brighente, Federico Carboni, Mauro Conti, Francesco Marchiori, Luca Pajola, Radha Poovendran, Bhaskar Ramasubramania, Dinuka Sahabandu, Mariano Sciacco, Matteo Soldà, Federico Turrin, and Jianying Zhou.

Not all the ideas came from hard study in the papers, but also from socializing and discussing with your peers. I would like to thank everyone at 731 for the wonderful time we had together. This includes Federico, Francesco, Gabriele, Luca, Matteo, Pier Paolo, Stefano, Tommaso, and all the people who stopped by for an espresso. A special thanks also to Elena for her support with the Italian paperwork.

Last but not least, the support of your loved ones is essential to go through a hard journey like a PhD. My big thanks to my parents and Gloria, who have always supported me through this adventure.

Bibliography

[1] ABDENNOUR, N., OUNI, T., AND AMOR, N. B. Driver identification using only the can-bus vehicle data through an rcn deep learning approach. *Robotics and Autonomous Systems 136* (2021), 103707.

[2] ABU-GELLBAN, H., NGUYEN, L., MOGHADASI, M., PAN, Z., AND JIN, F. Livedi: An anti-theft model based on driving behavior. In *Proceedings of the 2020 ACM Workshop on Information Hiding and Multimedia Security* (2020), pp. 67–72.

[3] ACHARYA, S., DVORKIN, Y., PANDŽIĆ, H., AND KARRI, R. Cybersecurity of smart electric vehicle charging: A power grid perspective. *IEEE Access 8* (2020), 214434–214453.

[4] AGARAP, A. F. Deep learning using rectified linear units (relu), 2019.

[5] AGENCY, U. S. E. P. "Sources of Greenhouse Gas Emissions". https://www.epa.gov/ghgemissions/sources-greenhouse-gas-emissions, 2016. Accessed: May 20, 2024.

[6] AGRAWAL, K., ALLADI, T., AGRAWAL, A., CHAMOLA, V., AND BENSLIMANE, A. Novelads: A novel anomaly detection system for intra-vehicular networks. *IEEE Transactions on Intelligent Transportation Systems 23*, 11 (2022), 22596–22606.

[7] AHMAD, A., LEE, S., AND PEINADO, M. Hardlog: Practical tamper-proof system auditing using a novel audit device. In *2022 IEEE Symposium on Security and Privacy (SP)* (2022), IEEE Computer Society, pp. 1554–1554.

[8] AHMADI-ASSALEMI, G., AL-KHATEEB, H. M., MAPLE, C., EPIPHANIOU, G., HAMMOUDEH, M., JAHANKHANI, H., AND PILLAI, P. Optimising driver profiling through behaviour modelling of in-car sensor and global positioning system data. *Computers & Electrical Engineering 91* (2021), 107047.

[9] AHMADVAND, M., PRETSCHNER, A., AND KELBERT, F. A taxonomy of software integrity protection techniques. In *Advances in Computers*, vol. 112. Elsevier, 2019, pp. 413–486.

[10] AIZAWA, A. An information-theoretic perspective of tf–idf measures. *Information Processing & Management 39*, 1 (2003), 45–65.

[11] AKHTAR, T., POLITIS, I., AND KOTSOPOULOS, S. Wireless channel characterisation over simulations for an indoors environment at 2.4 GHz. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST 263* (2019), 387–397.

[12] ALCARAZ, C., CUMPLIDO, J., AND TRIVINO, A. Ocpp in the spotlight: threats and countermeasures for electric vehicle charging infrastructures 4.0. *International Journal of Information Security 22*, 5 (2023), 1395–1421.

[13] ALCARAZ, C., LOPEZ, J., AND WOLTHUSEN, S. Ocpp protocol: Security threats and challenges. *IEEE Transactions on Smart Grid 8*, 5 (2017), 2452–2459.

[14] ALI, M., SHIAELES, S., BENDIAB, G., AND GHITA, B. Malgra: Machine learning and n-gram malware feature extraction and detection system. *Electronics 9*, 11 (2020).

[15] AMBROSIN, M., CONTI, M., LAZZERETTI, R., RABBANI, M. M., AND RANISE, S. Collective remote attestation at the internet of things scale: State-of-the-art and future challenges. *IEEE Communications Surveys & Tutorials 22*, 4 (2020), 2447–2461.

[16] ANEGAWA, T. Characteristics of chademo quick charging system. *World Electric Vehicle Journal 4*, 4 (2010), 818–822.

[17] ANTOUN, J., KABIR, M. E., MOUSSA, B., ATALLAH, R., AND ASSI, C. A Detailed Security Assessment of the EV Charging Ecosystem. *IEEE Network 34*, 3 (2020), 200–207.

[18] ANTOUN, J., KABIR, M. E., MOUSSA, B., ATALLAH, R., AND ASSI, C. A detailed security assessment of the ev charging ecosystem. *IEEE Network 34*, 3 (2020), 200–207.

[19] ARDUINO. Arduino: open-source electronic prototyping platform enabling users to create interactive electronic objects. https://www.arduino.cc/, February 2023.

[20] ASHRAF, J., BAKHSHI, A. D., MOUSTAFA, N., KHURSHID, H., JAVED, A., AND BEHESHTI, A. Novel deep learning-enabled lstm autoencoder architecture for discovering anomalous events from intelligent transportation systems. *IEEE Transactions on Intelligent Transportation Systems 22*, 7 (2021), 4507–4518.

[21] ATTANASIO, L., CONTI, M., DONADEL, D., AND TURRIN, F. Miniv2g: An electric vehicle charging emulator. In *Proceedings of the 7th ACM on Cyber-Physical System Security Workshop* (New York, NY, USA, 2021), CPSS '21, Association for Computing Machinery, p. 65–73.

[22] AUTOSAR. Specification of secure onboard communication protocol. https://www.autosar.org/fileadmin/standards/R20-11/FO/AUTOSAR_PRS_SecOcProtocol.pdf, 2020. Accessed: Jan. 02, 2024.

[23] AZADANI, M. N., AND BOUKERCHE, A. Performance evaluation of driving behavior identification models through can-bus data. In *2020 IEEE Wireless Communications and Networking Conference (WCNC)* (2020), IEEE, pp. 1–6.

[24] AZADANI, M. N., AND BOUKERCHE, A. Driver identification using vehicular sensing data: A deep learning approach. In *2021 IEEE Wireless Communications and Networking Conference (WCNC)* (2021), IEEE, pp. 1–6.

[25] BABU, P. R., PALANISWAMY, B., REDDY, A. G., ODELU, V., AND KIM, H. S. A survey on security challenges and protocols of electric vehicle dynamic charging system. *Security and Privacy 5*, 3 (2022).

[26] BAFNA, P., PRAMOD, D., AND VAIDYA, A. Document clustering: Tf-idf approach. In *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)* (2016), IEEE, pp. 61–66.

[27] BAHRAMI, A. Ev charging definitions, modes, levels, communication protocols and applied standards. *Changes 1* (2020), 10–01.

[28] BAI, Y.-S., AND ZHANG, C.-N. 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.

[29] BAILEY, K. O., OKOLICA, J. S., AND PETERSON, G. L. User identification and authentication using multi-modal behavioral biometrics. *Computers & Security 43* (2014), 77–89.

[30] BAKER, R., AND MARTINOVIC, I. Losing the car keys: Wireless phy-layer insecurity in EV charging. In *28th USENIX Security Symposium (USENIX Security 19)* (Santa Clara, CA, Aug. 2019), USENIX Association, pp. 407–424.

[31] BALASUBRAMANIAN, K., BARAGUR, A. G., DONADEL, D., SAHABANDU, D., BRIGHENTE, A., RAMASUBRAMANIAN, B., CONTI, M., AND POOVENDRAN, R. CANLP: A NLP-based intrusion detection system for CAN. In *Proceedings of the 2024 ACM/ SIGAPP Symposium on Applied Computing* (2024), pp. 212–214.

[32] BAO, K., VALEV, H., WAGNER, M., AND SCHMECK, H. A threat analysis of the vehicle-to-grid charging protocol ISO 15118. *Computer Science - Research and Development 33*, 1-2 (2018), 3–12.

[33] BARABAS, I., TODORUŢ, A., CORDOŞ, N., AND MOLEA, A. Current challenges in autonomous driving. In *IOP conference series: materials science and engineering* (2017), vol. 252, IOP Publishing, p. 012096.

[34] BARANDAS, M., FOLGADO, D., FERNANDES, L., SANTOS, S., ABREU, M., BOTA, P., LIU, H., SCHULTZ, T., AND GAMBOA, H. Tsfel: Time series feature extraction library. *SoftwareX 11* (2020), 100456.

[35] BARLETTA, V. S., CAIVANO, D., NANNAVECCHIA, A., AND SCALERA, M. A kohonen som architecture for intrusion detection on in-vehicle communication networks. *Applied Sciences 10*, 15 (2020).

[36] BARRÉ, A., DEGUILHEM, B., GROLLEAU, S., GÉRARD, M., SUARD, F., AND RIU, D. A review on lithium-ion battery ageing mechanisms and estimations for automotive applications. *Journal of Power Sources 241* (2013), 680–689.

[37] BEHL, D., HANDA, S., AND ARORA, A. A bug mining tool to identify and analyze security bugs using naive bayes and tf-idf. In *2014 International Conference on Reliability Optimization and Information Technology* (2014), pp. 294–299.

[38] BEKTAŞ, S., AND ŞIŞMAN, Y. The comparison of l1 and l2-norm minimization methods. *International Journal of the Physical Sciences 5*, 11 (2010), 1721–1727.

[39] BERA, P. K., AND ISIK, C. Identification of stable and unstable power swings using pattern recognition. In *2021 IEEE Green Technologies Conference (GreenTech)* (2021), pp. 286–291.

[40] BERE, G., OCHOA, J. J., KIM, T., AND AENUGU, I. R. Blockchain-based firmware security check and recovery for battery management systems. In *2020 IEEE Transportation Electrification Conference & Expo (ITEC)* (2020), IEEE, pp. 262–266.

[41] BHATIA, R., KUMAR, V., SERAG, K., CELIK, Z. B., PAYER, M., AND XU, D. Evading voltage-based intrusion detection on automotive can. In *NDSS* (2021).

[42] BITBANE. CANT, 2016. Defcon 26 Car Hacking Village.

[43] BLECH, T. Chademo dc charging standard: evolution strategy and new challenges, 2019.

[44] BLOOM, G. Weepingcan: A stealthy can bus-off attack. In *Workshop on Automotive and Autonomous Vehicle Security* (2021).

[45] BLUM, A. F., AND LONG JR, R. T. *Fire Hazard Assessment of Lithium Ion Battery Energy Storage Systems*. Springer, 2016.

[46] BOGOSYAN, S., AND GOKASAN, M. Novel strategies for security-hardened bms for extremely fast charging of bevs. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)* (2020), IEEE, pp. 1–7.

[47] BOZDAL, M., SAMIE, M., ASLAM, S., AND JENNIONS, I. Evaluation of can bus security challenges. *Sensors 20*, 8 (2020), 2364.

[48] BOZDAL, M., SAMIE, M., AND JENNIONS, I. A survey on can bus protocol: Attacks, challenges, and potential solutions. In *International Conference on Computing, Electronics & Communications Engineering* (2018), IEEE, pp. 201–205.

[49] BRANDS, S., AND CHAUM, D. Distance-bounding protocols. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 765 LNCS* (1994), 344–359.

[50] BREIMAN, L. Random forests. *Machine learning 45*, 1 (2001), 5–32.

[51] BREIMAN, L., FRIEDMAN, J., STONE, C. J., AND OLSHEN, R. A. *Classification and regression trees*. CRC press, 1984.

[52] BRIGHENTE, A., CONTI, M., DONADEL, D., POOVENDRAN, R., TURRIN, F., AND ZHOU, J. Electric vehicles security and privacy: Challenges, solutions, and future needs. *arXiv preprint arXiv:2301.04587* (2023).

[53] BRIGHENTE, A., CONTI, M., DONADEL, D., AND TURRIN, F. Evscout2.0: Electric vehicle profiling through charging profile. *ACM Transactions Cyber-Physical Systems* (sep 2022).

[54] BRIGHENTE, A., CONTI, M., DONADEL, D., AND TURRIN, F. Hyperloop: A cybersecurity perspective. In *2024 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)* (2024), IEEE, pp. 352–360.

[55] BRIGHENTE, A., CONTI, M., AND SADAF, I. 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* (2021), Springer, pp. 651–667.

[56] BURTON, A., PARIKH, T., MASCARENHAS, S., ZHANG, J., VORIS, J., ARTAN, N. S., AND LI, W. Driver identification and authentication with active behavior modeling. In *2016 12th International Conference on Network and Service Management (CNSM)* (2016), IEEE, pp. 388–393.

[57] BUSCEMI, A., TURCANU, I., CASTIGNANI, G., PANCHENKO, A., ENGEL, T., AND SHIN, K. G. A survey on controller area network reverse engineering. *IEEE Communications Surveys & Tutorials* (2023).

[58] BUSCHLINGER, L., SPRINGER, M., AND ZHDANOVA, M. Plug-and-patch: Secure value added services for electric vehicle charging. In *Proceedings of the 14th International Conference on Availability, Reliability and Security* (2019), pp. 1–10.

[59] CAPUDER, T., SPRČIĆ, D. M., ZORIČIĆ, D., AND PANDŽIĆ, H. Review of challenges and assessment of electric vehicles integration policy goals: Integrated risk analysis approach. *International Journal of Electrical Power & Energy Systems 119* (2020), 105894.

[60] CARBONI, F., CONTI, M., DONADEL, D., AND SCIACCO, M. If you're scanning this, it's too late! a qr code-based fuzzing methodology to identify input vulnerabilities in mobile apps. In *International Conference on Applied Cryptography and Network Security* (2023), Springer, pp. 553–570.

[61] CASAGRANDE, M., CESTARO, R., LOSIOUK, E., CONTI, M., AND ANTONIOLI, D. E-spoofer: Attacking and defending xiaomi electric scooter ecosystem. In *Proceedings of the 16th ACM Conference on Security and Privacy in Wireless and Mobile Networks* (2023), pp. 85–95.

[62] CASAGRANDE, M., CONTI, M., AND LOSIOUK, E. Contact tracing made unrelay-able. In *Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy* (2021), pp. 221–232.

[63] CAVDAR, D., AND TOMUR, E. 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*, May (2015), 1308–1312.

[64] CHAKRABORTY, N. Intrusion detection system and intrusion prevention system: A comparative study. *International Journal of Computing and Business Research (IJCBR) 4*, 2 (2013), 1–8.

[65] CHAN, A. C.-F., AND ZHOU, J. On smart grid cybersecurity standardization: Issues of designing with nistir 7628. *IEEE Communications Magazine 51*, 1 (2013), 58–65.

[66] CHAN, A. C.-F., AND ZHOU, J. Cyber–physical device authentication for the smart grid electric vehicle ecosystem. *IEEE Journal on Selected Areas in Communications 32*, 7 (2014), 1509–1517.

[67] CHAN, A. C.-F., AND ZHOU, J. A secure, intelligent electric vehicle ecosystem for safe integration with the smart grid. *IEEE Transactions on Intelligent Transportation Systems 16*, 6 (2015), 3367–3376.

[68] CHANDWANI, A., DEY, S., AND MALLIK, A. Cybersecurity of onboard charging systems for electric vehicles—review, challenges and countermeasures. *IEEE Access 8* (2020), 226982–226998.

[69] CHATZOGLOU, E., KAMBOURAKIS, G., AND KOULIARIDIS, V. A multi-tier security analysis of official car management apps for android. *Future Internet 13*, 3 (2021), 58.

[70] CHECKOWAY, S., MCCOY, D., KANTOR, B., ANDERSON, D., SHACHAM, H., SAVAGE, S., KOSCHER, K., CZESKIS, A., ROESNER, F., AND KOHNO, T. Comprehensive experimental analyses of automotive attack surfaces. In *20th USENIX security symposium (USENIX Security 11)* (2011).

[71] CHEN, B., YIN, J., CHEN, S., CHEN, B., AND LIU, X. An adaptive model ensemble adversarial attack for boosting adversarial transferability. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2023), pp. 4489–4498.

[72] CHEN, J., WU, Z., AND ZHANG, J. Driver identification based on hidden feature extraction by using adaptive nonnegativity-constrained autoencoder. *Applied Soft Computing 74* (2019), 1–9.

[73] CHEN, M., AND RINCON-MORA, G. Accurate electrical battery model capable of predicting runtime and i-v performance. *IEEE Transactions on Energy Conversion 21*, 2 (2006), 504–511.

[74] CHENG, L., LIU, F., AND YAO, D. Enterprise data breach: causes, challenges, prevention, and future directions. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 7*, 5 (2017), e1211.

[75] CHISCOP, I., GAZDAG, A., BOSMAN, J., AND BICZÓK, G. Detecting message modification attacks on the can bus with temporal convolutional networks. *arXiv preprint arXiv:2106.08692* (2021).

[76] CHO, K., KIM, J., CHOI, D. Y., YOON, Y. H., OH, J. H., AND LEE, S. E. An fpga-based ecu for remote reconfiguration in automotive systems. *Micromachines 12*, 11 (2021), 1309.

[77] CHO, K.-T., AND SHIN, K. G. Error handling of in-vehicle networks makes them vulnerable. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (2016), pp. 1044–1055.

[78] CHO, K.-T., AND SHIN, K. G. Fingerprinting electronic control units for vehicle intrusion detection. In *25th USENIX Security Symposium (USENIX Security 16)* (2016), pp. 911–927.

[79] CHOI, W., LEE, S., JOO, K., JO, H. J., AND LEE, D. H. An enhanced method for reverse engineering can data payload. *IEEE Transactions on Vehicular Technology 70*, 4 (2021), 3371–3381.

[80] CHRIST, M., BRAUN, N., NEUFFER, J., AND KEMPA-LIEHR, A. W. Time Series FeatuRe Extraction on basis of Scalable Hypothesis tests (tsfresh – A Python package). *Neurocomputing 307* (2018), 72–77.

[81] CHRIST, M., KEMPA-LIEHR, A. W., AND FEINDT, M. Distributed and parallel time series feature extraction for industrial big data applications. *arXiv preprint arXiv:1610.07717* (May 2016).

[82] CHUNG, M. Y., JUNG, M. H., LEE, T. J., AND LEE, Y. Performance analysis of HomePlug 1.0 MAC with CSMA/CA. *IEEE Journal on Selected Areas in Communications 24*, 7 (2006), 1411–1420.

[83] CLARITY, V. "Reference Implementation Supporting the Evolution of the Vehicle-2-Grid communication interface ISO 15118". https://v2g-clarity.com/rise-v2g/, 2020. Accessed: May 14, 2021.

[84] CLEMENT-NYNS, K., HAESEN, E., AND DRIESEN, J. The impact of vehicle-to-grid on the distribution grid. *Electric Power Systems Research 81*, 1 (2011), 185–192.

[85] COMMAAI. opendbc: democratize access to car decoder rings. https://github.com/commaai/opendbc, 2023. Accessed: Aug. 13, 2023.

[86] COMMISSION, I. E. Plugs, socket-outlets, vehicle connectors and vehicle inlets - Conductive charging of electric vehicles - Part 1: General requirements. Standard, International Electrotechnical Commission, Geneva, CH, 2014.

[87] COMMITTEE, S. E. C. S., ET AL. Sae electric vehicle conductive charge coupler. *SAE, California* (2001).

[88] CONTI, M., DONADEL, D., POOVENDRAN, R., AND TURRIN, F. Evexchange: A relay attack on electric vehicle charging system. In *Computer Security – ESORICS 2022* (2022), Springer International Publishing, pp. 488–508.

[89] CONTI, M., DONADEL, D., AND TURRIN, F. A survey on industrial control system testbeds and datasets for security research. *IEEE Communications Surveys & Tutorials 23*, 4 (2021), 2248–2294.

[90] CONTI, M., MANCINI, L. V., SPOLAOR, R., AND VERDE, N. V. Can't you hear me knocking: Identification of user actions on android apps via traffic analysis. In *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy* (2015), pp. 297–304.

[91] CONTI, M., NATI, M., ROTUNDO, E., AND SPOLAOR, R. 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.

[92] COPPERHILL TECHNOLOGIES. PiCAN 2 - CAN Bus Interface for Raspberry Pi. copperhilltech.com/pican2-duo-can-bus-board-for-raspberry-pi/, 2023. Accessed: Aug. 23, 2023.

[93] CORBETT, C., SCHOCH, E., KARGL, F., AND PREUSSNER, F. Automotive ethernet: Security opportunity or challenge? *Sicherheit 2016-Sicherheit, Schutz und Zuverlässigkeit* (2016).

[94] CORTI, F., REATTI, A., PICCIRILLI, M. C., GRASSO, F., PAOLUCCI, L., AND KAZIMIERCZUK, M. K. Simultaneous wireless power and data transfer: Overview and application to electric vehicles. In *2020 IEEE International Symposium on Circuits and Systems (ISCAS)* (2020), IEEE, pp. 1–5.

[95] COSTANTINO, G., DE VINCENZI, M., MARTINELLI, F., AND MATTEUCCI, I. Electric vehicle security and privacy: A comparative analysis of charging methods. In *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)* (2023), IEEE, pp. 1–7.

[96] COWARD, C. Learn how to dump your car's ecu rom. https://www.hackster.io/news/learn-how-to-dump-your-car-s-ecu-rom-06e58cb30b80, 2021. Hackster.io. Accessed: 2024-05-09.

[97] CRONIN, P., GAO, X., YANG, C., AND WANG, H. Charger-Surfing: Exploiting a power line Side-Channel for smartphone information leakage. In *30th USENIX Security Symposium (USENIX Security 21)* (Aug. 2021), USENIX Association, pp. 681–698.

[98] CULLER, M., AND BURROUGHS, H. Cybersecurity considerations for grid-connected batteries with hardware demonstrations. *Energies 14*, 11 (2021), 3067.

[99] CURA, A., KÜÇÜK, H., ERGEN, E., AND ÖKSÜZOĞLU, I. B. Driver Profiling Using Long Short Term Memory (LSTM) and Convolutional Neural Network (CNN) Methods. *IEEE Transactions on Intelligent Transportation Systems 22*, 10 (Oct. 2021), 6572–6582. Conference Name: IEEE Transactions on Intelligent Transportation Systems.

[100] DANIEL LEUSSINK, K. K. More than 2 million toyota users face risk of vehicle data leak in japan. https://www.reuters.com/business/autos-transportation/toyota-flags-possible-leak-more-than-2-mln-users-vehicle-data-japan-2023-05-12/, 2023. Reuters. Accessed: 2024-05-09.

[101] DAS, H. S., RAHMAN, M. M., LI, S., AND TAN, C. Electric vehicles standards, charging infrastructure, and impact on grid integration: A technological review. *Renewable and Sustainable Energy Reviews 120* (2020), 109618.

[102] DE ARAUJO-FILHO, P. F., PINHEIRO, A. J., KADDOUM, G., CAMPELO, D. R., AND SOARES, F. L. An efficient intrusion prevention system for can: Hindering cyber-attacks with a low-cost platform. *IEEE Access 9* (2021), 166855–166869.

[103] DE FAVERI TRON, A., LONGARI, S., CARMINATI, M., POLINO, M., AND ZANERO, S. Canflict: exploiting peripheral conflicts for data-link layer attacks on automotive networks. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security* (2022), pp. 711–723.

[104] DEL CAMPO, I., FINKER, R., MARTINEZ, M. V., ECHANOBE, J., AND DOCTOR, F. A real-time driver identification system based on artificial neural networks and cepstral analysis. In *2014 International Joint Conference on Neural Networks (IJCNN)* (2014), IEEE, pp. 1848–1855.

[105] DELOITTE. Electric vehicles: setting a course for 2030. https://www2.deloitte.com/us/en/insights/focus/future-of-mobility/electric-vehicle-trends-2030.html, July 2020. Accessed: June 15, 2021.

[106] DENG, H., RUNGER, G., TUV, E., AND VLADIMIR, M. A time series forest for classification and feature extraction. *Information Sciences 239* (2013), 142–153.

[107] DESTA, A. K., OHIRA, S., ARAI, I., AND FUJIKAWA, K. Rec-cnn: In-vehicle networks intrusion detection using convolutional neural networks trained on recurrence plots. *Vehicular Communications 35* (2022), 100470.

[108] DEY, S., CHANDWANI, A., AND MALLIK, A. Real time intelligent data processing algorithm for cyber resilient electric vehicle onboard chargers. In *2021 IEEE Transportation Electrification Conference & Expo (ITEC)* (2021), IEEE, pp. 1–6.

[109] DI PINTO, A. A., DRAGONI, Y., AND CARCANO, A. Triton: The first ics cyber attack on safety instrument systems. In *Blackhat USA* (2018), pp. 1–26.

[110] DOAN, T. P., AND GANESAN, S. Can crypto fpga chip to secure data transmitted through can fd bus using aes-128 and sha-1 algorithms with a symmetric key. Tech. rep., SAE Technical Paper, 2017.

[111] DONADEL, D., MARCHIORI, F., PAJOLA, L., AND CONTI, M. Can LLMs Understand Computer Networks? Towards a Virtual System Administrator . In *2024 IEEE 49th Conference on Local Computer Networks (LCN)* (Los Alamitos, CA, USA, Oct. 2024), IEEE Computer Society, pp. 1–10.

[112] DRIMER, S., AND MURDOCH, S. J. Keep your enemies close: Distance bounding against smartcard relay attacks. *16th USENIX Security Symposium* (2007), 87–102.

[113] DUDEK, S., DELAUNAY, J.-C., AND FARGUES, V. V2g injector: Whispering to cars and charging units through the power-line. In *Proceedings of the SSTIC (Symposium sur la sécurité des technologies de l'information et des communications), Rennes, France* (2019), pp. 5–7.

[114] DWORK, C. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation* (2008), Springer, pp. 1–19.

[115] EFATINASAB, E., MARCHIORI, F., DONADEL, D., BRIGHENTE, A., AND CONTI, M. When authentication is not enough: On the security of behavioral-based driver authentication systems. *arXiv e-prints* (2023), arXiv–2306.

[116] EL MEKKI, A., BOUHOUTE, A., AND BERRADA, I. Improving driver identification for the next-generation of in-vehicle software systems. *IEEE Transactions on Vehicular Technology 68*, 8 (2019), 7406–7415.

[117] ELEND, B., AND ADAMSON, T. Cyber security enhancing can transceivers. In *Proceedings of the 16th International CAN Conference* (2017).

[118] ERZIN, E., YEMEZ, Y., TEKALP, A. M., ERÇIL, A., ERDOGAN, H., AND ABUT, H. Multimodal person recognition for human-vehicle interaction. *IEEE MultiMedia 13*, 2 (2006), 18–31.

[119] EZZINI, S., BERRADA, I., AND GHOGHO, M. Who is behind the wheel? driver identification and fingerprinting. *Journal of Big Data 5*, 1 (2018), 1–15.

[120] FASTNED. What is Autocharge? https://support.fastned.nl/hc/en-gb/articles/115012747127-What-is-Autocharge-. Accessed: Sept. 06, 2024.

[121] FERRI, C., HERNÁNDEZ-ORALLO, J., AND MODROIU, R. An experimental comparison of performance measures for classification. *Pattern Recognition Letters 30*, 1 (2009), 27–38.

[122] FIJALKOWSKI, B. Media oriented system transport (most) networking. In *Automotive Mechatronics: Operational and Practical Issues*. Springer, 2011, pp. 73–74.

[123] FONTES, R. R., AFZAL, S., BRITO, S. H., SANTOS, M. A., AND ROTHENBERG, C. E. Mininet-wifi: Emulating software-defined wireless networks. In *2015 11th International Conference on Network and Service Management (CNSM)* (2015), IEEE, pp. 384–389.

[124] FOSTER, I., PRUDHOMME, A., KOSCHER, K., AND SAVAGE, S. Fast and vulnerable: A story of telematic failures. In *9th USENIX Workshop on Offensive Technologies (WOOT 15)* (2015).

[125] FRAIJI, Y., AZZOUZ, L. B., TROJET, W., AND SAIDANE, L. A. Cyber security issues of internet of electric vehicles. In *2018 IEEE Wireless Communications and Networking Conference (WCNC)* (2018), IEEE, pp. 1–6.

[126] FRANCILLON, A., DANEV, B., AND CAPKUN, S. Relay attacks on passive keyless entry and start systems in modern cars. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)* (2011), Eidgenössische Technische Hochschule Zürich, Department of Computer Science.

[127] FRANCIS, L., HANCKE, G., MAYES, K., AND MARKANTONAKIS, K. Practical NFC Peer-to-Peer Relay Attack Using Mobile Phones. In *Radio Frequency Identification: Security and Privacy Issues* (Berlin, Heidelberg, 2010), S. B. Ors Yalcin, Ed., Springer Berlin Heidelberg, pp. 35–49.

[128] FRANKE, U., AND WERNBERG, J. A survey of cyber security in the swedish manufacturing industry. In *2020 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)* (2020), IEEE, pp. 1–8.

[129] FREUND, Y., AND SCHAPIRE, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences 55*, 1 (1997), 119–139.

[130] FUCHS, A., KERN, D., KRAUSS, C., AND ZHDANOVA, M. HIP: HSM-based Identities for Plug-and-Charge. In *Proceedings of the 15th International Conference on Availability, Reliability and Security* (New York, NY, USA, aug 2020), ACM, pp. 1–6.

[131] FUGIGLANDO, U., MASSARO, E., SANTI, P., MILARDO, S., ABIDA, K., STAHLMANN, R., NETTER, F., AND RATTI, C. Driving behavior analysis through can bus data in an uncontrolled environment. *IEEE Transactions on Intelligent Transportation Systems 20*, 2 (2018), 737–748.

[132] FUGIGLANDO, U., SANTI, P., MILARDO, S., ABIDA, K., AND RATTI, C. Characterizing the" driver dna" through can bus data analysis. In *Proceedings of the 2nd ACM International Workshop on Smart, Autonomous, and Connected Vehicular Systems and Services* (2017), pp. 37–41.

[133] GAHR, B., LIU, S., KOCH, K., BARATA, F., DAHLINGER, A., RYDER, B., FLEISCH, E., AND WORTMANN, F. Driver identification via the steering wheel. *arXiv preprint arXiv:1909.03953* (2019).

[134] GAHR, B., RYDER, B., DAHLINGER, A., AND WORTMANN, F. Driver identification via brake pedal signals—a replication and advancement of existing techniques. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)* (2018), IEEE, pp. 1415–1420.

[135] GANGWAR, A., AND SAHU, S. A survey on anomaly and signature based intrusion detection system (ids). *International Journal of Engineering Research and Applications 4*, 4 (2014), 67–72.

[136] GAO, C., WANG, G., SHI, W., WANG, Z., AND CHEN, Y. Autonomous driving security: State of the art and challenges. *IEEE Internet of Things Journal 9*, 10 (2021), 7572–7595.

[137] GAO, Y., AL-SARAWI, S. F., AND ABBOTT, D. Physical unclonable functions. *Nature Electronics 3*, 2 (2020), 81–91.

[138] GARCIA, F. D., OSWALD, D. F., KASPER, T., AND PAVLIDÈS, P. Lock it and still lose it-on the (in) security of automotive remote keyless entry systems. In *USENIX security symposium* (2016), vol. 53.

[139] GAROFALAKI, Z., KOSMANOS, D., MOSCHOYIANNIS, S., KALLERGIS, D., AND DOULIGERIS, C. Electric vehicle charging: A survey on the security issues and challenges of the open charge point protocol (ocpp). *IEEE Communications Surveys & Tutorials* (2022).

[140] GENNARO, R., LYSYANSKAYA, A., MALKIN, T., MICALI, S., AND RABIN, T. Algorithmic tamper-proof (atp) security: Theoretical foundations for security against hardware tampering. In *Theory of Cryptography Conference* (2004), Springer, pp. 258–277.

[141] GHOLAMI, A., KIM, S., DONG, Z., YAO, Z., MAHONEY, M. W., AND KEUTZER, K. A survey of quantization methods for efficient neural network inference. In *Low-Power Computer Vision*. Chapman and Hall/CRC, 2022, pp. 291–326.

[142] GHOSAL, A., SAGONG, S. U., HALDER, S., SAHABANDU, K., CONTI, M., POOVENDRAN, R., AND BUSHNELL, L. Truck platoon security: State-of-the-art and road ahead. *Computer Networks 185* (2021), 107658.

[143] GIRALDO, J., SARKAR, E., CARDENAS, A. A., MANIATAKOS, M., AND KANTARCIOGLU, M. Security and privacy in cyber-physical systems: A survey of surveys. *IEEE Design & Test 34*, 4 (2017), 7–17.

[144] GIRMA, A., YAN, X., AND HOMAIFAR, A. Driver identification based on vehicle telematics data using lstm-recurrent neural network. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)* (2019), IEEE, pp. 894–902.

[145] GONG, X., WANG, Q., CHEN, Y., YANG, W., AND JIANG, X. Model extraction attacks and defenses on cloud-based machine learning models. *IEEE Communications Magazine 58*, 12 (2020), 83–89.

[146] GOODFELLOW, I., BENGIO, Y., AND COURVILLE, A. *Deep Learning*. MIT Press, Cambridge, MA, 2016.

[147] GORDON, P. Netherlands aims to ban conventionally-fueled vehicles by 2050. https://www.smart-energy.com/industry-sectors/electric-vehicles/netherlands-aims-to-ban-conventionally-fueled-vehicles-by-2050/, Feb. 2019. Accessed: Sept. 11, 2024.

[148] GOTTUMUKKALA, R., MERCHANT, R., TAUZIN, A., LEON, K., ROCHE, A., AND DARBY, P. Cyber-physical system security of vehicle charging stations. In *2019 IEEE Green Technologies Conference (GreenTech)* (2019), IEEE, pp. 1–5.

[149] GROVER, K., LIM, A., AND YANG, Q. Jamming and anti–jamming techniques in wireless networks: a survey. *International Journal of Ad Hoc and Ubiquitous Computing 17*, 4 (2014), 197–215.

[150] GROZA, B., AND MURVAY, P.-S. Security solutions for the controller area network: Bringing authentication to in-vehicle networks. *IEEE Vehicular Technology Magazine 13*, 1 (2018), 40–47.

[151] GROZA, B., MURVAY, S., VAN HERREWEGE, A., AND VERBAUWHEDE, I. Libra-can: A lightweight broadcast authentication protocol for controller area networks. In *Cryptology and Network Security: 11th International Conference, CANS 2012, Darmstadt, Germany, December 12-14, 2012. Proceedings 11* (2012), Springer, pp. 185–200.

[152] GUNASEKHAR, T., RAO, K. T., SAIKIRAN, P., AND LAKSHMI, P. A survey on denial of service attacks. *International Journal of Computer Science and Information Technologies 5*, 2 (2014), 2373–2376.

[153] GUO, G., WANG, H., BELL, D., BI, Y., AND GREER, K. Knn model-based approach in classification. In *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE* (Berlin, Heidelberg, 2003), R. Meersman, Z. Tari, and D. C. Schmidt, Eds., Springer Berlin Heidelberg, pp. 986–996.

[154] GUO, L., AND YE, J. Cyber-physical security of electric vehicles with four motor drives. *IEEE Transactions on Power Electronics 36*, 4 (2020), 4463–4477.

[155] GUO, L., YE, J., AND YANG, B. Cyberattack detection for electric vehicles using physics-guided machine learning. *IEEE Transactions on Transportation Electrification 7*, 3 (2020), 2010–2022.

[156] GUO, R., LU, L., OUYANG, M., AND FENG, X. Mechanism of the entire overdischarge process and overdischarge-induced internal short circuit in lithium-ion batteries. *Scientific reports 6*, 1 (2016), 1–9.

[157] HALABI, J., AND ARTAIL, H. A lightweight synchronous cryptographic hash chain solution to securing the vehicle can bus. In *2018 IEEE International Multidisciplinary Conference on Engineering Technology (IMCET)* (2018), IEEE, pp. 1–6.

[158] HALDER, S., GHOSAL, A., AND CONTI, M. Secure over-the-air software updates in connected vehicles: A survey. *Computer Networks 178* (2020), 107343.

[159] HALLAC, D., SHARANG, A., STAHLMANN, R., LAMPRECHT, A., HUBER, M., ROEHDER, M., LESKOVEC, J., ET AL. Driver identification using automobile sensor data from a single turn. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)* (2016), IEEE, pp. 953–958.

[160] HAMMERSCHMIDT, C. Number of automotive ecus continues to rise. https://www.eenewseurope.com/en/number-of-automotive-ecus-continues-to-rise/, May 2019. Accessed: Dec. 13, 2023.

[161] HAN, M. L., KWAK, B. I., AND KIM, H. K. Anomaly intrusion detection method for vehicular networks based on survival analysis. *Vehicular communications 14* (2018), 52–63.

[162] HANCKE, G. P., MAYES, K. E., AND MARKANTONAKIS, K. Confidence in smart token proximity: Relay attacks revisited. *Computers and Security 28*, 7 (2009), 615–627.

[163] HARRINGTON, P. *Machine Learning in Action*. Simon and Schuster, New York, NY, USA, 2012.

[164] HARTKOPP, O., REUBER, C., AND SCHILLING, R. Macan-message authenticated can. In *10th Int. Conf. on Embedded Security in Cars (ESCAR 2012)* (2012).

[165] HASSAN, M. U., REHMANI, M. H., AND CHEN, J. Differential privacy techniques for cyber physical systems: a survey. *IEEE Communications Surveys & Tutorials 22*, 1 (2019), 746–789.

[166] HEMMINGER, S. "bridge - show / manipulate bridge addresses and devices". https://man7.org/linux/man-pages/man8/bridge.8.html, 2012. Accessed: June 16, 2021.

[167] HENZL, M., HANACEK, P., AND KACIC, M. Preventing real-world relay attacks on contactless devices. *Proceedings - International Carnahan Conference on Security Technology*, October (2014), 13–18.

[168] HILL, K. Automakers are sharing consumers' driving behavior with insurance companies. www.nytimes.com/2024/03/11/technology/carmakers-driver-tracking-insurance.html, 2024. The New York Times. Accessed: Apr. 20, 2024.

[169] HÖFER, C., PETIT, J., SCHMIDT, R., AND KARGL, F. POPCORN: Privacy-preserving charging for emobility. *Proceedings of the ACM Conference on Computer and Communications Security* (2013), 37–48.

[170] HOSMER JR, D. W., LEMESHOW, S., AND STURDIVANT, R. X. *Applied logistic regression*, vol. 398. John Wiley & Sons, 2013.

[171] HOSSAIN, M. D., INOUE, H., OCHIAI, H., FALL, D., AND KADOBAYASHI, Y. Long short-term memory-based intrusion detection system for in-vehicle controller area network bus. In *IEEE Annual Computers, Software, and Applications Conference* (2020), pp. 10–17.

[172] HOUSER, R. L., KEMPTON, W., MCGEE, R., KIAMILEV, F., AND WAITE, N. Ev fingerprinting. Tech. rep., SAE Technical Paper, 2017.

[173] HOWARD, J. D., AND LONGSTAFF, T. A. A common language for computer security incidents. Tech. rep., Sandia National Lab. (SNL-NM), 1998.

[174] HUANG, X., XU, C., WANG, P., AND LIU, H. Lnsc: A security model for electric vehicle and charging pile management based on blockchain ecosystem. *IEEE access 6* (2018), 13565–13574.

[175] HUMAYED, A., LIN, J., LI, F., AND LUO, B. Cyber-physical systems security—a survey. *IEEE Internet of Things Journal 4*, 6 (Dec. 2017), 1802–1831.

[176] HWANG, H., JUNG, G., SOHN, K., AND PARK, S. A study on mitm (man in the middle) vulnerability in wireless network using 802.1 x and eap. In *2008 International Conference on Information Science and Security (ICISS 2008)* (2008), IEEE, pp. 164–170.

[177] IEEE. IEEE Standard Technical Specifications of a DC Quick Charger for Use with Electric Vehicles. *IEEE Std 2030.1.1-2015* (2016), 1–97.

[178] IEHIRA, K., INOUE, H., AND ISHIDA, K. Spoofing attack using bus-off attacks against a specific ecu of the can bus. In *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)* (2018), IEEE, pp. 1–4.

[179] INFLUX TECHNOLOGY. Understanding CAN DBC. www.influxtechnology.com/post/understanding-can-dbc, 2021. Accessed: Aug. 23, 2023.

[180] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO). Road vehicles — Vehicle-to-Grid Communication Interface — Part 2: Network and application protocol requirements. Standard, International Organization for Standardization (ISO), Geneva, CH, Mar. 2014.

[181] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO). ISO 11898:2015: Road vehicles — Controller area network (CAN). Standard, International Organization for Standardization (ISO), Geneva, CH, Dec. 2015.

[182] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO). Road vehicles — Local Interconnect Network (LIN) — Part 8: Electrical physical layer (EPL) specification: LIN over DC powerline (DC-LIN). Standard, International Organization for Standardization (ISO), Mar. 2019.

[183] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO). Road vehicles — Vehicle-to-Grid Communication Interface — Part 1: General information and use-case definition. Standard, International Organization for Standardization (ISO), Geneva, CH, Mar. 2019.

[184] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO). Road vehicles — Vehicle to grid communication interface — Part 20: 2nd generation network layer and application layer requirements. Standard, International Organization for Standardization (ISO), Geneva, CH, 2021.

[185] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO). ISO 15118-8:2020: Road vehicles — vehicle to grid communication interface — part 8: Physical layer and data link layer requirements for wireless communication. Standard, International Organization for Standardization (ISO), Sept. 2022.

[186] IQBAL, S., HAQUE, A., AND ZULKERNINE, M. Towards a security architecture for protecting connected vehicles from malware. In *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)* (2019), IEEE, pp. 1–5.

[187] JAFARNEJAD, S., CASTIGNANI, G., AND ENGEL, T. Towards a real-time driver identification mechanism based on driving sensing data. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)* (2017), IEEE, pp. 1–7.

[188] JIANG, L., XIE, S., MAHARJAN, S., AND ZHANG, Y. Blockchain empowered wireless power transfer for green and secure internet of things. *IEEE Network 33*, 6 (2019), 164–171.

[189] JIN, S., CHUNG, J.-G., AND XU, Y. Signature-based intrusion detection system (ids) for in-vehicle can bus network. In *2021 IEEE International Symposium on Circuits and Systems (ISCAS)* (2021), IEEE, pp. 1–5.

[190] JO, H. J., AND CHOI, W. A survey of attacks on controller area networks and corresponding countermeasures. *IEEE Transactions on Intelligent Transportation Systems* (2021).

[191] JURDAK, R., DORRI, A., AND VILATHGAMUWA, M. A trusted and privacy-preserving internet of mobile energy. *IEEE Communications Magazine 59*, 6 (2021), 89–95.

[192] KALOGRIDIS, G., EFTHYMIOU, C., DENIC, S. Z., LEWIS, T. A., AND CEPEDA, R. Privacy for smart meters: Towards undetectable appliance load signatures. In *2010 First IEEE International Conference on Smart Grid Communications* (2010), pp. 232–237.

[193] KANG, H., KWAK, B. I., LEE, Y. H., LEE, H., LEE, H., AND KIM, H. K. Car hacking and defense competition on in-vehicle network. In *Workshop on Automotive and Autonomous Vehicle Security* (2021).

[194] KANG, Y. G., PARK, K. H., AND KIM, H. K. Automobile theft detection by clustering owner driver data. *arXiv preprint arXiv:1909.08929* (2019).

[195] KARTHIK, T., BROWN, A., AWWAD, S., MCCOY, D., BIELAWSKI, R., MOTT, C., LAUZON, S., WEIMERSKIRCH, A., AND CAPPOS, J. Uptane: Securing software updates for automobiles. In *International Conference on Embedded Security in Car* (2016), pp. 1–11.

[196] KEEN SECURITY LAB OF TENCENT. Car Hacking Research: Remote Attack Tesla Motors. https://keenlab.tencent.com/en/2016/09/19/Keen-Security-Lab-of-Tencent-Car-Hacking-Research-Remote-Attack-to-Tesla-Cars/, 2016. Accessed: Aug. 02, 2024.

[197] KEMPTON, W., TOMIC, J., LETENDRE, S., BROOKS, A., AND LIPMAN, T. 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).

[198] KETKAR, N. *Introduction to Keras*. Apress, Berkeley, CA, 2017, pp. 97–111.

[199] KHALID, A., SUNDARARAJAN, A., HERNANDEZ, A., AND SARWAT, A. I. Facts approach to address cybersecurity issues in electric vehicle battery systems. In *2019 IEEE Technology & Engineering Management Conference (TEMSCON)* (2019), IEEE, pp. 1–6.

[200] KHAN, A. K., AND MAHANTA, H. J. Side channel attacks and their mitigation techniques. In *2014 First International Conference on Automation, Control, Energy and Systems (ACES)* (2014), IEEE, pp. 1–4.

[201] KHAN, M. A. Challenges facing the application of iot in medicine and healthcare. *International Journal of Computations, Information and Manufacturing (IJCIM) 1*, 1 (2021).

[202] KHANDELWAL, S., AND SHREEJITH, S. A lightweight multi-attack can intrusion detection system on hybrid fpgas. In *International Conference on Field-Programmable Logic and Applications* (2022).

[203] KHANJANI, Z., WATSON, G., AND JANEJA, V. P. How deep are the fakes? focusing on audio deepfake: A survey. *arXiv preprint arXiv:2111.14203* (2021).

[204] KILEY, P. Reverse Engineering the Tesla Battery Management System to Increase Power Available. *Blackhat* (2020), 28.

[205] KIM, T., OCHOA, J., FAIKA, T., MANTOOTH, A., DI, J., LI, Q., AND LEE, Y. An overview of cyber-physical security of battery management systems and adoption of blockchain technology. *IEEE Journal of Emerging and Selected Topics in Power Electronics* (2020).

[206] KINGMA, D. P., AND BA, J. Adam: A method for stochastic optimization, 2017.

[207] KINNEY, S. L. *Trusted platform module basics: using TPM in embedded systems*. Elsevier, 2006.

[208] KLAPWIJK, P., AND DRIESSEN-MUTTERS, L. Exploring the public key infrastructure for ISO 15118 in the EV charging ecosystem. Tech. rep., ElaadNL, 2018.

[209] KÖHLER, S., BAKER, R., STROHMEIER, M., AND MARTINOVIC, I. Brokenwire: Wireless disruption of ccs electric vehicle charging. *arXiv preprint arXiv:2202.02104* (2022).

[210] KORAK, T., AND HUTTER, M. On the power of active relay attacks using custom-made proxies. *2014 IEEE International Conference on RFID* (2014), 126–133.

[211] KOSCHER, K., SAVAGE, S., ROESNER, F., PATEL, S., KOHNO, T., CZESKIS, A., MCCOY, D., KANTOR, B., ANDERSON, D., SHACHAM, H., ET AL. Experimental security analysis of a modern automobile. In *2010 IEEE Symposium on Security and Privacy* (2010), IEEE Computer Society, pp. 447–462.

[212] KROEZE, R. C., AND KREIN, P. T. Electrical battery model for use in dynamic electric vehicle simulations. In *2008 IEEE Power Electronics Specialists Conference* (2008), pp. 1336–1342.

[213] KRUEGEL, C., AND TOTH, T. Using decision trees to improve signature-based intrusion detection. In *International workshop on recent advances in intrusion detection* (2003), Springer, pp. 173–191.

[214] KT SECURE. Software Code Signing. https://ktsecure.co.uk/services/software-code-signing/, 2023. Accessed: Jan. 15, 2024.

[215] KUERBAN, M., TIAN, Y., YANG, Q., JIA, Y., HUEBERT, B., AND POSS, D. Flowsec: Dos attack mitigation strategy on sdn controller. In *2016 IEEE International Conference on Networking, Architecture and Storage (NAS)* (2016), IEEE, pp. 1–2.

[216] KULANDAIVEL, S., JAIN, S., GUAJARDO, J., AND SEKAR, V. Cannon: Reliable and stealthy remote shutdown attacks via unaltered automotive microcontrollers. In *2021 IEEE Symposium on Security and Privacy (SP)* (2021), IEEE, pp. 195–210.

[217] KWAK, B. I., WOO, J., AND KIM, H. K. Know your master: Driver profiling-based anti-theft method. In *2016 14th Annual Conference on Privacy, Security and Trust (PST)* (2016), IEEE, pp. 211–218.

[218] La Cour, A. S., Afridi, K. K., and Suh, G. E. Wireless charging power side-channel attacks. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security* (2021), pp. 651–665.

[219] Langner, R. Stuxnet: Dissecting a cyberwarfare weapon. *IEEE Security & Privacy 9*, 3 (2011), 49–51.

[220] Lee, C.-H., and Yang, H.-C. A privacy-preserving learning method for analyzing hev driver's driving behaviors. *IEEE Access* (2023).

[221] Lee, J.-H., Lim, S., and Ahn, C. W. Automotive ecu data-based driver's propensity learning using evolutionary random forest. *IEEE Access 7* (2019), 51899–51906.

[222] Lee, S., Park, Y., Lim, H., and Shon, T. 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* (2014), 6–9.

[223] Lee, Z. J., Lee, G., Lee, T., Jin, C., Lee, R., Low, Z., Chang, D., Ortega, C., and Low, S. H. Adaptive charging networks: A framework for smart electric vehicle charging. *IEEE Transactions on Smart Grid 12*, 5 (2021), 4339–4350.

[224] Lee, Z. J., Li, T., and Low, S. H. Acn-data: Analysis and applications of an open ev charging dataset. In *Proceedings of the Tenth ACM International Conference on Future Energy Systems* (New York, NY, USA, 2019), e-Energy '19, Association for Computing Machinery, p. 139–149.

[225] Leu, P., Camurati, G., Heinrich, A., Roeschlin, M., Anliker, C., Hollick, M., Capkun, S., and Classen, J. Ghost peak: Practical distance reduction attacks against {HRP}{UWB} ranging. In *31st USENIX Security Symposium (USENIX Security 22)* (2022), pp. 1343–1359.

[226] Li, H., He, Y., Sun, L., Cheng, X., and Yu, J. Side-channel information leakage of encrypted video stream in video surveillance systems. In *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications* (2016), IEEE, pp. 1–9.

[227] Li, Y., Wang, Y., Wu, M., and Li, H. Replay Attack and Defense of Electric Vehicle Charging on GB/T 27930-2015 Communication Protocol. *Journal of Computer and Communications 07*, 12 (2019), 20–30.

[228] Liao, H.-J., Lin, C.-H. R., Lin, Y.-C., and Tung, K.-Y. Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications 36*, 1 (2013), 16–24.

[229] Lin, C.-W., and Sangiovanni-Vincentelli, A. Cyber-security for the controller area network (CAN) communication protocol. In *2012 International Conference on Cyber Security* (2012), IEEE, pp. 1–7.

[230] Lin, X., Zhang, K., Cao, W., and Zhang, L. Driver evaluation and identification based on driving behavior data. In *2018 5th International Conference on Information Science and Control Engineering (ICISCE)* (2018), IEEE, pp. 718–722.

[231] LIU, H., SPOLAOR, R., TURRIN, F., BONAFEDE, R., AND CONTI, M. Usb powered devices: A survey of side-channel threats and countermeasures. *High-Confidence Computing 1*, 1 (2021), 100007.

[232] LIU, Q., YILDIRIM, K. S., PAWEŁCZAK, P., AND WARNIER, M. Safe and secure wireless power transfer networks: Challenges and opportunities in RF-based systems. *IEEE Communications Magazine 54*, 9 (2016), 74–79.

[233] LIU, W., WEN, Y., YU, Z., AND YANG, M. Large-margin softmax loss for convolutional neural networks. *arXiv preprint arXiv:1612.02295* (2016).

[234] LIU, X.-Y., FANG, Y., YANG, L., LI, Z., AND WALID, A. High-performance tensor decompositions for compressing and accelerating deep neural networks. In *Tensors for Data Processing*, Y. Liu, Ed. Academic Press, 2022, pp. 293–340.

[235] LOKMAN, S.-F., OTHMAN, A. T., AND ABU-BAKAR, M.-H. Intrusion detection system for automotive controller area network (can) bus system: a review. *EURASIP Journal on Wireless Communications and Networking 2019* (2019), 1–17.

[236] LOTTO, A., MARCHIORI, F., BRIGHENTE, A., AND CONTI, M. A survey and comparative analysis of security properties of can authentication protocols. *IEEE Communications Surveys & Tutorials* (2024), 1–1.

[237] LUNDBERG, S. M., AND LEE, S.-I. A unified approach to interpreting model predictions. *Advances in neural information processing systems 30* (2017).

[238] MACHURA, P., DE SANTIS, V., AND LI, Q. Driving range of electric vehicles charged by wireless power transfer. *IEEE Transactions on Vehicular Technology 69*, 6 (2020), 5968–5982.

[239] MADHANI, A., AND KRISHER, T. Biden Pushes Electric Vehicle Chargers as Energy Costs Spike. https://www.usnews.com/news/business/articles/2021-11-17/biden-pushes-electric-vehicle-chargers-as-energy-costs-spike, Nov. 2021. Accessed: June 05, 2024.

[240] MAGGI, F. A vulnerability in modern automotive standards and how we exploited it. *Trend Micro* (2017).

[241] MAHALINGAM, M., DUTT, D. G., DUDA, K., AGARWAL, P., KREEGER, L., SRIDHAR, T., BURSELL, M., AND WRIGHT, C. Virtual extensible local area network (vxlan): A framework for overlaying virtualized layer 2 networks over layer 3 networks. *RFC 7348* (2014), 1–22.

[242] MAKOWITZ, R., AND TEMPLE, C. Flexray-a communication network for automotive control systems. In *2006 IEEE International Workshop on Factory Communication Systems* (2006), IEEE, pp. 207–212.

[243] MARCHEGIANI, L., AND POSNER, I. Long-term driving behaviour modelling for driver identification. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)* (2018), IEEE, pp. 913–919.

[244] MARCHETTI, M., AND STABILI, D. Anomaly detection of can bus messages through analysis of id sequences. In *2017 IEEE Intelligent Vehicles Symposium (IV)* (2017), IEEE, pp. 1577–1583.

[245] MARCHIORI, F., AND CONTI, M. Your battery is a blast! safeguarding against counterfeit batteries with authentication. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security* (2023), pp. 105–119.

[246] MARCUS TAN, Y. X., IACOVAZZI, A., HOMOLIAK, I., ELOVICI, Y., AND BINDER, A. Adversarial attacks on remote user authentication using behavioural mouse dynamics. In *2019 International Joint Conference on Neural Networks (IJCNN)* (2019), pp. 1–10.

[247] MARRA, F., YANG, G. Y., TRÆHOLT, C., LARSEN, E., RASMUSSEN, C. N., AND YOU, S. Demand profile study of battery electric vehicle under different charging options. In *2012 IEEE Power and Energy Society General Meeting* (2012), pp. 1–7.

[248] MARRONE, S., AND SANSONE, C. Adversarial perturbations against fingerprint based authentication systems. In *2019 International Conference on Biometrics (ICB)* (2019), pp. 1–6.

[249] MARTINELLI, F., MERCALDO, F., ORLANDO, A., NARDONE, V., SANTONE, A., AND SANGAIAH, A. K. Human behavior characterization for driving style recognition in vehicle system. *Computers & Electrical Engineering 83* (2020), 102504.

[250] MATSUMOTO, T., HATA, M., TANABE, M., YOSHIOKA, K., AND OISHI, K. A method of preventing unauthorized data transmission in controller area network. In *2012 IEEE 75th Vehicular Technology Conference (VTC Spring)* (2012), IEEE, pp. 1–5.

[251] MEIRING, G. A. M., AND MYBURGH, H. C. A Review of Intelligent Driving Style Analysis Systems and Related Artificial Intelligence Algorithms. *Sensors 15*, 12 (Dec. 2015), 30653–30682. Number: 12 Publisher: Multidisciplinary Digital Publishing Institute.

[252] MICROCHIP. *SN65HVD230: 3.3 V CAN Transceiver with Standby Mode*, 4 2018. Rev. 0.

[253] MICROCHIP. *MCP2515: Stand-Alone CAN Controller with SPI Interface*, 4 2021. Rev. K.

[254] MILLER, C., AND VALASEK, C. Adventures in automotive networks and control units. *Def Con 21*, 260-264 (2013), 15–31.

[255] MILLER, C., AND VALASEK, C. Remote exploitation of an unaltered passenger vehicle. *Black Hat USA 2015*, S 91 (2015), 1–91.

[256] MINAWI, O., WHELAN, J., ALMEHMADI, A., AND EL-KHATIB, K. Machine learning-based intrusion detection system for controller area networks. In *Proc. ACM Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications* (2020), p. 41–47.

[257] MINAWI, O., WHELAN, J., ALMEHMADI, A., AND EL-KHATIB, K. Machine learning-based intrusion detection system for controller area networks. In *Proceedings of the 10th ACM Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications* (2020), pp. 41–47.

[258] MITCHELL, R., AND CHEN, I.-R. A survey of intrusion detection techniques for cyber-physical systems. *ACM Computing Surveys (CSUR) 46*, 4 (2014), 1–29.

[259] MIYAJIMA, C., NISHIWAKI, Y., OZAWA, K., WAKITA, T., ITOU, K., TAKEDA, K., AND ITAKURA, F. Driver modeling based on driving behavior and its evaluation in driver identification. *Proceedings of the IEEE 95*, 2 (2007), 427–437.

[260] MOLINA-MARKHAM, A., SHENOY, P., FU, K., CECCHET, E., AND IRWIN, D. 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, 2010), Association for Computing Machinery, p. 61–66.

[261] MÖRCHEN, F. Time series feature extraction for data mining using dwt and dft, 2003.

[262] MOTALLEBIGHOMI, M., FERNANDES, E., AND RANGANATHAN, A. {MakeShift}: Security analysis of shimano di2 wireless gear shifting in bicycles. In *18th USENIX WOOT Conference on Offensive Technologies (WOOT 24)* (2024), pp. 75–88.

[263] MUSTAFA, M. A., ZHANG, N., KALOGRIDIS, G., AND FAN, Z. Smart electric vehicle charging: Security analysis. *2013 IEEE PES Innovative Smart Grid Technologies Conference, ISGT 2013*, February (2013), 7.

[264] MÜLTIN, M. Iso 15118 as the enabler of vehicle-to-grid applications. In *2018 International Conference of Electrical and Electronic Technologies for Automotive* (2018), pp. 1–6.

[265] NASH, D. C., MARTIN, T. L., HA, D. S., AND HSIAO, M. S. Towards an intrusion detection system for battery exhaustion attacks on mobile computing devices. In *Third IEEE international conference on pervasive computing and communications workshops* (2005), IEEE, pp. 141–145.

[266] NASR, T., TORABI, S., BOU-HARB, E., FACHKHA, C., AND ASSI, C. Power jacking your station: In-depth security analysis of electric vehicle charging station management systems. *Computers & Security 112* (2022), 102511.

[267] NASR, T., TORABI, S., BOU-HARB, E., FACHKHA, C., AND ASSI, C. Chargeprint: A framework for internet-scale discovery and security analysis of ev charging management systems. In *NDSS* (2023).

[268] NILSSON, D. K., PHUNG, P. H., AND LARSON, U. E. Vehicle ecu classification based on safety-security characteristics. In *IET Road Transport Information and Control-RTIC 2008 and ITS United Kingdom Members' Conference* (2008), IET, pp. 1–7.

[269] NOEL, L., ZARAZUA DE RUBENS, G., KESTER, J., AND SOVACOOL, B. K. The Technical Challenges to V2G. In *Vehicle-to-Grid*. Springer International Publishing, Cham, 2019, pp. 65–89.

[270] 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 4*, 1 (2015), 246–253.

[271] NÜRNBERGER, S., AND ROSSOW, C. –vatican–vetted, authenticated can bus. In *Cryptographic Hardware and Embedded Systems–CHES 2016: 18th International Conference, Santa Barbara, CA, USA, August 17-19, 2016, Proceedings 18* (2016), Springer, pp. 106–124.

[272] NXP. *Secure TJA115x CAN Transceiver Family*, 1 2020. Rev. 3.

[273] PALANCA, A., EVENCHICK, E., MAGGI, F., AND ZANERO, S. A stealth, selective, link-layer denial-of-service attack against automotive networks. In *Detection of Intrusions and Malware, and Vulnerability Assessment: 14th International Conference, DIMVA 2017, Bonn, Germany, July 6-7, 2017, Proceedings 14* (2017), Springer, pp. 185–206.

[274] PARK, K. H., AND KIM, H. K. This car is mine!: Automobile theft countermeasure leveraging driver identification with generative adversarial networks. *arXiv preprint arXiv:1911.09870* (2019).

[275] PARK, S. B., JO, H. J., AND LEE, D. H. Flooding attack mitigator for in-vehicle can using fault confinement in can protocol. *Computers & Security 126* (2023), 103091.

[276] PAUL, L. Y., BARAS, J. S., AND SADLER, B. M. Physical-layer authentication. *IEEE Transactions on Information Forensics and Security 3*, 1 (2008), 38–51.

[277] PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A., MICHEL, V., THIRION, B., GRISEL, O., BLONDEL, M., PRETTENHOFER, P., WEISS, R., DUBOURG, V., VANDERPLAS, J., PASSOS, A., COURNAPEAU, D., BRUCHER, M., PERROT, M., AND DUCHESNAY, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research 12* (2011), 2825–2830.

[278] PEDREIRA, V., BARROS, D., AND PINTO, P. A review of attacks, vulnerabilities, and defenses in industry 4.0 with new challenges on data sovereignty ahead. *Sensors 21*, 15 (2021), 5189.

[279] PELLETIER, S., JABALI, O., LAPORTE, G., AND VENERONI, M. Battery degradation and behaviour for electric vehicles: Review and numerical analyses of several models. *Transportation Research Part B: Methodological 103* (2017), 158–187.

[280] PESÉ, M. D., STACER, T., CAMPOS, C. A., NEWBERRY, E., CHEN, D., AND SHIN, K. G. Librecan: Automated can message translator. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security* (2019), pp. 2283–2300.

[281] PHIL-EQTECH. CH-Workshop. https://github.com/phil-eqtech/CH-Workshop, 2020. Accessed: Dec. 13, 2023.

[282] PIĘTAK, A., AND MIKULSKI, M. On the adaptation of can bus network for use in the ship electronic systems. *Polish Maritime Research 16*, 4 (2009), 62–69.

[283] PINTO, J., MONTEIRO, V., GONÇALVES, H., EXPOSTO, B., PEDROSA, D., COUTO, C., AND AFONSO, J. L. Bidirectional battery charger with grid-to-vehicle, vehicle-to-grid and vehicle-to-home technologies. In *IECON 2013-39th Annual Conference of the IEEE Industrial Electronics Society* (2013), IEEE, pp. 5934–5939.

[284] POSITIVE_ANT583. Entering ecu boot mode and dumping firmware via can. https://www.reddit.com/r/CarHacking/comments/14dxxsr/entering_ecu_boot_mode_and_dumping_firmware_via/, 2023. Reddit. Accessed: Sept. 05, 2024.

[285] PRATT, R. M., AND CARROLL, T. E. Vehicle charging infrastructure security. In *2019 IEEE International Conference on Consumer Electronics (ICCE)* (2019), IEEE, pp. 1–5.

[286] PROJECT, L. F. Automotive grade linux: a fully open software stack for the connected car, with linux at its core. https://www.automotivelinux.org/, 2023.

[287] PUNDE, A. Understanding risks in over the air firmware upgrade for automotives including evs. https://www.einfochips.com/blog/understanding-risks-in-over-the-air-firmware-upgrade-for-automotives-including-evs/, January 2023. Accessed: Sept. 05, 2024.

[288] RAHIM, M. A., ZHU, L., LI, X., LIU, J., ZHANG, Z., QIN, Z., KHAN, S., AND GAI, K. Zero-to-stable driver identification: A non-intrusive and scalable driver identification scheme. *IEEE transactions on vehicular technology 69*, 1 (2019), 163–171.

[289] RAMEY, J. "Honda Will Go Electric- and Fuel Cell-Only by 2040". https://www.autoweek.com/news/green-cars/a36230978/honda-electric-and-fuel-cell-by-2040/, 2021. Accessed: May 05, 2021.

[290] RAMOS, J., ET AL. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning* (2003), vol. 242, Citeseer, pp. 29–48.

[291] RASPBERRY PI FOUNDATION. Raspberry Pi: Putting the power of computing and digital making into the hands of people all over the world. https://www.raspberrypi.org/, February 2023.

[292] RAVI, C., TIGGA, A., REDDY, G. T., HAKAK, S., AND ALAZAB, M. Driver identification using optimized deep learning model in smart transportation. *ACM Transactions on Internet Technology 22*, 4 (2022), 1–17.

[293] RAVI, S. S., AND AZIZ, M. Utilization of electric vehicles for vehicle-to-grid services: progress and perspectives. *Energies 15*, 2 (2022), 589.

[294] RIVEST, R. L., SHAMIR, A., AND WAGNER, D. A. Time-lock puzzles and timed-release crypto. *LCS Technical Reports (1974 - 2003)* (1996).

[295] RIZZO, N., SPRISSLER, E., HONG, Y., AND GOEL, S. Privacy preserving driving style recognition. In *2015 International Conference on Connected Vehicles and Expo (ICCVE)* (2015), IEEE, pp. 232–237.

[296] ROMAN, L. F., AND GONDIM, P. R. Authentication protocol in ctns for a cwd-wpt charging system in a cloud environment. *Ad Hoc Networks 97* (2020), 102004.

[297] ROSCHER, R., BOHN, B., DUARTE, M. F., AND GARCKE, J. Explainable machine learning for scientific insights and discoveries. *Ieee Access 8* (2020), 42200–42216.

[298] ROTH, C., THANH DINH, N., AND KESDOĞAN, D. 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.

[299] RUBIO, J. E., ALCARAZ, C., AND LOPEZ, J. Recommender system for privacy-preserving solutions in smart metering. *Pervasive and Mobile Computing 41* (2017), 205–218.

[300] RUGHOOBUR, P., AND NAGOWAH, L. A lightweight replay attack detection framework for battery depended iot devices designed for healthcare. In *2017 International Conference on Infocom Technologies and Unmanned Systems (Trends and Future Directions)(ICTUS)* (2017), IEEE, pp. 811–817.

[301] SAGONG, S. U., YING, X., POOVENDRAN, R., AND BUSHNELL, L. Exploring attack surfaces of voltage-based intrusion detection systems in controller area networks. In *Proc. ESCAR Eur.* (2018), pp. 1–13.

[302] SAHABANDU, D., MERTOGUNO, S., AND POOVENDRAN, R. A natural language processing approach for instruction set architecture identification. *IEEE Trans. Information Forensics and Security* (2023).

[303] SALDAÑA, G., SAN MARTÍN, J. I., ZAMORA, I., ASENSIO, F. J., AND OÑEDERRA, O. Analysis of the current electric battery models for electric vehicle simulation. *Energies 12*, 14 (2019), 2750.

[304] SALTAFORMAGGIO, B., CHOI, H., JOHNSON, K., KWON, Y., ZHANG, Q., ZHANG, X., XU, D., AND QIAN, J. Eavesdropping on fine-grained user activities within smartphone apps over encrypted network traffic. In *10th USENIX Workshop on Offensive Technologies (WOOT 16)* (2016).

[305] SANI, A. S., YUAN, D., BERTINO, E., AND DONG, Z. Y. Crypto-chain: A relay resilience framework for smart vehicles. In *Annual Computer Security Applications Conference* (New York, NY, USA, 2021), ACSAC, ACM, p. 439–454.

[306] SARIEDDINE, K., SAYED, M. A., TORABI, S., ATALLAH, R., AND ASSI, C. Investigating the security of ev charging mobile applications as an attack surface. *ACM Transactions on Cyber-Physical Systems 7*, 4 (2023), 1–28.

[307] SARIEDDINE, K., SAYED, M. A., TORABI, S., ATALLAH, R., AND ASSI, C. Edge-based detection and localization of adversarial oscillatory load attacks orchestrated by compromised ev charging stations. *International Journal of Electrical Power & Energy Systems 156* (2024), 109735.

[308] SCALAS, M., AND GIACINTO, G. Automotive cybersecurity: Foundations for next-generation vehicles. In *2019 2nd International Conference on new Trends in Computing Sciences (ICTCS)* (2019), IEEE, pp. 1–6.

[309] SCHELL, O., AND KNEIB, M. Valid: Voltage-based lightweight intrusion detection for the controller area network. In *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)* (2020), IEEE, pp. 225–232.

[310] SCHMIDHUBER, J. Deep learning in neural networks: An overview. *Neural Networks 61* (Jan 2015), 85–117.

[311] SCHMITTNER, C., AND MACHER, G. Automotive cybersecurity standards-relation and overview. In *International Conference on Computer Safety, Reliability, and Security* (2019), Springer, pp. 153–165.

[312] SCHNEEGASS, S., PFLEGING, B., BROY, N., HEINRICH, F., AND SCHMIDT, A. A data set of real world driving to assess driver workload. In *Proceedings of the 5th international conference on automotive user interfaces and interactive vehicular applications* (2013), pp. 150–157.

[313] SCHREIBER, T., AND SCHMITZ, A. Discrimination power of measures for non-linearity in a time series. *Phys. Rev. E 55* (May 1997), 5443–5447.

[314] SEEEDSTUDIO. CAN-BUS Shield V2 - high-performance MCP2515 controller & MCP2551 transceiver. https://www.seeedstudio.com/CAN-BUS-Shield-V2.html, February 2023.

[315] SEO, E., SONG, H. M., AND KIM, H. K. Gids: Gan based intrusion detection system for in-vehicle network. In *Annual Conference on Privacy, Security and Trust (PST)* (2018), IEEE, pp. 1–6.

[316] SERAG, K., BHATIA, R., FAQIH, A., OZMEN, M. O., KUMAR, V., CELIK, Z. B., AND XU, D. ZBCAN: A zero-byte can defense system. In *32nd USENIX Security Symposium (USENIX Security 23)* (2023), pp. 6893–6910.

[317] SERAG, K., BHATIA, R., KUMAR, V., CELIK, Z. B., AND XU, D. Exposing new vulnerabilities of error handling mechanism in can. In *30th USENIX Security Symposium (USENIX Security 21)* (2021), pp. 4241–4258.

[318] SERAG, K., KUMAR, V., CELIK, Z. B., BHATIA, R., PAYER, M., AND XU, D. Attacks on can error handling mechanism. In *International Workshop on Automotive and Autonomous Vehicle Security (AutoSec)* (2022).

[319] SHAHAN, Z. 16 Countries Now Over 10% Plugin Vehicle Share, 6 Over 20%. https://cleantechnica.com/2021/09/05/16-countries-now-over-10-plugin-vehicle-share-6-over-20/, 2021. Accessed: June 03, 2024.

[320] SHATERI, M., MESSINA, F., PIANTANIDA, P., AND LABEAU, F. Real-time privacy-preserving data release for smart meters. *IEEE Transactions on Smart Grid 11*, 6 (2020), 5174–5183.

[321] SHODAN. https://www.shodan.io/. Accessed: Sept. 05, 2024.

[322] SHOKRI, R., STRONATI, M., SONG, C., AND SHMATIKOV, V. Membership inference attacks against machine learning models, 2017.

[323] SHRIVASTAVA, S. Blackenergy-malware for cyber-physical attacks. *iTrust - Analysis Report 74* (2016), 115.

[324] SIBLINI, W., FRÉRY, J., HE-GUELTON, L., OBLÉ, F., AND WANG, Y.-Q. Master your metrics with calibration. In *International Symposium on Intelligent Data Analysis* (2020), Springer, pp. 457–469.

[325] SIDDIQUI, A. S., GUI, Y., PLUSQUELLIC, J., AND SAQIB, F. Secure communication over canbus. In *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)* (2017), IEEE, pp. 1264–1267.

[326] SINHA, S. State of iot 2024: Number of connected iot devices growing 13 https://iot-analytics.com/number-connected-iot-devices/, September 2024. Accessed: Sept. 09, 2024.

[327] SK PANG ELECTRONICS. PiCAN2 Duo CAN-Bus Board for Raspberry Pi. https://copperhilltech.com/pican2-duo-can-bus-board-for-raspberry-pi/, February 2023.

[328] SOCIETY OF AUTOMOTIVE ENGINEERS (SAE). SAE J1772: Electric Vehicle and Plug in Hybrid Electric Vehicle Conductive Charge Coupler. Standard, Society of Automotive Engineers (SAE), Oct. 2017.

[329] SOCIETY OF AUTOMOTIVE ENGINEERS (SAE). SAE J2464: Electric and Hybrid Electric Vehicle Rechargeable Energy Storage System (RESS) Safety and Abuse Testing. Standard, Society of Automotive Engineers (SAE), Aug. 2021.

[330] SOLANO, J., LOPEZ, C., RIVERA, E., CASTELBLANCO, A., TENGANA, L., AND OCHOA, M. Scrap: Synthetically composed replay attacks vs. adversarial machine learning attacks against mouse-based biometric authentication. In *Proceedings of the 13th ACM Workshop on Artificial Intelligence and Security* (New York, NY, USA, 2020), AISec'20, Association for Computing Machinery, p. 37–47.

[331] SONG, H. M., KIM, H. R., AND KIM, H. K. Intrusion detection system based on the analysis of time intervals of can messages for in-vehicle network. In *2016 international conference on information networking (ICOIN)* (2016), IEEE, pp. 63–68.

[332] SONG, H. M., WOO, J., AND KIM, H. K. In-vehicle network intrusion detection using deep convolutional neural network. *Vehicular Communications 21* (2020), 100198.

[333] SORTOMME, E., AND EL-SHARKAWI, M. A. Optimal charging strategies for unidirectional vehicle-to-grid. *IEEE Transactions on Smart Grid 2*, 1 (2011), 131–138.

[334] SRIPAD, S., KULANDAIVEL, S., PANDE, V., SEKAR, V., AND VISWANATHAN, V. Vulnerabilities of electric vehicle battery packs to cyberattacks. *arXiv preprint arXiv:1711.04822* (2017).

[335] SRIVASTAVA, N., HINTON, G., KRIZHEVSKY, A., SUTSKEVER, I., AND SALAKHUTDINOV, R. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research 15*, 1 (2014), 1929–1958.

[336] STANDFORD. Useful Plots to Diagnose Deep Neural Network. cs231n.github.io/neural-networks-3/. Accessed: Aug. 30, 2023.

[337] STMICROELECTRONICS. Stm32 nucleo-144 development board with stm32h743zi mcu. https://www.st.com/en/evaluation-tools/nucleo-h743zi.html/, February 2023.

[338] STMICROELECTRONICS. Interrupt overview - stm32mpu. https://wiki.st.com/stm32mpu/wiki/Interrupt_overview, 2024. Accessed: June 8, 2023.

[339] SUN, C., LI, T., LOW, S. H., AND LI, V. O. Classification of electric vehicle charging time series with selective clustering. *Electric Power Systems Research 189* (2020), 106695.

[340] SUN, Z., HAN, Y., WANG, Z., CHEN, Y., LIU, P., QIN, Z., ZHANG, Z., WU, Z., AND SONG, C. Detection of voltage fault in the battery system of electric vehicles using statistical analysis. *Applied Energy 307* (2022), 118172.

[341] SUSAN, S., AND KUMAR, A. The balancing trick: Optimized sampling of imbalanced datasets—a brief survey of the recent state of the art. *Engineering Reports 3*, 4 (2021), e12298.

[342] SUTCLIFFE, E. A look at electric car explosions. https://www.evfiresafe.com/post/electric-car-explosions. Accessed: Spet. 05, 2024.

[343] SUYKENS, J. A., AND VANDEWALLE, J. Least squares support vector machine classifiers. *Neural processing letters 9*, 3 (1999), 293–300.

[344] TAHA, B., SEHA, S. N. A., HWANG, D. Y., AND HATZINAKOS, D. Eyedrive: A deep learning model for continuous driver authentication. *IEEE Journal of Selected Topics in Signal Processing* (2023).

[345] TAKADA, M., OSADA, Y., AND MORII, M. Counter attack against the bus-off attack on can. In *2019 14th Asia Joint Conference on Information Security (AsiaJCIS)* (2019), IEEE, pp. 96–102.

[346] TAYLOR, A., JAPKOWICZ, N., AND LEBLANC, S. Frequency-based anomaly detection for the automotive can bus. In *2015 World Congress on Industrial Control Systems Security (WCICSS)* (2015), IEEE, pp. 45–49.

[347] TENSORFLOW. Tensorflow Lite. https://www.tensorflow.org/lite, 2023. Accessed: Aug. 30, 2023.

[348] TENSORFLOW. Tensorflow Lite C API. www.tensorflow.org/lite/api_docs/c, 2023. Accessed: Aug. 30, 2023.

[349] THAKKAR, A., AND LOHIYA, R. A review of the advancement in intrusion detection datasets. *Procedia Computer Science 167* (2020), 636–645.

[350] THAKKAR, A., AND LOHIYA, R. A review on machine learning and deep learning perspectives of ids for iot: recent updates, security issues, and challenges. *Archives of Computational Methods in Engineering 28*, 4 (2021), 3211–3243.

[351] THE GUARDIAN. "Electric vehicles on world's roads expected to increase to 145m by 2030". https://www.theguardian.com/environment/2021/apr/29/electric-vehicles-on-worlds-roads-expected-to-increase-to-145m-by-2030, 2021. Accessed: May 20, 2021.

[352] THOMPSON, S. *Application of controller area network bus and CANopen protocol in Industrial Automation*. PhD thesis, Murdoch University, 2018.

[353] THORPE, C., TOBIN, J., AND MURPHY, L. An ISO/IEC 7816-4 Application Layer Approach to Mitigate Relay Attacks on near Field Communication. *IEEE Access 8* (2020), 190108–190117.

[354] THUNDERSKY. Instruction manual for LFP/LCP/LMP lithium power battery. Tech. rep., Thunder Sky, 2007.

[355] TINDELL, K. CAN Bus Security - Attacks on CAN bus and their mitigations. *Canis Labs White Paper* (2019).

[356] TINDELL, K. Can injection: keyless car theft. https://kentindell.github.io/2023/04/03/can-injection/, April 2023. Accessed: Sept. 05, 2023.

[357] TIPPENHAUER, N. O., LUECKEN, H., KUHN, M., AND CAPKUN, S. 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).

[358] TROEPFER, C. SAE electric vehicle conductive charge coupler, SAE J1772. Tech. rep., Society of Automotive Engineers, 2009.

[359] TRSTENJAK, B., MIKAC, S., AND DONKO, D. Knn with tf-idf based framework for text categorization. *Procedia Engineering 69* (2014), 1356–1364.

[360] TSENG, C.-Y., BALASUBRAMANYAM, P., KO, C., LIMPRASITTIPORN, R., ROWE, J., AND LEVITT, K. A specification-based intrusion detection system for aodv. In *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks* (2003), pp. 125–134.

[361] TSENG, P.-Y., LIN, P.-C., AND KRISTIANTO, E. Vehicle theft detection by generative adversarial networks on driving behavior. *Engineering Applications of Artificial Intelligence 117* (2023), 105571.

[362] UN-NOOR, F., PADMANABAN, S., MIHET-POPA, L., MOLLAH, M. N., AND HOSSAIN, E. A comprehensive study of key electric vehicle (EV) components, technologies, challenges, impacts, and future direction of development. *Energies 10*, 8 (2017), 1217.

[363] UNAL, C., YIRIK, E., ÜNAL, E., CUMA, M., ONUR, B., AND TÜMAY, M. A review of charging technologies for commercial electric vehicles. *International Journal Of Advances On Automotive And Technology* (01 2018), 61–70.

[364] U.S.D.O.E. EV everywhere grand challenge: Road to success. Tech. rep., U. S. Department of Energy, Jan. 2014.

[365] VALENZUELA, J., WANG, J., AND BISSINGER, N. Real-time intrusion detection in power system operations. *IEEE Transactions on Power Systems 28*, 2 (2012), 1052–1062.

[366] VAN AUBEL, P., AND POLL, E. Security of ev-charging protocols. *arXiv preprint arXiv:2202.04631* (2022).

[367] VAN DEN BOSSCHE, P. *Electric Vehicle Charging Infrastructure*. Elsevier B.V, 2010.

[368] VAN DER MAATEN, L. Barnes-hut-sne. *arXiv preprint arXiv:1301.3342* (2013).

[369] VAN DER MAATEN, L., AND HINTON, G. Visualizing data using t-sne. *Journal of machine learning research 9*, 11 (2008).

[370] VERMA, M. E., IANNACONE, M. D., BRIDGES, R. A., HOLLIFIELD, S. C., MORIANO, P., KAY, B., AND COMBS, F. L. Addressing the lack of comparability & testing in can intrusion detection research: A comprehensive guide to can ids data & introduction of the road dataset. *arXiv preprint arXiv:2012.14600* (2020).

[371] VETTER, J., NOVÁK, P., WAGNER, M. R., VEIT, C., MÖLLER, K.-C., BESENHARD, J. O., WINTER, M., WOHLFAHRT-MEHRENS, M., VOGLER, C., AND HAMMOUCHE, A. Ageing mechanisms in lithium-ion batteries. *Journal of Power Sources 147*, 1 (2005), 269–281.

[372] WANG, H., AND CHENG, K. W. E. An improved and integrated design of segmented dynamic wireless power transfer for electric vehicles. *Energies 14*, 7 (2021), 1975.

[373] WANG, L., QIN, Z., SLANGEN, T., BAUER, P., AND VAN WIJK, T. Grid impact of electric vehicle fast charging stations: Trends, standards, issues and mitigation measures-an overview. *IEEE Open Journal of Power Electronics 2* (2021), 56–74.

[374] WANG, N., TANG, L., AND PAN, H. A global comparison and assessment of incentive policy on electric vehicle promotion. *Sustainable Cities and Society 44* (2019), 597–603.

[375] WILLIAM J. BROAD, JOHN MARKOFF, D. E. S. Israeli Test on Worm Called Crucial in Iran Nuclear Delay. https://www.nytimes.com/2011/01/16/world/middleeast/16stuxnet.html, 2011. Accessed: Aug. 08, 2024.

[376] WOODWARD, M., WALTON, B., AND HAMILTON, J. "Electric vehicles: Setting a course for 2030", 2020. Accessed: Nov. 23, 2024.

[377] WU, C., SUN, J., ZHU, C., GE, Y., AND ZHAO, Y. Research on overcharge and overdischarge effect on lithium-ion batteries. In *2015 IEEE Vehicle Power and Propulsion Conference (VPPC)* (2015), pp. 1–6.

[378] WU, H., PANG, G. K. H., CHOY, K. L., AND LAM, H. Y. An optimization model for electric vehicle battery charging at a battery swapping station. *IEEE Transactions on Vehicular Technology 67*, 2 (2017), 881–895.

[379] XIE, N., LI, Z., AND TAN, H. A survey of physical-layer authentication in wireless communications. *IEEE Communications Surveys & Tutorials 23*, 1 (2020), 282–310.

[380] XU, J., LI, Z., DU, B., ZHANG, M., AND LIU, J. Reluplex made more practical: Leaky relu. In *2020 IEEE Symposium on Computers and communications (ISCC)* (2020), IEEE, pp. 1–7.

[381] XU, T., LU, X., XIAO, L., TANG, Y., AND DAI, H. Voltage based authentication for controller area networks with reinforcement learning. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)* (2019), IEEE, pp. 1–5.

[382] XUN, Y., LIU, J., KATO, N., FANG, Y., AND ZHANG, Y. Automobile driver fingerprinting: A new machine learning based authentication scheme. *IEEE Transactions on Industrial Informatics 16*, 2 (2019), 1417–1426.

[383] YANG, B., GUO, L., LI, F., YE, J., AND SONG, W. Vulnerability assessments of electric drive systems due to sensor data integrity attacks. *IEEE Transactions on Industrial Informatics 16*, 5 (2019), 3301–3310.

[384] YANG, L., MOUBAYED, A., AND SHAMI, A. MTH-IDS: A multitiered hybrid intrusion detection system for internet of vehicles. *IEEE Internet of Things Journal 9*, 1 (jan 2022), 616–632.

[385] YANG, Q., GASTI, P., ZHOU, G., FARAJIDAVAR, A., AND BALAGANI, K. S. On inferring browsing activity on smartphones via USB power analysis side-channel. *IEEE Transactions on Information Forensics and Security 12*, 5 (2016), 1056–1066.

[386] YANG, T., KONG, L., XIN, W., HU, J., AND CHEN, Z. Resisting relay attacks on vehicular Passive Keyless Entry and start systems. In *2012 9th International Conference on Fuzzy Systems and Knowledge Discovery* (may 2012), IEEE, pp. 2232–2236.

[387] YE, J., GUO, L., YANG, B., LI, F., DU, L., GUAN, L., AND SONG, W. Cyber–physical security of powertrain systems in modern electric vehicles: Vulnerabilities, challenges, and future visions. *IEEE Journal of Emerging and Selected Topics in Power Electronics 9*, 4 (2020), 4639–4657.

[388] YOSHIZAWA, T., SINGELÉE, D., MUEHLBERG, J. T., DELBRUEL, S., TAHERKORDI, A., HUGHES, D., AND PRENEEL, B. A survey of security and privacy issues in v2x communication systems. *ACM Computing Surveys 55*, 9 (2023), 1–36.

[389] YOUNG, C., ZAMBRENO, J., OLUFOWOBI, H., AND BLOOM, G. Survey of automotive controller area network intrusion detection systems. *IEEE Design & Test 36*, 6 (2019), 48–55.

[390] YUAN, H., TANG, Y., SUN, W., AND LIU, L. A detection method for android application security based on tf-idf and machine learning. *PloS one 15*, 9 (2020), e0238694.

[391] ZAX, D. Many cars have a hundred million lines of code. https://www.technologyreview.com/2012/12/03/181350/many-cars-have-a-hundred-million-lines-of-code/. Accessed: Sept. 06, 2024.

[392] ZHANG, F., SHEN, L., AND WU, G. Notes on the security of certificateless aggregate signature schemes. *Information Sciences 287* (2014), 32–37.

[393] ZHANG, H., XIAO, X., MERCALDO, F., NI, S., MARTINELLI, F., AND SANGAIAH, A. K. Classification of ransomware families with machine learning based onn-gram of opcodes. *Future Generation Computer Systems 90* (2019), 211–221.

[394] ZHANG, J., WU, Z., LI, F., XIE, C., REN, T., CHEN, J., AND LIU, L. A deep learning framework for driving behavior identification on in-vehicle can-bus sensor data. *Sensors 19*, 6 (2019), 1356.

[395] ZHANG, M., AND MASRUR, A. Improving timing behavior on encrypted can buses. In *2019 IEEE 25th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)* (2019), IEEE, pp. 1–6.

[396] ZHANG, Z., CHAU, K., QIU, C., AND LIU, C. Energy encryption for wireless power transfer. *IEEE Transactions on Power Electronics 30*, 9 (2014), 5237–5246.

[397] ZHANG, Z., PANG, H., GEORGIADIS, A., AND CECATI, C. Wireless power transfer—an overview. *IEEE Transactions on Industrial Electronics 66*, 2 (2018), 1044–1058.

[398] ZHAO, Q., CHEN, M., GU, Z., LUAN, S., ZENG, H., AND CHAKRABORY, S. Can bus intrusion detection based on auxiliary classifier gan and out-of-distribution detection. *ACM Trans. Embed. Comput. Syst. 21*, 4 (sep 2022).