University of Padova

Department of Mathematics "Tullio Levi-Civita"

Master Thesis in Cybersecurity

A review of electric vehicles charge while driving authentication protocols and CAN bus covert channel solutions

Supervisor
Prof. Alessandro Brighente
University of Padova

Co-supervisor
Prof. Mauro Conti
University of Padova

Master Candidate
Claudio Stefani

Student ID
2021131

Academic Year
2022-2023

"All truths are easy to understand once they are discovered; the point is to discover them."
— Galileo Galilei

Abstract

During recent years the consideration of environmental issues is becoming increasingly important. For this reason, transportation sector is finding means to replace conventional fuel engines. Electric vehicles (EVs) are emerging as a valid alternative to internal combustion engines vehicles. Researchers are starting to design models for recharging batteries while the EV is in motion, namely Charge While Driving (CWD) methods. To begin a dynamic wireless charging session and during the whole process, the EV has to communicate with different entities in order to exchange information about its identity, the charging level, billing data and other depending on the architecture of the system. Considering the sensitivity of the exchanged information, the research community developed various authentication protocols to protect the communications. In order to provide full connection security, the intra-vehicle communication plays an important role. However, security is often not implemented due to the usage of legacy systems such as Controller Area Network (CAN) bus. Therefore, it is fundamental to develop secure solutions that however do not impact the behavior of traditional systems. In this work we review the main authentication protocols for CWD technology, and CAN bus covert channels available in the literature.

Contents

Listing of figures

Listing of tables

Listing of acronyms

AES ..... Advanced Encryption Standard

ARINC ..... Aeronautical Radio Incorporated

AXI ..... Advanced eXtensible Interface

BEV ..... Battery Electric Vehicle

BPSK ..... Binary phase-shift keying

CAN ..... Controller Area Network

CA ..... Certification Authority

CCS ..... Company Charging Server

CC ..... Charging Company

CD+AMP ..... Collision Detection with Arbitration on Message Priority

CMC ..... Charging Management Center

CP ..... Charging Pad

CRC ..... Cyclic Redundancy Check

CSMA ..... Carrier Sense Multiple Access

CSPA ..... Charging Service Providing Authority

CSP ..... Charging Service Provider

CWD ..... Charge While Driving

C&C ..... Command and Control

DC ..... Direct Current

DDoS ..... Distributed Denial of Service

DH ..... Diffie-Hellman

DLC . . . . . Data Length Code

DMV . . . . . Department of Motor Vehicles

DSRC . . . . . Dedicated Short Range Communication

DSSS . . . . . Direct-Sequence Spread Spectrum

DoD . . . . . Department of Defense

DoS . . . . . Denial of Service

ECU . . . . . Electronic Control Unit

EH-DWC . . . . . Energy Harvesting-Dynamic Wireless Charging System

EKE . . . . . Encrypted-Key Exchange

EOF . . . . . End-of-frame

EV . . . . . Electric Vehicle

FADEC . . . . . Fast Authentication for Dynamic EV Charging

FLPA . . . . . Fast and Lightweight Privacy-aware Authentication

FPGA . . . . . Field Programmable Gate Array

FSK . . . . . Frequency-shift keying

GNSS . . . . . Global Navigation Satellite System

GUI . . . . . Graphical User Interface

IAT . . . . . Inter-Arrival Time

IDE . . . . . Identifier Extension

IDPS . . . . . Intrusion Detection Prevention System

IDS . . . . . Intrusion Detection System

IFS . . . . . Inter-frame Space

ISO . . . . . International Organization for Standardization

IoV . . . . . Internet of Vehicles

LIN . . . . . Local Interconnect Network

LSB . . . . . Least Significant Bit

MAC . . . . . Message Authentication Code

MITM . . . . . Man-in-the-middle

MOST . . . . . Media Oriented Systems Transport

MiTM . . . . . Man in The Middle

NSA . . . . . National Security Agency

OBD . . . . . On-Board Diagnostic

OBU . . . . . On-Board Unit

OLEV . . . . . Online Electric Vehicle

OPAMP . . . . . Operational Amplifier

PAT . . . . . Payment Authorization Token

PCB . . . . . Printed circuit board

PHEV . . . . . Plug-in Hybrid Electric Vehicle

PO . . . . . Pad Owner

PSS . . . . . Power Supply Station

QAM . . . . . Quadrature Amplitude Modulation

RA . . . . . Registration Authority

RCA . . . . . Regional Certification Authority

RRC . . . . . Root-raised-cosine

RSU . . . . . Roadside Unit

RTA . . . . . Registration Trusted Authority

RTR . . . . . Remote Transmission Request

SAE . . . . . Society of Automotive Engineers

SDR . . . . . Software-defined radio

SNR . . . . . Signal-to-noise ratio

SOF . . . . . Start Of Frame

SPEKE . . . . . Simple Password Exponential Key Exchange

SPI . . . . . Serial Peripheral Interface

SS . . . . . Spread-Spectrum

SoC . . . . . State of charge

SoC . . . . . System-on-Chip

TA . . . . . Trusted Authority

TCSEC . . . . . Trusted Computer Security Evaluation Criteria

TOT . . . . . Time-based One-time Tokens

TPMS . . . . . Tire Pressure Monitoring System

TPM . . . . . Trusted Platform Module

TRANSEC . . . . Transmission Security

TRM . . . . . Tamper Resistant Module

TTP . . . . . Trusted Third Party

V2I . . . . . Vehicle to Infrastructure

V2P . . . . . Vehicle to Pedestrian

V2V . . . . . Vehicle to Vehicle

VANET . . . . . Vehicular Ad-Hoc Network

WPT-CWD . . . . Wireless Power Transfer-Charge While Driving

WPT . . . . . Wireless Power Transfer

1

Review of Authentication Protocols for Charge While Driving Wireless Power Transfer in Electric Vehicles

1.1 Introduction

The largest contributor of EU greenhouse gas emissions is the transport sector [11]. A recent proposed EU legislation sets the target to cut 100% emissions from vans and cars by 2035 [11]. An increase of usage of Electric Vehicles (EVs) will be necessary to meet these goals. Battery Electric Vehicles (BEVs) accounted for 9.0% of total new car registrations in 2021, and the share of electric vehicles, both BEVs and Plug-in Hybrid Electric Vehicles (PHEVs) in national new car registrations increased in all countries (EU-27, Iceland, Norway) compared with 2020 [11]. In the context of EVs, one big challenge is represented by the charging phase. Up to date, EV producers focused on producing batteries that could guanratee driving over long ranges and having a long life span. However, range anxiety is still one of the main concerns of drivers, who are afraid not being able tor each a charging station before running out of battery [12]. A possible solution to reduce range anxiety is the use of Charge While Driving (CWD) technologies, i.e., building a dedicated infrastructure to allow drivers charging their vehicle while driving along a dedicated lane. This concept is growing fast in the EVs world, and led to evaluation studies

and testbed implementations [13][14][15]. By combining the idea of CWD with the concept of Wireless Power Transfer (WPT) in what we refer to as CWD-WPT, we obtain a very robust solution to the biggest problem against EVs, the driving autonomy. The infrastructure of a CWD-WPT model includes a series of charging coils embedded in the road surface, commonly referred to as *pads*. These pads provide the power to charge the EV by turning on only when required, and they can power the EV in a few milliseconds, depending on the speed of the vehicle. In this way, a user could use her own EV everyday and every time, without worrying too much about the battery level. To support the CWD-WPT model, Vehicular Ad-Hoc Network (VANET) technology provides means for communications. VANET alone however is not ready to satisfy the security requirements of a charging application. In particular, it needs to satisfy some basic security requirements, such as identify unequivocally the sender of the message, ensure data confidentiality due to the sensitiveness of the exchanged data, and make sure that the received bit sequence is not corrupted or manipulated by a malicious entity. So, we need an authentication protocol that respects some basic requirements: secure authentication, data integrity, confidentiality, access control, non-repudiation, and availability. Without these requirements the protocol may be exposed to cyber attacks that might impair the system at different levels, including unfair billing or theft of charging power. The perfect protocol needs to be computationally and communication efficient, but also very secure and robust against possible external attacks. To this aim, many authors proposed different approaches to secure communications in CWD-WPT. All of them however come with different system entities, different terminologies, and different cryptographic primitives, making their comparison a challenging task. Their comparison however is a fundamental need in deciding which protocol to implement and to understand whether the current state-of-the-art solutions provide sufficient guarantees for a real life implementation.

In this paper, we review and compare the ten most relevant protocols available in the scientific literature highlighting their strengths and weakness. Our work starts by building a meta system and threat model that creates an holistic overview of the CWD-WPT scenario, a basic need for a fair comparison. We then review the details of each authentication protocol, starting from the protocol overview to its implementation and performance. By creating a common baseline of execution time of cryptographic primitives, we then compare protocols based on their computational and communication complexities. We also compare them in terms of resistance over the attacks defined in our meta threat model. Although other works in the literature review authentication protocols for this application, none of them provides a common baseline and a fair comparison.

The contribution of this paper can be summarized as follows.

- We present 10 of the most state-of-the-art Wireless Power Transfer-Charge While Driving (WPT-CWD) authentication protocols available in the scientific literature.
- We analyze the resistance of each protocol against common cyber attacks as well as its prevalent security properties support.
- We compare the computational and communication costs of each protocol, measuring the execution time of cryptographic primitives used by authors on a real computer.
- Ultimately we draw conclusions about which protocol is the best among all.

The rest of the paper is organized as follows. Section 1.2 is about related or past works. Section 1.3 introduces the general model of CWD technologies. Section 1.4 addresses the structure of each analyzed work, a global and brief summary of how it works and what type of cryptographic primitives are used. Section 1.5 will present a comparison of the performance, in terms of computational and communication costs. Section 1.6 will present an overview about the security, followed by a comparison. Finally Section 1.7 is related to the conclusions and an overall view of the work.

1.2 Related Work

As we illustrate in this section, the literature does not offer many works similar to our. Babu et al. [16] published a survey on security challenges and protocols of EV dynamic charging system. They clustered existing papers into four groups: authentication protocols, privacy preserving protocols, lightweight protocols, and secure billing and payment protocols. For each group they created a summary of characteristics of the protocols in question, performed a computation cost analysis and a comparison of security features. We believe that grouping papers is a good choice to be clearer, but we preferred to make a one-to-one comparison between methods' characteristics and to focus in making a clear division between resistance to cyber attacks and security properties. For example, Babu et al. put location privacy and resistance to replay and insider attacks into the same cluster. Furthermore, we analyzed computational costs for each entity of the papers, while Babu et al. considered only the total computation overhead. We also computed the total communication cost for different protocols in bytes and described the phases in detail, in order to make the reader able to understand the differences between different methods.

The other related works focus mainly on state-of-the-art of CWD systems, there are no papers about a review of authentication and/or billing protocols applied to dynamic wireless charging technology except the one mentioned above. In particular, most of the papers introduce a paragraph dedicated to "previous works", but they just present briefly the structure of the models used for inspiration, it is not performed a real and detailed comparison. Jang et al. [17] contains a subsection dedicated to the billing infrastructure, but is not complete, as it considers only two papers out of many available in the scientific community.

1.3 System and Threat Model

In this section, we provide a general system and threat model that summarizes the most relevant entities considered in CWD-WPT state-of-the-art solutions. In particular, we introduce the system model in Section 2.4.1 and discuss the attacker's aims and capabilities in Section 2.4.2.

1.3.1 Meta System Model

A WPT-CWD system consist of subsequent charging coils embedded into the road concrete powered with high voltage, high frequency AC source, and compensation circuits. Each coil is inside a so called Charging Pad (CP), and CPs are usually placed one after another in a dedicated roadway. The EV mounts a receiving coil and, when it passes over the CP, it converts the magnetic field induced in the receiver coil to DC to charge the battery. An On-Board Unit (OBU) is a device installed on EVs, through which they can communicate with both the Roadside Units (RSUs) and other vehicles establishing an Internet of Vehicles (IoV). The RSUs are communication units located aside the roads, working as gateways between the OBU of the vehicles and the CPs, as each RSU is connected to a group of them. RSUs can also communicate to the Charging Service Providing Authority (CSPA), i.e., the entity responsible for deploying the charging infrastructure and billing the EV driver for the energy consumed while charging. It can communicate with the EVs using cellular communication link or via WiFi through RSUs. The CSPA also communicates with CPs. The Registration Authority (RA) generates the system parameters, e.g., public/private keys and signing/verifying keys for other parties. The RA generates for each registered EV a verifiable pseudonym to ensure anonymity of the EV to other systems in the VANET. We provide a picture of the system model in the Figure 1.1. To clarify the nomenclature and provide a unified framework, we grouped similar entities of different system models under a unified name, as described in Table 1.1. We can see that there are five

groups, each of which includes entities that perform the same or very similar functions, but which authors of different papers have called by distinct names. In addition to the entities already described above, we also find in the grouping table the Charging Company (CC), i.e., the company involved in the investment for the construction of the charging infrastructure, which is an offline entity.

Table 1.2 lists the entities of the model of each paper analyzed in this work, ✓ means that the particular entity is present in the model, while X not present.

Figure 1.1: Meta system model

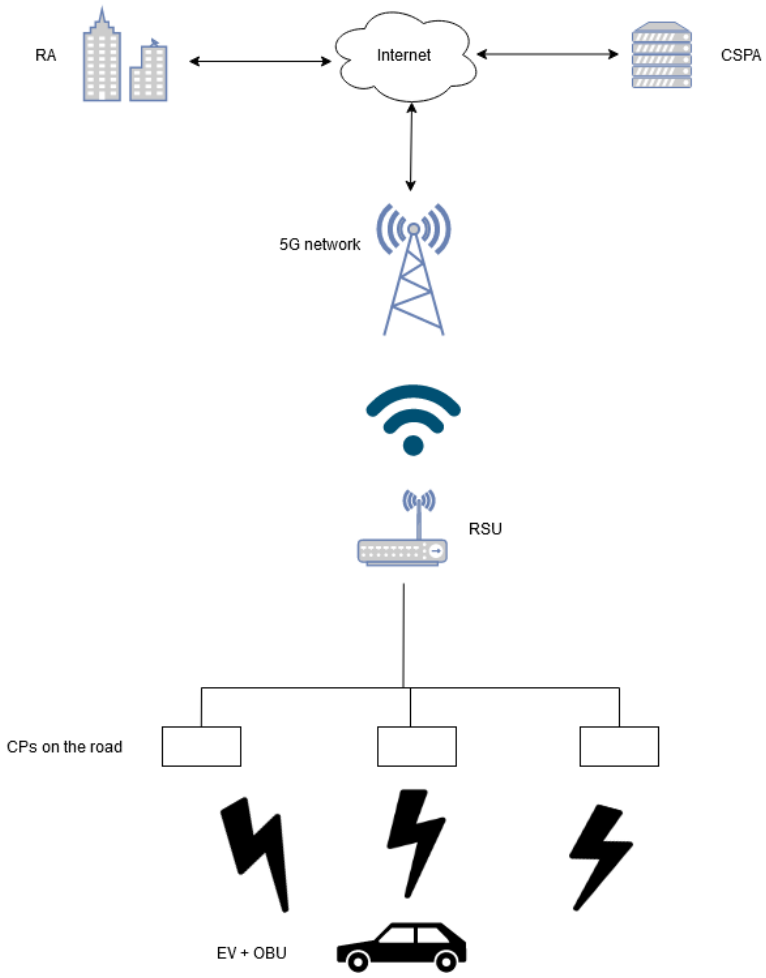

Table 1.1: Model entities grouping

List	Group name
Utility, CSPA, Charging Service Provider (CSP), Charging Management Center (CMC), Company Charging Server (CCS)	CSPA
Charging Controller, RSU, Pad Owner (PO)	RSU
Charging Company (CC), Power Supply Station (PSS)	CC
Registration Trusted Authority (RTA), Trusted Authority (TA), Trusted Third Party (TTP)	RA
Coins, Tickets	Coins

1.3.2 Meta Threat Model

The works by While Li et al. (2013) [1], Hussain et al. [2], Hamouid et al. [8] and Wu et al. [9] do not include a threat model, other works consider mainly attacks against the privacy of the EV and its driver[3][4][6] plus attacks against the payment system [3][5][6], where a malicious costumer succeeds in charging her/his EV for free or paying less than the actual rate. For privacy attacks, authors consider mainly a strong adversary model, where attackers can be both entities of the system model, i.e. CSPA, RSUs, CPs, the OBU, EVs, and external eavesdroppers. Papers consider generally impersonation, coin forging, double spending and replay attacks using old coins as methods for circumventing the payment system. The attacker is computationally bounded, s/he can compose and replay the eavesdropped messages, but s/he cannot decipher the encrypted data without knowing the key, and s/he is not able to reverse one-way functions. EVs are curious about sensitive information of other vehicles, such as drivers identities, State of charge (SoC), battery type etc.

We take into account the resistance of protocols against the following attacks.

Injection Attack. Injection attacks aims at breaking the system performance or stability injecting fake data without being detected. Protocols should always check if data inside received packets are correct and in the right format before performing other operations.

Replay Attack. Replay attacks are attacks on a security protocol using replay of messages from different context into the intended (or original and expected) context, thereby fooling the honest participants into thinking they have successfully completed the protocol run [18]. We check if the surveyed protocols verify timestamps of packets and use random numbers, discarding

Table 1.2: System model entities of the different works

Paper / Entity	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]
CSPA	✓	✓	✓	✓	✓	✓	X	✓	X	X
RSU	✓	X	✓	✓	✓	✓	✓	✓	✓	✓
CC	X	X	X	✓	X	X	✓	X	✓	✓
CA	✓	✓	X	X	X	X	X	X	X	X
EV	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CP	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RA	X	X	X	X	X	X	X	✓	✓	X
Fog Server	X	X	X	X	X	X	✓	X	X	✓
Billing Authority	X	✓	✓	X	X	✓	X	✓	✓	X

data if they are too old or invalid.

Known-key Attack By known-key attack, an attacker is able to charge her/his own EV using old session keys s/he might know. Protocols should renew session keys when needed as countermeasure and store the old ones into a blacklist.

Denial of Service Attack. A Denial of Service (DoS) attack aims at blocking legitimate users' system access by reducing system availability [19]. This category of attacks is very general, as the attacker could intervene even physically by cutting the communication cables to have success in breaking the system.

Man in the Middle Attack. A Man in The Middle (MiTM) attack consists in intercepting, sending and receiving data meant for someone else, or not meant to be sent at all, without either outside party knowing until it is too late [20]. Protocols should authenticate and verify the integrity of data to mitigate this vulnerability, for example through digital signatures.

Impersonation and Spoofing Attacks. We call impersonation attack, or spoofing attack when an attacker assumes the identity of another party in the network, receiving messages directed to the node it fakes [21]. This type of attack is usually prevented by signing the messages with each entity private key, computationally unfeasible to obtain by an attacker due to discrete log problem.

Double Spending Attack. In the papers that introduce the use coins or credits to get energy from the CPs, double spending attack happens when an attacker succeeds in reusing the same coin more than once.

Leakage Attack. Pseudo-Random Number
Generators (PRNGs) are cryptographic algorithms used to generate apparent random numbers. These generators contain a secret state, and if an attacker doesn't know it, he cannot predict the output of the algorithm. If the attacker is able to extract the secret state, he could predict future outputs. Resistance against random number leakage attacks means making sure that the attacker cannot access directly the state and the outputs, as well as selecting the first seed smartly.

Insider Attack. Insider attacks happen when a trusted user, or anyone operating inside the security perimeter, with access to sensitive information and information systems performs malevolent actions [22]. The papers should propose security policies and mechanisms to help the companies involved in the Wireless Charging System to prevent such attacks or mitigate damages.

Masquerade Attack. In masquerade attacks the attacker poses as another legitimate user. In our case the difference with impersonation attack is that in masquerade the attacker exchanges messages pretending to be one entity of the system model, while in impersonation the attacker pretends to be the EV user.

Coin Forging Attack. Coin forging consists in generating fake digital coins and use them as credit to charge the EV for free.

Eavesdropping Attack. In eavesdropping attack, a malicious user overhears communications happening between legitimate entities in the network. This represents a threat as the attacker may be able to gather sensitive users' information.

Jamming Attack. In jamming attack, an attacker generates a signal to transmit over the victim's channel to degrade the communication quality up to the point where the legitimate receiver is not able to correctly receive packets from the victim.

1.4 Protocols Description

In this section, we present firstly a meta protocol description, then the structure and characteristic of 10 different protocols in chronological order of their appearance in the literature.

1.4.1 Meta protocol description

Roman et al. [7], Babu et al. [10], Hussain et al. [2], Wu et al. [9] begin with the System Initialization phase, during which the CCS, DMV, or the TA, generates the master private

key and the global public key, in addition with key pairs for each communicating entity in the system model. The first step of Gunukula et al. [3] and Pazos-Revilla et al. [6] is the coins purchase, which are necessary to buy energy from the charging system. Li et al. (2013) [1], Li et al.(2017) [5] and Hamouid et al. [8] protocols start with the pre-distribution of some data, that are used later. This phase is very close to the System Initialization described above, in fact CSPAs or RTAs generate the keys for the CPs and pseudonyms for the EVs during low traffic hours and distributes them daily, even before vehicles enter the charging sections. During the second phase, EV registers itself with the CSPA, usually through a secure channel. After the registration, model entities authenticate each other and the EV sends a charging request to the CSPA including a coin or a Payment Authorization Token (PAT). At this point, if the coin is valid and not already used, the CSPA authorizes the EV to start the charging process. To achieve a fast authentication with each CP at high speeds on the road while driving, the authentication using an hash chain is usually preferred in considered papers.

1.4.2 Li et al. (2013)[1]

The first protocol introduced in this section is called Fast Authentication for Dynamic EV Charging (FADEC) [1]. The architecture of this protocol is made comprises five distinct entities: the CSPA, the RA, a set of CPs, RSUs and EVs equipped with OBU. It relies on Dedicated Short Range Communication (DSRC), a medium range wireless technology based on IEEE802.11p standard, for the communication between the EV and the RSUs. FADEC proceeds as follows. First of all, the EV establishes a session key in order to authenticate to the CSPA, by using as intermediate the RSUs. This step is performed by using HMAC to sign the exchange of messages, in order to protect the protocol against MiTM attacks. Additionally, the protocol uses JFK (Just Fast Keying), a state-of-the-art key-exchange method. Similarly, the communication between EV and RSUs is based on a second session key and it is encrypted and signed as previously discussed. This key is used to communicate with all the subsequent RSUs, by using a broadcast-and-discard method for key dissemination. In this way, the first RSU shares the key to all its neighbouring RSUs, which store it for a determined amount of time, estimated as the maximum time in which the EV could move to the involved RSU.

In terms of performance, two different scenarios are presented. The simulation is made by using a four-lane single-direction straight road segment of 3km, with a total of five RSUs involved. It uses the mobility trace of 300 EVs that travel at a maximum speed of 75km/h. The first scenario, indicated as *resource rich scenario*, assumes that both the EV and the RSUs have

a strong CPU, so that the operation of signing and verifying the digital signature take almost 20ms. In the second scenario, called *resource constrained scenario*, at least 200ms are required to perform the same operations as above. In terms of security, by observing some studies and trials, some problems with replay attacks and preservation of user's privacy arises. In replay attacks, the attacker could replay an EV's message to an RSU, in order to jeopardize the integrity of the billing system. In this protocol, this type of attack can be blocked by simply introducing a timestamp or a random nonce in the exchange of messages between the two entities, technique used in almost every other authentication protocol. FADEC is not susceptible to other attacks like MiTM, DoS or impersonation, thanks to the use of JFK messages (that guarantees also integrity protection) and HMAC authentication. Talking about privacy, FADEC doesn't provide location or user privacy at all. In fact, a CP owner or a possible malicious entity posing as him, can follow the EV movement at every time. This privacy problem can be solved by using some encryption mechanism, in order to hide the non-essential data to the entities or some pseudonym scheme, but this would require a partial re-design of the protocol's model.

1.4.3 Hussain et al. [2]

The protocol proposed by Hussain et al. [2] is referred to as "Hussain Protocol" in some different IEEE papers, like [7], so in the following we abide to this convention. This protocol is focused on mutual authentication. It is inspired by a technology called Online Electric Vehicle (OLEV), already in use in south Korea to name vehicles that receives electricity from a network placed under the road surface. Based on the typical concept of VANET, the OBU-equipped EV communicates with the CSPA using DSRC (similarly to the FADEC protocol). Additionally, it introduces a Tamper Resistant Module (TRM) to store confidential information about the EV. The RA is the entity in charge of registering and revoking the credentials of the whole system and each of these authorities has to be registered to the Department of Motor Vehicles (DMV), that is the entity at the top of the hierarchy. For the authentication of the EV to the CPs, the authors proposes two different approaches: the first one is based on a *hash and XOR* technique of the key, while the second is based on *hash chains*. This protocol is divided in two different steps:

1. **Initialization.** The first phase is system initialization, where, in order to protect the privacy of the user, the pseudonyms of the EV are generated. In this step, the different keys (private and public) are generated and stored by the RA using ElGamal encryption over elliptic curve cryptography. After, we have TRM installation, where the TRM is physically installed in the vehicle by the DMV, then it is initialized with system parameters.

The successive phase is pseudonym assignment, where the DMV, that has generated n pseudonyms for each vehicle, saves these values in the TRM and sends them to the RA. Here it is used ElGamal encryption with a master public key.

2. **Direct Mutual Authentication (DMA).** The OBU installed on the car authenticates with the CPs using as intermediate the Charging Service Providing Authority (CSPA). This authentication is based on an exchange of messages between CSPA and DMV, all signed and encrypted using hash chains and public key encryption. Once a secure channel between the OBU and the CSPA is created, the EV registers itself with the CSPA and finally the OBU is ready to be authenticated by the CPs using the previously generated pseudonyms. In this step, it is performed a mutual authentication between the OBU and the CPs;

3. **Pure Hash Chain based Authentication (PHA).** This stage is an alternative to the DMA presented above. As said before, the DMA stage includes hash and XOR operations, while here we introduce hash chain-signed messages in the authentication step between CPs and OBU. This method requires more computation which can be a disadvantage for a fast-moving vehicle, creating a computing delay. So, this mechanism is preferred for low-speed vehicles, as it provides more security against possible attacks.

By basing the protocol on hash/hash chains, the authors made it robust in term of integrity and privacy protection. In fact, given m (a generic message), it is easy to compute $h(m)$ (the corresponding hash function), while calculating the value of m giving $h(m)$ is an hard-computation problem. Using DMA (that we can consider as a lightweight authentication mechanism), the computational cost is relatively low, but the overall security is guaranteed, as it is based on hash primitives. The integrity and the privacy of the user is preserved by the use of pseudonyms, technique that will be also taken up in more recent protocols. The problem with this protocol is that the authors focused mainly on ensuring mutual authentication and preservation of the privacy, so that they neglected further studies on the security and the response of the system against attacks like MiTM or DoS.

1.4.4 Gunukula et al. [3]

In the work [3], Gunukula et al. introduce the *bank*, i.e., a new entity that sells charging coins to the user in order to enable the EV charging.

The hierarchy of this model is made by three macro entities: first of all, we have the bank, which sells the coins; then, we have the Charging Station, which includes a CSPA and various RSUs. Each RSU has several CPs. Finally, there is the EV.

The protocol works following several steps:

1. The EV purchases a selected amount of charging coins with its real identity. In order to protect the privacy of the buyer, the bank cannot link the coins to the user. To do this, it uses partial blind signature during the purchase stage.
2. When the EV needs to charge, it sends the first coin to the CSP, which contacts the bank and makes sure that the coin is valid by verifying the signature. As highlighted before, the bank has information only about the coin (valid/invalid, new/used), whereas it has no information about who is using it or used it in the past.
3. Once the coin is verified, the CSP sends two secret tokens to the EV, in order to make it able to compute the secret key by hashing these two tokens and XORing the hashes. This key will be shared with the selected RSU, by using Diffie-Hellman key agreement.
4. By using these keys, the EV authenticates to each RSU and obtains a secret token, this time to compute the secret key useful for authentication with the CPs.
5. Finally, the EV can authenticate to the CPs and get charged.

The system limits the number of RSUs (and consequently of CPs) that can be used for charging the vehicle by limiting the number of keys that the EV can generate. This number is based on the number of coins spent in the charging request. The computation times indicated are in the order of milliseconds for the generation of the tokens, while the hashing operation is in the order of microseconds, depending on the CPU used. By observing some studies, we conclude that the use of blind signature, Diffie-Hellman key agreement and hash and XOR mechanism while generating and sharing the keys make the protocol secure against leakage of user information, also supported by anonymous authentication and non-correlation between coins and users. The authors affirm that the protocol is secure against attacks like MiTM or attacks to the payment service, and provide a detailed description of how the system reacts to these treats. The real problem is that the protocol is susceptible to attacks that affect the overall functionality of the systems, like DoS or Distributed Denial of Service (DDoS), that weren't considered in the authors security reviews.

1.4.5 Rabieh et al. [4]

Rabieh et al. [4] propose an authentication scheme for pre-paid EVs CWD services that can run without the need of a TTP. The objective of the work is to perform mutual authentication while protecting the privacy of the drivers. To attain location privacy, vehicles use blind signatures to anonymously authenticate themselves to the charging stations and, for authenticating to the CPs, a hash chain.

The model considers the EV, the CC, from which the EV purchases charging coins, CSPA, which authenticates EVs to allow recharging, RSUs, deployed on roads and connected to the CSPA and the CPs, not connected to each other, but they can connect via wireless with EVs.

The protocol is divided into the following four phases.

1. **System Bootstrap & Coins Purchasing.** The CC generates the required cryptographic public and private keys. The EV purchases the charging coins offline from the CC using blinded pseudonyms, in a similar way in which the EV buys coins from the bank in [3]. All coins are signed by the CC with the timestamp at the time of buying.

2. **Anonymous Charge Permission.** The EV verifies the signature of the coins and unblinds them. Then, the EV sends to the CSPA a charging permission request packet containing the anonymous ticket. The CSPA verifies the anonymous ticket using public parameters and, if the verification succeeds, the CSPA revokes the EV pseudonym associated with the current ticket to prevent reuse. The CSPA finally generates the session key and the key confirmation code. The confirmation code is used by the EV to be sure that the computed session key is correct. Finally, The CSPA replies to the EV with a granting permission packet.

3. **Charging Request / Reply.** The EV verifies the signature of the granting permission packet and, if it succeeds, EV and CSPA mutually authenticate each other. After that, the EV computes the session key and verifies if it is correct using the confirmation code received from the CSPA. If the check is successful, both EV and CSPA start an encrypted session. The EV then sends to the CSPA a charging request packet containing the charging parameters. Once received the packet from the EV, the CSPA generates a hash chain with a number of hash keys sufficient to charge the battery of the vehicle. The CSPA signs the last key of the chain (the root key) and sends it back to the EV inside an encrypted packet.

4. **EV Charging.** The EV uses the hash chain from the CSPA to authenticate itself to the CPs. The EV sends the root key, signed by the CSPA to the charging controller. The controller verifies CSPA's signature and publishes the root key to the CPs. After, the EV sends the next key of the chain to the next pad, which applies the hash function multiple times until reaching the root key to be compared with the one published by the charging controller. If the computed root key is equal to the key from the controller, the CP checks if the received key is inside a revocation list to prevent double spending and if not, it switches on the charging. The used key is then revoked and the process is repeated for each CP.

The protocol is designed on purpose for CPs with limited computational and communication resources. The authors consider a deeper attack model that takes into account the collusion problem too. The EV location and real identity remain anonymous even if, for example,

the CC colludes with the charging station and other vehicles. Computation and communication overheads are addressed in the paper but these metrics are not compared to related works. In addition, the authors performed a privacy preservation and security analysis to investigate the security features and resistance against common attacks provided by the protocol.

1.4.6 Li et al. (2017) [5]

This protocol, called Portunes+[5], is a work of Hongyang Li, Györgi Dàn and Klara Nahrstedt, the same authors of FADEC, that in September 2017 propose a new idea for EV authentication. This time, the main aim of the protocol is to provide privacy-preserving authentication, even at the expense of a less-optimized charging process. Portunes+ is divided in 3 main steps:

1. **Key Predistribution (daily).** The CSPA generates pseudonyms and session keys for each EV. Then, it sends these couples to the RSU, while the traffic is little (usually during the night). Finally, the RSU disseminates these session keys to all its CPs.

2. **Authentication.** Explicit authentication is performed between the EV and the CSPA and between EV and CPs. It is important to highlight that the real identity of the EV is not revealed to the CPs during the authentication (pseudonyms are used), in order to protect the privacy of the user in case of MiTM attacks. On the opposite, *implicit authentication* is done between a CP and the successive one.

3. **Accounting.** The RSU collects the amount of energy supplied from the CPs to each EV, then sends this information to the CSPA, that reports the bill to the EV.

Note that all the messages exchanged between the entities are signed with the private key of the sender (using ECDSA and MAC) and encrypted with the public key of the receiver (using AES). It is also frequent the use of timestamps in order to make the message exchange robust against possible forging attacks. In this protocol a fourth intermediate step can be added, called EV location estimation. In this step, the protocol needs to estimate the location of the vehicle, in order to inform which CP has to be switched on. Usually, to perform this operation, other protocols relies simply on the vehicle GPS information, but this could have several drawbacks. For example, the stability of the signal could be compromised by a gallery, or it can be forged by a location attack, which may cause an erroneous estimation of the real position of the EV. A more practical drawback regards the precision of the GPS, that usually, for a civil use, is around 5/10 meters of accuracy error. So, the authors decided to use a more secure, but also more time consuming, technique. In fact, a value lp is included in the messages exchanged between a

CP and the EV. From this and other values already present in the messages, the entity could calculate the horizontal distance between the CP and the EV at a specific time instant t, then it can estimate the speed. Then, by sending a new beacon at time t', it can estimate the exact position of the car, by using the speed and the location estimation computed before. Talking about security, it was found by the authors themselves a vulnerability during the exchange of messages between EV and CPs in the authentication step, where an attacker could intercept a certain message, posing as the EV. This is possible, but it becomes very expensive and difficult for the attacker, as if she wants to impersonate as another EV, she has to be really near to the EV involved. The range should be in the order of a couple of meters at a high speed (more than 100 km/h) for a target vehicle long around 5 meters, which is almost infeasible. Even if the attacker replays the captured beacon to the next CP (so that she can anticipate the target vehicle), she has to be really quick to capture every new beacon from the CPs, in the order of 200ms, that is again almost infeasible

1.4.7 Pazos-Revilla et al. [6]

The paper [6] presents a five phases protocol, which implements hierarchical authentication, physical-layer-based authorization between EV and CPs to speed up the process, anonymous coins designed to support efficient payment, anonymous authentication and key establishment. A bank, a charging station and EVs are part of the model. The charging stations include CSPA, RSUs and CPs. EVs should purchase coins from the bank before issuing a charging request to the CSPA. To perform physical-layer-based authorization the authors used Autocorrelation Demodulation (ACD). In this way, secret Time-based One-time Tokens (TOT) exchanged between EV and CPs have high probability of correct reception and the attacker is not able to intercept them. Below we describe the phases of the Pazos-Revilla protocol.

1. **Purchasing of Coins.** When it needs to charge, the EV buys coins from the bank using its real identity. Partial blind signatures are used for coin computation to ensure unlinkability between the EV and the coin. The authors assume that each coin corresponds to a specific monetary amount enough to recharge from a certain number of CPs. To do this, the EV sends a a Coin Purchase Request (CPReq) to the bank. Once receiving the request, the bank performs some checks and withdraws a certain amount of money from the EV owner's account. Then generates the coin by partially blind signing a value sent from the EV in the request packet together with the current date. The bank sends the coin back to the EV inside a Coin Purchase Response (CPRes) packet. The EV unblinds the partially blinded signature to get the bank signature and the date, then sends the anonymous coin to the CSPA when it needs to charge.

2. **Charging Request.** This phase occurs when the EV has to recharge its battery. It contacts the CSPA for the authentication and to determine a shared key for the mutual communication. The EV uses the shared key to encrypt and send the anonymous coin to the CSPA. Once received and deciphered, the CSPA sends the coin to the bank to check if it is already used and to deposit the payment into CSPA account. If not already used, the bank saves the hash of the coin signature inside the used coins table and informs the CSPA about the validity of the coin. The CSPA encrypts and sends to the EV two seed tokens needed for the authentication to the RSUs

3. **Efficient Keys Distribution and Hierarchical Authentication.** First, the CSPA distributes keys to RSUs. The CSPA selects two random master secrets and broadcasts them encrypted with a shared group key to the RSUs. RSUs use the master seeds to compute the secret tokens to be shared to the EV. The second phase consists in key distribution from CSPA to EVs. During the third phase, the keys are distributed from RSU to CPs and EVs. Lastly, the paper uses a hierarchical authentication method where an EV should authenticate itself first to the CSPA. The CSPA sends secret tokens to compute the keys needed to authenticate the EV to the RSUs. RSUs sends a token to the EV needed to authenticate the EV to the CPs. The paper proposes a fast challenge-response authentication mechanisms to mutually authenticate RSUs and EVs each other.

4. **Physical-Layer-Based Authorization.** The proposed authorization scheme exploits channel diversity in frequency and time to provide an high probability of successful authorizations. The EV sends shared TOTs to each CP to perform the authorization process at the physical layer. The transmitter installed on the EV and the CPs support N frequency tones and N consecutive TOTs shared with the N next CPs are transmitted at the same time. The authors assume a model with only two tones f_1 and f_2 as example, in which each CP except the first and last one are hard-wired to their following and preceding ones. The EV transmits in parallel over f_1 the TOT related to the k-th CP and over f_2 the TOT shared with the $k+1$-th CP. The k-th CP demodulates the TOT waveforms obtaining amplitude-varying and noisy versions of the two TOTs. CP k receives also a third waveform from the $k-1$-th CP, that is associated to the k-th TOT too. Then, the CP equal-weight combines the two waveforms associated to the k-th TOT. Given n TOTs from the token pool of the k-th CP, the TOT m passes authorization depending on an hypothesis decision.

The main advantage of this scheme is that it guarantees high authorization success rates under different weather conditions and it prevents eavesdropping and jamming attacks given a good transmitting power. Furthermore, Pazos-Revilla protocol is the only one of the papers analyzed in this review exploiting physical layer properties for authentication.

1.4.8 Roman et al. [7]

Roman et al. [7], introduce the concept of cloud VANET, where they add to the typical concept of VANET the concept of cloud computing to reduce the latency during the request of data and less expensive communication between the entities, thanks to the use of geographically near servers and fog ones. It is divided into the following four macro steps.

1. **Initialization.** A Pseudorandom random number generators creates the nonces and the seeds that will be used to generate the keys. The CSPA chooses a master private key and calculate the relative public key, that becomes global. The CSPA computes also an own private key and an own public key. Finally, the CSPA defines an elliptical curve in order to select all the parameter needed in the next steps.

2. **EV Registration.** The EV registers to the CSPA. Every user chooses his random private key and calculate the public key. The public key, the ID, and the charging parameters of the user's vehicle are sent to the CSPA. The CSPA finally creates a secure certificate using the previous computed keys, then sends the information to the EV.

3. **Coin Purchasing.** The EV has now to purchase a selected amount of coins. Every ticket represents a specified amount of energy to be emitted. The user purchases coins through the CSPA, by telling it the number of ticket that is interested in buying. The EV generates n blindly signed random pseudonyms, that will be also signed by the CSPA. Once the coins are signed and delivered, the EV computes two signature verification values that are be placed inside the coins.

4. **Charging Request.** In this step, the system performs all the required steps in order to verify the validity of the coins, where the RSUs checks all the session keys, the verification keys, and the signatures. Then, the Fog Server performs a check on the validity of the pseudonym used and of the timestamp, in order to find incongruences. The first step is the authentication step, where the EV sends to the interested RSU the pseudonyms and other information in order to get correctly authenticated by the entity. Finally, if the authenticating RSU accepts the EV, it creates a session with it and then, each time the EV connects to a CP, this checks if the shared key used by the EV is legit, then revokes it in order for it not to be reused. The same process continues with the next CPs, until the session ends or the key is rejected.

This protocol includes a very well done study on mutual authentication between the entities, preservation of privacy (thanks to the use of pseudonyms that make impossible to correlate the real identity of the EV to the messages), protection to integrity (thanks to the use of hash functions and digital signatures), and perfect secrecy (thanks to the use of random values during

the creation of the keys, that can not be retrieved and make the keys independent between them). For example, in the creation of each session key two random values are used every time so, even if a session key is leaked for any reason, the attacker could not read or retrieve previous or next messages. It is included also a good report of how the protocol react and resist against various types of attacks, like MiTM, impersonation, and DoS. Up to now, this is a very secure protocol, validated also by the use of the AVISPA [23] tool. The only problem reported by external analysis is that the authors haven't addressed the procedures for exchanging primary data between communication entities. Talking about performance, this protocol is in the order of milliseconds for the authentication processes. We report a more precise comparison table the next section.

1.4.9 Hamouid et al. [8]

The paper [8] introduces Fast and Lightweight Privacy-aware Authentication (FLPA) scheme for EVs CWD. The system is based on verifiable encryption, authenticated-pairwise-keys, and coin hash chain. The model of the system comprises a bank or broker, a CSPA, CPs, RA, RSUs, and EVs. The EV authenticates itself to the CSPA first using a strong anonymous authentication based on pseudonym's authenticity verification, then to the CPs using a fast and lightweight authentication mechanism based on Authenticated-Key-Agreement protocol and hash chains. The work includes also a payment authorization phase in which the EV sends to the bank a request of Payment Authorization Token (PAT) to a specified CSPA for a given amount M. We briefly describe all phases of the protocol as follows.

1. **Verifiable pseudonyms pre-distribution.** During this phase the RA generates for each registered EV, a verifiable pseudonym. Pseudonyms are necessary to ensure EV privacy during authentication and charging, and they are generated applying a composition of two cryptographic functions to the real identity of the EV. In addition,
   pseudonyms meet verifiability, unforgeability, unlinkability and traceability properties. More in detail, the RA computes a signature on the EV's ID and applies the verifiable encryption to the output of the signature.

2. **Payment authorization phase.** The EV sends a request to the bank containing its pseudonym, a valid bank account details, the CSP identifier, a timestamp, and a session key. The bank computes a PAT by signing with its private key and encrypting with the session key all data received from the EV except for the session key and the account details. The bank further saves the token to prevent double spending.

3. **Anonymous authentication and recharging permission.** This phases includes two sub-phases, i.e, i) anonymous authentication and token validation, and ii) recharging permission. In anonymous authentication and token validation the EV sends to the CSP a charging request encrypted with the CSP public key, containing the PAT from the bank, its charging parameters, and a secret session key to encrypt packets from the CSP to the EV. The CSP authenticates the EV's pseudonym by executing the Proof-of-Knowledge (PoK) verification protocol, then validates the token by verifying the token issuer signature. During the second sub-phase the CSP generates a recharging coin chain by picking the last coin and computing N times a private one-way hash function on it to generate each coin. The quantity N of coins, that are the recharging units, depends directly on the amount specified in the PAT. Then, relying on a bilinear pairing scheme, the EV uses a CSP-generated ephemeral customer secret-key, to get a pairwise key with each of the CPs. As the ephemeral customer secret-key depends on the timestamp associated to the current recharging session, the EV can use that key only for one session. Ultimately, the CSP checks that token is not included in a locally stored Spent Token List (STL) and if not, it sends to the EV a charging permissions containing the coin chain, the signed chain root, the ephemeral customer secret-key, the public values of CPs, and the timestamp of the current recharging session. All data are signed and encrypted.

4. **EV charging process.** In the charging process, the EV sends to the next CP the next unspent coin of the chain, the signed chain root, and the public value related to that CP. The message is encrypted with the corresponding authenticated-pairwise key between the EV and the CP. If the CP can decrypt the packet from the EV, the vehicle is implicitly authenticated as only the real holder of the EV pseudonym should be able to compute the pairwise key. Then, after checking the validity of the sent coin, the CP sends one unit of energy to the EV and saves the coin temporarily to prevent double spending.

5. **Billing and redemption.** When the EV sends a termination notice to CPs or when the coin chain timeout is reached, CPs send the last Spent Coin Record (SRC) containing the EV pseudonymn, the first and the last coin of the chain to the CSP. Then, the CSP calculates the amount owed as a function of last spent coin and sends a redemption request to the bank including billing information and EV PAT.

The authors performed a simulation on MATLAB and compared their protocol with [24] and [5] considering the communication/computation overhead and the authenticated charging efficiency metrics. The usage of lightweight cryptographic operations resulted in smaller computational cost than the others. Also, the communication cost is of one message with respect to 9 and 3 messages in the other two papers. FLPA performs better even in terms of Charging Efficiency Ration (CER), that is the ratio of the number of successful EV-CP authentications to the total number of CPs. Moreover it requires only a Required Minimal Inter-Pads

Distance (RMIPD) to have CER of 100% considering 1 meter inter-CP distance even at high speeds. In terms of security, no formal or informal security analysis has been addressed in the paper.

1.4.10 Wu et al. [9]

The work in [9] is slightly different from the others because it combines energy harvesting technology with WPT-CWD. The system can convert harvested solar, wind, or other forms of energy into electricity and then use electromagnetic induction to recharge EVs in motion. But the system can also work in the opposite way, i.e., the EVs can release electricity in the network during special weather conditions such as rainy and cloudy. The model is composed of RA, Energy Harvesting-Dynamic Wireless Charging System (EH-DWC), EV and Blockchain Payment Platform. The EH-DWC further includes the PSS, the RSU, CP. while the EV is equipped with an OBU for communication. A payment phase is also included, which works over a Blockchain network. The proposed protocol works over three states depending on the weather conditions.

1. **High power state.** In this state, all EVs can be charged according to the first-arriving-first-charging policy by converting harvested energy into electricity.
2. **Medium power state.** The PSS allocates charging power for the requested EV according to its reputation value based on bargain game. Then each EV pays the virtual currency for the PSS in the Blockchain platform. The EV obtains the token signed by the PSS which is used to authenticate with RSU and CPs respectively when the receipt has been verified by the PSS.
3. **Low power state.** The EV can discharge to the PSS to alleviate power shortages of the EH-DWC system after authenticating with RSU and CPs respectively. The PSS pays the amount of cash for the EV through the Blockchain network and updates the reputation of the EV with help of the RA when the discharging phase is finished.

Each of the three states described above contain at least four phases, while the low power state has also an additional phase, called reputation update phase. The common phases are charging/recharging negotiation, hierarchy authentication, payment and charging/recharging.

Furthermore, the paper includes a security analysis to prove mutual authentication, secure communication, fair energy allocation, anonymity, resistance against replay and man-in-middle attacks. In addition, the authors evaluated the performance of the protocol in terms of computation cost, energy allocation and reputation update, but without taking into account the related works.

1.4.11 Babu et al. (2021) [10]

The work in Babu et al. [10] presents an authenticated key agreement protocol using elliptic curve cryptography and hash functions. The proposed protocol main entities are CSPA, EV, charging station, Fog Server, RSU, and CPs, and consists of five phases: 1) system initialization phase, 2) Fog Server registration phase, 3) EV registration phase, 4) login phase, and 5) mutual authentication phase. We briefly describe the phases of the protocol.

1. **System Initialization Phase.** During this phase, the CSPA generates the required system parameters and publishes some of them.

2. **Fog Server Registration Phase.** In the second phase, the Fog Server registers with the CSPA generating a key pair and sending its ID and public key.

3. **EV Registration Phase.** During the third phase the owners of the EVs registers their vehicles at the CSPA. To do so, EVs send to the CSPA an hashed version of their ID and a calculated public key. The CSPA replies with a list of parameters that the EV has to store in a tamper-proof onboard memory.

4. **Login Phase.** In phase four, the EV initiates a login request to the Fog Server via a public channel when the owner wishes to charge it.

5. **Mutual Authentication Phase.** During the last phase, the most complex one, EV, Fog Server and RSU follow a protocol to mutually authenticate EV-Fog Server, EV-RSU and agree on a session key for communicating over an insecure channel.

The paper uses the Dolev-Yao threat model [25] as an adversary model to analyze the security of the proposed protocol. Dolev-Yao assumes that the attacker can listen to all communication on a network and can also alter, intercept, and erase all exchanged data. The paper considers also another adversary model, the Canetti and Krawczyk's [26], which assumes that the adversary could not have the capabilities of the Dolev-Yao model, but he can compromise private information as secret and session keys through session-hijacking. Adopting these two threat models, the authors conducted a formal and informal security analysis on the protocol, to demonstrate resistance against MiTM, impersonation attacks, replay attacks, and insider attacks, and to guarantee user anonymity and untraceability.

1.5 Performance Comparison

In this section, we report the communication cost (Section 1.5.1 and computation costs (Section 1.5.2) of the previously revised protocols. We then provide a discussion on their comparison in Section 2.11

1.5.1 Communication costs

In Table 1.3, we report the Communication Costs (CC) computed for each protocol. It is important to note that, in order to have a fair comparison, we decided to involve in the calculation just an EV, connected to the first RSU and the first CP. In this way, we obtain an homogeneous result in bytes for each protocol. The majority of these results were computed by an analysis of the protocol in its entirety, while some other were extrapolated from a previous work made by the authors and adapted to our context. For example, Gunukula [3] and Roman [7] protocols, report a very complete formulation for calculating this cost, that takes into account every possible variable involved. For the sake of completeness, we report these two formulas (in Bytes). For Gunukula [3] protocol we have

$$CC = n \times (464 + \tau(116 + 32 \times \varphi)), \tag{1.1}$$

while for Roman [7] protocol we have

$$CC = n \times (488 + \tau(232 + 32 \times \varphi)), \tag{1.2}$$

where n is the number of EVs, τ is the number of RSUs, and φ is the number of CPs involved. All the numeric elements are obtained by summing all the costs in bytes of every message, operation that vary from protocol to protocol based on the number of messages exchanged, the number of element per message, and the real dimension of each element. In the Hussain protocol [2], we should make a distinction between the use of DMA (Direct Mutual Authentication) and PHA (Pure Hash chain based Authentication). In the first case, we shall consider the dimension u of the pseudonym used. In the second case, we have a fixed operation, composed by 6 bytes for the timestamp, 1 byte for the charging request and 64 bytes both for the encrypted and the hashed values. The communication cost for FADEC is described in the protocol itself. It sends 1024 bits/s of information, transmitted every 5 seconds. This means that every 5 seconds transmits $1024 \times 5 = 5120$bit$= 640$ bytes. In Rabieh & Wei [4] protocol, we ob-

tain the communication cost by summing the interested value between the wide computation done by the authors. In fact, we have that the charging permission request requires around 230 bytes, the granting permission packet requires 250 bytes, the charging request packet is made by 128 bytes, the authentication packet is made by 200 bytes, and the key packet size is 32 bytes. So, the overall computation is made by 840 bytes. In Portunes+ [5], the CC is composed as $160+128\times2 = 416\text{bits} = 52$ bytes, where 128 is both the size of the two keys derived from the one-way function, of dimension 160 bytes each. For Pazos-Revilla protocol [6], the communication overhead is made by 64 bytes for communication between CSP and EV and between CSP and RSU, 32 bytes for communication between RSU and CP and between RSU and CP, 32 bytes for authentication between EV and RSU and between EV and CP, as described in the paper. Finally, in Babu protocol [10], the communication cost is divided in 160 bytes for EV to Fog Server/acCSPA communication, 224 bytes for EV to RSU communication, 32 bytes for EV to CP communication, 40 bytes for RSU to CP communication and 112 bytes for Fog Server/CSPA to RSU communication. In this table are not reported Hamouid [8] and Wu [9] protocol, as in the last one it isn't reported any study about the communication cost, nor theoretical or practical, while in Hamouid it is just indicated that is better than other protocols, without any report on that, so we avoid to report it in the table, as we want to provide a fair comparison.

Table 1.3: Communication costs for different protocols

Protocol	**Communication Cost**
FADEC [1]	640 bytes
Hussain with DMA [2]	$77+u$ bytes
Hussain with PHA [2]	135 bytes
Gunukula[3]	612 bytes
Rabieh e Wei. [4]	840 bytes
Portunes+[5]	52 bytes
Pazos-Revilla[6]	256 bytes
Roman[7]	752 bytes
Babu et al. [10]	568 bytes

1.5.2 Computational costs

In this section, we report the computational costs of the protcols, i.e., the time needed to perform the operations between the entities. In Table 1.4, we report a brief index of the different execution times for every operation involved in each protocol. It is important to note that these times could be different from paper to paper, based on the CPU involved and on the general testing scheme. In Table 1.5, we reported a comparison between the different protocols, where n is the number of EVs involved. It is important to note that the estimated values are here just for a numeric comparison, the important part of the table are the arithmetical expression, that are independent from the environment. We ignored the cost of bit-wise XOR operation, because it is usually very fast and efficient due to the fact that hardware operates directly on the binary representation of numbers. We can see that some papers also take into account the number of charging pads and RSUs.

Table 1.4: Symbol Description

Symbol	**Description**	**Measured exectution time (ms)**
T_{sig}	Execution time for performing the digital signature generation	0.632
T_{ver}	Execution time for performing the digital signature verification	0.073
T_{h}	Execution time for performing the hash function	0.015
T_{h2}	Execution time for performing the hash mapping	Not tested
T_{exp}	Execution time for performing the modular exponentiation	30.326
T_{pair}	Execution time for performing bilinear pairing	834.963
T_{eca}	Execution time for performing the bilinear point addition	10.164
T_{ecm}	Execution time for performing the bilinear point multiplication	64.466
T_{pe}	Execution time for performing public key encryption	0.127
T_{pd}	Execution time for performing public key decryption	0.543
T_{enc}	Execution time for performing encryption	0.017
T_{dec}	Execution time for performing decryption	0.011
T_{rn}	Execution time for selecting random number	0.002
T_{pok}	Execution time for adapted version of the Camenisch's Proof of Knowledge protocol	Not Tested

To calculate the execution time of computations performed by entities of each analyzed paper to populate Table 1.5, we clocked the execution time of cryptographic primitives listed

Entity	[1]*	[2] (DMA)*	[2] (PHA)*	[3]	[4]	[5]*	[6]*	[7]	[8]	[10]	[9]
EV	N.A.	$3T_h$	N.A.	$2T_{exp} + 5T_h \approx 60.727ms$	$3T_{exp} + 4T_{ocm} + T_{eca} + 2T_{ver} \approx 359.15ms$	$2T_{sig} + 2T_{pe} + T_{enc}$	$(3n_{rsu} - 2)T_h$	$3T_{ecm} + 2T_{pair} + T_{ver} + 5T_h \approx 1863.472ms$	$2T_{pair} + T_h + T_{enc}$	$T_{ecm} + 4T_h \approx 64.526ms$	$2T_{ver} + 2T_{pd} + T_{pe} + T_{sig} + T_{rn} + T_{ecm} + T_{auth-ph} \approx 65.395ms*$
CSPA	N.A.	N.A.	$T_h + T_{enc}$	$2T_{exp} + T_{sig} + T_{ver} \approx 61.357ms$	$2T_{pair} + 4T_{exp} + 2T_{sig} + T_h \approx 1792.509ms$	$T_{sig} + T_{ver} + T_{pe} + T_{pd}$	N.A.	$T_{ecm} + 2T_{pair} + T_{sig} + 4T_h + \approx 1735.084ms$	$T_{PoK} + T_{ver} + 2T_{sig}$	$2T_{ecm} + 2T_h \approx 128.962ms$	$2T_{sig} + 2T_{pe} + 2T_{ecm} + T_{ver} + T_{pd} + T_{rn} + T_{auth-ph} \approx 131.068ms$
RSU	N.A.	N.A.	N.A.	$2n^2T_h - 4nT_h + 3T_h \approx 0.03n^2 - 0.06n + 0.045ms$	−−	N.A.	$4(n - 1)(n_{rsu} - 1)T_h^2$	$5T_h \approx 0.075ms$	N.A.	$6T_h \approx 0.09ms$	$T_{rn} + 2T_{ecm} + 2T_h + 3T_{h2} + T_{dec} + T_{pe} + T_{pd} + T_{sig} + T_{ver} \approx 0.015n + 130.38$
CP	N.A.	$6T_h$	T_{enc}	$n^2T_h - 2nT_h + T_h + nT_h \approx 0.015n^2 - 0.015n + 0.015ms$	$(n(n + 1)/2)T_h \approx 0.0075n^2 + 0.0075n$	$T_{h2} + T_{dec} + T_{enc}$	$4n(n_{pad} - 1)T_h^2$	$(n(n + 1)/2)T_h \approx 0.0075n^2 + 0.0075n$	$2T_{pair} + 2T_h + T_{dec} + T_{ver}$	$(n(n + 1)/2)T_h \approx 0.0075n^2 + 0.0075n$	−−

Table 1.5: Computational costs for different protocols

in Table 1.4 using Python 3.10 programming language on a 8 GB RAM, Intel(R) Core(TM) i5-10300H CPU @ 2.50GHz, Windows 11 laptop. We have not calculated the cost of hash mapping as it strictly depends on the native libraries of the programming language in which the operation is performed, e.g. Python uses dictionaries to perform hash mapping, while in Java one has to use the HashMap class. The empty columns represents operations whose values were infinitesimal (for reason related to the model), and hence negligible.

Hussain [2], Portunes+ [5] and in the Hamouid [8] protocols present only the authentication cost or speed, not the one of the whole process, while FADEC [1] and Pazos-Revilla [6] protocols, report a really complete description of the simulation done by the authors. However, although very descriptive in all the steps, they lack a formal computation of costs and results. Data in the table referred to these protocols is a computation done by us from the operations described by authors in their works, hence may be inaccurate.

1.5.3 Cost Comparison

In this section, we discuss the costs of the different models considering the joint computational and communication costs. From Table 1.3, we see that the more efficient is Portunes+ [5], with a total payload of only 52 bits. Other protocols are more expensive, for example Rabieh

and Wei [4] (that is the most expensive) uses 840 bits. This total payload is influenced by different factors, the most important of all is the security measures involved. Then, we should combine this consideration with the total computational costs, described in Table 1.5. In this case, we can consider n (the number of EVs involved) as one, in order to have an easier comparison. In Figure 1.2 there is the plot of the computational cost function of protocols that increase their computational complexity based on the number of EVs on the road. As we can see, this computational cost is sort of "uncorrelated" from the communication one, as, for example, Gunukula [3] requires 140 bits less than Roman (612 vs 752 bits), but they have the same computational cost function. We can see that the protocol of Huassain et. al.[2] is very efficient since the computational cost of CPs is not related to the number of vehicles in the system and requires only to perform an hash function or an encryption depending on DMA or PHA versions. Again, this computational value is very influenced by the security mechanism involved. For example, the signature of all messages exchanged requires more time than signing only specific ones, but it guarantees an higher level of security against attacks like masquerades or impersonation, by assuring the integrity of all the messages in every phase. At the same time, some messages' content is not useful for an attacker, so the sender could avoid to protect them and save some milliseconds without repercussions. By looking at these tables, an interested user should think about his needs. If he just need a fast authentication protocol, without real needs in security scope, he could base his search on Table 1.5 and look for the one that requires less time. Otherwise, if he wants to maintain little exchange-messages, in order to use little storage memories or to save data for other information, he should look at the Table 1.3 and find the one that sends less bits. In general, this is not a good way of reasoning. We should base our observation for our ideal protocol also on security. In fact, usually it's better to find a trade-off between speed and security, in order to find a protocol that guarantees security against possible attacks, respecting our speed needs.

1.6 Security Comparison

In this section, we compare the security properties and guarantees of the previously reviewed protocols.

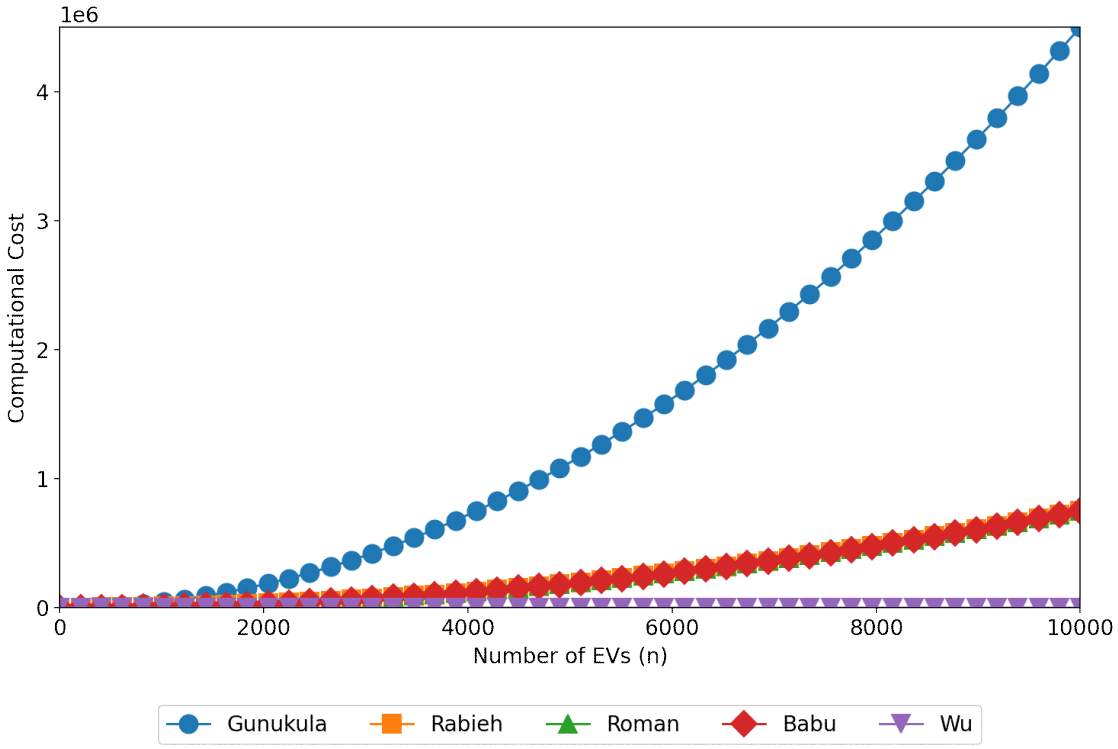

Figure 1.2: Computational cost in terms of number of EVs n

1.6.1 Security properties

Besides the analysis of the possible attacks discussed in Section2.4.2, we also analyze the protocols according to the following security properties.

A system implements mutual authentication if every entity verifies each other identity. Key Agreements are means for two parties to agree upon a shared secret in such a way that the secret will be unavailable to eavesdroppers. The most famous key agreement protocol is Diffie-Hellman's [27]. We adopt the well known definitions of confidentiality, integrity, and avilability.

By privacy, we mean the capability of the system to implement techniques to protect the identity of the EV user from disclosure. Also, the location privacy of the EV should be guaranteed. A protocol provides forward secrecy if after an attacker compromises long term keys, session keys generated before the attack are not compromised too [28].

Non-repudiation concerns with preventing the denial by one of the entities involved in a communication of having participated in all or part of the communication. The evidence informa-

tion of non-repudiation is given by the proof of receipt to prove that the recipient received the data and by proof of origin to prove that the originator sent the data [29]. As for the integrity property, non-repudiation can be achieved implementing digital signatures, but they are not enough alone. In fact, often trusted third parties are involved to assist participants to generate, verify, or transfer non-repudiation evidence and resolve disputes [30]. *Unlinkability* instead deals with the impossibility from the attacker side to infer which parties are communicating one another and are involved in the process [31].

1.6.2 Resistance to Cyber-Attacks and Seurity Features

In Table 1.6, we report a summary of the various cyber-attacks treated by each of the previously reviewed paper. For each cell, a checkmark (✓) indicates that the attack is analyzed directly or indirectly in the work, a X mark indicates that the paper explicitly states the vulnerability with respect to that attack and finally Not Treated (N.T.) if the paper does not mention the attack at all or if it doesn't provide enough elements to perform an adequate study. By analyzing directly, we mean that the resistance against that particular attack is addressed in the paper through a formal or an informal security analysis, while by indirectly we mean that the paper at least mentions the resistance against that attack in one or more of its sections. Notice that, since DoS attack is too generic, we always put N.T. if it is not explicitly mentioned in the paper. Table 1.7 instead reports a comparison of the security properties of the surveyed protocols.

1.6.3 Security Comparison

In terms of resistance against common cyber attacks, we can see from Table 1.6 that almost all papers focus their efforts on protecting the parties involved in the model against replay, impersonation and masquerades. On the other hand, only Pasos-Revilla et al. [6] implements a defense system to mitigate eavesdropping and jamming attacks exploiting physical layer properties. If other protocols are vulnerable to interception it does not mean they are not safe, but that an attacker could in theory read the exchanged packets. Since usually encryption is performed at an higher level than physical, the packets would appear meaningless to the attacker, or they will contain only public information specially designed to be available to everyone.

Another aspect that stands out is that Roman et al. [7] is resistant against almost all attacks, they in fact used a different approach with respect to the other related works. Instead of focusing on two or three security goals, for example location privacy and impersonation, they listed

Table 1.6: Attack resistance treated in different protocols

Attack type	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]
Injection	N.T.	N.T.	N.T.	N.T.	N.T.	N.T.	✓	✓	N.T.	N.T.
Replay	X*	✓	✓	✓	✓**	✓	✓	✓	✓	✓
Known key	N.T.	N.T.	✓	N.T.	N.T.	✓	✓	✓	N.T.	N.T.
Denial-of-service	✓	N.T.	X	✓	X	N.T.	✓	N.T.	N.T.	✓
Man-in-the-middle	✓	N.T.	✓	X	N.T.	✓	✓	✓	✓	✓
Impersonation	✓	✓	✓	X	✓	✓	✓	✓	✓	✓
Double spending	N.T.	N.T.	✓	✓	N.T.	✓	✓	✓	✓***	N.T.
Random number leakage	N.T.	N.T.	N.T.	N.T.	N.T.	N.T.	✓	N.T.	✓	✓
Privileged insider	N.T.	N.T.	✓	✓	N.T.	N.T.	✓	N.T.	N.T.	✓
Masquerade	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Coin forging	N.T.	N.T.	N.T.	N.T.	N.T.	✓	N.T.	N.T.	N.T.	N.T.
Eavesdropping	X	X	X	X	X	✓	X	X	X	X
Jamming	X	X	X	X	X	✓	X	X	X	X
*Easily solvable by introducing timestamps										
**Vulnerable if attacker is fast enough										
***Employs transactions over Blockchain										

Table 1.7: Security properties treated in different protocols

Property name	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]
Mutual authentication	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Key agreement	✓	✓	✓	✓	X*	✓	✓	✓	✓	✓
Confidentiality	X	✓	✓	✓	✓	✓	✓	✓	✓	✓
Integrity	✓	N.T.	N.T.	N.T.	✓	✓	✓	✓	✓	✓
Privacy and Unlinkability	X	✓	✓	✓	✓	✓	✓	✓	✓	✓
Forward secrecy	✓	X	✓	✓	N.T.	✓	✓	✓	✓	✓
Non-repudiation	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
*authors used a key pre-distribution approach										

all common attacks and security properties and tried to design a protocol that accomplishes all of these. We believe this is the right manner to outline a safe protocol for CWD a-priori.

The Rabieh [4] protocol seems to be the most vulnerable, as man-in-the-middle and impersonation attacks are feasible against it. This might be a problem for real-world implementation as attackers can easily eavesdrop wireless channels.

About security properties, we have to say that every paper provides almost all of them, except Li et al. FADEC [1] approach, but it was the first literature work so it is easy to understand the reasons. The biggest problem with this protocol is that it doesn't guarantee any sort of privacy, as it doesn't use pseudonyms to protect the user identity, but it also doesn't protect the user location, that can be easily intercepted by an attacker with all the possible withdrawals. Even for these reasons, FADEC is the "base" paper, on which other authors bring developments and additional features.

Another "critical" approach is the one brought again by Li et al., called Portunes+ [5]. It brought some important developments in respect to FADEC, but, compared to other (recent and not) protocols, it lacks of some important properties. In the original paper, the security discussion wasn't very thorough, as it focuses mainly on replay attacks and how the system avoid them. By some studies and analysis of the structure, we denote that the biggest shortcoming of Portunes is the lack of a key agreement protocol. In fact, it relies on a key pre-distribution

approach, performed during the night, that is way less secure than the usual key agreement protocol. For the positive side, it solves the privacy problem of FADEC by using pseudonyms for each EV and by implementing a secure way to track the EVs position, that guarantees security and perfect accuracy, but it is way more expensive that using GPS + encryption.

In general, the main difficulty we encountered during this review is that every paper discusses the resistance against cyber attacks and its security properties in different manner. In particular some papers perform a formal analysis, others an informal one, or both, or neither. For example, Hamouid & Adi [8] have not dealt with a security analysis, while Babu et al. [10] did both a formal and informal analysis. Pazo-Revilla et al. [6] contains a general security analysis, without specifying if it is formal or informal. And that is not the only issue: even if papers performed security analysis, they they did it only on the properties and attacks that they deemed most appropriate to their particular case. Different system models are also another element that introduce challenges into the comparison. To solve this, scientific literature should develop models which can be implemented in real world, demonstrate their feasibility, then develop secure authentication and billing schemes over them. Finally, it would be appropriate to develop a framework to test the security of these types of protocols in a unique way, otherwise the comparisons between them may be subject to inaccuracies.

1.7 Conclusions

The target of this project was to provide an impartial review of security challenges in WPT for Charge-While-Driving approaches. We performed a discussion and a comparison about ten of the most important approaches (in our opinion), in order to highlight the differences but, most important, the strengths and weaknesses of every protocol.

From the comparisons, it can be see that some protocols are ”obsolete”, like for example FADEC [1], but they are important to note the improvement made by others and the feature shared in time. At the same time, we propose some very valid ”State of the art” alternatives, like Roman [7] and Babu [10], that are the most precise and the most discussed approaches since today. There are also some others very valid approaches, like Pazos-Revilla [6] one, that provides a very wide protection against attacks (as can be seen in Table 1.6), with a relatively small communication cost. So, thanks to the analytic comparisons performed, we can provide an ”easy to see” scheme to decide the most feasible approach for every different CWD network.

Of course, our target was not to find the greatest approach ever between the one we consider, as the choice is influenced by different factors personal to the network and based on the different

needs of the constructor, such that every protocol could be feasible than other to respect the needs.

2

A review of CAN bus covert channel solutions

2.1 Introduction

The CAN, International Organization for Standardization (ISO) 11898, is a widely used communication protocol in the automotive industry for the exchange of information between Electronic Control Units (ECUs). Passenger cars have different kinds of ECUs, e.g. for powertrain and chassis, infotainment, and body electronics, which are connected together by multuple CAN buses, as well as other communication systems such as Media Oriented Systems Transport (MOST), and Local Interconnect Network (LIN). The total number of ECUs in a car is variable, can be for example 40 inside a Volvo XC90 [32]. Despite its robustness and reliability, the CAN bus was not designed with security in mind, which makes it vulnerable to malicious attacks. Automotive cyber attacks can be classified into two branches: local access and remote access attacks. Local attacks require direct or indirect physical access to the CAN bus, while remote attacks can be performed using technologies such as Bluetooth, Wi-Fi, cellular networks, and Global Navigation Satellite System (GNSS) [33]. Remote attacks can be further classified into short range and long range attacks based on the range of wireless access [33]. CAN bus can be affected mainly by the following attack types, described in detail by [33]: DoS and DDoS, suspension attacks, black hole and gray hole attacks, Sybil attack, MiTM, impersonation, fab-

rication, malware inside vehicle software, replay, eavesdropping, fuzzy and timing attacks. The increasing connectivity in vehicles also poses a significant remote attack surface for Vehicle to Vehicle (V2V), Vehicle to Infrastructure (V2I), and Vehicle to Pedestrian (V2P) communications. To address these issues, researchers and engineers have been exploring the possibility of using covert channels to transport out-of-band cryptographic information over the CAN bus.

The CAN bus remains a widely used communication protocol in the automotive industry, even if we can find communication protocols with enhanced security features by design on the market. This is because of its advantages such as its low cost, high reliability, and its ability to support a large number of nodes in real-time control applications. Additionally, the CAN bus has a proven track record of being used in various automotive applications for several decades, and its usage has been standardized by international organizations such as ISO and Society of Automotive Engineers (SAE).

Covert channels are techniques that allow for the transmission of information in a manner that is not intended or detected by the system. Covert channels over a CAN could be useful in situations where secure communication is necessary, but the available bandwidth on the network is too limited to accommodate a more secure communication method. In such cases, a covert channel can be established by exploiting the inherent properties of the CAN bus, which can allow for the transmission of secure data even in the absence of a native secure communication protocol. For example, in the case of ECUs authentication, a covert channel can be established to transmit authentication messages securely over the network. Covert channels can also be used to exchange secrets, to carry out covert operations over the network, or in general to augment the bit rate of CAN while maintaining backwards compatibility.

In this paper, we provide a comprehensive overview of the current state of research on using covert channels for out-of-band transportation of information over the CAN bus. At the moment, there are various available solutions for implementing covert channels of different natures and for disparate purposes in a CAN, but nobody has ever thought of analysing the existing literature to compare them with each other using objective parameters. We have found short discussions in the "Related works" section of some papers [34] [35], but in our opinion they are not sufficient to give an overall view of this field.

We analyze and synthesize the existing studies on this topic to identify the challenges, opportunities, and future directions for this field of research. By conducting this literature review, we hope to contribute to the advancement of secure CAN bus systems and promote the development of new techniques for secure communication over CAN.

The contribution of this paper can be summarized as follows:

- We explain why in some cases it is useful to use covert channels in the CAN bus.
- We describe all the types of CAN covert channels we were able to find in the literature through a research by keywords and by searching inside the citations of works we found.
- We classified each covert channel described in the papers under analysis by type.
- We describe any cyber security measures that the authors of these covert channels have proposed to mitigate native CAN bus vulnerabilities.
- Finally, we compare the hardware required to implement each solution and the information that can be transmitted through each channel.

The paper is organized as follows: in section 2.2 we outline the CAN and covert channels. Section 2.3 is about related works. We try to depict a system and threat model for a covert channel implementation over CAN bus in section 2.4 and we describe covert channels for CAN in 2.5. We summarize papers under analysis in section 2.6, while inspecting their maximum bit rate in 2.7 and their proposed CAN security solutions in 2.8. We finish with explaining the hardware used for their channels setup in 2.9, information transmissible by the channel in 2.10 and a final comparison in 2.11.

2.2 Fundamentals

2.2.1 The CAN

The CAN is a message broadcast system developed initially by BOSCH. The maximum data rate of CAN is 1 Mbit/s with a bus length of 40 meters and 30 connected nodes at most and it is a multi-master, in fact the packets transmitted by the devices are broadcast to the entire network [36]. By Kang et al. [37], the rate limitation at 1 Mbit/s is due to three causes, the first one are bus characteristics, limiting the minimum clock pulse width, resulting in restricting the maximum clock rate too. Secondly, the circuit of the CAN limits the attenuation at high frequencies, so high frequency signal edges will be degraded and lastly in the CAN standard only binary signal is allowed, with a very low bandwidth utilization. The CAN standard is an International Standardization Organization (ISO) defined serial communications bus originally developed to be used in the automotive industry. The CAN communication protocol is a Carrier Sense Multiple Access (CSMA) with Collision Detection with Arbitration on Message Priority (CD+AMP) [36]. CSMA means that each node on a bus must wait for a prescribed

period of inactivity before attempting to send a message. CD+AMP means that collisions are resolved through a bit-wise arbitration, based on the priority of each message in specified in the identifier field of the data frame. A frame with the highest priority always gets bus access. Physically, the CAN bus circuit is composed by two wires, CAN_H, and CAN_L and two 120Ω termination resistors. Devices are linked to the bus with one terminal connected to CAN_H and another to CAN_L. We find a wiring diagram of the CAN bus in Figure 2.2.The logical zero at the physical layer is represented by the dominant state, in which $V_{CAN_H} - V_{CAN_L} \geq 2.5V$, while to represent the logical one, the bus has to be in recessive state, where $V_{CAN_H} - V_{CAN_L} = 0V$ [36].

There are three types of messages that can be transmitted on the bus: data, remote and error frames. Data frames are used to transmit data between devices and they consist of an identifier, which specifies the type of data being transmitted, and a payload, which contains the actual data. Remote frames are requests for transmission addressed to a specific device on the CAN bus and they consist of an identifier, but does not contain any payload data. Finally, error frames are transmitted by a device when it detects an error in the communication, they are composed of a special identifier and do not contain any payload data [36]. The standard CAN data frame contains 11 fields, while the extended data frame has 29 fields. The first field of standard CAN is Start Of Frame (SOF), which is one bit used for nodes synchronization after the bus idle state. After, there is the 11 bit identifier field, which denotes the message priority, lower is its value, higher the priority is. The third field is the Remote Transmission Request (RTR) which is set to dominant state when information is required from another node. All nodes connected to the bus receive the packet but the explicit recipient is specified in the identifier field. The fourth field is the Identifier Extension (IDE) bit, which is dominant if a standard CAN identifier with no extension is being transmitted. Then there is r0, one bit reserved for possible future use. At the sixth place of the frame fields there is the 4 bits Data Length Code (DLC) containing the number of data bytes being transmitted. After DLC there are from zero to 64 bits of application data, the effective payload. The eighth field is the Cyclic Redundancy Check (CRC), containing the 16 bit checksum for error correction of preceding data payload. CRCs are known also as polynomial codes, in fact they are a class of codes suited especially for the detection of burst errors. In such a code, we select the code words such that the associated polynomials are multiples of a certain generator polynomial $g(x)$. The generator polynomial therefore decides the error control properties of a CRC [38]. At the ninth place there is the 2 bits ACK field, composed of one acknowledgement bit and one delimiter bit. If a node receives an error free message, it overwrites the acknowledgement bit with a dominant bit, while in case

SOF	ID	RTR	IDE	r0	DLC	Payload (0-8 bytes)	CRC	ACK	EOF

Figure 2.1: The CAN frame

of errors the receiver discards the message and the sender repeats the packet after re-arbitration. The tenth field is End-of-frame (EOF), a 7 recessive bits sequence denoting the end of the CAN frame. Lastly, there is the Inter-frame Space (IFS), 3 recessive bits to allow the receiver to process the received frame and prepare for the next one. We can see a picture of the can frame fields in Figure 2.1.

2.2.2 Covert channels

U.S. Department of Defense (DoD) defines a covert channel as "any communication channel that can be exploited by a process to transfer information in a manner that violates the system's security policy" [39]. Vanderhallen et al. [40] instead gives this definition of covert channel: data transmission that (1) for its payload carrier leverages some behavioural property of an underlying form of non-covert communication and (2) in exploiting that carrier does not require write access to non-covert traffic data objects. In our case, the underlying form of non-covert communication is the CAN traffic, while the traffic data objects are the CAN frames. We can extend these definitions including the frame manipulation by writing into unused or reserved fields the information we want to transmit outside the allowed payload.

2.3 Related Work

There is no related work about specifically covert channels in CAN bus. Vanderhallen et. al [40] in Section 9 provides an overview of covert channels in CAN; the paper makes a distinction between storage channels and timing channels as written in the Trusted Computer Security Evaluation Criteria (TCSEC) [39], also known as the "Orange book", a set of deprecated criteria established by the National Computer Security Center, managed by the United States' National Security Agency (NSA). TCSEC definitions are the following: covert storage channels include all vehicles that would allow the direct or indirect writing of a storage location by one process and the direct or indirect reading of it by another. Covert timing channels include all vehicles that would allow one process to signal information to another process by modulating its own use of system resources in such a way that the change in response time observed by

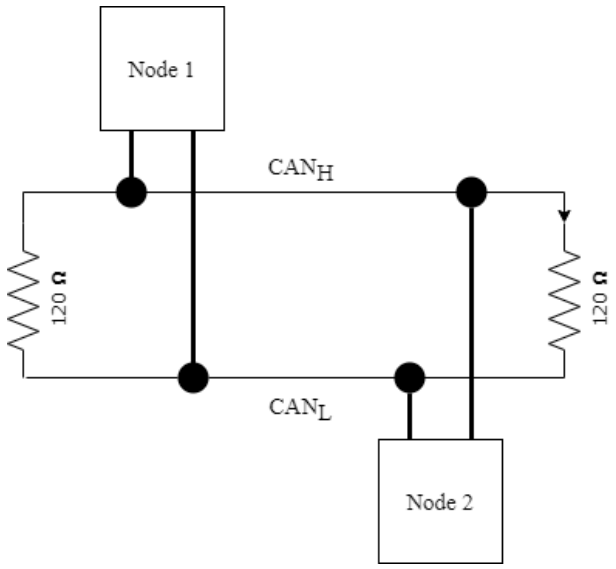

Figure 2.2: CAN bus wiring diagram

the second process would provide information. We believe these two categories are limiting in the automotive background, so we decided to not group covert channels in this way. Another flaw that we found in Vanderallen et al. is the lack of citations to the papers that implemented that particular channel within the paragraphs that describe the various types of channel. At the same time, authors did not consider what we call "gray zone covert channels" in Section 2.5.1 as covert channels, but physical extensions to CAN, while we prefer to consider them as such, given that they can embed additional information on the same medium of CAN bus without compromising its operation. Even voltage covert channels are not mentioned, which is comprehensible, since the first paper about them published after theirs. Analyzing similar contexts, we can find Zander et al.'s survey of covert channels and their countermeasures applied to computer network protocols [41] and the Carrara & Adams from University of Ottawa survey specifically based on out-of-band covert channels [42].

2.4 System and Threat Model

In this section we will try to offer a general system model of a CAN bus covert channel usage scenario, along with modelling threats of this system, based on some sources of scientific literature.

2.4.1 Meta System Model

The CAN bus is used for between intra-vehicular devices communications. The exchanged messages contain information about various aspects of the car's operation, such as engine speed, vehicle speed, and fuel level. An ECU is a computer that is used to control various electronic systems and devices in a vehicle. ECUs are responsible for managing a wide variety of functions, including engine control, transmission control, ABS brakes, and stability control, among others. There are many different ECUs in a typical vehicle, usually from 20 to 100 of them depending on the model [43], each one responsible for a specific set of functions. The ECUs in a vehicle are connected to each other and to other devices and systems using a communication network, such as the previously mentioned CAN. This allows the ECUs to exchange data and coordinate their activities in order to ensure that the vehicle is operating safely and efficiently.

2.4.2 Meta Threat Model

Bozdal et al.[44] did a vulnerability assessment of CAN protocol in their work. They carried out their assessment based on confidentiality, integrity and availability. Confidentiality is defined in Section 3542 of Title 44 of the U.S. Code as "authorized restriction of information access and disclosure, including the means for protecting personal privacy and proprietary information", integrity as the prevention of "improper information modification or destruction" and availability the "timely and reliable access and use of information". CAN protocol does not ensure confidentiality as it does not implement encryption methods naively, so this could could allow a privacy violation by a malicious user. CAN nodes can verify the integrity of the payload using CRC field of the data frame, but they cannot prevent data injection by attackers, thus CAN cannot provide integrity by definition. Availability is not guaranteed by CAN protocol too as transmitting always messages with the highest priority, low priority nodes would never be able to send their packets. In summary, CAN protocol doesn't have any measure against attacks. In the automotive context Bozdal et al.[44] describes two attack types, that are physical access and remote access attacks. The attacks of the first kind require an indirect or direct access to the CAN bus network. This can be done replacing an existing node with a malicious one or accessing the bus through the On-Board Diagnostic (OBD) port. Remote access attacks are possible in modern vehicles containing wireless interfaces for communicating with systems such as Tire Pressure Monitoring System (TPMS), Bluetooth, and radio, since these interfaces are usually connected to CAN too.

2.5 Covert channels in CAN

Implementing measures to ensure confidentiality, integrity and availability (i.e. cryptography and Message Authentication Codes (MACs)) in the CAN leads ECUs to transmit additional bits through the network, but given the limited 8 byte payload of the CAN frame, they have to send more frames, resulting in an increased busload. In addition, modifying the original payload might destroy backward compatibility with existing devices connected to the CAN bus. Lastly, an ECU is a limited-resource device, so performing expensive cryptographic computations could take a lot of time. With covert channels we can send data out-of-band, without modifying the original CAN protocol and overloading the bus.

2.5.1 CAN covert channels in the literature

Further on, we describe CAN bus covert channels sorts we were able to find across the literature and their method of operation. In Table 2.1 we can see the type of covert channel proposed by each paper we analysed. We put ✓ if the work proposes that particular covert channel, ✗ otherwise.

Timing channels Timing channels are covert channels where information is conveyed by timings of events, therefore the receiver should have access to an independent clock with which these timings may be measured [45]. In the CAN context, we can manipulate packet inter-arrival times of periodic messages to encode additional information, as done by Ying et al. [46].

Frame manipulation channel This type of channel is obtained by changing bits of a CAN message in a non-disruptive way. Ying et al. [46] uses the L Least Significant Bits (LSBs) of the payload to introduce custom data. This should be done only on data frames containing floating point numbers coming from sensor measurements, to not introduce significant degradation in accuracy. Additionally, we can tamper with the CRC field. As described in 2.2.1, the CRC field is used by the CAN hardware to detect errors inside the data field. It would be possible to insert covert data inside CRC field at first transmission attempt. The receiver would read covert data inside that field, then flag the frame as invalid and discard it. The problem with this approach is that the CRC field is not software-controllable. Alternatively, we can embed only one bit of information by simply manipulating randomly the CRC field. In this case, the receiver should only

check the error verification outcome and associate a binary state if passed or not. Murvay et al. [35] exploits CRC for improving the bit rate of their voltage covert channel.

Voltage channel CAN specifications allow a threshold band for $V_{CAN_H} - V_{CAN_L}$ in both recessive and dominant states, against voltage fluctuations induced by electromagnetic noise and by the connected devices during the arbitration phase. We can create a voltage-based covert channel by encoding additional information as different voltage levels, always remaining within the limits set by the CAN standard, as done in [35].

Modulated signal overlap channels We can create a covert channel by applying a carrier modulated signal on top of the standard CAN signal. Kang et al. [37] proposed an high-speed CAN controller sending both high-speed CAN bits and standard CAN frame bits to the transmitter. High-speed bits are modulated and overlapped to standard CAN bits. Michaels et al. [47] instead chose a Binary phase-shift keying (BPSK)-based Direct-Sequence Spread Spectrum (DSSS) signal with traditional Root-raised-cosine (RRC) filtering. The modulated signal is injected into both the CAN_H and CAN_L lines, creating a spread spectrum underlay in the common mode path. At the receiver side, the watermark is extracted by performing a normalized summation of the bus voltages, i.e. $V_{watermark} = V_{CAN_H} + V_{CAN_L} - 2V_{recessive}$.

Gray zone channel CAN bit transmission interval is divided into three zones, synchronization zone, sampling zone and gray zone [48]. Synchronization zone is the interval in which a CAN node expects to detect an edge if a bit flip happens, so asserting edges at wrong time could lead to synchronization issues. Sampling zone is the time range in which the actual bit value is determined. Lastly, gray zone is the time between synchronization and sampling zones. Altering the signal in this zone does not affect the overall operation of the CAN, therefore we can create a covert channel by injecting higher data rate information during gray zone interval, as done in [48].

Hybrid channels We can implement a hybrid channel by combining together more than one of the channels mentioned previously. For example, Ying et al. [46] created a hybrid channel by merging a timing channel with a frame manipulation channel.

2.5.2 CAN covert channels not present in the literature

Vanderhallen et al. [40] describes other possible channels, as they comply with covertness definition of some research papers about covert channels [49][50][39]. We write them here for the sake of completeness, although there would be problems in practical implementation.

Dedicating ID bits It is possible to dedicate some bits of ID field of the CAN frame to transport additional data if some applications rely on fewer IDs than the maximum possible.

Manipulating arbitration collision frequency Nodes can cause arbitration collisions on purpose and covert payload can be embedded into collision frequency. This technique is difficult to implement as CAN driver software does not allow government of arbitration collisions.

Manipulating packet size We can use the DLC field of the CAN frame to transport covert data. Accordingly, data field should be sized properly to match the defined length. Using this kind of covert channel may affect data accuracy, as it removes bits from payload of the original message in some cases.

Data field padding If the payload of a CAN message contains less than 8 bytes of data, it can be padded with covert data. This technique has the downside of increasing the bus load, thus interfere with real-time transmissions.

Packet reordering We can generate a sequence of values for the ID field to send packets with a specific order from which covert information can be decoded. That practice obviously impacts real-time behaviour and thus cannot be considered generally applicable. This approach can be combined with a inter-arrival times covert channel to create an hybrid channel.

2.6 Summary of analyzed papers

We will now summarize briefly below the main concepts advocated by each work in question. Papers are ordered ascending by date of publishing.

Ziermann et al. [48] The paper describes the design and implementation of CAN+, which is a new and backward-compatible protocol that improves the data rate of the traditional

Table 2.1: Type of covert channel for CAN bus proposed by each paper.

Paper	Timing	Frame manipulation	Voltage	Modulated signal overlap	Gray Zone	Hybrid
Ziermann et al. [48]	✗	✗	✗	✗	✓	✗
Kang et al. [37]	✗	✗	✗	✓	✗	✗
Groza et al. (2019-03) [51]	✓	✗	✗	✗	✗	✗
Ying et al. [46]	✓	✓	✗	✗	✗	Possible but not implemented
Groza et al. (2019-07) [52]	✓	✗	✗	✗	✗	✗
Canis Automotive Labs Ltd.[53]	✗	✗	✗	✗	✓	✗
Groza et al. (2021) [54]	✓	✗	✗	✗	✗	✗
Vanderhallen et al. [40]	✓	✗	✗	✗	✗	✗
Murvay et al. [35]	✗	✓	✓	✗	✗	✓
Soderi et al. [34]	✗	✗	✗	✓	✗	✗
Michaels et al. [47]	✗	✗	✗	✓	✗	✗

CAN. CAN+ uses the gray zone covert channel described in section 2.5 to increase the data rate of the standard CAN bus system by up to 16 times. The authors evaluate the performance and compatibility of the CAN+ protocol through simulations and experiments, showing that it can provide high data rates while maintaining compatibility with existing CAN devices. They also demonstrate that the CAN+ protocol can be used in real time applications such as video transmission. To sum up, the CAN+ protocol provides a new solution for increasing the data rate of the CAN bus, leveraging the advantages of backward compatibility and covert channels to achieve high performance with no overhead.

Kang et al. [37] The paper proposes new scheme for enhancing the speed of CAN, where a carrier modulated signal is introduced on top of the existing CAN signal, whereby the data rate can be enhanced over 100 Mb/s. The proposed scheme is compatible with the existing CAN network and accordingly enables seamless upgrade of the existing network to support high-speed demand using CAN protocol.

Groza et al. (2019-03) [51] The paper introduces INCANTA, a protocol for intrusion detection in CAN that uses a covert timing channel to embed authentication tags. The protocol is designed for cyclic frames, which are typical in CAN bus traffic, and assumes the existence of a shared secret key between nodes. The protocol consists of two functions executed by each node and does not address on-event frames or the sharing of the secret key. Frames that are on-event can be treated distinctly provided that there is a reference frame for computing the delay. For example one can use the delay toward the previous cyclic frame as a covert channel.

Ying et al. [46] This work presents TACAN, which is new approach for secure transmitter authentication in CAN bus systems that leverages covert channels for secure communication. TACAN consists of three different covert channels for ECU authentication: Inter-Arrival Time (IAT)-based, leveraging the IATs of CAN messages; offset-based, exploiting the clock offsets of CAN messages; LSB-based, concealing authentication messages into the LSBs of normal CAN data. The authors evaluate the performance and security of the TACAN approach through simulations and experiments, showing that it can provide secure transmitter authentication with low overhead and low latency. They also demonstrate that TACAN is resilient against various attacks, such as forgery, replay and masquerade. In conclusion, TACAN provides a new and practical solution

for secure transmitter authentication in CAN bus systems, leveraging the advantages of timing and frame bit manipulation covert channels to achieve high security with low overhead.

Groza et al. (2019-07) [52] This paper proposes TRICKS, a Time Triggered Covert Key Sharing for Controller Area Networks, or a solution for secure communication between ECUs in a vehicle using the CAN protocol and timing covert channels. The solution assumes that ECUs share a weak secret key and can be implemented at the application layer without modifying the CAN protocol stack. The system involves negotiating a session key in a secure manner using timer-counter circuits and sending CAN frames. The paper discusses four variations of the solution based on data or remote frames, identifier priority and timings. The last two versions of the schemes use the Diffie-Hellman (DH) version of the Encrypted-KeyExchange (EKE) protocol plus Simple Password Exponential Key Exchange (SPEKE) [55] to boost efficiency and are secure against guessing attacks. The solution covers both scenarios with mid to high-end automotive grade controllers and low-end cores.

Canis Automotive Labs Ltd. [53] Canis Labs is a company which develops software, hardware and hardware IP technology for CAN bus, focused on security [56]. They published a white paper about CAN-HG, a technique to augment classic CAN by adding fast bits inside of a standard CAN bit, reaching a bitrate of 10 Mbit/s. They use the same technique of Ziermann et al. [48] to add fast bits by taking advantage of a feature of the CAN specification. In fact, CAN requires that the bus signal to be ignored after a bit is sampled as a zero until the next sample point. The main problem of this high bit rate is that CAN physical layer introduces bit asymmetry and noise due to reflections from impedance mismatches, so they had to use a digital filtering system combined with a dynamic bit decoding scheme to overcome it. Additionally, they proposed a central security node that contains CAN-HG Intrusion Detection System (IDS) hardware.

Groza et al. (2021) [54] The paper "CANTO" presents a method for covert authentication using timing channels over optimized traffic flows in the CAN communication protocol, carrying 3-5 additional bits per frame. The method uses timing differences in CAN messages to transmit authentication information, allowing for secure communication without the need for encryption or visible changes in the network traffic. The authors evaluate the performance and security of the proposed method through simulations and

show that it can provide secure authentication while maintaining the performance of the CAN network. They moreover designed four optimization algorithms for traffic allocation and prove their effectiveness by both theoretical models/simulation and experimental data.

Vanderhallen et al. [40] The authors of this paper are advocating for the use of timing-based covert communication in automotive control networks and evaluate its implementation on the CAN. They indeed implement a channel similar to the one described in TACAN [46]. They propose a timing-based nonce synchronization scheme for message authentication in VulCAN [57], which improves the robustness against message loss. The authors evaluate the security properties of their design and provide evidence that it does not harm the security guarantees of VulCAN in the presence of a powerful network-level attacker. They also perform a comprehensive analysis of covert and covert-like bandwidth sources applicable to the CAN protocol.

Murvay et al. [35] The paper proposes a voltage-based covert channel for communication in CAN, which can be implemented at the application layer with minimal hardware modifications and does not affect normal CAN traffic. The authors create small voltage peaks by letting multiple transceivers send data at the same time, which do not affect the CAN standard behaviour and are not viewed as a form of information transfer. The channel remains visible to an adversary but is used only as a transportation layer for secure cryptographic material. This voltage-based covert channel has a higher data rate compared to covert timing channels. The data rate is further increased by using the bits of the CRC field to transport covert information. The authors demonstrate its suitability for CAN authentication data transportation and key agreements on both low and high-end automotive embedded platforms through a proof-of-concept implementation.

Soderi et al. [34] The paper proposes SENECAN, a mechanism to securely distribute keys over a CAN bus communication network by exploiting watermarking and jamming at the physical layer. The system combines watermarking and jamming to ensure confidentiality, integrity, authentication, and anti replay attack capabilities without modifying the protocol architecture. We can say that SENECAN is a covert channel of the "Modulated signal overlap" kind. The authors implement and test the proposed scheme and show its effectiveness. The mechanism uses spread-spectrum watermarking techniques to embed information in the original message and travels within the CAN pro-

tocol frames. The authors survey related works and compare the proposed mechanism with them, highlighting that it provides more robust security properties without requiring additional nodes. The system modulates part of the secret message using a spreading sequence and combines it with the host signal. The recipient disrupts part of the message using an additional CAN transceiver and reconstructs the clean message using the knowledge of the disrupted part. The watermark is extracted using a matched filter that uses the same spreading code as in transmission.

Michaels et al. [47] The paper proposes a low-cost authentication mechanism to improve message validation on wired buses, particularly the CAN bus. The mechanism uses a co-channel watermark that is time-aligned with primary bus messages. This physical-layer watermark technique provides additional protection against attacks that replay a message and attempt to isolate and disqualify a node from communicating on the CAN bus. The watermark also helps mitigate various other types of attacks such as blackhole, grey-hole, rushing, Sybil, message flooding, command injection, impersonation/masquerade, fuzzing, and false data attacks. The proposed approach adapts physical-layer watermarking techniques that are based on DSSS used in wireless communications to improve CAN bus message authentication and integrity. The watermark is introduced to the CAN bus as a common mode signal to minimize its impact on the CAN message processing. This paper also suggests potential extensions of the proposed technique to other wired message buses such as LIN [58], Automotive Ethernet [59], or the Aeronautical Radio Incorporated (ARINC) 429 bus [60].

2.7 Maximum covert channel bit rate

In this section we will outline the performance of the covert channels we are analyzing in terms of the maximum bit rate of the transported data. We have to say that some evaluated papers [52] [40] [47] do not contain the bit rate value of the channel they have designed. In Groza et al. (2019-07) [52], authors exploit both delays and the non-destructive arbitration of CAN to achieve a secure key exchange without additional hardware. So, their covert channel is only used as key exchange mechanism and not for general data transmission. They discuss four variation of their protocol based on data or remote frames, identifier priority and timings. Unfortunately they do not talk about the throughput of their solutions since most methods implemented are probabilistic. Michaels et al. [47] does not specify the exact bit rate of the covert

channel proposed in the paper. However, authors state that the chosen watermark method applied on the CAN signal needs only convey a small amount of data, for example in the simplest form, a single bit is conveyed via the presence or absence of a valid watermark. Additionally, in the Analysis of Co-Channel Watermark section, the paper says that the Signal-to-noise ratio (SNR) range of [+10, +30] dB obtained in tests can be suitable for a rate adaptive protocol with a 1-100 bps speed. Groza et al. (2019-03) [51] did not calculate their covert channel bit rate, however in Related Work section of [54], same authors computed this protocol data-rate, obtaining 57 bps for a single frame ID. Note that their calculation did not take into account covert data transmitted through multiple IDs. It is improbable that performance will increase linearly with the number of IDs because lower-priority IDs may experience more unpredictable arrival times. This is likely due to the increased complexity introduced by the additional IDs. Vanderhallen et al. [40] set a CAN baud rate of 500 kbit/s for their tests. They additionally broke 1/5 of packets on purpose to simulate message loss and test their covert channel resilience. They did their performance evaluation sending a sequence of 10000 authenticated message, resulting in 2000 dropped messages for each experiment. Despite this, they did not compute nor measure the bit rate of their timing covert channel. They only state to have used a timing channel setup similar to the one introduced by Ying et al. [46], so we can assume that the bit rate could be similar. There have also been cases of practical problems in implementing the experiment, such as Soderi et al [34]. Due to a limitation in the simulation environment of this work, the maximum achievable bit rate for the proposed channel is 7200 bps. This happens in consequence of a bottleneck in the serial data exchange between Matlab software and Arduino hardware used in the experiment, making each frame delivered every 15 ms. Ying et al. [46] introduces two different covert channels for CAN, we have to talk about them separately. In the IAT-based channel, the time to transmit one bit is $T_{bit} = LT$, where T is the period of a CAN message and L is the window length. The window length is the size of a sliding window that is used to extract information from the offsets of the packets being transmitted. Larger windows allow for more information to be transmitted, but also increase the likelihood of detection, as they are more likely to deviate from normal network behavior. On the other hand, smaller windows result in a lower data rate, but are less likely to be detected. From the bit time we can extract the bit rate as $R_{IAT} = \frac{1}{T_{bit}} = \frac{1}{LT}$. In the other channel proposed, based on LSBs, the bit transmission time is $T_{bit} = \frac{T}{N_{LSB}}$, where N_{LSB} is the number of LSBs chosen for constructing the channel. The bit rate in this case is $R_{LSB} = \frac{N_{LSB}}{T}$. The most performing channel is from Kang et al. [37], in fact the maximum bit rate obtained for the modulated signal overlapping channel described in the paper, working with the standard CAN protocol, is 85.3 Mbit/s, done by superimposing a

64-Quadrature Amplitude Modulation (QAM) signal over frames. On the other hand, Groza et al. (2021) [54] is very slow. The covert channel implementation introduced in this paper is able to transmit 3-5 bits per CAN frame. Hence, the 24-bit security level demanded by current standards may be reached transmitting covert bits over 6 CAN frames. Over the entire CAN bus traffic, the total bit rate is approximately 5 kbit/s. The voltage covert channel of Murvay et al. [35] is working over a 2500 frames/second CAN rate achieving a throughput of 80 kbit/s. If the CRC field is used as a carrier for covert bits too, creating an hybrid channel, the bit rate can be increased to 97.5 kbit/s. Authors of CAN+ protocol, Ziermann et al. [48], show experimentally that an increase of up to 16x the data rate of a conventional CAN standard (1 Mbit/s) is possible, so the maximum bit rate of their covert channel is 16 Mbit/s. Finally CAN-HG, the solution brought by Canis Automotive Labs Ltd. [53], reached augmented data (covert data) bit rate high as 10 Mbit/s with a CAN bit rate of 0.5 Mbit/s.

2.8 Proposed CAN cyber security measures

The covert channels examined in this work were conceived in the minds of their authors for purposes different from each other. Some channels would only serve as an integration or enhancement of the traditional CAN protocol, while others were designed specifically to give the CAN those security features that it does not have innately. In this section we will investigate what possible cyber security measures our surveyed papers illustrate. We begin talking about general purpose covert channels, not intended to increase the security of the CAN protocol. CAN+, the system proposed by Ziermann et al. [48], is only meant to increase the available throughput of standard CAN, thus no security measures are discussed. The same goes for Kang et al. [37]: security concerns are not analyzed, since this work's purpose is to enhance the speed of CAN protocol. Groza et al. (2021) [54] presents a general purpose timing covert channel, even if it is meant to transport CAN frames authentication data. Thus, no security measures are described. The covert voltage channel introduced by Murvay et al. [35] is generic too. Then there is the interesting case of CAN-HG, the solution offered by Canis Automotive Labs Ltd. [53]. This covert channel uses the same technique as [48] but additionaly provides for a central security node that contains CAN-HG IDS hardware and uses fast bits (covert bits) to communicate security information. CAN-HG guarantees message integrity and bus availability and while providing security mechanisms in hardware. Message integrity is safeguarded by "Bus Guardian", a device added to a node between the CAN controller RX/TX lines and the CAN transceiver. The Bus Guardian augments a CAN frame by adding CAN-HG headers

to it, inserting them using the grey zone covert channel technique explained in Section 2.5.1. The header contains the source address of the sending node. The central IDS operates as follows: the CAN-HG IDS hardware decodes the CAN-HG header before the rest of the CAN frame is fully received and passes it to the IDS software. The software examines the CAN ID and the source address in the header. If they do not match, the software considers the frame as spoof and destroys it. The CAN protocol has three stages of frame transmission: arbitration, transmitting the DLC, data, CRC, and message acceptance. If an error occurs during transmission, the protocol stops receiving the frame and all CAN nodes re-synchronize. IDS software that actively protects the bus is called an Intrusion Detection Prevention System (IDPS). The IDPS approach provides authentication of the message directly in hardware, and the spoof is detected because of where it comes from, and the address identifying that is injected by the Bus Guardian hardware at the source node. Bus Guardian provides additional security measures beyond injecting a CAN-HG header, such as destroying a frame if it sees its own source address being used on the bus, which prevents a device from forging a CAN-HG header. In terms of bus availability measures, the Bus Guardian can detect CAN protocol attacks from the host and temporarily stop the signals to the CAN transceiver, preventing further attacks. Additionally, the central IDPS can broadcast a "cease" command on the CAN with the highest priority ID, causing Bus Guardian to block the attacking host's CAN controller signals until further notice. That command has special protections, and Bus Guardian detects any attempt to spoof CAN ID 0, temporarily taking the host off the bus for long enough to confirm a permanent block using an IDPS cease command. Finally, CAN-HG provides security mechanisms in hardware, such as providing events with metadata throughout a CAN frame, destroying a frame and handling IDPS commands.

Now we are going to look into channels made for security purposes. The protocol designed by Groza et al. (2019-03) [51], named INCANTA, provides time-covert cryptographic authentication for messages on the CAN bus. Protocol assumes a shared secret key k exists on each node and consists of two procedures, $SendCyclic(id, m)$, executed by the sender node and $RecCyclic(id, m)$, ran by the receiver. Both functions are triggered at fixed delays for cyclic messages, as most of CAN traffic is of this kind. In *SendCyclic* function, the sender computes a MAC tag for the message m with the specified id and sets a timeout, T, before broadcasting the message. In *RecCyclic*, the receiver checks if the message m with id was received within an acceptable time frame and if the computed MAC tag matches the expected tag. If these checks are not satisfied, the function drops the received frame and returns *Intrusion*. Authors assume that the CAN message having a certain id is sent at delay δ while the expected arrival time differs

by a small constant ε which compensates for both synchronization error and propagation/computation delays. They also allow to set the desired security level with a parameter ℓ.
Ying et al. [46] introduced TACAN, a message authentication scheme that extracts and verifies authentication messages embedded in the timing or LSBs of normal messages to authenticate the transmitting ECU and also serves as an intrusion detection mechanism. TACAN can detect attacks that interrupt the transmission of normal CAN messages, as well as attacks in which attackers fail to generate valid authentication messages. The protocol features include securely storing master and session keys in the ECU's Trusted Platform Module (TPM), using monotonic counters for generating session keys and authentication messages, and requiring continuous transmission of unique authentication messages to enable real-time transmitter authentication. TACAN can detect forgery attacks, replay attacks, and masquerade attacks. For forgery attacks, the attacker has to forge a valid digest for each local counter value without the session key, and the probability of a successful forgery is $1/2^M$, where M is the number of condensed digest bits. Replay attacks are detected by the use of monotonic counters, and masquerade attacks are detected by forcing the attacker to perform a forgery or replay attack.
The work of Groza et al. (2019-07) [52], known as TRICKS, is more substantial, because describes not a single one, but four methods to exploit both delays and the non-destructive arbitration of CAN to achieve a secure key exchange between ECUs. The first solution is called "Data vs. remote frame negotiation" and it relates to the Mueller and Lothspeich principle [61]. In this protocol, an array of bits is randomly generated on each node, and each bit in the arrays establishes if the frame to be sent is a data frame or a remote frame. Both nodes broadcast frames simultaneously at intervals Δ. If the bits on the nodes are the complement of each other, a data frame will appear on the bus. The ID that is broadcast is a fixed value for all nodes. The security of the bit is compromised if both nodes have sent a remote frame, which is visible to the adversary. Half of the frames from each node contribute to the entropy of the key, resulting in $H_{average}(k) = k/2$. Experimental results on this approach are also discussed, including figures that show frames on each node and frames as they arrived on the bus. All frames arrived at the expected time. The second method's name is "Minimax Negotiation". The minimax key exchange is a protocol used to establish a shared key between two nodes, ECU_A and ECU_B, using randomized IDs. Each node generates two arrays of k random IDs and sends them on the bus at delay $i\Delta$. The protocol continues with the rest of the bits even if there is a small probability that the IDs are identical on both nodes, which is visible to an adversary, and occurs with a probability of $Pr_{bad} = 1/211$ or $1/229$ for standard and extended IDs, respectively. The protocol avoids adding random bytes in the data-field as it increases the transmission time

and leads to a larger time difference Δ. The experiments show that Δ can be lowered to 125μs, which is sufficient to send two frames on the bus with no data-field. Each frame carries exactly one bit of entropy and the protocol runtime is $T(k) = 2k\Delta$. The experiments demonstrate that the protocol works well, with no collisions observed for the 211 bit IDs. There is also a variation of Minimax Negotiation named "Time-Triggered Minimax Negotiation". This method involves generating random bits which are kept secret by two CAN nodes in addition to generating IDs identical to the Minimax protocol. Frames with random IDs are sent at specific intervals according to the value of the bit in the array. If the bits are 10 or 01, they can be easily separated by the sender and receiver, but if they are 00 or 11, half of the bits are lost and the probability of ID collision is reduced to $Pr_{bad} = 1/2$. The delay bit is visible when frames arrive one after another, resulting in half of the bits being disclosed to an adversary. The protocol can be improved by using EKE to bootstrap the session key, and additional bits can be harvested from the ID that wins arbitration to improve the entropy of the exchanged key. The entropy of the exchanged key can be further improved by collecting additional bits based on the principle of the previous Minimax scheme. The last key exchange method described by authors is Randomized time-triggered one. Firstly, the two CAN nodes generate k random values in the interval $[1...\ell]$, where $k < \ell$. They broadcast frames with random identifiers at specific times determined by the values they generated. The key is then extracted based on the delays between distinct frames from the same sender. Due to randomized delays, collisions are expected. The probability of frames being sent at the same time depends on ℓ and is estimated to be k/ℓ. To extract the key bits, an adversary must guess which frame was sent by which node. The entropy $H_{average}(k) = -log_2((k!)^2/(2k)!)$ accounts for the average number of frames that collide, as well as the length of the interval ℓ. As ℓ increases, the chance of a collision decreases, but the duration of the key-exchange protocol also increases. The time to send all ℓ frames is $T(k) = l\Delta$. Authors also cryptographically enforced the Randomized time-triggered Minimax key exchange method (the third one described above) using DH [62] version of the EKE [63] protocol, which allows bootstrapping an authentic key based on a low-entropy common secret. They do it by piggybacking frames with parts of a DH based keys shares of EKE. The work asserts that both the time-triggered minimax and the randomized time-triggered key exchange are suitable for piggy-backing the EKE-DH frames, while the minimax negotiation is not. Finally, authors explain also how the proposed key exchange methods can be extended to more than a single pair of CAN nodes.

Vanderhallen et al. [40] instead did not create his own solution from scratch, but built an extension of VulCAN [57], an open-source CAN authentication suite. In VulCAN, the sender

node first transmits the message with no modifications, then it calculates the MAC of the message, and without delay transmits the MAC as authentication payload of a new message. In the developed extension, the sender additionally encodes the N least significant bits of the MAC nonce by delaying transmission of the authentication payload. A busy-waiting assembly loop is used to create the appropriate delay. Covert payload encoding is performed by multiplying the N least-significant nonce bits with a timing interval ∂. At the receiver side, the original VulCAN design requires the receiving nodes to first receive a message, calculate an expected MAC value using their local nonce value, and then receive the corresponding authentication frame. If the expected MAC matches the received MAC, authentication succeeds and the message is processed; otherwise, it is discarded. The extension to the design adds a circular IAT buffer to hold inter-arrival timings of messages and enables the recovery from message authentication failure by constructing a new tentative nonce based on the inter-arrival time of the most recent authentication frame. The new nonce is used to calculate a second MAC value that is compared to the received MAC value. If they match, authentication succeeds, and message is processed at application level. If authentication fails again, the receiver's nonce remains at its original value, and the message is discarded. In this case, the receiver may initiate a resynchronization protocol with the sender to establish a new shared session key. Then we have Soderi et al. [34], which proposes an authenticated key distribution system for CAN based on a combination of a Spread-Spectrum (SS) watermarking [64] technique and jamming receivers. The key distribution phase is performed in a dedicated time slot to establish synchronization and avoid simultaneous jamming from multiple nodes. During this phase, every node sends keys to other nodes they are supposed to communicate with, and each node stores a list that associates every other node with a specific KEY-ID number. The KEY-ID is included in the message sent to enable receiving nodes to associate the corresponding decryption key with the sender. Receiving nodes with the same arbitration ID must also share the same key for a specific KEY-ID. The key transmission is performed using Frequency-shift keying (FSK) modulation and watermarking to prevent re-transmission due to frames collision. The receiving node reconstructs the original message and saves the couple (KEY-ID, key) in a tamper-resistant memory. This procedure can be performed during vehicle pre-sale or successive sessions to refresh keys, and it is repeated for every ordered couple of nodes. The schema allows distributing keys securely in a one-way transmission without requiring a response by the receiving node.
In conclusion, Michaels et al. [47] offers a low-cost authentication mechanism for CAN bus by employing a co-channel underlay watermark that arrives time-aligned with the primary bus messages. The watermark is designed to be minimally intrusive to the primary CAN messages

and is extracted via a normalized summation of the bus voltages. The frequency domain self-interference of the watermark is driven primarily by the chosen spread ratio, and a reasonable frequency response up to 10 MHz is achievable. To improve the security of the watermark, a Transmission Security (TRANSEC) engine was integrated to dynamically change the generated spreading code on a user-configurable basis. The base key within the device is installed during production and serves as a foundational security component for device-unique communication of vehicle or bus-specific Advanced Encryption Standard (AES) keys and a relatively large 2048 bit watermark base key, from which individual 160 bit session keys are derived.

2.9 Hardware and software required for covert channel setup

While some CAN bus covert channels are easy to implement on off-the-shelf cheap micro-controllers such as Raspberry or Arduino equipped with CAN communication network cards, some of them, especially the ones who guarantee high bit rates, require custom build hardware or high-end dedicated automotive micro-controllers. In this section we will analyze the hardware and, if explicitly stated, even the software used in test beds of covert channels proposed by all papers we considered. We start with solutions that require dedicated hardware design to be implemented.

Ziermann et al. [48] had to build a new CAN transceiver from scratch, since commercial-off-the-shelf models do not support the high data rates, up to 60 Mbit/s, required for CAN+.

The core CAN-HG protocol brought by Canis Automotive Labs Ltd. [53] is implemented in a Verilog hardware IP block called "hgmac". This device contains a CAN protocol engine plus the logic to augment an external CAN signal with CAN-HG data, which synthesises to just under 5000 gates. The Bus Guardian can be implemented in a standard-alone device, such as an Field Programmable Gate Array (FPGA) or integrated into a SoC device, or into a CAN transceiver. Regarding the CAN-HG controller, the company offers some deployment options, such as a hardware IP block on a hybrid FPGA with an integrated micro-controller subsystem and connected to the micro-controller via Advanced eXtensible Interface (AXI) bus, a hardware IP block on a SoC connected to the rest of the system via an AXI bus too and finally the Mercury CAN-HG controller, a standalone small FPGA providing a Serial Peripheral Interface (SPI) interface to the host CPU.

Now we will list the papers that used low-end commercial hardware for their experiments. Ying

et al. [46] implemented TACAN in a testbed consisting of two experimental ECUs Raspberry Pi 3 plus PiCAN 2 boards, and 8 stock ECUs of a 2016 Chevrolet Camaro (University of Washington EcoCAR). They wrote the code of their covert channel in Python language.
Meanwhile, Soderi et al. [34] used two Arduino UNO boards equipped with a CAN shield from SeedStudio and a Raspberry Pi 3 with a PiCAN2 DUO board to collect CAN traffic using Wireshark during their experiments. Furthermore, one of the two Arduino boards is connected to an additional MCP2551 CAN transceiver, used to perform jamming interference.
In the performance evaluation section of Vanderhallen et al. [40], authors evaluated the reliability of their proposed solution using hardware equipped with off-the-shelf CAN transceivers. Their micro-controller setup is the same as VulCAN [57], consisting of six Xilinx Spartan-6 FPGAs, each synthesized with a Sancus-enabled OpenMSP430 core and MCP2515 CAN transceiver chips.
All other jobs not mentioned so far, except Kang et al. [37], have implemented their covert channel on high-end micro-controllers, specialized for real-time use in the automotive industry. Groza et al. (2019-03) [51] employed two AURIX development boards featuring Infineon AURIX micro-controllers and a Vector VN1610 PC to CAN adapter to build their experimental setup, thus no dedicated devices are required to set up the presented time-covert authentication method. The same micro-controllers of Infineon AURIX family alongside a Vector VN adapter connected to a PC running the CANoe environment are used in both Groza et al. (2019-07) [52] and Groza et al. (2021) [54] too.
Murvay et al. [35] implemented their voltage covert channel mechanism on two micro-controller classes, one is from the mid end NXP S12XF family, while the second is from the Infineon Aurix family, designed for high-performance applications. The mid end controller achieved a maximum bit rate of 500 kbps, but its processing capabilities were pushed to the limit, so the higher class one was required to encode covert bits up to 1 Mbit/s.
To conclude this part, the testbed built by Michaels et al. [47] consists of two units each of two different commercial CAN transceiver development cards, namely the NXP MPC5748G, with an NXP TJA1044 transceiver, the Microchip SAM E54, with a MCP ATA6561 transceiver, and a Red Pitaya STEMLAB 125-14 Software-defined radio (SDR). The SDR allowed for the deployment of the dynamic watermarking CAN transceiver, connected to the shared CAN bus. However, a custom Printed circuit board (PCB) with high-speed Operational Amplifier (OPAMP) circuits was needed to match the impedance and power levels of the CAN bus to the SDR transmitter outputs. To enable communication and control of the testbed, a common sockets-based Command and Control (C&C) interface was established to the micropro-

cessors within each of the development cards. This interface was then plugged to an Ethernet switch to allow for addressable communication to each node within the testbed, logging of messages/events, and testing of bit/frame error rates. A Graphical User Interface (GUI) was also developed to enable low-level control of the TRANSEC engine, message rates, and monitoring of communication links across the CAN bus. Finally, we want to talk about Kang et al, which requires a separate discussion. The high speed CAN transmission scheme proposed by them is implemented only in computer simulation. However authors provide with a scheme of the custom high-speed CAN system transmitter, signal generator and receiver which should be required to set up the covert channel.

2.10 Information transmissible by the covert channel

In this section we will analyze the possible information transmissible by analyzed covert channels. We want to devote some space to this topic because some covert channels do not allow transmission of any data other than strictly information to add security measures to the CAN protocol. Ziermann et al. [48], Kang et al. [37], Groza et al. (2019-03) [51], Canis Automotive Labs Ltd. [53] and Murvay et al. [35] are general purpose covert channels by design, this means that they can transport any kind of data outside the CAN payload.
Ying et al. [46] and Vanderhallen et al. [40] use timing/frame manipulation covert channels for transmitting authentication messages, but their method could be used to exchange data of another nature as well.
Soderi et al. [34] use watermarking and jamming to securely send a key between two CAN nodes in the predefined time slot. The time slot length is fixed and corresponds exactly to the estimated time to complete the key transmission. If we put a sequence of bits containing other data instead of the key, probably nothing would prevent us from using this type of channel to transmit any other kind of information.
The watermarking technique of Michaels et al. [47] is used for authenticating CAN bus messages, and even in this case it could probably be adapted to transmit general bits as well.
A separate discourse deserve Groza et al. (2019-07) [52] and Groza et al. (2021) [54]. The first one only exploits delays of CAN frames for key exchange purposes using specifically the four protocol versions discussed by authors, therefore their covert channel is not usable for other intentions. In Groza et al. (2021) [54] frame delays are used to encode MACs of cyclic messages,

consequently a general usage is not feasible.

2.11 Final Comparison

We created Table 2.2 so that we have at a glance a direct comparison between the various covert channels we analysed above. We have grouped the various channels by type and for each type, we have sorted them by date of publication. From the data in Section 2.7 we can see a huge gap between different types of covert channels. In particular, the covert channels of timing type, i.e. [51] [46] [52] [54] [40], in general offer a very low bit rate, while those of type "gray zone" [48] [53] and one of "modulated signal overlap" kind [37] have obtained the best results. In the middle of the ranking we find the voltage covert channels [35]. Timing covert channels can be implemented on cheap low-end hardware, but have the problem of having a limited bit rate, that depends both on the period of the CAN message and the length of the window L, which cannot be increased too much otherwise the covert transmission can be detected by IDSs. Furthermore, they only work on cyclic CAN messages, because the delay is measured relative to the previous transmission of the same message. The highest-performance covert channel [37] was only computer-simulated, it means that the bit rate achieved is theoretical, hence its feasibility and performance on real hardware should be evaluated. Grey zone covert channels provide performance beyond the maximum allowed by the CAN protocol itself (1 Mbit/s), at the cost of using customised hardware, the construction of which requires both engineering and monetary effort. Moreover, [53] is a work produced by a private company, probably subject to patents and therefore not in the public domain.

2.12 Conclusions and future work

In this paper we reviewed all covert channels solutions for CAN bus we were able to find in the scientific literature. We have grouped the covert channels by type and described the general operation of each type of channel. Then we summarized each paper analyzed and for each of them we reported the performance in terms of bit rate and disclosed any security measures proposed by the authors to extend the standard CAN protocol, since natively except the payload CRC, it does not provide other measures to ensure confidentiality, integrity and availability of communication. We also gave particular importance to the hardware and software that were used for the practical implementation of the proposed covert channel, to understand the possible monetary effort that would be necessary in case of large-scale implementation of the solution.

Table 2.2: Covert channels comparison

Paper	Covert type	Max Bitrate	Security features	Required Hardware type	Information conveyed
Groza et al. (2019-03) [51]	Timing	57 bit/s	Message Authentication	Off-the-Shelf / High-end	ALL
Ying et al. [46]	Timing	$\frac{1}{T_{bit}} = \frac{1}{LT}$	Message authentication, IDS	Off-the-Shelf / Low-end	Authentication Messages, but usable for every kind of traffic
Groza et al. (2019-07) [52]	Timing	Not specified	Secure key exchange	Off-the-Shelf / High-end	Key exchange information
Groza et al. (2021) [54]	Timing	5 kbit/s	NONE	Off-the-Shelf / High-end	Encoded MAC
Vanderhallen et al. [40]	Timing	Not specified	Message Authentication	Off-the-Shelf / Low-end	Authentication messages
Ziermann et al. [48]	Gray Zone	16 Mbit/s	NONE	Custom	ALL
Canis Automotive Labs Ltd.[53]	Gray Zone	10 Mbit/s	Integrity, Bus Availability, IDS	Custom + Off-the-Shelf / High-end	ALL
Kang et al. [37]	Modulated Signal Overlap	85.3 Mbit/s	NONE	NONE (computer simulation)	ALL
Soderi et al. [34]	Modulated Signal Overlap	7200 bit/s	Secure key distribution	Off-the-Shelf / Low-end	Cryptographic keys
Michaels et al. [47]	Modulated Signal Overlap	Not specified	Message authentication	Custom PCB + Off-the-Shelf / High-end	Authentication Messages
Murvay et al. [35]	Voltage + Frame manipulation	97.5 kbit/s	NONE	Off-the-Shelf / Medium to high-end	ALL
Ying et al. [46]	Frame manipulation	$\frac{N_{LSB}}{T}$	Message authentication, IDS	Off-the-Shelf / Low-end	Authentication Messages, but usable for every kind of traffic

Ultimately, we checked the types of information each covert channel can carry, since some of them can convey any type of bit sequence, while others are specially designed to encode security information. For example, a timing covert channel could be used to translate a CAN frame MAC into a small transmission delay. We can say that there is no covert channel better than the others in an absolute sense, but it depends on the needs of our application. To add some safety features to the standard CAN bus, we could opt for a timing covert channel built with inexpensive hardware, while if we want to increase the bit rate of the CAN, e.g. to transmit video footage from high-definition cameras in older vehicles without creating a parallel transmission system, we could opt for customised hardware and a grey zone covert channel type system. We have noticed that no paper has tried to implement its solution on a real ECU of a vehicle using CAN bus as the internal communication system. Only Ying et al. [46] tested their solution by connecting the two Raspberry Pi used for experiments to the OBD-II port of a car. As future work, therefore, it would be possible to evaluate the covert channels analysed here on a case-by-case basis and test their feasibility by modifying the original software of some ECUs to implement them.

References

[1] Hongyang Li, G. Dan, and K. Nahrstedt, "FADEC: Fast authentication for dynamic electric vehicle charging," in *2013 IEEE Conference on Communications and Network Security (CNS)*. IEEE, pp. 369–370. [Online]. Available: http://ieeexplore.ieee.org/document/6682732/

[2] R. Hussain, D. Kim, M. Nogueira, J. Son, A. Tokuta, and H. Oh, "A new privacy-aware mutual authentication mechanism for charging-on-the-move in online electric vehicles," in *2015 11th International Conference on Mobile Ad-hoc and Sensor Networks (MSN)*. IEEE, pp. 108–115. [Online]. Available: http://ieeexplore.ieee.org/document/7420932/

[3] S. Gunukula, A. B. T. Sherif, M. Pazos-Revilla, B. Ausby, M. Mahmoud, and X. S. Shen, "Efficient scheme for secure and privacy-preserving electric vehicle dynamic charging system," in *2017 IEEE International Conference on Communications (ICC)*. IEEE, pp. 1–6. [Online]. Available: http://ieeexplore.ieee.org/document/7997252/

[4] K. Rabieh and M. Wei, "Efficient and privacy-aware authentication scheme for EVs pre-paid wireless charging services," in *2017 IEEE International Conference on Communications (ICC)*. IEEE, pp. 1–6. [Online]. Available: http://ieeexplore.ieee.org/document/7996868/

[5] H. Li, G. Dan, and K. Nahrstedt, "Portunes+: Privacy-preserving fast authentication for dynamic electric vehicle charging," *IEEE Transactions on Smart Grid*, vol. 8, no. 5, pp. 2305–2313, 2017.

[6] M. Pazos-Revilla, A. Alsharif, S. Gunukula, T. N. Guo, M. Mahmoud, and X. Shen, "Secure and privacy-preserving physical-layer-assisted scheme for ev dynamic charging system," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 4, 2018.

[7] L. F. Roman and P. R. Gondim, "Authentication protocol in ctns for a cwd-wpt charging system in a cloud environment," *Ad Hoc Networks*, vol. 97, p. 102004, 2020.

[8] K. Hamouid and K. Adi, "Privacy-aware authentication scheme for electric vehicle in-motion wireless charging," in *2020 International Symposium on Networks, Computers and Communications (ISNCC)*. IEEE, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/9297199/

[9] X. Wu, G. Li, and J. Zhou, "A lightweight secure management scheme for energy harvesting dynamic wireless charging system," *IEEE Access*, vol. 8, pp. 224 729–224 740, 2020.

[10] P. R. Babu, R. Amin, A. G. Reddy, A. K. Das, W. Susilo, and Y. Park, "Robust authentication protocol for dynamic charging system of electric vehicles," vol. 70, no. 11, pp. 11 338–11 351, conference Name: IEEE Transactions on Vehicular Technology.

[11] "New registrations of electric vehicles in europe." [Online]. Available: https://www.eea.europa.eu/ims/new-registrations-of-electric-vehicles

[12] N. Rauh, T. Franke, and J. F. Krems, "Understanding the impact of electric vehicle driving experience on range anxiety," *Human factors*, vol. 57, no. 1, pp. 177–187, 2015.

[13] H. Wang, U. Pratik, A. Jovicic, N. Hasan, and Z. Pantic, "Dynamic wireless charging of medium power and speed electric vehicles," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 12, pp. 12 552–12 566, 2021.

[14] J. M. Miller, O. C. Onar, C. White, S. Campbell, C. Coomer, L. Seiber, R. Sepe, and A. Steyerl, "Demonstrating dynamic wireless charging of an electric vehicle: The benefit of electrochemical capacitor smoothing," *IEEE Power Electronics Magazine*, vol. 1, no. 1, pp. 12–24, 2014.

[15] R. Tavakoli and Z. Pantic, "Analysis, design, and demonstration of a 25-kw dynamic wireless charging system for roadway electric vehicles," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 6, no. 3, pp. 1378–1393, 2017.

[16] P. R. Babu, B. Palaniswamy, A. G. Reddy, V. Odelu, and H. S. Kim, "A survey on security challenges and protocols of electric vehicle dynamic charging system," *Security and Privacy*, vol. 5, no. 3, p. e210, 2022.

[17] Y. J. Jang, "Survey of the operation and system study on wireless charging electric vehicle systems," *Transportation Research Part C: Emerging Technologies*, vol. 95, pp. 844–866, 2018.

[18] S. Malladi, J. Alves-Foss, and R. B. Heckendorn, "On preventing replay attacks on security protocols," IDAHO UNIV MOSCOW DEPT OF COMPUTER SCIENCE, Tech. Rep., 2002.

[19] G. Carl, G. Kesidis, R. Brooks, and S. Rai, "Denial-of-service attack-detection techniques," *IEEE Internet Computing*, vol. 10, no. 1, pp. 82–89, 2006.

[20] A. Mallik, "Man-in-the-middle-attack: Understanding in simple words," *Cyberspace: Jurnal Pendidikan Teknologi Informasi*, vol. 2, no. 2, pp. 109–134, 2019.

[21] L. Tamilselvan and D. V. Sankaranarayanan, "Prevention of impersonation attack in wireless mobile ad hoc networks," *International Journal of Computer Science and Network Security (IJCSNS)*, vol. 7, no. 3, pp. 118–123, 2007.

[22] C. W. Probst, R. R. Hansen, and F. Nielson, "Where can an insider attack?" in *International Workshop on Formal Aspects in Security and Trust*. Springer, 2006, pp. 127–142.

[23] A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuellar, P. Hankes Drielsma, P.-C. Héam, O. Kouchnarenko, J. Mantovani, S. Mödersheim, D. Von Oheimb, M. Rusinowitch, J. Santos Santiago, L. Vigano, M. Turuani, and L. Vigneron, "The AVISPA Tool for the automated validation of internet security protocols and applications," in *17th International Conference on Computer Aided Verification - CAV 2005*, ser. Lecture Notes in Computer Science, K. Etessami and S. K. Rajamani, Eds., vol. 3576. Edinburgh, Scotland/UK, France: Springer, Jul. 2005, pp. 281–285. [Online]. Available: https://hal.inria.fr/inria-00000408

[24] R. Hussain, J. Son, D. Kim, M. Nogueira, H. Oh, A. O. Tokuta, and J. Seo, "Pbf: A new privacy-aware billing framework for online electric vehicles with bidirectional auditability," *Wireless Communications and Mobile Computing*, vol. 2017, 2017.

[25] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983.

[26] R. Canetti and H. Krawczyk, "Universally composable notions of key exchange and secure channels," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2002, pp. 337–351.

[27] E. Rescorla *et al.*, "Diffie-hellman key agreement method," 1999.

[28] D. Park, C. Boyd, and S.-J. Moon, "Forward secrecy and its application to future mobile communications security," in *International Workshop on Public Key Cryptography*. Springer, 2000, pp. 433–445.

[29] T. Coffey and P. Saidha, "Non-repudiation with mandatory proof of receipt," *ACM SIGCOMM Computer Communication Review*, vol. 26, no. 1, pp. 6–17, 1996.

[30] J. Zhou and D. Gollmann, "Observations on non-repudiation," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 1996, pp. 133–144.

[31] A. Pfitzmann and M. Hansen, "A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management," 2010.

[32] K. H. Johansson, M. Törngren, and L. Nielsen, "Vehicle applications of controller area network," *Handbook of networked and embedded control systems*, pp. 741–765, 2005.

[33] F. Fakhfakh, M. Tounsi, and M. Mosbah, "Cybersecurity attacks on can bus based vehicles: a review and open challenges," *Library hi tech.*, vol. 40, no. 5, 2022-11-22.

[34] S. Soderi, R. Colelli, F. Turrin, F. Pascucci, and M. Conti, "SENECAN: Secure KEy DistributioN OvEr CAN Through Watermarking and Jamming," *IEEE Trans. Dependable and Secure Comput.*, pp. 1–1, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9786611/

[35] P.-S. Murvay, L. Popa, and B. Groza, "Securing the controller area network with covert voltage channels," *Int. J. Inf. Secur.*, vol. 20, no. 6, pp. 817–831, Dec. 2021. [Online]. Available: https://link.springer.com/10.1007/s10207-020-00532-5

[36] S. C. HPL, "Introduction to the controller area network (can)."

[37] S. Kang, S. Han, S. Cho, D. Jang, H. Choi, and J.-W. Choi, "High speed CAN transmission scheme supporting data rate of over 100 Mb/s," *IEEE Commun. Mag.*, vol. 54, no. 6, pp. 128–135, Jun. 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7498099/

[38] T. Ramabadran and S. Gaitonde, "A tutorial on crc computations," *IEEE Micro*, vol. 8, no. 4, pp. 62–75, 1988.

[39] D. C. Latham, "Department of defense trusted computer system evaluation criteria," *Department of Defense*, vol. 198, 1986.

[40] S. Vanderhallen, J. Van Bulck, F. Piessens, and J. T. Mühlberg, "Robust authentication for automotive control networks through covert channels," *Computer Networks*, vol. 193, p. 108079, Jul. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1389128621001699

[41] S. Zander, G. Armitage, and P. Branch, "A survey of covert channels and countermeasures in computer network protocols," *IEEE Commun. Surv. Tutorials*, vol. 9, no. 3, pp. 44–57, 2007. [Online]. Available: https://ieeexplore.ieee.org/document/4317620

[42] B. Carrara and C. Adams, "Out-of-Band Covert Channels—A Survey," *ACM Comput. Surv.*, vol. 49, no. 2, pp. 1–36, Jun. 2017. [Online]. Available: https://dl.acm.org/doi/10.1145/2938370

[43] Q. Wang and S. Sawhney, "Vecure: A practical security framework to protect the can bus of vehicles," in *2014 International Conference on the Internet of Things (IOT)*. Cambridge, MA, USA: IEEE, Oct 2014, p. 13–18. [Online]. Available: https://ieeexplore.ieee.org/document/7030108

[44] M. Bozdal, M. Samie, S. Aslam, and I. Jennions, "Evaluation of can bus security challenges," *Sensors*, vol. 20, no. 8, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/8/2364

[45] J. C. Wray, "An analysis of covert timing channels," *Journal of Computer Security*, vol. 1, no. 3-4, pp. 219–232, 1992.

[46] X. Ying, G. Bernieri, M. Conti, L. Bushnell, and R. Poovendran, "Covert channel-based transmitter authentication in controller area networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 4, pp. 2665–2679, 2021.

[47] A. Michaels, V. S. Palukuru, M. Fletcher, C. Henshaw, S. Williams, T. Krauss, J. Lawlis, and J. Moore, "CAN Bus Message Authentication via Co-Channel RF Watermark," *IEEE Trans. Veh. Technol.*, vol. 71, no. 4, pp. 3670–3686, Apr. 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9684736/

[48] T. Ziermann, S. Wildermann, and J. Teich, "CAN+: A new backward-compatible Controller Area Network (CAN) protocol with up to 16× higher data rates." in *2009 Design, Automation & Test in Europe Conference & Exhibition*. Nice: IEEE, Apr. 2009, pp. 1088–1093. [Online]. Available: https://ieeexplore.ieee.org/document/5090826

[49] V. Berk, A. Giani, and G. Cybenko, "Detection of covert channel encoding in network packet delays," 2005.

[50] S. Zander, G. Armitage, and P. Branch, "A survey of covert channels and countermeasures in computer network protocols," *IEEE Communications Surveys & Tutorials*, vol. 9, no. 3, pp. 44–57, 2007.

[51] B. Groza, L. Popa, and P.-S. Murvay, "INCANTA - INtrusion Detection in Controller Area Networks with Time-Covert Authentication," in *Security and Safety Interplay of Intelligent Software Systems*, B. Hamid, B. Gallina, A. Shabtai, Y. Elovici, and J. Garcia-Alfaro, Eds. Cham: Springer International Publishing, 2019, vol. 11552, pp. 94–110, series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/978-3-030-16874-2_7

[52] ——, "Tricks—time triggered covert key sharing for controller area networks," *IEEE Access*, vol. 7, pp. 104 294–104 307, 2019.

[53] C. C. A. L. Dr. Ken Tindell, "Can-hg overview," https://canislabs.com/downloads/1905-2020-12-14-CAN-HG-overview.pdf.

[54] B. Groza, L. Popa, and P.-S. Murvay, "CANTO - Covert AutheNtication With Timing Channels Over Optimized Traffic Flows for CAN," *IEEE Trans.Inform.Forensic Secur.*, vol. 16, pp. 601–616, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9171355/

[55] D. P. Jablon, "Strong password-only authenticated key exchange," *ACM SIGCOMM Computer Communication Review*, vol. 26, no. 5, pp. 5–26, 1996.

[56] "Canis labs home." [Online]. Available: https://canislabs.com/

[57] J. Van Bulck, J. T. Mühlberg, and F. Piessens, "Vulcan: Efficient component authentication and software isolation for automotive control networks," in *Proceedings of the 33rd Annual Computer Security Applications Conference*, 2017, pp. 225–237.

[58] H.-C. von der Wense, "Introduction to local interconnect network," SAE Technical Paper, Tech. Rep., 2000.

[59] K. Matheus and T. Königseder, *Automotive ethernet*. Cambridge University Press, 2021.

[60] R. K. Chun, "Arinc 429 digital data communications for commercial aircraft," *Journal of Guidance, Control, and Dynamics*, vol. 6, no. 2, pp. 120–123, 1983.

[61] A. Mueller, T. Lothspeich, and R. Bosch, "Plug-and-secure communication for can," 2015.

[62] M. Hellman, "New directions in cryptography," *IEEE transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[63] S. M. Bellovin and M. Merritt, "Encrypted key exchange: Password-based protocols secure against dictionary attacks," 1992.

[64] S. Soderi, L. Mucchi, M. Hämäläinen, A. Piva, and J. Iinatti, "Physical layer security based on spread-spectrum watermarking and jamming receiver," *Transactions on emerging telecommunications technologies*, vol. 28, no. 7, p. e3142, 2017.