POLITECNICO DI TORINO

Master's Degree in Mechatronic Engineering

Master's Degree Thesis

Evolution of In-Vehicle Communication Systems: Automotive Ethernet and the role of MACsec in Cybersecurity

Supervisor:
Prof. Marco Rapelli

Candidate:
Paolo Scalone

Internship Tutor
Teoresi S.p.A.:
Eng. Andrea Orlando

Academic year 2025/2026

Acknowledgements

At the conclusion of this academic journey, I would like to express my sincere gratitude to all the people who made this thesis possible.

First of all, I would like to thank my supervising tutor, Ing. Andrea Orlando, for his constant support, guidance, and patience throughout this work. His availability and encouragement were fundamental, especially during the most challenging moments.

I would also like to thank Teoresi for giving me the opportunity to carry out this thesis within a professional environment and for allowing me to take my first steps into the real working world. My sincere thanks go as well to all my colleagues at Teoresi, who were always available and willing to help whenever needed.

I am deeply grateful to my supervisor, Prof. Ing. Marco Rapelli, for accepting to guide me in this project and for giving me the opportunity to approach the automotive field through such an interesting and stimulating topic.

A special thanks goes to my family, especially to my parents, Angelo and Angela, and to my brother Giuseppe, for their constant support and for always believing in me.

I would also like to thank my friends, who have always supported and encouraged me throughout these years.

Last but not least, I would like to thank Giorgia for her continuous support, patience, and for always standing by my side during this important period of my life.

Abstract

With the continuous advancement of technology and the increasing demand for advanced systems inside vehicles, the number of Electronic Control Units (ECUs) has significantly grown. This evolution requires new communication systems capable of handling large amounts of data while ensuring high speed and low latency. Modern vehicles increasingly integrate applications such as Advanced Driver Assistance Systems (ADAS), including Automatic Emergency Braking (AEB) and Adaptive Cruise Control (ACC), as well as Vehicle-to-Everything (V2X) communication, which enables vehicles to connect with other vehicles and surrounding infrastructure. For this reason, traditional in-vehicle communication systems such as CAN, MOST, and LIN, each with their own characteristics, are gradually being replaced by Automotive Ethernet.

Automotive Ethernet is a communication technology based on the IEEE 802.3 standard that provides high data rates while maintaining low latency even under high data traffic. Like standard Ethernet, it follows the OSI reference model. The main difference between standard Ethernet and Automotive Ethernet lies in the physical layer. In particular, Automotive Ethernet uses unshielded twisted pair (UTP) cables, typically consisting of a single pair of wires, allowing high transmission speeds while reducing wiring weight and overall vehicle cost.

With the adoption of Automotive Ethernet, the need for robust security mechanisms has become fundamental in order to protect the data exchanged between ECUs. One of the most suitable security protocols for in-vehicle networks is MACsec, standardized under IEEE 802.1AE. MACsec operates at the Data Link Layer (Layer 2 of the OSI model) and provides confidentiality, integrity, and authentication by modifying the Ethernet frame structure through the addition of a SecTAG and an Integrity Check Value (ICV). Secure communication is established through a key exchange mechanism that allows ECUs to authenticate each other before transmitting data. This ensures protection against various attacks such as eavesdropping, man-in-the-middle, packet modification, and replay attacks.

The effectiveness of MACsec was evaluated through an experimental setup using tools provided by Intrepid Control Systems, specialized in Automotive Ethernet

communication analysis. The practical implementation allowed the verification of MACsec behavior in a realistic in-vehicle communication scenario.

Table of Contents

List of Figures

List of Tables

Acronyms

AH Authentication Header.
AN Association Number.
AVB Audio Video Bridging.
AVTP Audio/Video Transport Protocol.

BMCA Best Master Clock Algorithm.

C Changed Text.
CA Connectivity Association.
CAK Connectivity Association Key.
CAN Controller Area Network.
CBS Credit-Based Shaper.
CKN Connectivity Association Key Name.

DoIP Diagnostic over IP.
DOS Denial-of-Service.
DTLS Datagram Transport Layer Security.

E Encrypted payload.
EAP Extensible Authentication Protocol.
EAPoL Extensible Authentication Protocol over LAN.
ES End Station.
ESP Encapsulating Security Payload.

FCS Frame Check Sequence.
FQTSS Forwarding and Queueing for Time-Sensitive Stream.

gPTP generalized Precision Time Protocol.

H-MTD High-Speed Modular Twisted-Pair Data.

IANA Internet Assigned Numbers Authority.
ICK Integrity Check Key.
ICV Integrity Check Value.
IEEE Institute of Electrical and Electronics Engineers.
IETF Internet Engineering Task Force.
IP Internet Protocol.
IPsec IP security.
IPv4 Internet Protocol version 4.
IPv6 Internet Protocol version 6.

KEK Key Encryption Key.

LLC Logic Link Control.

MAC Media Access Control.
MACsec MAC Security Protocol.
MII Media Independent Interface.
MKA MACsec Key Agreement.
MKPDU MACsec Key Agreement Protocol Data Unit.

OPEN One-Pair EtherNet.
OSI Open System Interconnection.

PN Packet Number.
PSK Pre-Shared Keys.

SAK Security Association Key.
SC Security Channel.
SCB Single Copy Broadcast.
SCI Secure Channel Identifier.
SCP SCI presence.

SecTAG Security TAG.
SFP Small Form-factor Pluggable.
SL Short Length.
SOME/IP Scalable service-Oriented MiddlewarE over IP.
SRP Stream Reservation Protocol.

TCI Tag Control Information.
TCP Transmission Control Protocol.
TLS Transport Layer Security.
TSN Time Sensitive Network.

UDP User Datagram Protocol.

V Version Number.
VSB Vehicle Spy Binary.

WAN Wide Area Network.

Chapter 1

Introduction

This first chapter of the thesis provides an overview of the evolution of the automotive industry, describing how vehicles have progressively evolved into systems characterized by electronic control units and network-based communication. Over the years, the increasing integration of electronic components has significantly changed the internal architecture of vehicles, leading to the development of advanced communication systems and new technological challenges.

In the next section, the goals of the thesis and its structure are introduced.

1.1 Goals and Thesis Structure

This work of thesis begins with a historical overview of the evolution of in-vehicle systems, describing the transition from a limited number of Electronic Control Units to numerous interconnected modules distributed following new in-vehicle architectures.

After introducing the most widely used in-vehicle communication protocols, the focus moves to Automotive Ethernet, analyzing its fundamental principles.

Furthermore, the thesis will concentrate on cybersecurity, with specific interest on the security protocol MACsec. The working principle of the MACsec is explained, and its behavior and performance are evaluated through an experimental setup, simulating an Automotive Ethernet environment.

The thesis is structured as follows:

Chapter 1 provides a historical overview of in-vehicle networks and their

evolution.

Chapter 2 focuses on automotive electronics, discussing system requirements and electronic architectures.

Chapter 3 displays the most used in-vehicle communication protocols.

Chapter 4 focuses on Automotive Ethernet, describing its characteristics, differences from standard Ethernet, and its operation principle across the OSI layers.

Chapter 5 introduces MACsec, explaining its mechanisms and its role in order to protect sensitive data transmitted within the vehicle network.

Chapter 6 describes the experimental setup and the tools used to evaluate the effectiveness of MACsec.

Chapter 7 contains the data collected during the simulation.

Chapter 8 concludes the thesis and suggests possible future improvements.

1.2 Literature Review

In the last years, with the large usage of Ethernet in vehicle domain and not, cybersecurity has been the subject of many studies from industry side, and also by academia.

Ethernet connections, even if they are useful for sending and receiving large amounts of data, were not originally designed with security in mind. This limitation became very important, in particular when networks usage expanded beyond simple local environments. For this reason, both academia and industry have focused their attention on developing and studying common security protocols. This protocol has been relevant since the 1990s, when the Internet started to become widely used in private and commercial applications. An important example is given by the article cited in [1], where the authors, in the early 2000s, highlighted the need for enhanced security mechanism in Local Area Networks, by discussing the common weakness of traditional Ethernet and proposing possible solutions to protect data exchanged into the network.

Moreover, the problem of security became crucial in the automotive domain. With the increasing number of Electronics Control Units inside modern vehicles, the amount of exchanged data has significantly increased, leading to a higher need

for secure communication. For this reason, many automotive suppliers started to focus on improving security mechanisms inside the vehicle network.

At the beginning, security was not considered a priority in in-vehicle networks. As a consequence, it was difficult for automotive companies to introduce security protocols, since no specific standards were initially defined for this purpose. So, in the early 2000s, several research consortia created projects with the common goal of improving in-vehicle security. The most well-known projects are E-safety Vehicle Intrusion proTected Applications (EVITA) [2] in 2008, Secure Vehicle Communication (SeVeCom) [3] in 2006, PRivacy Enabled Capability In Co-Operative Systems and Safety Applications(PRECIOSA) [4], and PREparing SEcuRe V2X communication systEms (PRESERVE) in 2010 [5]. These projects focused on providing secure and trustworthy communication both inside the vehicle and between vehicles, as well as between vehicles and infrastructure.

Moreover, research in automotive security mainly focused on the different communication protocols used inside the vehicle. Several works analyzed the different threats affecting bus communication protocols used in a vehicle, by studying potential attacks and corresponding countermeasures. Many of these studies concentrated on the most widely used in-vehicle communication protocol, the Control Area Network (CAN). For example, in the work cited in [6], the SAE international authors present a detection method for CAN buses that identifies spoofing attacks without requiring modification to the ECUs. Later, security research was extended to other communication protocols, such as for Local Interconnect Network (LIN) [7], and FlexRay [8].

The Automotive Ethernet protocol represents a new solution for in-vehicle networks. For this reason, the security topic in Automotive Ethernet networks is not yet extensively studied as other traditional bus systems. Due to its similarity with standard Ethernet communication, researchers and manufacturers initially referred to existing Ethernet security solutions, such as those discussed in LAN security studies [9], and tried to adapt them to the automotive environment [10]. However, many traditional security methods were not fully compatible with automotive requirements, especially in terms of high bandwidth and low latency constraints [10].

Among the available solutions, MACsec is considered one of the most suitable protocols, since it provides confidentiality, authenticity, and data integrity at the

link layer. Nevertheless, only a limited number of works try to implement MACsec in an Automotive Ethernet scenario [11]. Another relevant contribution is presented in the paper cited in [12], where the authors evaluate the performance of MACsec in a simulated Automotive Ethernet environment.

Although several studies, including those mentioned above, analyze MACsec from a theoretical or simulation-based perspective, limited research has been conducted on experimental validation using real automotive hardware and industry tools. In particular, there is a lack of experimental studies performed with professional validation tools such as those developed by Intrepid Control Systems.

1.3 Historical Overview

Technological advancements in electronics had a profound impact on the automotive industry, transforming vehicles from simple transportation devices into sophisticated machines. The evolution of electronic systems in the latter half of the 20th century significantly advanced the automotive industry, enabling the integration of sophisticated technologies to improve vehicle performance, safety, and efficiency.

In particular, starting from the 1970s, a true technological revolution took place: many mechanical and hydraulic systems were gradually replaced by electronic systems, leading to significant improvements in both passenger comfort and vehicle safety.

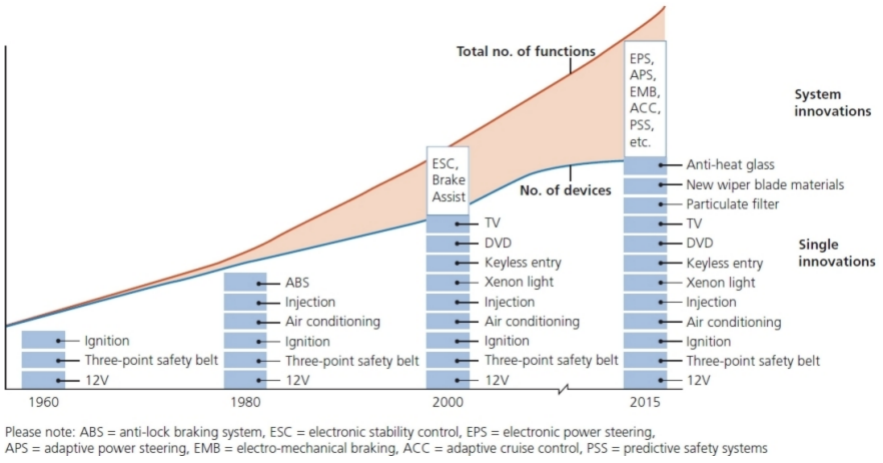


Figure 1.1: Evolution of Automotive Electronics over the years[13]

During these years, many electronic systems were added, which are also very important now, that assist the driver: they are regarding the steering, traction (control of the driving torque), or braking, like, for example, the ABS (Anti-lock Braking System), ESP (Electronic Stability Program) and EPS (Electric Power Steering). Moreover, the use of on-board systems also extends to the control of various devices in the vehicle, such as light, wipers, doors, and windows as well as infotainment and entertainment systems like radios, hands-free phones and navigation units.

These advancements are made possible by the widespread use of numerous Electronic Control Units (ECUs). Initially, the use of the ECUs was not very useful: until the 90s, all the ECUs were connected to each other, using a large amount of cables to compensate the large amount of communication channels. With the increasing of ECUs over the years, the number of wires and connectors also grew. The point-to-point method was not useful due to problems of weight, cost, complexity, and reliability.

In order to solve this problem, the automotive industries had to find a solution to reduce the amount of cabling inside a vehicle. For this reason, the use of industrial communication network or fieldbus became strictly needed.

In the beginning of the 1980s, Robert Bosch GmbH evaluated the serial bus systems to understand if they were suitable in the automotive industry. Since all the existing technologies were not appropriate, in 1986 Bosch presented "Automotive Serial Controller Area Network", the CAN. Initially, CAN was made in order to add new functionalities in the vehicle, but it was very useful to reduce all the cabling in the vehicle, replacing them with a shared network. The German automotive company BMW was the first firm to use the CAN protocol in their cars, in particular in 1986 with their 850 CSI model. Subsequently, other companies also used this protocol network, such as Mercedes-Benz, which in 1992 implemented it in all S series vehicles, where CAN was used for engine control and body and comfort electronics. After the invention of CAN by Bosch, other companies also tried to implement their own protocol on the basis of CAN, e.g. Volkswagen with the A-BUS (Automobile Bitserielle Universal-Schnittstelle) and Renault with VAN (Vehicle Area Network). For this reason, CAN was standardized initially in ISO 11519-2, and after two years with ISO 11898, in which there was an extended version of CAN.

In the following years, more topologies of CAN were created in order to surpass the problems of the original CAN. For example, to have more data on the bus, CAN-FD (Flexible Data) was created; or higher levels protocols were made, such as CAN Kingdom, CAN Open or DeviceNet, in order to have easier maintenance and more open to future developments.

In order to achieve different requirements, other network protocols were created, such as LIN, MOST, FlexRay and Automotive Ethernet. All the characteristics of these fieldbus protocols will be discussed later.

Figure 1.1 shows the evolution of the automotive electronics over the years. It highlights how the growing number of Electronic Control Units (ECUs) is closely related to the increasing integration of safety systems, as well as comfort and infotainment features. In fact, a typical family-size car contains around 70 ECUs, a number that increases in the case of luxury vehicles, which can feature up to 100 ECUs [14].

Nowadays, the complexity of the electronics in the cars increases day by day. The new technologies present in the car now are:

- **ADAS (Advanced Driver-Assistance)**: It is a technology that thanks to components like sensors (LiDAR, Radar), cameras and artificial intelligence help the driver during the road trip. It is possible to divide ADAS into six different levels, as shown in Figure 1.2.

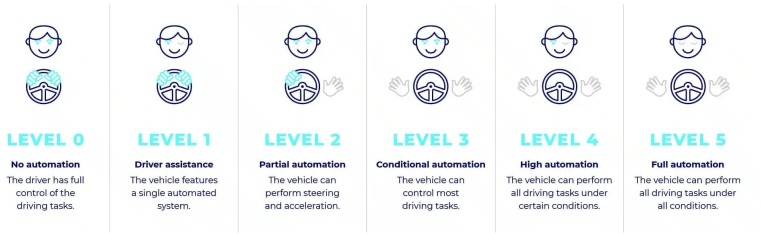


Figure 1.2: ADAS Levels [15]

- **V2X (Vehicle-To-Everything)**: this technology thanks to sensors, cameras and wireless connectivity (e.g. Wi-Fi or 5G), allows cars to communicate and share information with each other. Table 1.1 summarizes the main types of V2X communication.

  By combining all the information, the goal of V2X technology is to increase

Table 1.1: Different V2X communications[16]

Name	Description
V2V	Vehicle-to-Vehicles
V2I	Vehicle-to-Infrastructure
V2P	Vehicle-to-Pedestrian
V2N	Vehicle-to-Network

safety for both drivers and pedestrians, as well as to improve efficiency by, for example, optimizing driving routes and reducing fuel consumption [16].

The increasing complexity of in-vehicle networks, combined with the lack of built-in security mechanisms in legacy protocols, highlights the need for secure and high-bandwidth communication solutions. For this reason, the thesis will focus on Automotive Ethernet and, in particular, on the MACsec protocol as a candidate technology to guarantee confidentiality, integrity, and authentication.

Chapter 2

Automotive Electronics and Fundamental Requirements

2.1 Domain in Modern Vehicle

In the automotive industry, fundamental domains refer to the most important functional areas into which a vehicle's electronic components are logically divided. Organizing all the ECUs of a vehicle into well-defined domains is very important for managing development, integration, and safety.

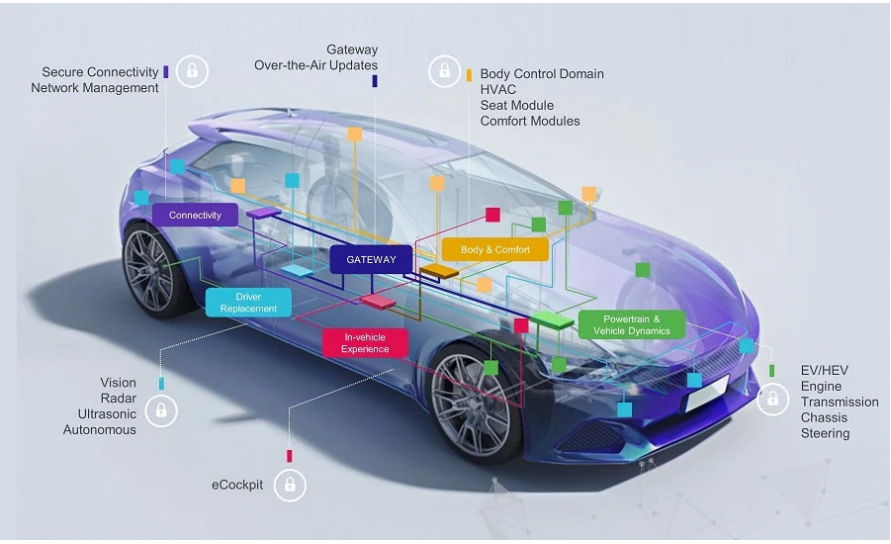


Figure 2.1: Graphical representation of automotive domains[17]

Each domain represents a specific area of vehicle functionality. The most traditional functional domains are[18]:

- **Powertrain**: This domain contains all the systems of a vehicle that play a crucial role in vehicle propulsion. Therefore, components such as the engine, transmission, and gearbox are part of it. Optimizing all systems in this domain is essential to improve parameters such as drivability, driving comfort, and fuel consumption.

  In order to improve the power and efficiency of the vehicle, advanced onboard computing units are needed to handle the data gathered by a large number of sensors. As a result, precise timing becomes essential: systems must be able to process and respond to information in microseconds, guaranteeing minimal latency and real-time operation[19, 20].

- **Chassis**: This domain includes all vehicle systems related to the interaction between the road and the vehicle, including all electronics that play an important role in vehicle driving dynamics, stability, and maneuverability. Typical components involved in this domain include wheels, suspension, steering, and braking systems.

  Like in the powertrain domain, sensors play a key role in this area. For this reason, it is essential that the system can respond in real time with minimal delay.

- **Body and Comfort**: This domain contains all the features that do not affect the movement or the drivability of the vehicle, but concentrate to the features that help the passengers during a trip. Climate control, roof, windows, door control, seat control are an example of system included in this domain.

  Since these features do not have an important role, they are not subject to strict performance constraints. As a result, these systems can operate using communication networks characterized by low bandwidth and higher latencies, making them a more economical solution[19, 20].

- **Multimedia, Infotainment, HMI**: This domain contains all the systems of a vehicle dedicated to the interaction between the driver and passengers

with the vehicle itself, in terms of infotainment, entertainment and control. This domain is different from the others because its goal is not the driving and the stability of the vehicle, but is the user experience. All of this is possible thanks to the Human-Machine Interface (HMI) that comprehends touchscreen, physical buttons and also vocal commands.

Infotainment includes systems that provide functionalities such as navigation, radio, audio/video playback, smartphone connectivity (e.g., via Bluetooth), and access to online services and apps. On the other hand, the multimedia part includes devices like central displays, rear-seat screens for passengers, and microphones used for phone calls.

Since these systems are not critical for vehicle safety, they do not require extremely fast response times. However, they do need a medium/high bandwidth to support video transmission, music streaming, and, especially system updates.

- **ADAS**: This domain includes all the features that can assist the driver during the driving process, as well as all the systems that can increase the safety, not only for the driver, but also for the passengers and pedestrians. Examples of driver assistance systems include navigation, cruise control, and automatic parking. On the other hand, systems such as the lane departure system, collision avoidance systems, blind spot detection, and intelligent speed adaption are more focused on increasing safety.

  All these safety systems receive data from the numerous sensors distributed around the vehicle. The collected signals are interpreted in order to support the driver to have the control of the vehicle and can act with almost all embedded vehicle systems. For these reasons, this domain requires a higher communication bandwidth with the sensors and must keep latencies very low to work correctly[19, 20].

A visual overview of the automotive domains introduced in this chapter is provided in Figure 2.1.

2.2 Requirements

As mentioned above, all the components in a vehicle have different tasks, some less important and others more important, such as those related to passenger safety. As a result, components in different domains are governed by distinct and domain-specific prerequisites[21]:

- **Fault Tolerance**: Faults, errors, and failure can cause a system malfunction. In order to have a system that can work also when a failure occurs, they are constructed using redundant hard- and software architectures. This can improve the ability to provide error containment.

- **Latency**: It is the control of the timing about the transmission and the receiving of the messages. A determinist communication system has the ability to know the timeline of a message.

- **Bandwidth**: It takes into account the amount of messages that a system has to transmit. High bandwidth requires higher cost, so in many cases cheaper communication buses with lower bandwidth are sufficient.

- **Flexibility**: It is the ability to deal with numerous event- and time-triggered messages and to manage large amount and/or number of messages on the network.

- **Security**: Very important parameter, fundamental to ensure the security of the system, especially when the messages exchanged are reachable from outside the vehicle.

Nowadays, in order to fulfill all these requirements, different field-buses communication technologies are used.

Table 2.1 presents all the domains previously introduced in Section 2.1 along with their main requirements. In particular, it includes specific values for latency and bandwidth. Among all domains, only ADAS includes cybersecurity as a system requirement.

Table 2.1: Communication Requirements by Automotive Domain[20, 21]

Domain	Fault Tolerance	Latency	Bandwidth	Flexibility	Security
Powertrain	Some	Yes $< 10\,\mu s$	Yes (Low)	Some	No
Chassis	Yes	Yes $< 10\,\mu s$	Some (Low)	No	No
Body & Comfort	No	Some $< 10\,ms$	Some (Low)	Yes	No
Multimedia	No	Some $< 10\,ms$	Some (Varies by system, increasing)	Yes	No
ADAS	No	Some $< 20\,\mu s$ or $< 1\,ms$	Yes (20–100 Mbps per camera)	Yes	Yes

2.3 Architectures

In the last decades, technological progress in the field of electronics had a strong impact on the automotive industry. Modern vehicles contain a large number of Electronic Control Units (ECUs), sensors and communication interfaces, increasing the overall complexity of the vehicle.

This has made the role of system architecture more and more important. It is not just about organizing components, but a well-designed architecture can help lower overall costs and can strongly affect aspects such as performance, energy efficiency, and the ability to meet the requirements mentioned in the previous section 2.2[22].

In order to make vehicles more efficient and high-performing in all aspects, the industry has moved from using domain architectures to adopting zonal architectures[23].

2.3.1 Domain Architecture

The Domain Architecture is represented in figure 2.2. All the electronics in the vehicle are organized into function groups, the domains, such as Powertrain, Body and Comfort, ... (see section 2.1).

Every domain in this architecture has a dedicated controller or gateway, that process all the data within that function. The presence of these controllers helps to reduce the overall loads and this makes it possible to solve more specific tasks.

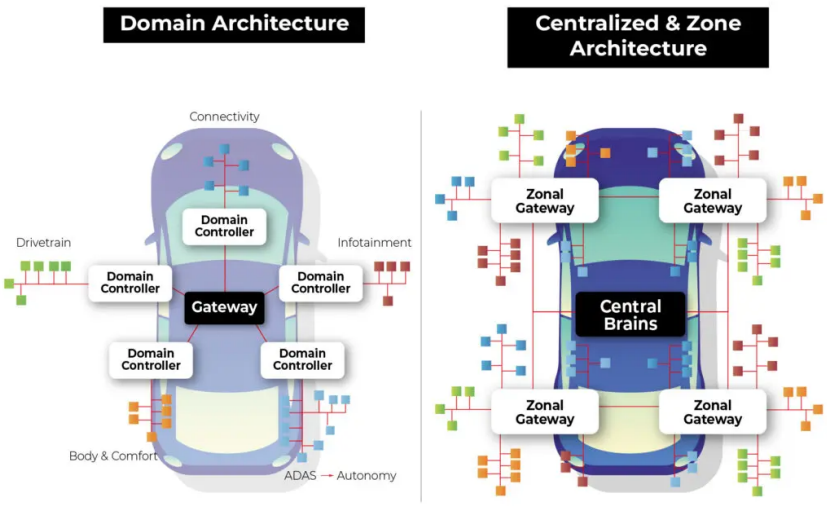


Figure 2.2: Domain (left) and Zonal (right) architectures

For this reason, this type of architecture is well suited for vehicles with specific functions that can work independently from each other.

This specific architecture does not reduce the amount of physical layers in the vehicle, such as cabling. Instead, its main advantage is the ability to simplify updates, expansion, and testing of a singular domain, without affecting the others. This means that changes can be made without modifying the architecture itself, or impacting the vehicle's global performance[22].

2.3.2 Zonal Architecture

To enhance the overall functionality of modern vehicles, automotive companies are increasingly pushing towards the adoption of zonal architectures(figure 2.2). The main goal of this approach is to divide the vehicle into different physical or geographical zones, such as front, rear, and sides. Each of these zones is managed by a zonal controller, responsible for handling all specific functions within its area, including lighting, sensors, and climate control.

All the zonal gateways are interconnected through a central processing module and communicate via Automotive Ethernet, enabling real-time data processing across the entire vehicle. Standards like 100BASE-T1 and 1000BASE-T1 are used to ensure high speed communication while keeping the amount of wiring low but

optimized for the vehicle[22].

Compared to domain-based architectures, which are more rigid and harder to scale, zonal architectures offer a modular and flexible design. This allows manufacturers to update, test, or expand individual zones without affecting the rest of the system. As a result, the total amount of cabling is significantly reduced (up to 40% less cabling), which helps to lower vehicle weight (20-30% weight reduction) and increase energy efficiency[24].

In fact, the zonal architecture gives more control to Original Equipment Manufacturers (OEMs) regarding the vehicle's software lifecycle. It makes possible high level maintenance operations through over-the-air (OTA) and firmware-over-the-air (FOTA) updates, and it is always connected on cloud to make it easier to introduce new features and/or improve existing ones, such as autonomous driving capabilities. This can help OEMs to adopt a service-oriented software architecture, for example by relocating real time control loops to the zone modules. The modular setup further simplifies system updates and diagnostics, as interventions can be performed directly in the affected zone, improving both scalability and maintainability. Additionally, zonal modules allow for more efficient power distribution and make it possible to power down unused components — a significant advantage for electric and hybrid vehicles[22, 23].

Alongside these advantages, however, there is an important issue related to the use of Automotive Ethernet: it is crucial to establish a secure communication channel. Today, it is pretty easy to find guides online on how to perform hacking attacks on Ethernet networks, and these can be replicated in vehicles. Figure 2.3 shows the main areas of the vehicle that can be hacked without cybersecurity protection. If security is compromised in the vehicle's network, all communication between ECUs becomes unreliable. For this reason, cybersecurity cannot be treated as a simple add-on, it needs to be approached in a holistic way, covering the entire vehicle lifecycle, from design to maintenance. To address this, new standards like ISO/SAE 21434 have been introduced specifically for automotive cybersecurity engineering, similar to ISO 26262, that is used for functional safety.

Due to the variety of data types and bandwidth requirements, it is not enough to apply traditional software-based solutions like IPsec for everything. While IPsec can work well for low-bandwidth control data, streaming audio or high resolution sensor data requires continuous communication and fast authentication. Doing all

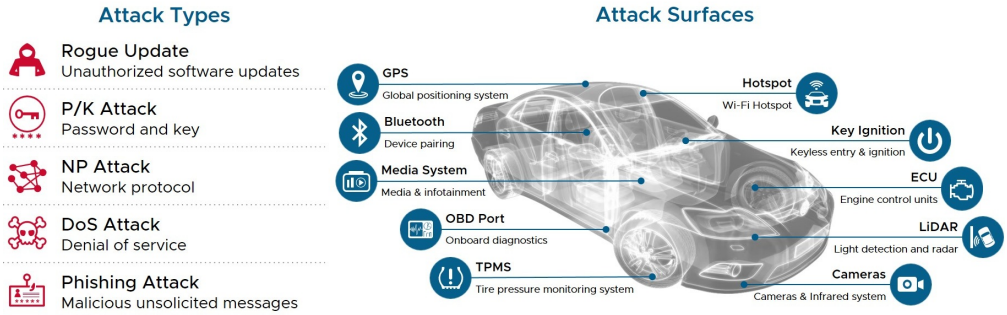


Figure 2.3: In-Vehicle attack points and surfaces[25]

this in software would add too much processing overhead. For this reason, more efficient and hardware-supported security solutions are being explored to ensure both safety and performance in modern vehicle networks.

One promising solution to overcome these problems is to use a lower-level encryption and authentication protocol like MACsec. It operates in Layer 1 or 2 of the Ethernet protocol stack and can be integrated directly into the Ethernet MAC or physical layer, to enable authentication, encryption of the payload, or both, without affecting the line rate[23].

Chapter 3

Communication protocols

As mentioned before, to ensure that all the systems in a vehicle, such as the engine control, braking system, infotainment, and body electronics, can communicate with each other, they are connected using linguistic frameworks, also known as automotive communication protocols. Without these communication languages, a vehicle would only be a collection of isolated ECUs, unable to exchange data, which would significantly reduce its overall cohesion and safety.

There are various communication protocols, some of which are more widely used than others, and each has its own characteristics. For this reason, different types of protocol can coexist within a vehicle. The choice of a specific communication protocol depends on the requirements of a given domain, such as bandwidth or latency.

3.1 CAN

The *Controller Area Network* (CAN) was first introduced by Bosch in 1986 and, thanks to its numerous advantages, it remained the most widely used communication network for more than 30 years.

Being an open ISO standard,CAN can be adapted by manufacturers to achieve specific requirements, which makes it highly flexible. Moreover, it is very easy to implement, since it is a "plug and play" solution. It is characterized by a UTP (*Unshielded Twisted Pair*) that lowers the cost of the network inside a vehicle. The two cables are designed as CAN High (CAN_H) and CAN Low (CAN_L), both

starting from a nominal voltage of 2.5 V. The communication is based on a binary signaling system with two possible states: the dominant state (logical 0), where CAN_H rises to 3.5 V and CAN_L drops to 1.5 V, producing a differential voltage of 2 V, and the recessive state (logical 1), where both lines remain at their nominal 2.5 V[26].

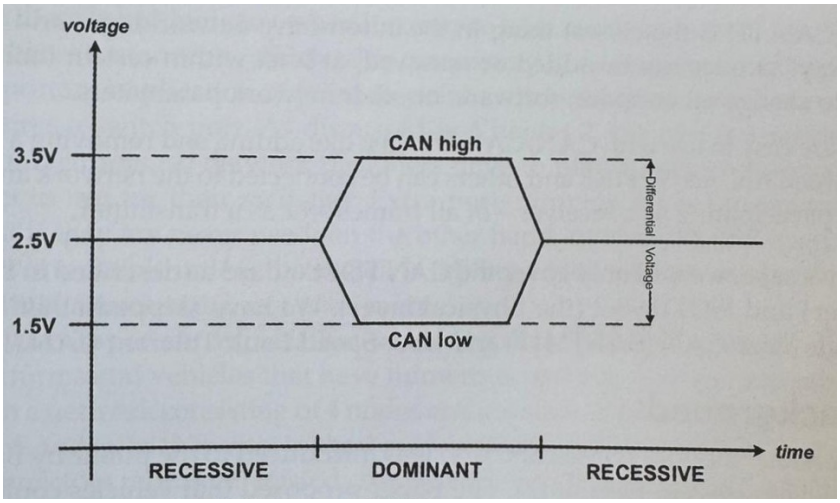


Figure 3.1: Nominal voltage for CAN High and CAN Low [26]

The maximum data rate of CAN goes from 125 Kbps up to 1 Mbps. It depends mainly on the bus length (higher the speed, shorter the length) with 1 Mbps achievable for 40 meters bus length. Secondly, the performance also depends on the number of connected nodes, which is typically limited to about 30 in order to maintain high-speed communication[18, 26].

To overcome these limitations, two extensions of the protocol were developed: CAN FD and CAN XL. Both provide higher data rates compared to standard CAN, with CAN FD supporting up to 5 Mbps and CAN XL up to 20 Mbps against 1 Mbps of regular CAN. They also allow for larger payloads: up to 64 bytes for CAN FD and up to 2048 bytes for CAN XL, compared to the 8-byte limit of classical CAN[18].

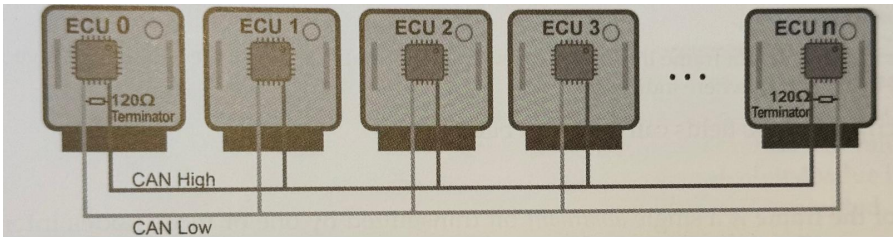


Figure 3.2: Linking of ECUs via the CAN bus[26]

3.2 LIN

The *Local Interconnect Network* (LIN) protocol was first introduced in the late 1990s, when companies such as BMW and Volvo considered CAN too complex and expensive for certain applications due to its UTP cabling. LIN was therefore developed as a low-cost solution, based on a single wire and without the need for a dedicated controller[26].

This network is based on master-slave architecture, where all of them share the same physical medium. An optimum configuration consists of a master and up to sixteen slave nodes, with a maximum bus length of 40 meters[26]. The single master node controls the communication by sending frame headers, while the slave nodes can only respond when requested by the master[27].

Due to its low cost and single-wire design, LIN supports a maximum speed of 20 Kbps, which is the main reason why this network is only used for low-bandwidth applications[27].

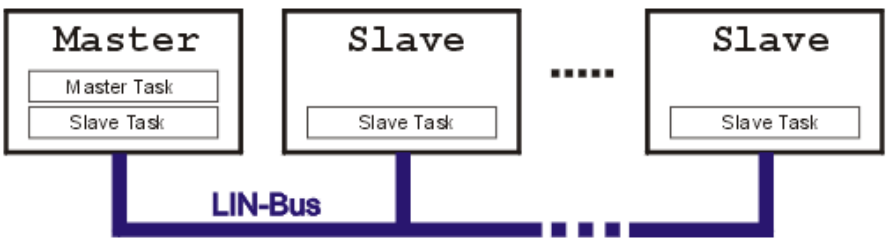


Figure 3.3: LIN topology [28]

3.3 MOST

The *Media Oriented System Transport* (MOST) was developed in 1998 by BMW, Harman and Daimler. As the name suggests, this network was specifically designed to handle in-vehicle infotainment systems, such as GPS, audio, speakers, and display screens.

MOST is based on a ring topology, in which data circulates sequentially through all nodes in the network. In the ring, one device acts as the Timing Master, responsible for clock synchronization and for continuously sending frames to the other devices, which operate as Timing Slaves. The maximum distance between ECUs is 20 m, and up to 64 nodes can be connected within a single ring. Using plastic optical fiber (POU) as a transmission medium, MOST can reach a speed rate of 150 Mbps[26].

Over the years, standard MOST has evolved to increase bandwidth, leading to different versions such as MOST25, MOST50, and MOST150. In particular, MOST150 introduced, in addition to synchronous, asynchronous and control channels, a dedicated channel for Ethernet communication[29].

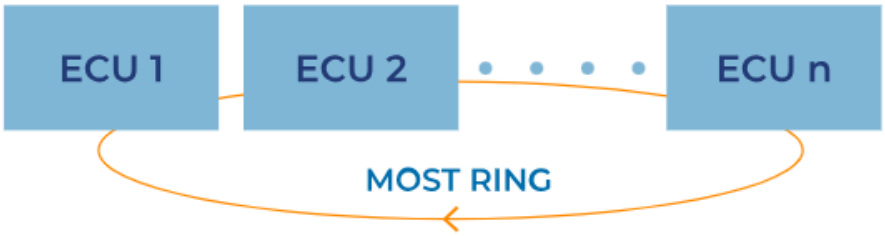


Figure 3.4: MOST topology [30]

3.4 FlexRay

In 2000, a consortium including companies such as Volvo, Daimler and Bosch, developed a new communication network to overcome the limitations of CAN in x-by-wire applications, such as steer-by-wire, brake-by-wire, and so on.

FlexRay can be configured in different topology options, including linear bus, passive star, active star, and point-to-point. In the linear bus topology, the network can reach a maximum length of 24 m with up to 22 connected nodes; these values

can be increased when using an active star configuration. FlexRay achieves data rates up to 10 Mbps and allows payloads of up to 256 bytes.

The main limitation of FlexRay is that is was designed primarly for x-by-wire applications; Therefore, it does not provide the bandwidth or protocol support needed for uses outside this domain. For instance, it is not suitable for infotainment systems, which makes it less flexible. Due to the low versatility, FlexRay is expected to be replaced by Automotive Ethernet[26].

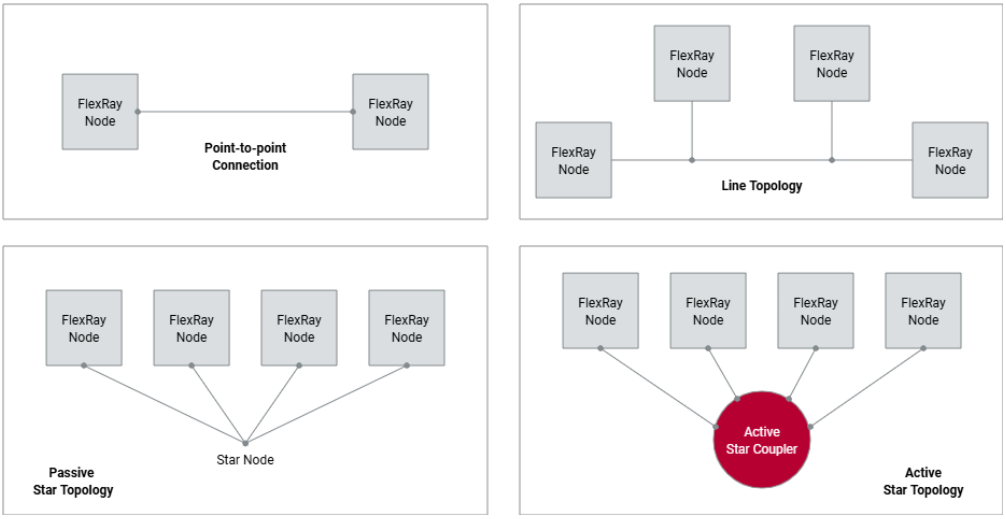


Figure 3.5: Different FlexRay topologies [31]

Chapter 4

Automotive Ethernet

Automotive Ethernet is an adaptation of the traditional Ethernet technology for the automotive domain, with its main differences lying in the physical layer. This layer is specifically designed to meet the fundamental requirements for automotive applications, such as high-speed communication between the various electronic components of a vehicle. For decades, standard Ethernet has been the reference technology for *Local Area Networks* (LANs). As a result, numerous transmission methods and protocols have already been developed, including TCP/IP, a protocol that enables Internet connectivity and supports essential services such as e-mail, web access, file transfer, and instant messaging. Another example is *Audio Video Bridging* (AVB), a communication standard designed to run over Ethernet, that allows high-quality real-time audio and video streaming. These protocols, each with different functions, are defined in standards developed by organizations such as the *Institute of Electrical and Electronics Engineers* (IEEE) and the *Internet Engineering Task Force* (IETF). Since protocols and utilities are designed to be compatible with any form of Ethernet, they are also available in Automotive Ethernet. The large availability of protocols and applications makes it possible to design innovative in-vehicle solutions, such as high-quality audio and video streaming or vehicle-to-infrastructure communication.

4.1 Standard Ethernet

The invention of Ethernet is attributed to Robert Metcalfe, who, together with colleagues at Xerox PARC, developed it in 1972. Initially, Ethernet was conceived as an experimental network to connect Alto computers and laser printers, with a maximum data rate of 2.94 Mbps. Later, in 1979, Xerox, together with Digital Equipment Corporation and Intel Corporation, formed an association with the goal of standardizing Ethernet and making it available to all the companies. This consortium, known as "DIX", launched the first version of Ethernet, the *Ethernet Version 1* or *DIX Ethernet Stardard Version 1*, operating at 10 Mbps.

In parallel, the IEEE initiated the development of IEEE 802.3 standard, which introduced some differences compared to the DIX specification, mainly at the Data Link and Physical Layer levels. In the 1990s, IEEE 802.3 defined the 100 Mbps standard, called 100BASE-TX, and in the early 2000s, the 10 Gbps (GbE) Ethernet standard was introduced.

4.2 OSI Layer

As mentioned before, the main differences between the standard Ethernet and the Automotive Ethernet are in the physical layer. It is defined by the IEEE 802.3 and OPEN (One-Pair EtherNet) Alliance standards. In order to understand all the functionalities of AE and its integration within the in-vehicle communication architecture, it is very useful to introduce the OSI reference model. The Open System Interconnection (OSI) model divides the communication network in 7 different layers, from the most concrete to the most abstract: Physical, Data Link, Network, Transport, Session, Presentation, and Application. It is important to note that the OSI reference model represents a conceptual framework, because in practice several protocols span across multiple layers. For example, the SOME/IP protocol operates across the Session, Presentation and Applications layers (5 to 7) and the AVB/TSN protocol does not use the layers from 3 to 7, as is possible to see from figure 4.1.

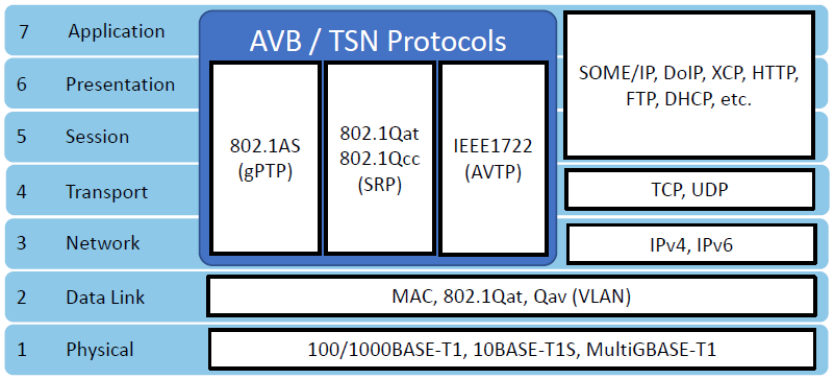


Figure 4.1: OSI Reference Model and Automotive Ethernet

4.2.1 Physical Layer

The first main difference between Standard Ethernet and Automotive Ethernet lies in the cabling. To meet the specific requirements of the automotive environment, the company Broadcom developed a new technology called *BroadR-Reach*, also known as *100BASE-T1*.

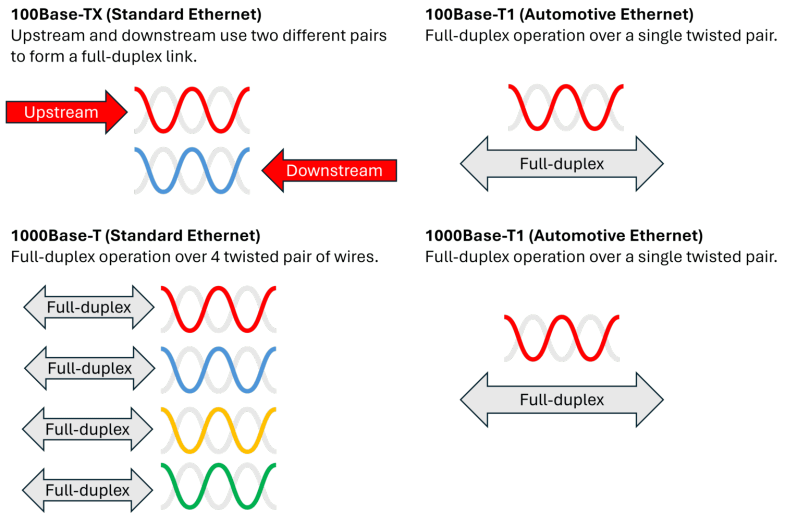


Figure 4.2: 100BASE-TX (Standard Ethernet) vs 100BASE-T1 (Automotive Ethernet)

Unlike other Ethernet standards, such as 100BASE-TX, the 100BASE-T1 standard complied with the radiated emission limits established by the Comité International Spécial des Perturbations Radioélectriques (CISPR) 25 Class 5[32].

For this reason, it is considered the first Ethernet standard suitable for automotive applications. It solves several in-vehicle challenges by providing higher data transmission speed (100 Mbps) using just one unshielded twisted pair (UTP) cable for sending and receiving data, as is possible to see in figure 4.2. This reduced the weight of the wiring and also lowered the overall cost of the vehicle's network.

In the following years, based on the model of the 100BASE-T1, new stardards were introduced, such as 1000BASE-T1 and MultiGBASE-T1. The notation of these standards follows a specific structure, which divides the acronym into three parts. For instance, in the format XXBASE-YZ, each element has a precise meaning:

- **XX**: indicates the data rate of the interface in Mbps. For higher speeds, the letter "G" is used to denote Gigabit;
- **BASE**: shows that the interface uses baseband signaling;
- **YZ**: defines the transmission medium. In particular, the "Y" can be "F" for fiber or "T" for twisted pair cabling. On the other hand, "Z" distinguishes among fiber or twisted-pair variations, and it can also be a number, that indicates the number of pairs being used: for example, T4 stands for four twisted pairs.

100BASE-T1

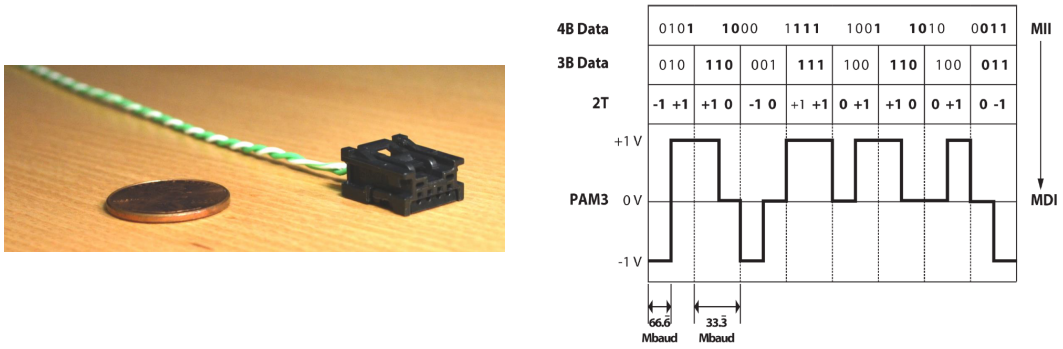
The 100BASE-T1, originally known as BroadR-Reach and later standardized by the IEEE as 802.3bw, was the first Ethernet standard specifically designed for automotive applications. As said before, 100BASE-T1 had a significant impact on in-vehicle networks: by using techniques such as superposition, encoding, and scrambling schemes, it reduced electromagnetic interference (EMI), cabling weight, cost, and footprint size. Moreover, it enabled data transmission and reception over a single unshielded twisted-pair cable, achieving a data rate of 100 Mbps with a maximum communication distance of 15 meters. With these characteristics, 100BASE-T1 allows the communication of different topologies of data, such as audio, video, connected car, firmware/software, and calibration data, in particular thanks to the Audio Video Bridging (AVB) standards.

100BASE-T1 uses a particular encoding process, involving 4-bit to 3-bit (4B3B), 3-bit to 2-ternary pair (3B2T) and three-level pulse amplitude modulation (PAM3).

A key role is played by the Media Independent Interface (MII), which links the MAC (Media Access Control to the PHY (Physical Layer) chip. In 100BASE-T1, the MII remains unchanged, as the standard preserves transparency at the MAC level. Four different variants of MII can be used, each with its own specific features:

- **MII**: 4-bit-wide data interface with receive and transmit controls and clocks;
- Reduced MII (**RMII**): 2-bit-wide data interface with receive and transmit controls and a single clock reference;
- Reduced Gigabit MII (**RGMII**): 4-bit-wide data interface with reduced pin count, including receive and transmit controls and clocks;
- Serial Gigabit MII (**SGMII**): 2-pin low-voltage differential signaling (LVDS) receive path and 2-pin LVDS on the transmit path.

After receiving the MAC data, the Ethernet PHY encodes, scrambles, and serializes the data, to prepare the transmission of the data over the single UTP cable.



(a) *Cable with connector.* **(b)** *PAM3.*

Figure 4.3: a) 100BASE-T1 cable and b) PAM3 encoding from MII to GMI

The first conversion, from 4 bit to 3 bit, is considered as a simply clock conversion, that converts the 4 bit sent by the MII at 25 MHz into 3-bit blocks with a clock rate of 33 1/3 MHz to maintain the 100 Mbps bit rate. After that, by using each group of three bits, pairs of ternary symbols are encoded, following a particular symbol mapping. 2 ternary symbols can have 9 possible values compared to 3 bits

that can have only 8: this leaves a single symbol pair unused. Therefore, the (0,0) combination is reserved only for control functions. In the end, the ternary pair vector is transmitted through 3-level pulse amplitude modulation, or PAM3. To maintain a nominal speed rate of 100 Mbps, 66 2/3 sets of ternary symbols per microsecond must be sent, yielding to a fundamental frequency of 66 2/3 MHz. The final signal is sent using three voltage levels ($+1\,V$, $0\,V$, $-1\,V$) with less than 2.2 V peak-to-peak.

1000BASE-T1

Even if 100Mbps was an enormous upgrade in automotive networks, in some cases this speed is insufficient. To address this limitation, in 2016 the 1000BASE-T1 standard (IEEE 802.3bp-2016) was introduced, allowing data rates of 1 Gbps over a single twisted-pair copper cable.

In general, 1000BASE-T1 is very similar to 100BASE-T1. A notable difference is the optional Auto-Negotiation layer, which allows the PHY to detect and adapt to the capabilities of the link partner. Regarding the Media Independent Interface (MII), only GMII, RGMII, and SGMII can be used, since the higher bandwidth of 1 Gbps exceeds the limits of the standard MII.

As in 100BASE-T1, after the reception of the data from the MAC through the MII, the Ethernet PHY encodes, scrambles, and serializes the data. The first step is an 80B/81B conversion: over ten clock cycles of data, 80 bits are encoded in 81-bit block containing both encoded data and control information. The first bit of the block indicates whether the content is purely data (0) or includes control information (1). After that, to compensate for bit errors, the Reed-Solomon Forward Error Correction (RS-FEC) is applied, adding 396 parity bits (or 44 symbols). In addition, 9 bits referring to Operations, Administration, and Maintenance (OAM) are inserted. The result is a PHY frame, also called a Reed–Solomon frame, which consists of 45 blocks of 81 bits plus the OAM and RS-FEC bits, for a total of 4050 bits. Finally, as for 100BASE-T1, 3B/2T encoding converts each set of 3 bits into a pair of ternary symbols.

The resulting symbols are then electrically transmitted using three voltage levels: low ($-1\,V$), zero ($0\,V$), and high ($+1\,V$), corresponding to the 3-level pulse amplitude modulation (PAM3)

To reach the nominal throughput of 1 Gbps, 1000BASE-T1 uses pairs of ternary symbols transmitted at 750 Mbps. This results in an effective nominal bandwidth greater than the target value, since $750\,\text{Mbps} \times (3\,\text{bits}/2\,\text{symbols}) = 1.125\,\text{Gbps}$.

MultiGBASE-T1

The IEEE 802.3ch standard covers three different speed grades: 2.5 Gbps, 5 Gbps, and 10 Gbps. Although these data rates may seem extremely high, they are particularly useful for transmitting uncompressed HD and 4K HD video streams across the vehicle.

Similarly to 1000BASE-T1, in the PHY architecture the Auto-Negotiation layer can be present. Regarding MII, in order to support the high data rate, it is fundamental to use at least one that supports this speed, such as XGMII, where the capital "X" is the Roman numeral 10, which stands for 10 Gbps.

After receiving data from the MAC, MultiGBASE-T1 performs a 64B/65B conversion. In this process, 32 bits of data over two clock cycles (64 bits in total) are mapped in a 65-bit block. The first bit, called *data/ctrl header* indicates if the block carries only data (1) or includes control information (0). Fifty 65B blocks are then divided into 10-bit units called *symbols*, to which an additional 10-bits of OAM information are added. Reed–Solomon Forward Error Correction (RS-FEC) is applied, adding 34 symbols, corresponding to 340 parity bits. Collectively, the fifty 65-bit blocks, the 10 OAM bits, and the 340 RS-FEC parity bits form a 3600-bit frame, known as a PHY frame or RS-FEC frame.

In addition, Side-Stream scrambling is applied to increase data randomization. After this step, Gray Mapping is used to prepare bit pairs for PAM4 modulation. In particular, in MultiGBASE-T1 the mapping works as follows: (0,0) $\mapsto$ 0, (0,1) $\mapsto$ 1, (1,1) $\mapsto$ 2 and (1,0) $\mapsto$ 3. These values are then associated with the four PAM4 signal levels: $0 \mapsto -1\,V$, $1 \mapsto -1/3\,V$, $2 \mapsto +1/3\,V$ and $3 \mapsto +1\,V$.

To achieve the nominal throughput of 2.5, 5, or 10 Gbps, MultiGBASE-T1 uses different clock frequencies for symbol transmission:

- 10G at 5.625 GHz
- 5G at 2.8125 GHz
- 2.5G at 1.41 GHz

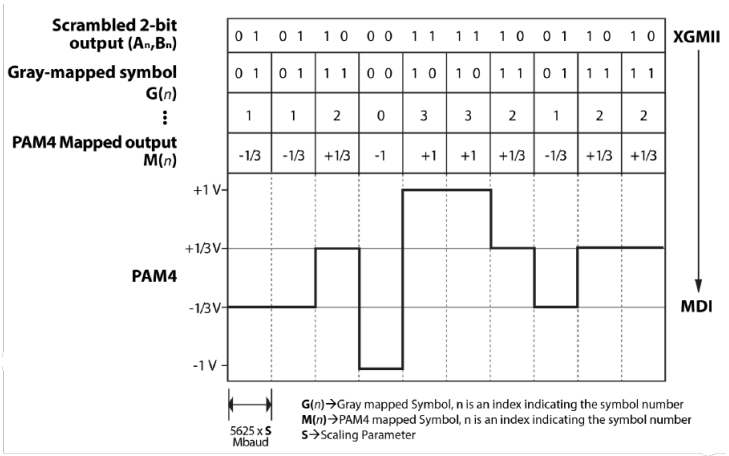


Figure 4.4: PAM4 encoding from XGMII to MDI

For istance, at 10Gbps the system operates at 5.625 GHz resulting in a data rate of $2\,\text{bits} \times 5.625\,\text{GHz} = 11.25\,\text{Gbps}$. Considering an overhead of about 10% for error correction, the effective rate is reduced to approximately 10 Gbps.

10BASE-T1S

Unlike the other standards, the 10BASE-T1S was not designed to increase bandwidth and/or latency, but to offer a slower speed, cheaper, and Ethernet-based option to the widely used automotive networking, such as CAN/CAN-FD or FlexRay standards.

10BASE-T1S, standardized by IEEE as 802.3cg, provides a maximum data rate of 10 Mbps over a bus length of 15-25 meters (the "S" stands for short-range). This makes it suitable for replacing many in-vehicle communication protocols, since more than 80% of them operate below 10 Mbps[33].

The main feature of 10BASE-T1S is its support for BUS or multi-drop topology, similar to traditional automotive networks. This removes the need for a switch and avoids the point-to-point limitation typical of standard Ethernet networks, as shown in Figure 4.5. The multi-drop approach also reduces the total number of PHYs required, lowering both costs and vehicle weight. Furthermore, adopting a single base technology for all in-vehicle networks removes the need for a complex gateway ECU, which is usually required to transfer data between different network types. Another benefit is that Ethernet message routing, MAC/IP remain consistent

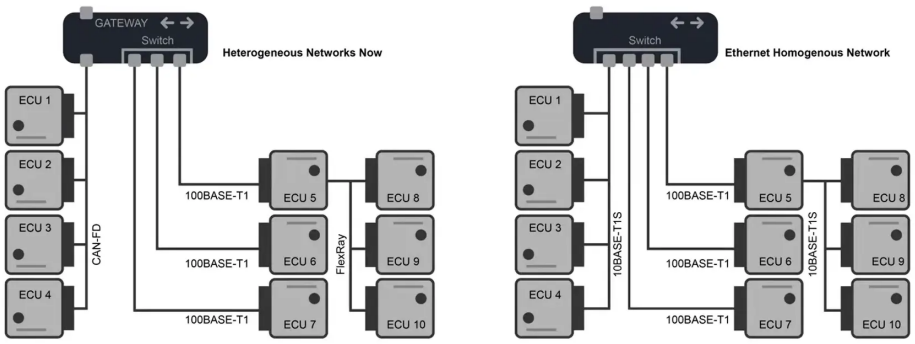


Figure 4.5: Example of Heterogeneous Networks (CAN/FlexRay/Ethernet) and Ethernet Homogeneous Network[26]

throughout the vehicle[34].

Since 10BASE-T1S supports multi-drop, three or more PHYs can share the same physical medium, meaning that they cannot all transmit simultaneously. To address this, a new mechanism, called Physical Layer Collision Avoidance (PLCA), was designed for 10BASE-T1S. PLCA coordinates the nodes, determining when each is allowed to transmit and preventing collisions. In PLCA each node is assigned an ID starting with 0. The first node, with ID 0, is responsible for sending a special data symbol, called a BEACON, which signals the start of a network cycle. During this cycle, each node is given a transmit opportunity (TO). If a node has data to send, it transmits a COMMIT during its TO to indicate that it will send data. After the COMMIT, the node sends data along with two other special symbols: *end of stream delimeter (ESD)* and *end of stream delimeter ok (ESDOK)* if the transmission occurs without errors. After a configurable bit times, typically 32, the TO expires, and the next node is given its TO. Once all nodes have transmitted, node ID 0 sends a new BEACON to start the next cycle. The duration of a PLCA cycle is not fixed: it can shrink or grow depending on how many nodes transmit during the cycle and also on the amount of data sent by each node. If no nodes transmit, as shown in Figure 4.6, the PLCA cycle its minimum, which is simply the sum of Node Count $\times$ TO (default 32 bits) plus the BEACON (20 bits).

On the other hand, as illustrated in Figure 4.7, the maximum PLCA cycle occurs when all nodes transmit the largest possible amount of data during their TO.

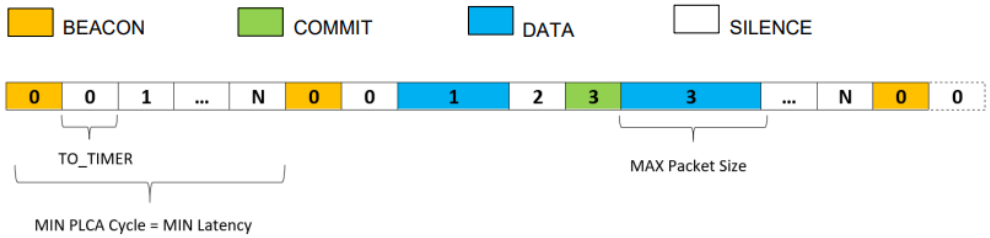


Figure 4.6: Min PLCA cycle[35]

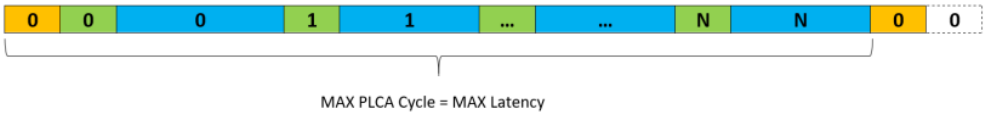


Figure 4.7: Max PLCA cycle[35]

At physical layer, 10BASE-T1S employs a simple two-level binary signal combined with *Differential Manchester Encoding (DME)*. The process begins with 4B/5B encoding, where four bits from the MII are converted to five bits, to enable the transmission of additional special symbols. Next, DME combines clock and data into a self-synchronizing data stream: each bit period always includes a transition in the middle, while the presence or absence of a transition at the beginning of the period distinguishes between a logical 0 and a logical 1. The final signal has a frequency of 25 MHz.

4.2.2 Data Link Layer

The second OSI layer, called Data Link Layer, is composed by the *Media Access Control (MAC)* and by the *Logical Link Control (LLC)*. The Data Link Layer is responsible for node-to-node delivery, framing, addressing and error detection[36]. In Ethernet-based networks, this functionality is realized through the transmission of data units known as *Ethernet frames*.

MAC Sublayer

The MAC sublayer acts as an interface to the Physical Layer, allowing the LLC and the upper OSI layers to exchange data without dealing with the specific characteristics of the transmission medium. Its main purpose is to detect eventual

signal collision, which occurs when multiple signals are transmitted at the same time, causing packet loss. This is achieved by performing multiple access resolutions and collision resolutions, ensuring that all signals are transmitted correctly without data loss.

In order to manage communication between multiple devices on the same network, each network interface must be identified in a unique way. This is possible thanks to the **MAC Address**.

The MAC address is 48 bits (6 bytes) long and is usually represented as six hexadecimal pairs. The address is divided into two parts:

- **Organizationally Unique Identifier (OUI)**: The first 3 bytes, assigned by the IEEE Registration Authority to each hardware manufacturer to ensure the global uniqueness.
- **Network Interface Controller (NIC) Specific**: The last 3 bytes, assigned by the manufacturer to uniquely identify each device.

The first byte of the OUI is used to determine whether a MAC address is locally or universally administered, as well as to indicate the type of addressing.

The second least significant bit is called the *U/L (universal/local) flag*. When this bit is set to 0, the MAC address is universally administered, while if it is set to 1, the MAC address is locally administered.

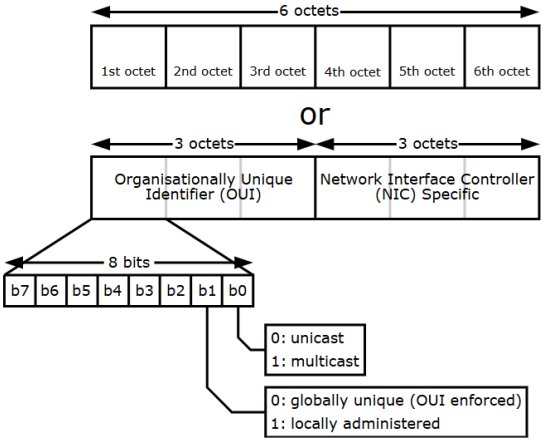


Figure 4.8: MAC Address[37]

Moreover, the first significant bit is called *I/G (individual/group) flag*. Its main

purpose is to identify the addressing type, which can be divided into three main categories[26]:

- **Individual**: The I/G flag is set to 0, meaning that the MAC address identifies a single device, and the message is sent as a unicast;
- **Group**: The I/G flag is set to 1, meaning that the MAC address is associated with a group of devices within a LAN (multicast);
- **Broadcast**: This MAC address represents all devices on a LAN. All bits are set to 1 (FF-FF-FF-FF-FF-FF), and its main purpose is to enable communication when the sender does not know the destination device.

Logic Link Control

The Logical Link Control (LLC) is defined by the standard IEEE 802.2. Its main function is to provide an interface between the lower sublayer of the Data Link Layer (the MAC sublayer) and the upper layer of the OSI model, the Network Layer.

The LLC provides conventional data link protocol functions, such as error control and flow control. Unlike other data link protocols, the LLC includes Service Access Point (SAP) information, which allows multiple applications on the same station to communicate simultaneously with applications on other stations within a network.

In a local network, the IEEE 802.2 LLC defines three different service types, known as *types of operation*, which indicate how protocol data units (PDUs) are sent and received[26, 38]:

- **Type 1 Operation: Unacknowledged Connectionless Service**: This is the simplest type of operation. There is no overhead to establish a connection, so PDUs are simply sent from one device to another. As the name suggests, there is no acknowledgment mechanism, meaning that the sender does not know whether the PDU has reached the destination or not.
- **Type 2 Operation: Acknowledged Connection-Oriented Service**: Unlike Type 1, this operation guarantees that PDUs are correctly exchanged between devices. Flow control and error detection mechanisms are also

implemented to ensure that the receiving device is not overwhelmed by the transmitting one.

- **Type 3 Operation: Acknowledged Connectionless Service**: This operation combines features of the previous two types. Although PDUs exchanged between devices are acknowledged, no prior connection is established between them.

Even if the main objective of LLC was to ensure connection between different types of network topologies, most LANs today, including Ethernet, primarily use the simplest mode of operation, Type 1, which essentially provides no link control between layers. In Ethernet, when connection establishment or acknowledgment is required, higher-layer protocols such as TCP in the TCP/IP stack are preferred.

Ethernet Frame

The Ethernet packet framework follows a precise structure, standardized by IEEE 802.3. It is composed by different fields, that provide information about the data being transmitted.

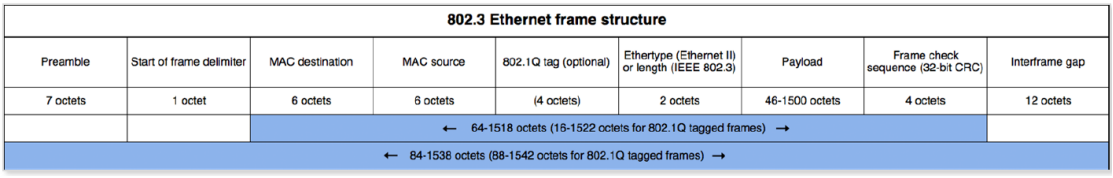


Figure 4.9: Ethernet Frame

In particular, as shown in figure 4.9, the frame can be divided in:

- **Preamble**

  It is a 7 bytes sequence that alternates 1s and 0s (10101010). Its main goal is to signal the transmission of a new packet and to synchronize the controller;

- **Start of Frame Delimiter (SFD)**

  It is a 1 byte sequence, with a fixed sequence 10101011 (0xAB). It is used to denote the start of the frame.

- **Destination Address**:

  It is a 6 byte hardware address which identifies the corresponding network node. The Destination address defines the receiver of the packet.

- **Source Address**:

  It is a 6 byte hardware address which identifies the corresponding network node. The Source address define the sender of the packet.

- **802.1Q or VLAN Tag**

  The VLAN Tag, defined by IEEE 802.1Q, is an optional 4 byte field inserted by the Source/Destination MAC address and the Length/Type field of an Ethernet frame. The use of the 802.1Q tagging brings several improvements, such as more efficient use of infrastructure, enhanced security, simplified network management, and improved performance[39]. It can be divided in 2 different parts: the *Tag Protocol Identifier (TPID)* and *Tag Control Information (TCI).*

  - the **TPDI** is a special Ethertype, which has a value of 0x8100, that marks the frame as VLAN-tagged.
  - the **TCI** is further divided into three subfields:
    * **Priority Code Point (PCP)**: 3 bits that indicate the frame priority level;
    * **Drop Eligible Indicator (DEI)**: 1 bit flag; if it is set to 1, the frame can be discarded in case of congestion;
    * **VLAN Identifier (VID)**: 12 bits that identify the VLAN, ranging from 0 to 4095 (0xFFF). A value of 0 means the frame is not associated with any VLAN, while 0xFFF is reserved.

- **Ethertype/Length**:

  It is a 2-byte field used to specify the type of data carried in the payload and to indicate the higher-layer protocol being used (for example, IPv4)[40]. Initially, this 16 bits was used to specify the length of the frame. Nowadays, if the value is lower than 1536, it indicates the length, otherwise it indicates the ethertype. The most common Ethertype values are shown in Table 4.1.

Table 4.1: EtherType Values by Protocol Category

Category	EtherType	Protocol
General Use	0x0800	IPv4
	0x86DD	IPv6
	0x0806	Address Resolution Protocol
	0x8100	VLAN - Single Tag
	0x9100	VLAN - Double Tag
AVB/TSN	0x22F0	IEEE 1722
	0x88F7	Generalized Precision Time Protocol
V2X	0x88DC	Wave Short Message Protocol

- **Data**

  It is a variable-length field that contains the actual data being transmitted. Its minimum length is 46 bytes without a VLAN tag, or 42 bytes with a VLAN tag [40]. Originally, the maximum payload length was 1500 bytes; however, larger sizes are now supported through extensions known as jumbo frames. In particular, a standard jumbo frame can carry up to 9000 bytes, while a super jumbo frame can reach up to 64,000 bytes [41].

- **Frame Check Sequence (FCS)**

  It is a 4 byte field primarily used for error detection in the frame. The sender computes the FCS using a Cyclic Redundancy Check (CRC) algorithm, and the receiver recalculates it to check if the frame was received correctly. If the calculated FCS does not match the FCS field in the frame, it indicates that the frame was corrupted during transmission, and the receiver discards it[26].

4.2.3 Network Layer

The third layer of the OSI model is the Network Layer. This layer performs several key functions, including routing control and congestion control. Routing control is responsible for maintaining routing tables and evaluating optimal routing paths for data transmission across the network. Congestion control manages the packets in the queue to avoid packet loss when there is no more space available. Moreover, the Network Layer handles multiplexing multiple logical connections over a single

data link to optimize bandwidth usage[38].

In modern network architectures, the Network layer relies primarily on the Internet Protocol (IP) to perform these functions. IP specifies how packets are addressed and routed across different networks. Over time, two main versions of this protocol have been developed: Internet Protocol version 4 (IPv4) and Internet Protocol version 6 (IPv6)[42].

Both protocols serve the same fundamental function, which is to assign a unique address to every device connected to a network. However, they differ in several important aspects that reflect the evolution of networking needs over time.

IPv4

IPv4 is the fourth version of IP and has been in use since the early 1980s, and it is still widely used today. An IPv4 address consists of 32-bit binary numbers. To make it easier to read, the address is first divided into four 8-bit sections, called octets. Each octet is then converted in hexadecimal and, to make it easily readable by humans, it is converted to a "Dotted Decimal" notation. In this format, an IPv4 address is written as four decimal numbers separated by dots, each ranging from 0 to 255. The smallest possible address is 0.0.0.0, while the largest is 255.255.255.255[26].

This configuration allows for a theoretical address space of 2^{32}, or 4,294,967,296 addresses. However, not all of these addresses can actually be used.

The 32-bit address can be divided into two components:

- **Network Identifier (Network ID)**: identifies, using a certain number of bits, the network where the host and/or other network interface is located;
- **Host Identifier (Host ID)**: identifies, using the remaining bits, the specific host within the network.

To distinguish between these two identifiers, a subnet mask is used.

As shown from the example represented in Figure 4.10, the IP address 192.168.1.34 can be divided into two parts using the subnet mask 255.255.255.0. In particular, the first part (192.168.1) represents the Network ID, while the (34) represents the Host ID.

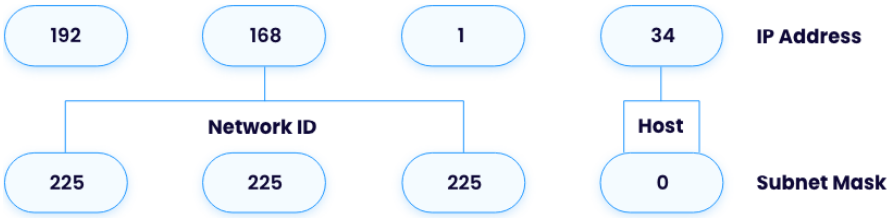


Figure 4.10: An example of Network ID, Host ID and Subnet mask[43]

Table 4.2: IPv4 address classes with their characteristics [26]

IP Address Class	Fraction of Total IP Address Space	Number of Network ID Bits	Number of Host ID Bits	IP Address Range
Class A	1/2	8	24	1.0.0.0 to 127.255.255.255
Class B	1/4	16	16	128.0.0.0 to 191.255.255.255
Class C	1/8	24	8	192.0.0.0 to 223.255.255.255
Class D	1/16	n/a	n/a	224.0.0.0 to 239.255.255.255
Class E	1/16	n/a	n/a	240.0.0.0 to 255.255.255.255

Furthermore, IP addresses are categorized into five classes: A, B, C, D, and E, each of which contains a subset of addresses assigned for specific purposes[44].

As shown in table 4.2, the first three classes (A, B, and C) occupy the majority of the total address space and are used for IP unicast addressing. The choice of class depends on the number of hosts an organization needs to connect to the Internet: Class A is intended for large organizations with hundreds of thousands or even millions of hosts, Class B for medium to large organizations with several hundred or thousands of hosts, and Class C for small organizations with up to around 250 hosts. The last two classes, D and E, are used for specific purposes: Class D is used for multicasting IP addressing, while Class E is reserved for experimental or future use[26, 45].

IPv6

With the rapid growth of the Internet after the introduction of IPv4, it became evident that this protocol could no longer provide enough unique addresses. This limitation led the Internet Engineering Task Force (IETF) to develop a new version of the protocol, known as IPv6[44].

IPv6 offers a larger address space than IPv4, in particular is composed by 128 bits. The theoretical number of possible addresses is now equal to 2^{128} possible combinations, approximately 340 trillion trillion trillion unique combinations[26]. The IPv6 addresses are represented as eight groups of 16-bit hexadecimal numbers, separated by colons.

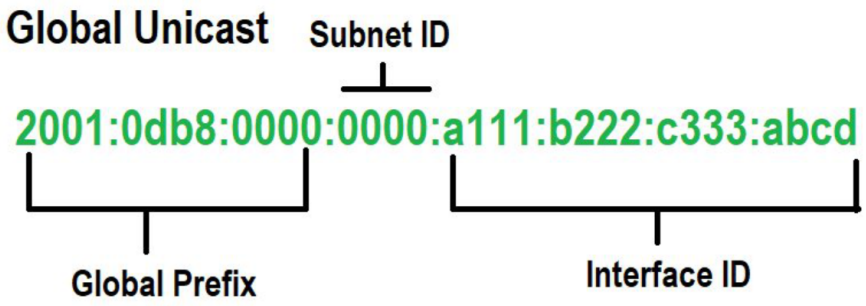


Figure 4.11: IPv6 Unicast Address

IPv6 addresses can be classified into three types: Unicast, Multicast and Anycast[26]. Figure 4.11 shows an example of a 16 bytes hexadecimal IPv6 address, in this case Unicast address.

A Unicast address is structured in three main parts[26]:

- **Global Routing Prefix**: Represents the network identifier or routing prefix. It is used for global routing and is typically 48 bits long;
- **Subnet ID**: Identifies a specific subnet within the organization or site. It has a length of 16 bits.
- **Interface ID**: A unique identifier assigned to each interface (host or device). It is unique within the given prefix and subnet, and its length is 64 bits.

Since the transition between IPv4 and IPv6 cannot happen instantly, IETF defined different methods to allow the two protocols to work together in a reliable

and automatic way, avoiding communication errors. In particular, three main techniques are used[44, 46]:

- **"Dual Stack" Devices**: Since IPv6 is an evolution of IPv4 and maintains many of its features, devices can be configured to support both protocols. This allows them to communicate with both IPv4 and IPv6 hosts;
- **IPv4/IPv6 Translation**: Some devices can receive requests from IPv6 hosts and convert them into IPv4 datagrams, making it possible to reach IPv4 destinations;
- **IPv4 Tunneling of IPv6**: To ensure the coexistence of IPv6 within the existing IPv4 infrastructure, IPv6 packets can be encapsulated inside IPv4 packets and transmitted through a point-to-point tunnel. In this configuration, IPv4 acts as a link layer for IPv6 traffic.

4.2.4 Transport Layer

The IP, described previously, is a connectionless, unreliable, and unacknowledged protocol. It provides an unreliable datagram service that must be implemented by all system addressable on the internet.

For this reason, Transport Layer plays a fundamental role: it provides an end-to-end communication service to applications running on the end hosts[47]. This is achieved through protocols such as the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP). Both operate above IP and are necessary to send data to the upper layers.

A typical host connected to the internet runs multiple processes simultaneously, each with the ability to send and receive data. Since all communication passes through the same network interface via the IP layer, the Transport Layer is responsible for managing the data flow between multiple applications. This is possible by performing multiplexing, which combines data from different applications for transmission, and demultiplexing, which ensures that the received data are delivered to the correct application.

During multiplexing and demultiplexing, ports are used at the Transport Layer to distinguish between different processes or services running on the same host. Each port number is 16 bits long, so they can range from 0 to 65,535. The

assignment of the port number is managed by the Internet Assigned Numbers Authority (IANA), to ensure efficient use and to provide flexibility for organizations that need custom or less common applications. For this reason, port numbers are divided into three ranges:

- **Well-Known (Privileged) Port Numbers**: From 0 to 1,023, representing ports reserved for widely used services (see Table 4.3);
- **Registered (User) Port Numbers**: From 1,024 to 49,151, representing port numbers registered to specific applications by IANA;
- **Private/Dynamic Port Numbers**: From 49,152 to 65,535, representing port numbers that are neither reserved nor registered by IANA.

Table 4.3: Common Well-Known port numbers and their corresponding protocols[48]

Port Number	Description
20/21	FTP data and FTP control
22	Remote login protocol secure shell (SSH)
25	Simple Mail Transfer Protocol (SMTP)
25	DNS protocol
80	HTTP (Hypertext Transfer Protocol)
143	Internet Message Access Protocol (IMAP)
443	HTTP over Secure Sockets Layer (SSL)
631	Internet Printing Protocol (IPP)

User Datagram Protocol

The UDP standard is defined by the IETF in RFC 768. It is a connectionless protocol, which makes it very fast, as reliability and efficiency in network communication are often in contrast: improving one typically reduces the other[49].

In other words, UDP is a lightweight protocol, that does not establish a connection before transmitting data. It does not provide acknowledgments, error detection, or guarantees that messages will arrive or be received in the correct order.

The main role of UDP is to take data from higher-layer protocols, encapsulate it into UDP datagram, and pass them to the IP layer for transmission.

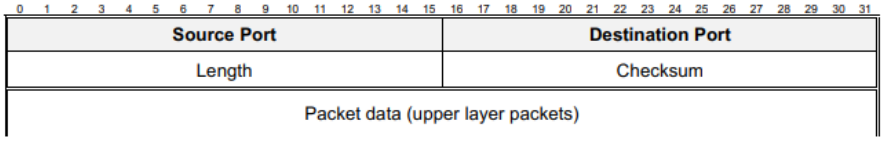


Figure 4.12: UDP Header and its fields[50]

The UDP header is 8 bytes long and consists of four distinct fields, as described in Table 4.4.

Table 4.4: Fields of an UDP Header

Field Name	Size (Bytes)	Description
Source Port	2	Identifies the port number of the sending process
Destination Port	2	Identifies the port number associated with the destination process
Length	2	Specifies the total length of the UDP datagram, including both header and data
Checksum	2	Verifies the integrity of the datagram

Transmission Control Protocol

UDP is a simple protocol, suitable for client-server interactions and multimedia transmissions. However, for most Internet applications, reliable delivery is crucial, which UDP does not provide. For this reason, TCP was introduced in 1981 through RFC 793.

The TCP was specifically designed to provide a reliable, end-to-end byte flow over an unreliable internetwork. Unlike a single network, an internetwork can include segments with different characteristics, such as topology, bandwidth, delay, or packet size. TCP dynamically adapts to these variations and maintains robust performance even in the presence of transmission error or network congestion.

The main responsibility of TCP is to transmit datagrams correctly, making efficient use of the available network capacity without causing congestion. It also retransmits lost segments and reorders any out-of-sequence data to reconstruct the original message correctly.

The TCP Header, shown in figure 4.13, is composed by 11 fields:

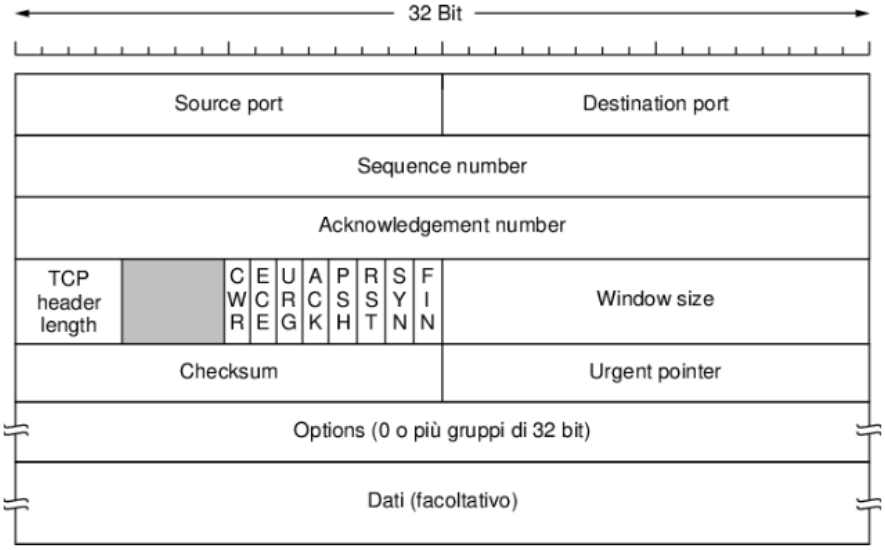


Figure 4.13: TCP Header[50]

- **Source/Destination Port**: 16 bits each; these fields identify the port number of the sending application and the port number of the receiving application, respectively.
- **Sequence Number**: 32 bits field that specifies the sequence number of the TCP segment, allowing the receiver to detect missing or duplicate segments.
- **Acknowledgment Number**: 32 bits field indicates the byte number that the receiver expects to receive next, not the last byte received correctly (Cumulative Acknowledgment).
- **TCP Header Length**: 4 bits field that indicates the length of the TCP header in 32-bit words. The header can range from 20 to 60 bytes, so this field can hold values from 5 (indicating 20 bytes) to 15 (indicating 60 bytes).
- **Reserved**: Field composed by 4 bits unused.
- **Flag Bits**: 8 bits, each one with a specific purpose. (Table 4.5)
- **Windows Size**: A 16 bits field that specifies the maximum number of bytes that can be sent starting from the last byte acknowledged.

Table 4.5: Flag Bits[51]

Name	Description
CWR	*Congestion Windows Reduced*: signals congestion control information between TCP endpoints.
ECE	*ECN-Echo*: set when the local TCP detects network congestion and requests the sender to reduce the transmission rate.
URG	*Urgent*: indicates that the Urgent Pointer field is valid, meaning certain data in the segment should be processed immediately.
ACK	*Acknowledgment*: when set to 1, the Acknowledgment Number field is valid; if 0, it is ignored.
PSH	*Push*: requests the receiver to deliver data to the application immediately upon arrival, without waiting to fill the buffer.
RST	*Reset*: used to reset an unstable connection or reject invalid segments or connection attempts.
SYN	*Synchronize*: used to establish a TCP connection.
FIN	*Finalize*: used to terminate a TCP connection (the sender has no more data to send).

- **Checksum**: 16 bits field that indicates the presence of error in the header and data.
- **Urgent Pointer**: 16 bit field, used when the URG flag is set. It indicates the end of urgent data within the segment.
- **Options**: A variable bit length field, always a multiple of 32, that allows to add additional features that are not included in the standard header.

4.2.5 Application Layer

The Application layer represents the higher level in the OSI model. It specifies the application programs or processes that provide services, independent of the physical location of the service provider or the user. The lower layers handle reliable data transport but operate below the user level, without providing direct services to end users. The main function of the Application Layer is to facilitate communication and data exchange between applications on different network devices. Through a variety of protocols, each designed for a specific purpose, this layer enables applications to interact and communicate efficiently across the network.

The most common Application Layer protocols include HTTP, SMTP, FTP, and SNMP[49].

In the context of Automotive Ethernet, the key protocols at this layer are SOME/IP and DoIP.

SOME/IP

Scalable service-Oriented MiddlewarE over IP (SOME/IP) is a service-oriented protocol that functions as automotive middleware, enabling communication between applications distributed across different ECUs in a vehicle. The main key aspects of the protocol are its service-based communication approach, small footprint, full compatibility with AUTOSAR, high scalability for use on both very small and very large platforms, and greater flexibility compared to other operating systems used in automotive applications, such as AUTOSAR, OSEK, QNX, and Linux [41].

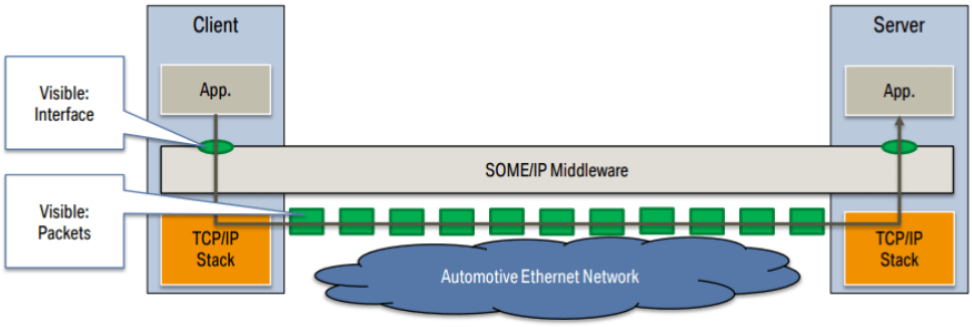


Figure 4.14: SOME/IP Overview [52]

As the name suggests, this protocol acts as a middleware, which means an intermediate software layer that provides all the necessary services to enable data exchange between independent software components, such as ECUs in the automotive field. It makes the underlying network transparent and simplifies both the visualization and the management of the exchanged messages.

One of the most important features of the SOME/IP protocol is the Service Discovery. Using IP multicast and the so-called SOME/IP-SD messages, this module allows ECUs to detect and announce the available services within the vehicle network. The Service Discovery enables dynamic communication between ECUs by sending and receiving multicast messages, such as “OfferService” and “FindService”. In this way, ECUs can automatically find each other and exchange information without a fixed configuration. This mechanism simplifies system

integration and allows a flexible architecture, where services can appear or disappear during operation.

The SOME/IP-SD header is added to the normal header configuration of the SOME/IP protocol, as shown in Figure 4.15.

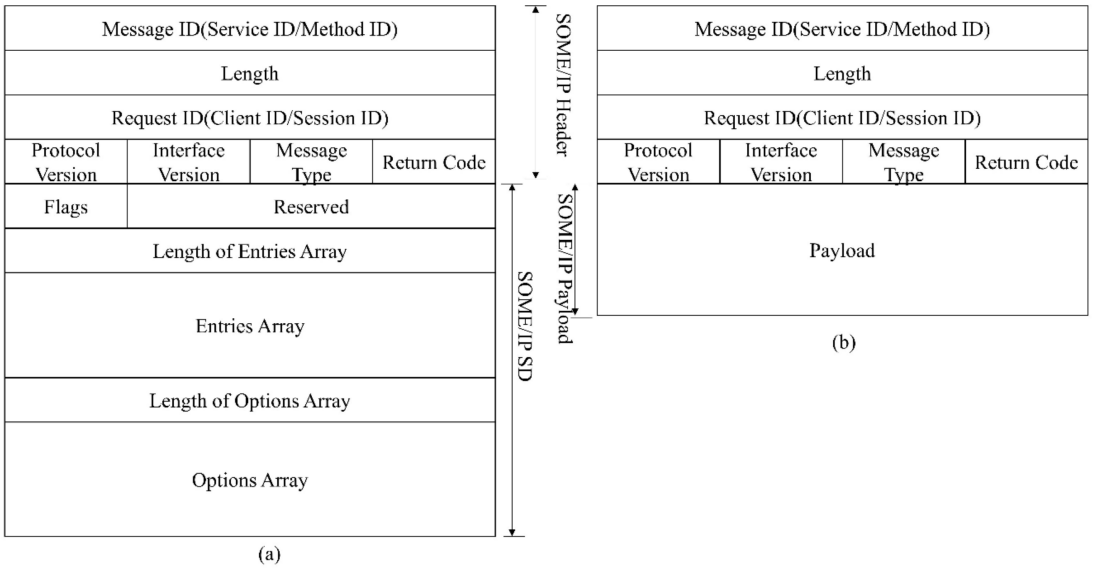


Figure 4.15: SOME/IP and SOME/IP-SD Header [**SomeIP-SD_Attack**]

The principal fields of an ordinary SOME/IP header can be explained as follows[41]:

- **Message ID**: 32 bits. The first 16 bits represent the Service ID, which uniquely identifies a service. A typical service consists of a set of methods, events, and fields, described by the Method ID, represented by the last 16 bits.
- **Length**: 32 bits. This field specifies the total number of bytes in the message, starting from the Request ID up to the Payload.
- **Request ID**: 32 bits. It is used to differentiate between multiple calls of the same method. The first 16 bits represent the Client ID, which identifies a specific client, while the remaining 16 bits represent the Session ID, which helps to distinguish sequential messages sent by the same sender.
- **Protocol Version**: 8 bits. It indicates the version of the SOME/IP protocol.

- **Interface Version**: 8 bits. It indicates the major version of the service interface.
- **Message Type**: 8 bits. It identifies the types of message.

Table 4.6: Message Types with their possible values [26]

Type Name	Value	Description
REQUEST	0x00	Request expecting a response
REQUEST_NO_RETURN	0x01	Fire & Forget Request
NOTIFICATION	0x02	Request of a Notification/Event callback
RESPONSE	0x80	Response message
ERROR	0x81	Response with an error

- **Return Code**: 8 bits. It indicates whether a request has been successfully processed.

SOME/IP is based on the activities of client and server. It is possible to differentiate different topologies of communication:

- **Request/Response**: A request message is sent by the client to call a function, while a response message is sent by the server to the client containing the result of the requested function.
- **Fire & Forget**: A message is sent by the client to the server to invoke a function, and no response is provided by the server.
- **Events**: When a specific event occurs, the server sends a message to the client with the corresponding information. Before that, the client must inform the server that it wants to receive those updates, like a subscription. Event messages work in a similar way to standard CAN messages.
- **Fields**: Fields represent properties that can be accessed remotely using Getter and Setter messages. The Getter is used to read the value of a field, while the Setter is used to modify it. When a field value is updated, a notification is sent.

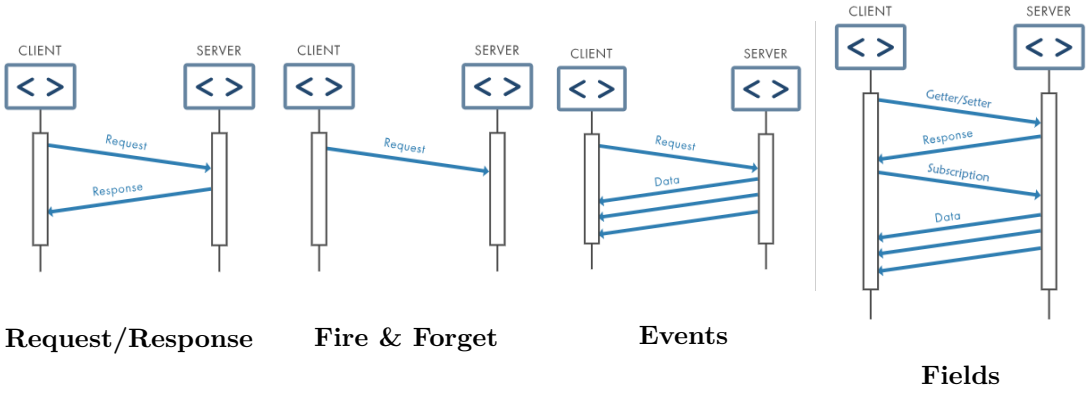


Figure 4.16: Four different SOME/IP communication methods [53].

DoIP

Diagnostic over Internet Protocol (DoIP) is an important protocol used in automotive Ethernet. Standardized by ISO 13400[54], DoIP has had a significant impact on the automotive industry, especially in diagnostics and communication between external diagnostic tools and vehicle components using IP, TCP and UDP.

Unlike other diagnostic protocols that depended on direct physical links, such as Unified Diagnostic Service (UDS) used over CAN, DoIP operates over high-speed Ethernet networks. This allows for faster diagnostics and software updates, reducing operational costs while maintaining reliability.

Moreover, DoIP enables communication over a network, allowing diagnostics and software updates to be performed remotely. This feature is particularly important for modern vehicles, which make use of advanced electronics requiring frequent software updates.

An example of DoIP architecture within a vehicle is explained in Figure 4.17. All DoIP entities must implement certain functionalities, as specified in ISO 13400, such as using the same IP version across all entities, ensuring unique MAC addresses, and implementing TCP according to RFC 793.

The DoIP communication sequence, shown in Figure 4.18, can be summarized in a few steps. First, the tester device sends a Vehicle Identification Request to the DoIP gateway, which responds with the Vehicle Identification Number (VIN), Entity ID (EID), and its logical address. After this identification phase over UDP, the tester sends a Routing Activation Request over TCP. Once validated by the

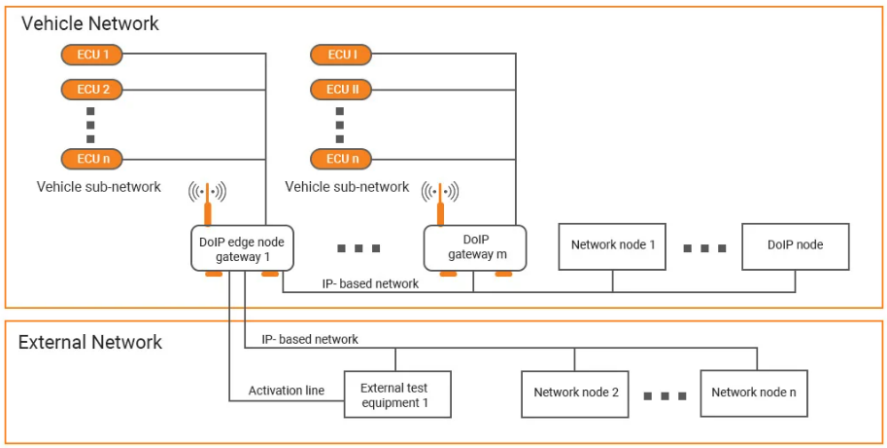


Figure 4.17: DoIP Architecture [55]

gateway, the communication between the tester and the target ECU is enabled and confirmed through a Routing Activation Response. After that, diagnostic messages between the tester and the ECU can be exchanged, with acknowledgments used to ensure correct delivery[56].

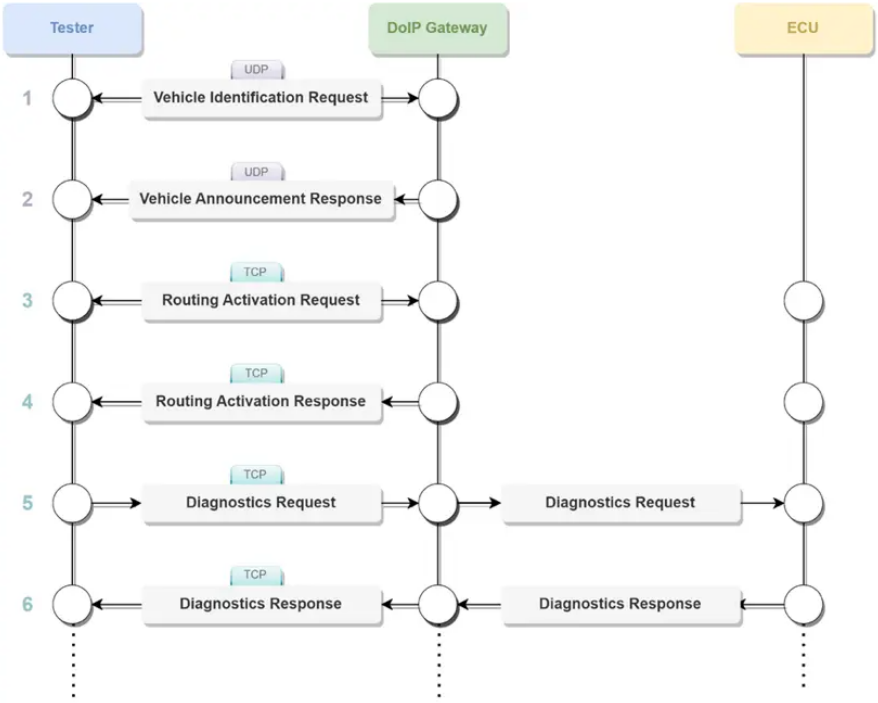


Figure 4.18: DoIP communication session example [56]

The general structure of a DoIP Ethernet frame is illustrated in Figure 4.19, which highlights the formats of the Ethernet frame, the DoIP message, and the DoIP payload.

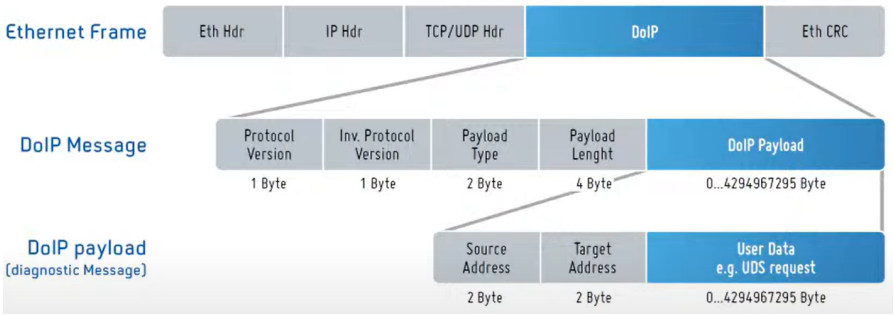


Figure 4.19: Typical structure of the DoIP Ethernet frame [26]

4.2.6 AVB/TSN

Nowadays, most vehicles include the need for high-end camera, advanced safety functions, and complex infotainment systems with ADAS being one of the most relevant features.

At first, Ethernet was not considered ideal for vehicle multimedia applications because traditional Ethernet offered non-deterministic packet delivery. In 2005, the IEEE formed the Audio Video Bridging (AVB) Task Group to enable reliable, low-latency, and synchronized streaming over Ethernet.

Moreover, the main challenges extended beyond the audio/video domain. In 2012, the AVB standard was renamed as the Time Sensitive Network (TSN) Task Group. Its main goal was to achieve low latency, high bandwidth, redundancy support, and deterministic communication, especially for Ethernet networks involved in safety-critical functions.

As is possible to notice in 4.1, the AVB/TSN protocol is designed to operate at Layer 2 of the OSI model, the Data Link Layer. This allows to avoid all the complexity of the upper layers, meaning that there are no IP addresses, ports, or sockets involved.

In an automotive network, it is possible to have a hybrid topology with both AVB and non-AVB ECUs. Inside an AVB domain, only AVB-capable endpoints and bridges can transmit or receive AVB streams. Non-AVB nodes can still be

connected to the same physical network, but they are not part of the AVB domain and treat AVB traffic as normal Ethernet frames, without any synchronization or quality of service guarantees.

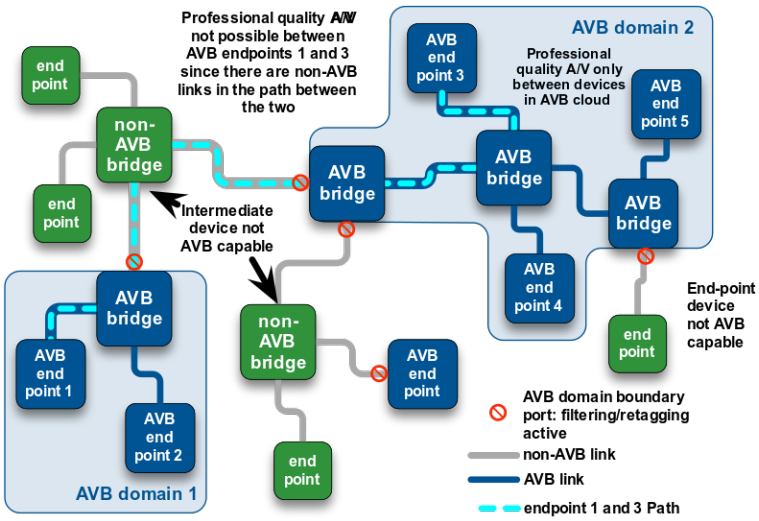


Figure 4.20: AVB Connection example [57]

Besides the AVB-capable switches or bridges, the AVB ECUs can be divided into two categories: The AVB "Talkers", which are the sources of the streaming data, and the AVB "Listeners", which are the devices that receive and use the data.

Regarding the AVB transport protocol, the first standard that defined the transmission of different raw and compressed audio/video formats was IEEE 1722-2011, also known as the Audio/Video Transport Protocol (AVTP). An example of an IEEE 1722 frame is shown in Figure 4.21. The AVB data are carried within an IEEE 1722 frame, called the AVBTP Payload, with a maximum frame size of 1476 bytes. The Ethertype typically used for IEEE 1722 is 0x22F0. Besides the AVBTP payload, additional information is included in the 1722 packet, such as:

- **Header**: indicates the type/format of AV data.
- **StreamID**: uniquely defines a specific data stream, derived from the Talker's MAC address.
- **AVB Timestamp**: provides the presentation time for synchronization purposes.

- **PacketInfo**: specifies the format of the data in the Payload.

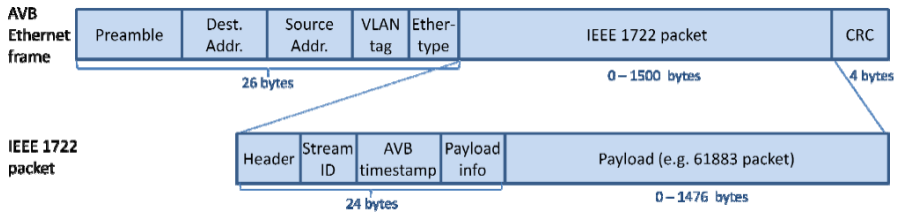


Figure 4.21: AVB Frame Format

Together with IEEE 1722, the AVB is completed with three foundational standards: 802.1AS, 802.1Qat, and 802.1Qav . These standards handle, respectively, the time synchronization, stream reservation, and traffic shaping.

IEEE 802.1AS is responsible for synchronizing all nodes within an AVB domain to a common reference clock. Also known as the generalized Precision Time Protocol (gPTP), this standard designates one principal node as the Grandmaster Clock, to which all other clocks in the network are synchronized. The selection of the Grandmaster is managed through the Best Master Clock Algorithm (BMCA), which automatically selects the most suitable clock source based on parameters such as priority and clock quality. Once the Grandmaster is chosen, all other devices synchronize their local clocks accordingly. Another main function of the algorithm is to deal with the clock spanning tree, the path where all the synchronization messages are distributed across the network. To ensure accurate timing, the link delay which represents the propagation delay between connected nodes, must be accurately measured and compensated by exchanging Sync and Follow Up messages between master and slave clocks, as shown in Figure 4.22.

To reserve bandwidth for specific applications and data streams within AVB network, the IEEE802.1Qat Stream Reservation Protocol (SRP) is used. By using SRP, the Talkers announce the availability of the stream they intend to transmit to all the AVB nodes, while the Listener can subscribe to those treams if they wish to receive them. Thanks to this mechanism, each switch along the path from the Talker to the Listeners checks whether the required bandwidth is available. If so, the bandwidth is reserved for the stream, otherwise the reservation request is rejected. The total bandwidth that can be reserved for an AVB stream is up to

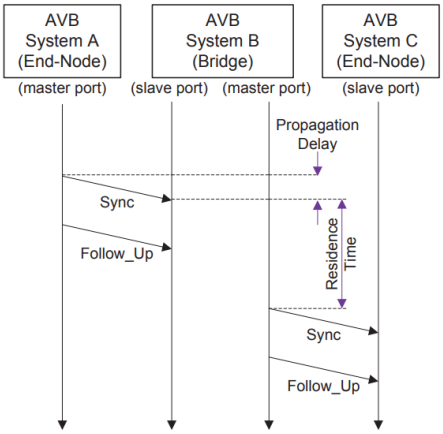


Figure 4.22: AVB Synchronization [58]

75%, and it is possible to differentiate between classes with different priority values, as shown in table 4.7.

Table 4.7: SRP priority classes[41, 58]

AVB Stream Class	**Priority Value**	**Frequency of packets** $[kHz]$	**Interval Time** $[\mu s]$
A	3	8	125
B	2	4	250

The last of the three standards, the IEEE 802.1Qav, known as Forwarding and Queuing for Time-Sensitive Streams (FQTSS), for regulating and scheduling the transmission of time-sensitive streams over time. This can be done by applying the Credit-Based Shaper (CBS). The working principle of CBS is based on assigning a credit value to each traffic queue. When the credit is equal or higher than 0, the packet is transmitted, and the credit decreases with a fixed send slope; when the credit is lower than 0, the packet must wait in the queue while the credit increases with a fixed idle slope. Figure 4.23 graphically illustrates the working principle of the CBS.

These protocols make AVB a standard capable of efficiently supporting audio/video streaming in the multimedia domain, but it is not suitable for time-critical and safety-critical control traffic. However, it is not suitable for time-critical and safety-critical control traffic, such as the communication between sensors within a vehicle [18]. For this reason, as mentioned before, in 2012 the AVB standard

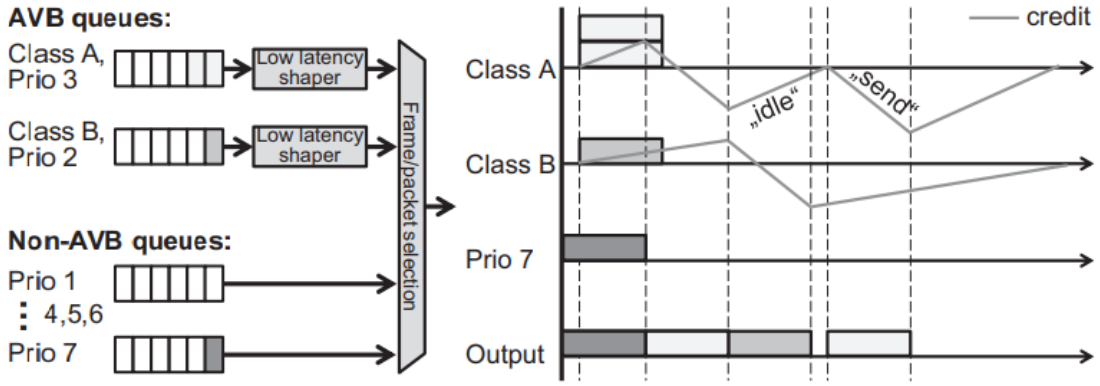


Figure 4.23: Illustration of CBS working principle [41]

evolved into the TSN Task group, with the aim of introducing new standards that extend the IEEE 802.1Q switches with additional features. The main goal of these new standards was to provide ultra-low latency, higher reliability, and improve clock synchronization robustness. The most relevant protocols introduced are summarized in Table 4.8.

Table 4.8: Main TSN standards for in-car communications[18]

Name	Description
802.1Qca-2015	Path Control & Reservation
802.1Qbv-2015	Enhancements for Scheduled Traffic
802.1Qbu-2016	Frame Preemption
802.1Qci-2017	Per-Stream Filtering and Policing
802.1Qch-2017	Cyclic Queuing and Forwarding
802.1QCB-2017	Frame Replication and Elimination for Reliability
802.1Qcc-2018	Stream Reservation Protocol (SRP) Enhancements and Performance Improvements
802.1Qcr-2020	Asynchronous Traffic Shaping
802.1QAS-2020	Timing and Synchronization for Time-Sensitive Application
P802.1QDG	Time-Sensitive Networking Profile for Automotive In-Vehicle Ethernet Communications
P802.1QAEdk	MAC Privacy Protection

Since audio/video and sensor data play a crucial role in modern vehicles, it is essential to have a security protocol that protects all the sensitive data and control messages. As it is possible to see in Table 4.8, the TSN standard includes the IEEE 802.1AEdk, an amendment of the IEEE 802.1AE-2018 standard, known as MAC

Security Protocol (MACsec). This protocol ensures the confidentiality, integrity, and authentication of Ethernet frames. The next chapter will focus on MACsec, describing its main features and working principles.

Chapter 5

MACsec

With the increasing connectivity of modern vehicles, ensuring secure data transmission has become a fundamental requirement. As introduced in the previous chapter, MACsec is the IEEE 802.1AE standard designed to provide secure communications at the Data Link Layer. This protocol plays a key role in protecting Ethernet networks from different security threats by ensuring confidentiality, integrity, and authentication. The main focus of this chapter will be the MACsec protocol, by analyzing its working principles, advantages and disadvantages, and potential application within the automotive domain.

5.1 Overview and Architecture

MACsec is a security protocol defined by IEEE 802.1AE standard that provides authentication, confidentiality, and integrity for data transmitted over point-to-point links within a LAN. It protects the network from unauthorized devices attempting to inject, modify, or sniff on Ethernet frames.

This security protocol operates at layer 2 of the OSI model, the Data Link Layer, since it provides hop by hop protection between directly connected devices, such as between two switches or between a host and a switch, by acting at the MAC address level. Its main working principle is to encrypt and authenticate every Ethernet frame transmitted over the physical link, regardless of the upper-layer protocols (IPv4, IPv6, VLAN, etc.), by adding a dedicated header and additional fields to the original frame.

Before the encryption and authentication process can take place, MACsec requires a mechanism to establish a secure communication channel through the exchange of cryptographic keys. This task is handled by the MACsec Key Agreement (MKA) protocol, defined in IEEE 802.1X. MKA is responsible for generating and distributing session keys among connected devices, in order to ensure that only authorized entities can participate in a secure link.

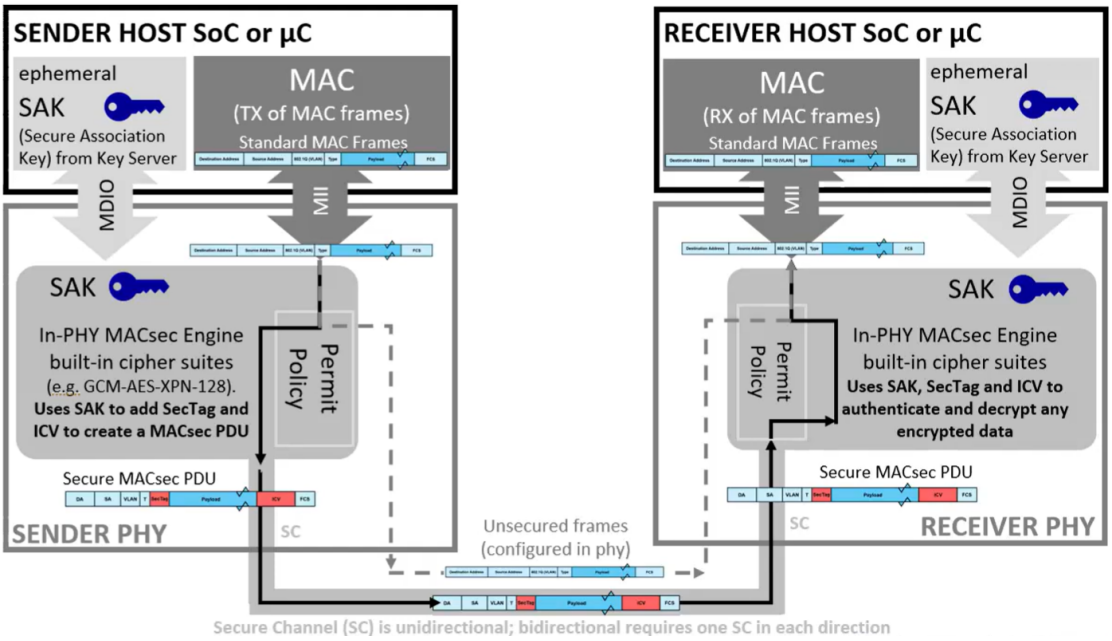


Figure 5.1: MACsec Architecture [59]

MACsec architecture is based on the creation of groups of trusted devices called Connectivity Associations (CAs). A CA is a logical construct that connects nodes within the same LAN that supports the MACsec protocol; all devices in the CA share the same cipher suite and security keys, which are established and managed by the MKA protocol. Once a CA is established, Security Channels (SCs) are created between the nodes to secure the communication links. Each SC is unidirectional, so a bidirectional communication requires two SCs, one for each direction. For each SC, one or more Secure Associations (SAs) are created. Each SA is linked to a Security Association Key (SAK), used to encrypt and authenticate the data according to the selected cipher suite. Every SAK is uniquely identified by a Secure Association Identifier (SAI), composed by the Secure Channel Identifier (SCI) and the Association Number (AN). Using this structure, together with the

Permit Policy, frames are protected by adding a Security TAG (SecTAG) and an Integrity Check Value (ICV), ensuring protection of the transmitted data.

A visual representation of this architecture and its main components is shown in Figure 5.1.

5.2 MACsec Frame

As mentioned above and shown in Figure 5.2, a MACsec frame format is based on a standard Ethernet frame with the addition of two main fields: the Security TAG (SecTAG) and the Integrity Check Value (ICV).

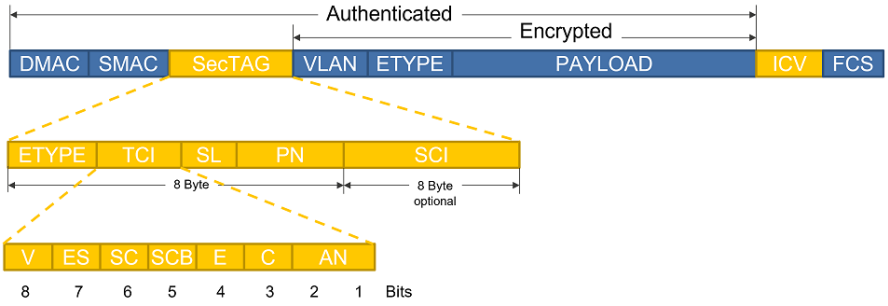


Figure 5.2: MACsec frame structure [60]

The SecTAG is a fundamental field within the MACsec frame, since it contains all the crucial security information required by the receiver for the decryption and verification of the data. It also includes a packet number, useful for replay protection. As shown in Figure 5.2, the SecTAG can be either 8 or 16 bytes long, depending on the presence of the Secure Channel Identifier (SCI), and it is divided into several subfields:

Table 5.1: Different fields inside the SecTAG

SecTAG fields	Length	Description
EtherType	2 Bytes	Contains the MACsec Ethertype, with a fixed value of $0x88E5$.
Tag Control Information (TCI)	1 Byte	Contains information about protection and encryption parameters.
Short Length (SL)	1 Byte	Specifies the length of the protected data field, specifically the number of bytes between the last byte of the SecTAG and the first byte of the ICV.
Packet Number (PN)	4 Bytes	Provides protection against replay attacks and is also used as Initialization Vector (IV) by the Cipher Suite.
Secure Channel Identifier (SCI)	8 Bytes	Identifies the Security Association to which the traffic belongs.

As detailed in Table 5.1, the TCI is further subdivided into several bits that provide additional information. A TCI is composed of[61]:

- **Version number (V)**: Version number of the MACsec protocol, currently always set to 0;
- **End Station (ES)**: Indicates whether the SCI matches the MAC source address;
- **SCI presence (SC)**: Indicates whether the SCI is present;
- **Single Copy Broadcast (SCB)**: Used to manage single copy broadcast in specific network topologies, without requiring the SCI;
- **Encrypted payload (E)**: Indicates whether the payload is encrypted;
- **Changed Text (C)**: Indicates whether the transmitted data differs encrypted data;
- **Association Number (AN)**: Identifies up to four different Secure Associations (SAs) within a Secure Channel.

Finally, the ICV field is appended at the end of the frame. It is 16 bytes long and its main function is to verify that the frame has not been altered during the transmission. The ICV is computed by the cipher suite algorithm and at the destination, it is recomputed and compared to the received value. If the values matches, the frame is accepted; otherwise, it is discarded.

5.3 MKA

The IEEE 802.1AE standard relies on authentication, generation, and exchange of cryptographic keys. In order to achieve this, it uses an external protocol, IEEE 802.1X, known as the MACsec Key Agreement (MKA). The main purpose of MKA is to detect the devices that support MACsec and to manage the negotiation of the keys to establish a secure link connection between them.

MKA supports two different mechanisms for the generation of key material:

- **Extensible Authentication Protocol (EAP)**

- **Pre-Shared Keys (PSK)**

In the automotive domain, EAP is not typically used, since MACsec used with PSK offers higher performance at startup and lower implementation complexity compared to EAP. Using PSK means that the initial key, i.e. Connectivity Association Key (CAK) and its respective Connectivity Association Key Name (CKN), are pre-installed on the communication participants in advance.

Each MACsec Key Agreement Protocol Data Unit (MKPDU) includes a CKN, which is used to process the received MKPDU.

From the CAK, MKA derives two additional keys called Integrity Check Key (ICK) and Key Encryption Key (KEK). The ICK is used to generate the Integrity Check Value (ICV) in MACsec, at both the receiver and the transmitter, and to verify the integrity of the MKPDU, ensuring that both parties possess the same ICK. On the other hand, the KEK is used to encrypt and distribute the SAKs.

The hierarchy of all MKA keys is shown in Figure 5.3.

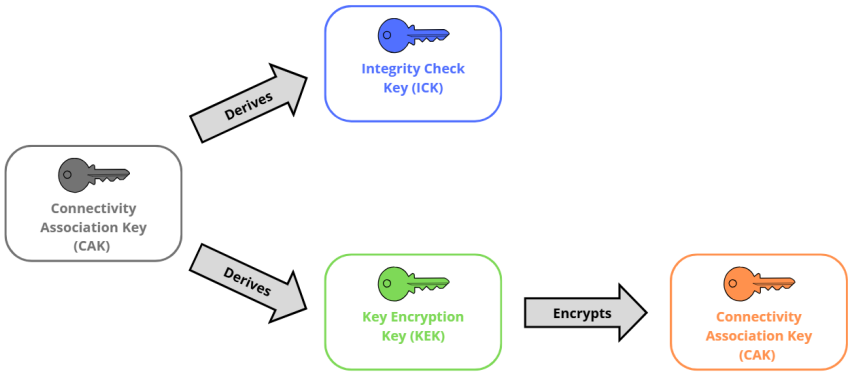


Figure 5.3: MKA Key Hierarchy

5.3.1 MKA Sequence

Usually in the MKA architecture, there are three different main role devices:

- **Supplicant**, also known as a client;
- **Authenticator**;

- **Authentication Server**.

Usually, in automotive applications, in order to reduce the extra communication with a server over internet or onboard and to reduce the delay introduced by the Authenticator and the Authenticator Server, there is a controller, called Key Server, that contains both the Authenticator and the Authentication Server.

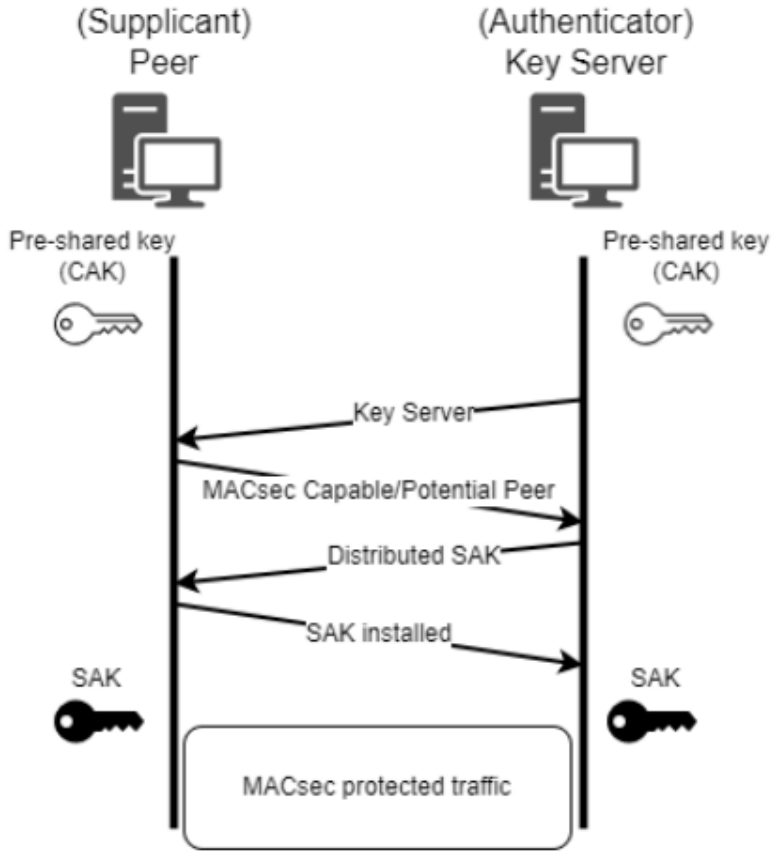


Figure 5.4: MKA Sequence [62]

The sequence of the MKA is shown in figure 5.4. It can be divided into different parts:

1. The Authenticator node, so in this case the Key Server, has the role to announce to the other devices that it is capable to use the MACsec protocol and tells the information needed to begin a new communication.

2. After receiving the packets from the Key Server, the nodes that belong to the same CA can take part to the MKA process. These nodes began the

Supplicant nodes and inform the Key Server with their MACsec capabilities and all the information needed to identify them.

3. The Authenticator and the Supplicants exchange all the information useful to authenticate each other in the communication.

4. Once that the authentication is done, the Key Server can distribute to the Supplicant nodes the SAK keys that are encrypted and integrity protected using the KEK and ICK keys derived by the CAK, as explained in section 5.3.

5. The SAKs are installed, and so both the Supplicants and Key Server are informed about the correct installation of the key for transmission and reception.

After this sequence, the MACsec protocol is ready to use.

5.4 MACsec Tx and Rx

After the MKA process is completed and the SAKs have been distributed among the connected devices, the MACsec protocol can start protecting Ethernet communication in the vehicle.

At this point, MACsec is responsible for performing encryption and authentication of Ethernet frames. These operations take place in two different phase of data flow: transmission (TX) and reception (RX).

5.4.1 MACsec Tx

In the transmitting MACsec PHY, a standard Ethernet frame is received by the host MAC, such as the one described in Chapter 4.2.2. The original FCS is discarded because the frame content will be modified, and therefore a new FCS must be calculated.

As previously mentioned, the PHY (or MACsec sublayer) assembles the SecTag, which contains an EtherType field with a fixed value of $0x88E5$. Beside the EtherType, the correct AN is inserted, and the PN is incremented by one, since a new MACsec frame is being transmitted.

Once completed, the SecTAG is inserted into the Ethernet frame.

The PN, combined with Padded Zeros, forms the Initialization Vector, which is provided as input into to the AES Crypto engine.

During transmission, the correct SAK is selected based on the AN contained in the SecTAG. The SAK is also used by the AES cipher to compute the ICV. This new value is then appended to the Ethernet frame, and finally a new FCS is calculated.

As shown in Figure 5.5, after completing this step, a secured but non-encrypted MACsec PDU is ready for transmission to the receiving device.

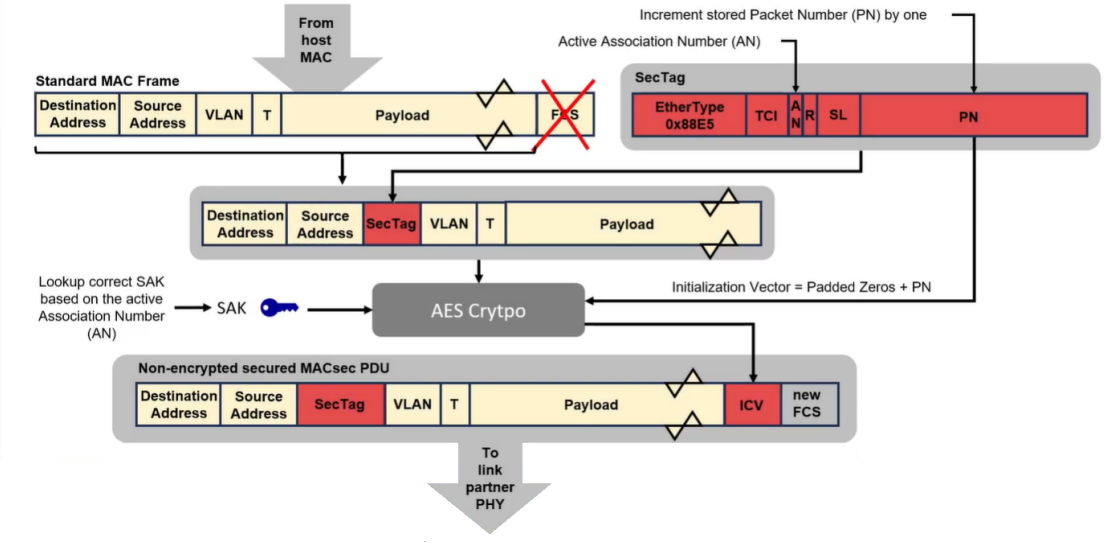


Figure 5.5: MACsec Transmit [59]

5.4.2 MACsec Rx

After transmission, the receiving device obtains the secured MACsec frame, which includes the SECtag and the ICV. The ICV plays a fundamental role in frame authentication.

From the SECtag, the AN is extracted in order to identify the correct active SAK. The PN is also read by the frame. The SAK key and the PN, used to build

the Initialization Vector, are provided as input in the AES crypto engine, following the same principle described for the TX case.

The received frame, excluding the ICV and FCS fields, is processed by the AES engine to compute a new ICV. If the frame has not be altered during transmission, the calculated ICV will match the received ICV. The receiving MACsec PHY then compares the two values. If they are identical, the frame is successfully authenticated.

After successful authentication, the SecTAG and the ICV are removed, and a new Ethernet FCS is calculated and appended to the frame.

Finally, the frame is restored to a standard Ethernet frame, ready to be transferred into the host MAC of the receiving device.

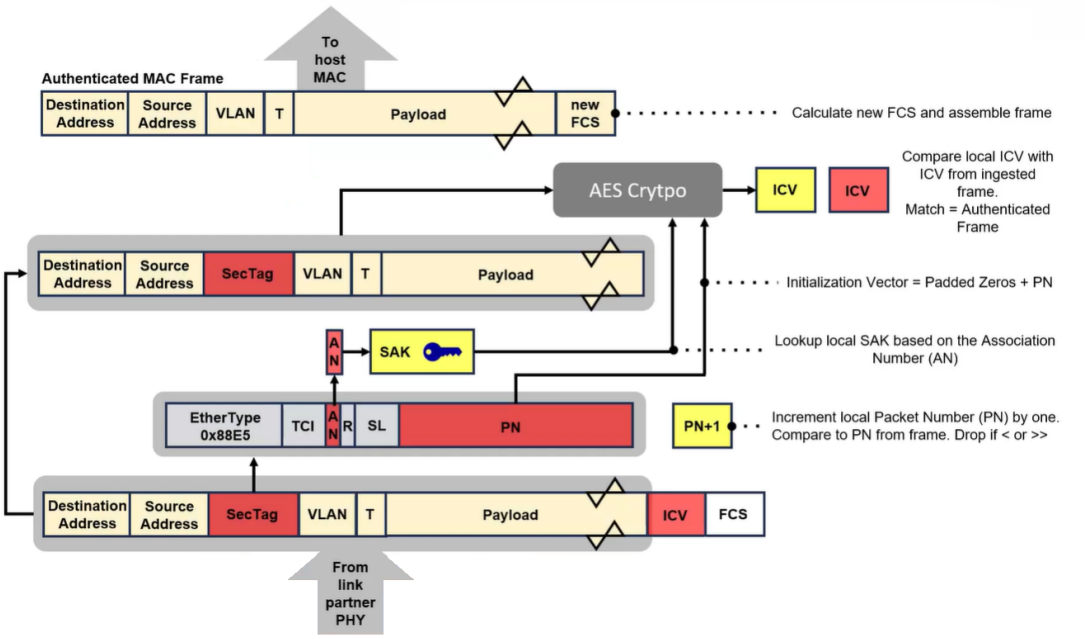


Figure 5.6: MACsec Receive [59]

5.5 Advantages of MACSec in Automotive Applications

As previously explained, MACsec is an effective solution for network security, but its main advantages go beyond the basic protection mechanisms.

For this reason, especially in the automotive field, it represents a very suitable technology for several applications. In particular:

- **Scalability and High Performance**: MACsec can be deployed in different network topologies, from simple point-to-point links to complex multi-node systems, thanks to its scalability. Moreover, the cryptographic processing is designed to support high data rates and low latency. In particular, header processing can start even before the entire frame has been received. This behavior is useful for real-time automotive applications.
- **Hardware-Based Implementation**: Unlike many other security protocols, MACsec can be entirely implemented in hardware. This reduces system complexity and potential vulnerabilities, while minimizing the need for software intervention. As a result, risks related to software development, updates and maintenance are also reduced.
- **Full Line-Rate Operation**: MACsec operates at the full speed of the network, meaning that security is provided without degrading communication performance. This is particularly important in Automotive Ethernet, where large volumes of data must be handled while guaranteeing maximum bandwidth and deterministic behavior.
- **Comprehensive Security Features**: In addition to performance and implementation benefits, MACsec provides different built-in security functions. These features protect the data exchange against different network threats, as summarized in Table 5.2.
- **Prevention of Unauthorized Access**: Only authenticated devices are allowed to exchange protected data, reducing the risk of untrusted ECUs or malicious actors gaining access into the vehicle network.
- **Compliance with Industry Standards**: Since MACsec is regularized by an IEEE standard, it is easier for manufacturers to apply regulatory and industry requirements for in-vehicle network security.

All these characteristics make MACsec highly suitable for automotive networks, where both high performance and strong protection against network attacks are required.

Table 5.2: Security features of MACsec

Feature	Description
Device-to-Device Security	MACsec protects data exchanged directly between devices, even if the network infrastructure itself is not trusted.
Connectionless Data Integrity	Since every MAC frame contains a personal integrity verification code, every change of the data is easily detected.
Data Origin Authenticity	Every protected frame received by a device is certainly originated from an authenticated device.
Replay Protection	MACsec detects and prevents replay attacks by using packet numbering and validation mechanisms. Limited tolerance is allowed only for frame reordering.
Bounder Receive Delay	MACsec is able to detect abnormal frame delays, for example those caused by man-in-the-middle attacks.

5.6 Attack to In-Vehicle Networks

As discussed in the previous chapters, the increasing adoption of Ethernet in the automotive domain has introduced new types of vulnerabilities.

Network attacks can originate from different causes, but usually they are initiated by a malicious actor who, by exploiting existing vulnerabilities or physical access, gains access to the in-vehicle Ethernet Network.

The main components that are more exposed to these attacks are:

- **ECUs**: Attacker may attempt to read or manipulate the communication between different ECUs, which are responsible for critical vehicle function, such as breaking, steering, and engine management;
- **ADAS**: ADAS handles sensitive data used for safety features such as collision avoidance, lane-keeping assistance, and adaptive cruise control. Compromising these systems can directly affect vehicle safety;
- **Body Control**: These systems control and monitor all the vehicle electronic function, ranging from door locking to interior lighting control. Unauthorized access may lead to loss of comfort function and, in some cases, safety issues.

The network attacks can be classified into two main categories: active and passive. Active attacks involve the modification, injection, or disruption of messages during transmission. In these cases, an attacker may first intercept a message and

then change or retransmit it in order to influence the system behavior. Passive attacks, on the other hand, aim only at observing and collecting all the information from the network without interfering with the communication protocol.

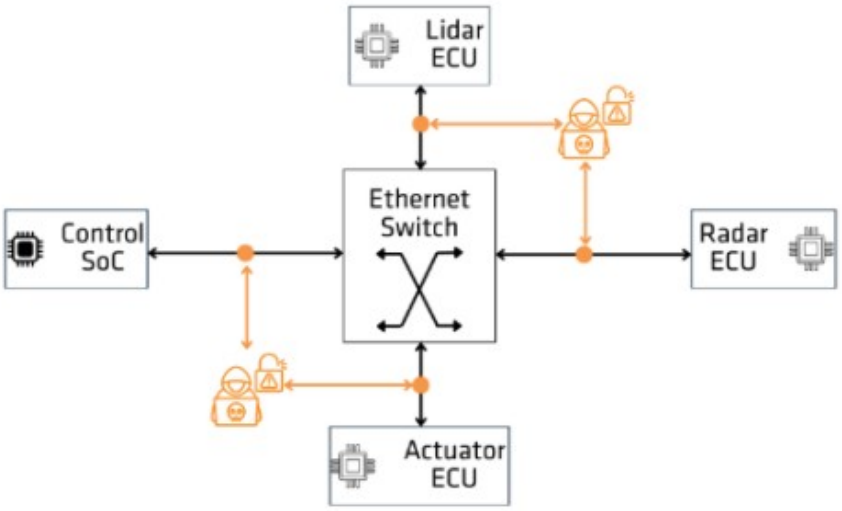


Figure 5.7: Threat Scenario in a vehicle network [63]

As shown in Figure 5.7, performing an attack may simply require access to Ethernet links or ports. The main goal of these attacks can include perform a Denial-of-Service (DoS) attack, which is particularly dangerous for a vehicle, or collecting sensitive information that may compromise the vehicle user privacy.

The most common security dangers in the automotive domain can be summarized into four main types: eavesdropping, Main-In-The-Middle (MITM), frame dropping and modify, and replay attacks.

5.6.1 Eavesdropping

In absence of security protocols, the data contained in network messages can be easily intercepted and read. For this reason, an attacker can observe and analyze the information exchanged between devices. Through eavesdropping, the attacker may identify the type of traffic, the ECUs or devices present in the network, and potentially sensitive data such as proprietary or private information. This knowledge can later be used for malicious activities.

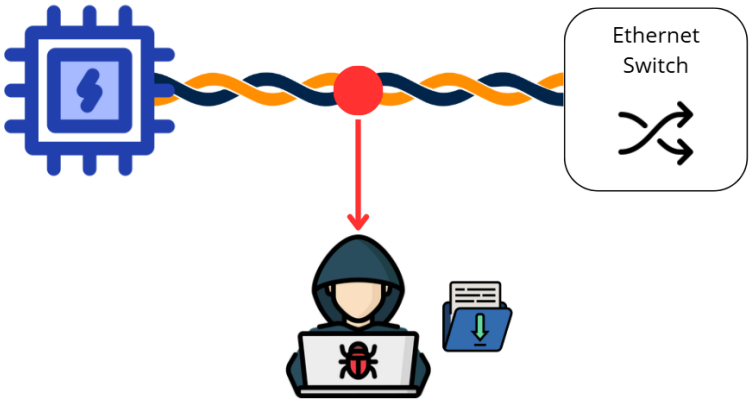


Figure 5.8: Eavesdropping attack

5.6.2 Man-in-the-middle

The Man-in-the-Middle attack occurs when an attacker stands itself between two devices in the vehicle network, pretending to be a legitimate node or a device. From this position, the attacker can intercept, modify, and inject fraudulent messages into the communication channel. These fake messages may contain wrong or manipulated information, allowing the attacker to influence the behavior of vehicle system. A simple example is the injection of fake data from sensor, radars, or camera in order to trigger unintended actions, such as activating the braking system.

This topology of attack is also particularly relevant in Time Sensitive Network. By injecting packets with a falsified high priority, an attacker can disrupt the scheduling mechanism explained in Chapter 4.2.6. This may lead to queue congestion, increase latency for critical traffic, and a reduction in network determinism.

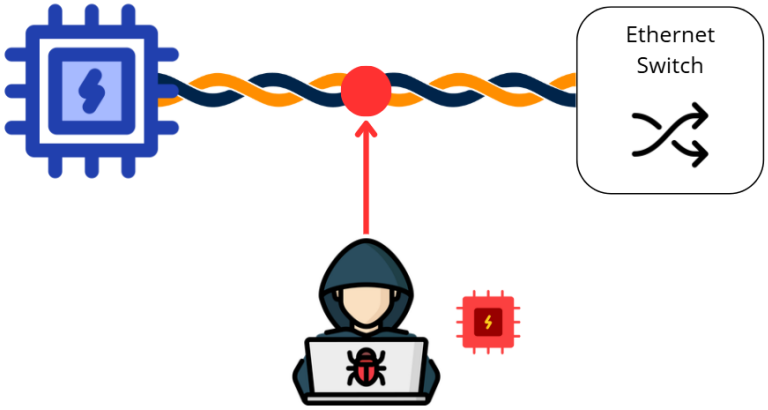


Figure 5.9: Man-in-the-middle attack

5.6.3 Drop and Modify

In this type of threat, the attacker gains access to the network and acts as a malicious gateway through which messages are routed. This fake gateway can selectively drop and modify the frames passing through it. The ECUs in the vehicle may only receive these manipulated or missing messages and, without any security mechanisms, trust them as legitimate. This can lead to unauthorized and abnormal system behavior. This topology of attack can be difficult to detect while still interfering with vehicle operations, reducing the overall system reliability. For example, an attacker could selectively block critical frames, causing service disruption or preventing their transmission.

Figure 5.10: Drop and Modify attack

5.6.4 Replay and Flooding

In a replay attack, an attacker captures a valid message and saves it for later retransmission. By injecting the same message again into the network, the attacker can create confusion at the receiving device, which may process multiple copies of the same frame. This behavior can also lead to a flooding scenario, where replayed messages are transmitted at a high rate, potentially overloading the ECU with excessive traffic. As a result, the device may experience degraded performance or malfunction due to the high volume of incoming frames. Examples of messages that could be replayed include commands to unlock doors, start the engine, or perform other car control actions.

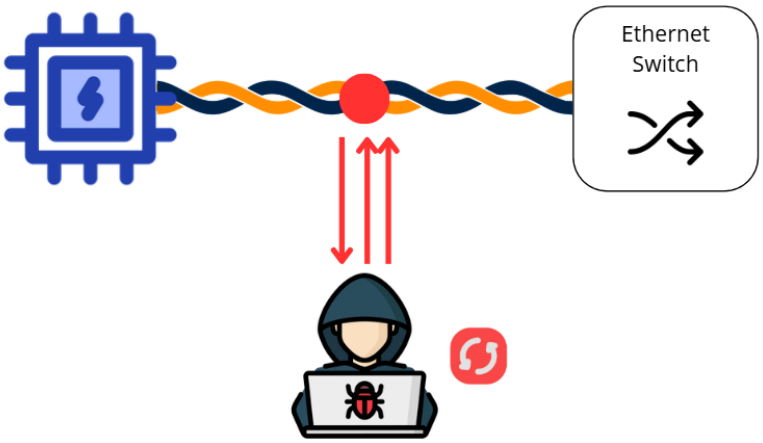


Figure 5.11: Replay and Flooding attack

5.7 MACsec vs Other Security Protocols

MACsec is not the only protocol that can provide secure communication over Ethernet. Other security solutions that can also be used in automotive Ethernet environment are Internet Protocol Security (IPsec), Transport Layer Security (TLS), and Datagram TLS (DTLS).

IPsec is a security protocol that operates at the network layer (Layer 3 of the OSI model). It is mainly composed by two protocols [64]:

- **Authentication Header (AH)**: This protocol provides authentication only, without providing encryption. It works by adding an additional header to

the IP packet, generated by applying a cryptographic algorithms, such as HMAC-MD5 or HMAC-SHA, to the IP header and its payload. This allows verification that the packet has not been modified during the communication through the network, and guarantees data integrity, origin authentication, and replay protection.

- **Encapsulating Security Payload (ESP)**: This protocol provides both encryption and data integrity. A new header, called ESP, is inserted after the original IP header, so routing can still be performed normally. The same cryptographic algorithms as AH are used. Unlike AH, ESP authenticates only the payload of the frame and not the entire IP packet.

IPsec can work in two different modes: Transport mode, where only the payload is protected, usually used on short links, and Tunnel mode, where the entire packet is encapsulated and protected, typically used in Wide Area Networks (WANs).

TLS is a protocol mainly used to protect the communication between applications and represents the evolution of the Secure Socket Layer (SSL) protocol. It is based on two types of cryptography [64]:

- **Symmetric cryptography**: It is used in order to encrypt and decrypt transmitted data, by exchanging secure keys between the sender and the receiver.

- **Asymmetric cryptography**: It is used to encrypt the data. Two different keys are used: a public key, used by the sender to encrypt the data, and a private key, used by the receiver to decrypt the data.

Keys are generated using algorithms such as Rivest–Shamir–Adleman (RSA) and Diffie-Hellman (DH) and are discarded at the end of the session. TLS also ensures data integrity by attaching a Message Authentication Code to each transmitted message, preventing tampering and replay attacks.

TLS only works over TCP connections, while DTLS is the equivalent protocol designed for UDP communications.

MACsec, IPsec and TLS/DTLS are different security protocols, used at different layers and for different purposes. As shown in Figure 5.12 MACsec works at the data link layer and protects every Ethernet frames, automatically securing the upper

layers. IPsec protects IP frames only along the network path, while TLS/DTLS secures only specific applications connections (TCP or UDP).

From a performance point of view, IPsec and TLS/DTLS require more processing because they are typically handled in software and must manage each communication session individually. At high data rates this becomes a limitation. MACsec instead is implemented in hardware and works point-to-point, requiring less processing and introducing minimal latency.

For this reason MACsec is particularly suitable for in-vehicle networks where low latency and deterministic communication are required, while IPsec and TLS/DTLS are typically used for higher-layer protection or external communications.

Protocol	Standard	Type/Layer	Authent.	Encryption	Comment
MACsec	IEEE 802.1AE	Hop-by-hop Data-Link	X	X	Requires crypto/keys at each network node
IPsec AH (Authentication Header)	IETF RfC 4302	End-to-End IP	X	-	
IPsec ESP (Encapsulating Security Payload)	IETF RfC 4303	End-to-End IP	X	X	
TLS 1.2 (Transport Layer Security)	IETF RfC 5246	End-to-End TCP	X	X	Does not work with UDP
SecOC	AUTOSAR	End-to-End PDUs	X	-	supports MACtruncation (works also with CAN / FlexRay)

Figure 5.12: Overview of different security protocols in automotive domain [65]

Another important aspect is the number of security handshakes required during vehicle startup. This strongly depends on the security protocol topology. MACsec scales linearly with the number of links, since one initialization is required for every Ethernet connection. Instead, TLS/DTLS and IPsec require a handshake for each communication session. For a vehicle with 20 ECUs, the number of initializations becomes significantly different, as shown in Table 5.3. The number of handshakes is very important, since a large number of handshakes increases startup time and may delay the activation of crucial systems such as ADAS or cameras[66].

Table 5.3: Handshakes in a vehicle network[66]

Protocol	**Startup per connection**	**Example with 20 ECUs**
MACsec	n per Ethernet link	20 MACsec startups
IPsec	n^2 per IP connection	800 IPsec startups (both direction)
TLS/DTLS	n^3 per UDP/TCP connection	1600 TLS/DTLS startup or more (TCP + UDP and both directions)

Chapter 6

Testing Environment

The main goal of this work of thesis is to analyze the behavior of MACsec, in order to understand its real operating principle and evaluate the performance of this security protocol. This chapter describes all the elements that made this study possible. In particular, it presents all the hardware tools used, such as RAD-Moon and RAD-Gigastar, together with the VehicleSpy3 software, all provided by Intrepid Control Systems.

Intrepid Control System is an American company mainly focused on developing advanced software and hardware tools for the automotive fields. Their products are designed to support engineers working in vehicle networks. The company has a long experience with communication technologies such as CAN, CAN FD, and LIN, and in the recent years has expanded its focus on automotive Ethernet, providing solutions for ECU testing, simulation, diagnostic, and data analysis.

6.1 Vehicle Spy 3

One of the most important software tools developed by Intrepid Control Systems is Vehicle Spy. Vehicle Spy 3 is a multifunctional tool that includes several features such as diagnostic, ECU/nodes simulation, data acquisition, automated testing, calibration, and vehicle network monitoring.

Besides its wide range of applications, one of the main strengths of this software is its simple and user-friendly interface. This choice reflects the company's focus on usability and helps improve user productivity.

Due to its versatility, Vehicle Spy is widely used in both research and industrial automotive environments. It supports different network protocols (CAN, CAN FD, J1939 for heavy vehicles to Automotive Ethernet) and allows the user to interact with a vehicle communication system in a flexible way. Thanks to its integrated functions, the software can be used during different phases of development, from testing and validation, with the ability to simulate an entire vehicle bus, to troubleshooting and system analysis.

The main applications of Vehicle Spy in the automotive field can be summarized as follows:

- **Message Setup and Monitoring**: It is the main application of the software. This functionality allows the user to view, save, and analyze the message traffic from all the networks simultaneously, even if different communication protocols are used.

- **Simulation**: The ability to simulate ECUs or network nodes makes Vehicle Spy useful for creating a complete test setup. Through Graphical Panels and Scripting, it is possible to reproduce specific customer functions. Another important feature is the Replay function, which allows pre-saved network traffic to be replayed during a live bus session.

- **Data Acquisition**: By recording communication data, the software offers a complete solution for data acquisition and analysis. Signals can be monitored in real time, while saved logs file can be analyzed in a second moment.

- **Diagnostics**: Vehicle Spy also supports ECU diagnostics through common automotive diagnostic protocols. It allows sending diagnostic request, receiving and interpreting responses, and monitoring parameters related to the system status. This feature is useful to read fault codes and to understand vehicle behavior during testing and validation.

- **Memory Edit/Calibration**: Vehicle Spy enables real time modification of ECU memory parameters, making it possible to adjust calibration values, test different configurations, and to analyze system behavior without modify the entire software.

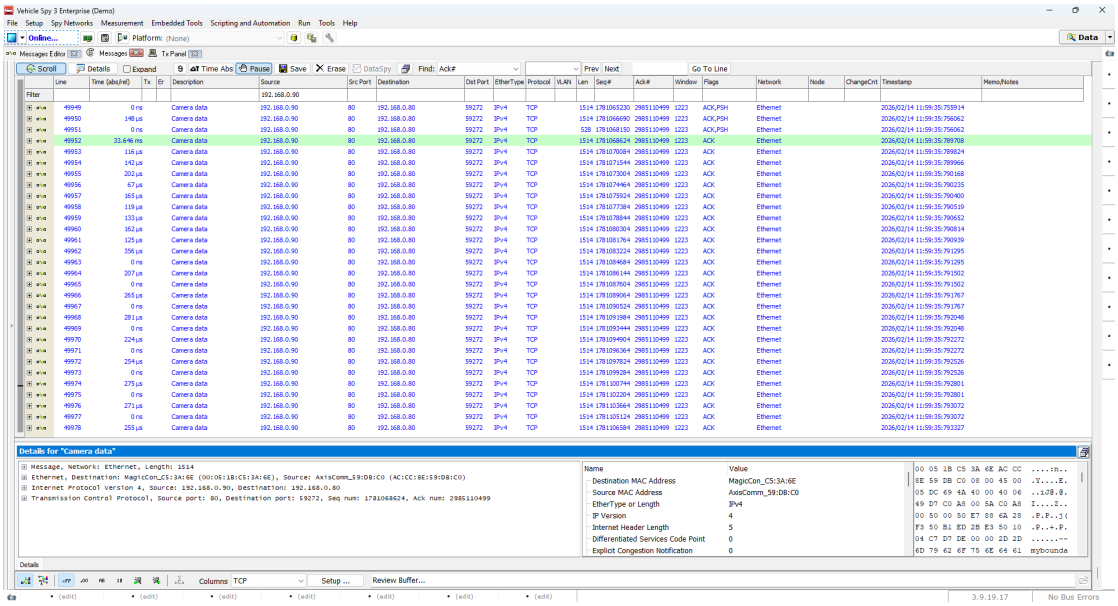


Figure 6.1: Example of Message View page of Vehicle Spy 3

All these functions can be easily executed thanks to the different views and panels available in the software. The most important tabs available in the main page are:

- **Message View**: This is the area where all active buses and signals can be monitored. In this view it is possible to filter and highlight messages sorted by specific parameters such as time, counter, source/destination IP or MAC address, data bytes, and other useful information (Figure 6.1).
- **Messages/Database Editor View**: In this section it is possible to configure and edit databases and messages. The tool makes it easy to create or modify messages by using the Messages Editor and adjusting messages or signals parameters.
- **Transmit Panel**: This panel allows the user to control the transmission of different types of messages, either manually or by setting a periodic transmission rate.
- **Graphical Panel**: This feature is useful for creating a custom user interface in order to interact with scripts, transmit messages, or visualize bus data in different ways.

6.2 RAD-Gigastar

Figure 6.2 shows the RAD-Gigastar, one of the most versatile tools developed by Intrepid Control Systems. It can operate in different modes, such as an active tap, media converter, vehicle interface, and Ethernet Data Logger.

Figure 6.2: RAD-Gigastar[67]

The device supports two 100/1000BASE-T1 or BroadR-Reach connections and also includes two Small Form-factor Pluggable (SFP) cages, allowing the use of optical or copper SFP transceiver. This feature is particularly interesting in the context of MACsec, since some SFP can integrate MACsec functionality, enabling the possibility to have a secure communication through the RAD-Gigastar.

Over Ethernet, the device can also be used as a programmable gateway, enabling communication between different networks, for example as a CAN-Ethernet gateway.

As shown in Figure 6.3, the RAD-Gigastar provides the following interfaces:

- 2x 100/1000BASE-T1 using the Marvell 88Q2112 PHY;
- 2x 100BASE-TX/1000BASE-T/1000BASE-X SFP ports;
- 6x ISO CAN FD channels with selectable on-board termination;
- 2x FlexRay receive channels with selectable on-board termination
- 1x LIN/K Line;

- 1x DoIP activation line;
- 1x 48V port for power supply.

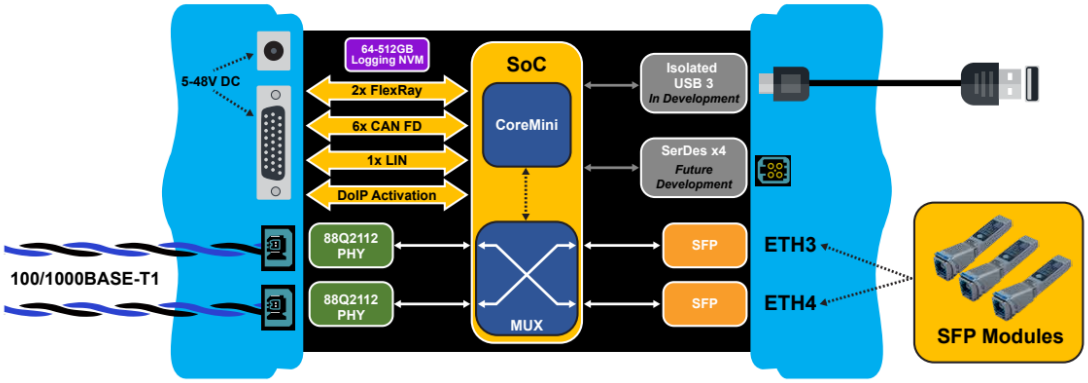


Figure 6.3: RAD-Gigastar Interfaces [67]

One of the most useful configurations of the RAD-Gigastar is its use as an active tap, meaning that the device is placed between two Automotive Ethernet nodes. These nodes can be ECUs or, for example, an ECU connected to a switch. The communication between the devices continues normally, so the presence of the RAD-Gigastar does not interfere with the network data flow. However, the tool allows all the traffic to be monitored and copied to a PC, where it can be analyzed using software such as Vehicle Spy.

6.3 88Q2221M 1000BASE-T1 with TC10 and MACsec SFP module

Beside the tools developed by the company, such as the previously described RAD-Moon and RAD-Gigastar, another important component is the Automotive Ethernet SFP module developed by Intrepid Control Systems. This module is based on a single-pair Ethernet PHY, specifically the Marvell 88Q2221M, which consents the communication over both 100BASE-T1 and 1000BASE-T1 links.

The Marvell 88Q2221M is an Automotive Ethernet PHY, meaning an integrated circuit that handles communication at physical layer, developed by the company

Marvell. In addition to signal transmission, this device includes security-related features[68]. In particular, it follows the OPEN Alliance TC17 specification and integrates MACsec functionality directly at the hardware level. This makes it possible to secure link-layer communication without impacting the high data rates required in Automotive Ethernet[69].

Figure 6.4: SFP module with MACsec, developed by Intrepid CS [70]

The SFP module shown in Figure 6.4 is increasingly used in automotive applications because it improves Layer 2 security while maintaining full communication speed. By implementing MACsec at the hardware lever, it helps to prevent possible intrusion such as man-in-the-middle and other network threats discussed in Chapter 5.6. In addition to these features, this SFP module also meets the Open Alliance TC10 requirements. This specification supports the wake-up and sleep mechanism for Automotive Ethernet, allowing the physical layer to enter in low-power states when communication is not needed, thus reducing energy consumption[71].

So, basically, the most important features of this SFP module are:

- Supports IEEE 802.3bw and IEEE 802.3bp standards;
- Complies with the Open Alliance TC10 requirements for Automotive Ethernet sleep and wake-up functions;
- Supports the Open Alliance TC17 specification for the MACsec security protocol;

- Supports user-selectable link modes, including master, slave, and auto-negotiation, with auto-negotiation set as the default mode.

6.4 Experimental Setup

In order to study the behavior of MACsec in an automotive Ethernet environment, the first step was to create an experimental setup for testing purposes. The configuration is composed by an Ethernet Camera, two RAD-Gigastar devices connected in series, both equipped with the SFP supporting MACsec, and finally a computer running Vehicle Spy 3. The complete setup is shown in Figure 6.5 below.

Figure 6.5: Complete Experimental Setup

The Ethernet camera was used in order to simulate an ECU inside a vehicle, since it transmits Ethernet frames containing image data over the network. The camera was powered using a standard power supply and connected through a standard Ethernet cable to the first RAD-Gigastar.

The first RAD-Gigastar acts as a media converter. The messages provided by the camera are transmitted over standard Ethernet and then converted into Automotive Ethernet frames. These frames are then processed by the SFP module supporting TC10 and MACsec. The SFP modules use High-Speed Modular Twisted-Pair Data (H-MTD) Automotive Ethernet cables as connection systems, which support high data rates up to 1000 Mbps. In this stage, the packets generated by the Axis camera are converted into Automotive Ethernet and protected using the MACsec security protocol.

As discussed in Chapter 5, secured frames must be encrypted in order to be correctly interpreted by the destination ECU. This operation is handled by the second RAD-Gigastar, where a second SFP module enables the MACsec communication between two terminals and manages the decryption of MACsec frames. Similarly to the first RAD-Gigastar device, this RAD-Gigastar also performs

Figure 6.6: H-MTD automotive Ethernet cable [72]

media conversion, but in the opposite direction, from Automotive Ethernet back to standard Ethernet. Both Rad-Gigastar devices are powered by 220 V.

Each RAD-Gigastar provides four Ethernet ports: two ports for 100/1000BASE-T1 connections and two SFP ports for pluggable modules. In this setup, only the two SFP ports were used: one SFP module for MACsec communication and another SFP module used only for signal conversion to standard Ethernet. Thanks to the RAD-Gigastar configuration, it is possible to control the direction of the data flow, which allowed the use of only SFP interfaces.

The data then travel through a standard Ethernet cable connected to the computer, allowing packet capture and analysis.

The computer plays a key role in the setup. Using Vehicle Spy 3, it is possible to configure all the main parameters of the simulation environment and monitor the data flow. In particular, it is also possible to define both transmitted (Tx) and received (Rx) messages, as well as to configure the Intrepid Control Systems tools. In particular, through Vehicle Spy, the user can:

- Specify the network through which the message is transmitted (e.g., SFP1) thanks to the settings of RAD-Gigastar;
- Insert a specific description to identify better particular messages;
- Define all the parameters of an automotive Ethernet message, such as source and destination MAC address or the Ethertype in order to define a particular type of message (IPv4, IPv6, ARP, AVB);

- Define the timing of sending the message, for example if a message is periodic it is possible to select from 0.005 to 5 seconds;
- Define the content of the message by changing the Ethernet Payload.

This experimental configuration described above was used not only to interconnect the devices, but mainly to evaluate the practical behavior of MACsec in Automotive Ethernet environment. In particular, this setup allowed the:

- observation of the MACsec encryption process at Layer 2;
- verification of encrypted vs. non-encrypted traffic;
- analysis of frame structure before and after MACsec protection;
- evaluation of latency and communication continuity when MACsec is enabled.

Thanks to this configuration and Vehicle Spy, it was possible to monitor the traffic and actively generate messages, in order to reproduce a realistic in-vehicle communication scenario and observe how MACsec affects data transmission in terms of security and performance.

6.5 Capturing and Save Data

Vehicle Spy, as explained previously, plays a crucial role in reading the information exchanged on the network bus. For this reason, it allows the user to record the communication and analyze it in a second moment.

Vehicle Spy can save captured data in two different formats: `.vsb` and `.pcap`.

The first one, the Vehicle Spy Binary (`.vsb`) file, is a proprietary format developed by Intrepid Control Systems. Its main characteristics are:

- Ability to store messages from different bus types, such as CAN, CAN FD, LIN, and Automotive Ethernet;
- Storage of database information, including message IDs, signal names, and data bytes, with the possibility of replaying specific messages inside the software;

- Storage of the complete logging configuration.

In other words, a `.vsb` file is not a simple frames capture but a complete measurement session.

The Packet Capture or `.pcap` file, on the other hand, is a standard format used by network analysis tools such as Wireshark, tepdumb, or tshark. Compared to the VSB format, it is simpler and it is not specifically designed for Automotive purposes. Its main application is to store raw network frames with a basic structure. Each packet typically contains information such as source and destination addresses, protocol type, packet size, and timestamp.

To analyze the traffic stored in the `.pcap` files, the open-source software Wireshark was used.

In practice, Vehicle Spy was mainly used to analyze the communication from an automotive point of view, to have a clear message identification an interpretation in order to evaluate the impact of MACsec on the system. Wireshark, instead, was used to inspect the packet structure in detail, analyzing specific fields such as SCI, AN, PN, and ICV.

The combined use of these two file formats allowed both functional validation and security verification.

During the experiments, the data acquisition was performed using hardware timestamp in order to have precise timing information. In order to have a comparison between the encrypted and non-encrypted traffic, both cases were recorded.

For this reason, the logging action was configured to:

- Record full Ethernet frames;
- Store timestamps;
- Capture all the traffic continuously during the communication stream;
- Avoid packet filtering in order to observe a complete network behavior.

This configuration permits the analysis of MACsec overhead and frame structure modifications.

For each test, mainly two different acquisitions were performed: one without MACsec enable and one with MACsec active. This methodology allowed the direct comparison of frame structure, payload visibility and protocol overhead.

Chapter 7

Experimental Results

After setting up the experimental environment described in the previous chapter and illustrated in Figure 6.5, this Chapter presents and discusses the results obtained from the experiments and simulations conducted during the testing phase.

The first objective of the experimental session was to explore the functionalities of the Intrepid Control Systems tools, in order to evaluate the effectiveness and capabilities of the instruments used in the experimental setup.

During the test, data were collected under different conditions in order to analyze various scenarios and better understand the behavior of the system. One of the main aspects considered was the comparison between scenarios with MACsec enabled and disabled, allowing an evaluation of the impact of the security protocol on network performance. Additionally, the number of transmitted data frames was varied in order to analyze how the system behaves under different traffic loads.

Based on these conditions, the performance of the system, particularly with MACsec encryption enabled, was evaluated according to the following metrics:

- MACsec Keys Exchange time;
- Latency;
- Jitter;
- Overhead and number of transmitted packets.

Although these performance metrics are closely related, they do not always vary in the same way. For example, the introduction of security mechanisms such

as MACsec may slightly increase latency due to the additional processing required for encryption and decryption operations. Similarly, variations in network load can influence jitter and bandwidth utilization, affecting the overall communication performance.

For this reason, evaluating these parameters together provides a more comprehensive understanding of the network behavior. The combined analysis of latency, jitter, and bandwidth allows a clearer evaluation of the communication performance and the overall efficiency of the network. Observing how these parameters vary under different traffic conditions makes it possible to better understand the behavior of the system and highlights potential performance limitations.

7.1 MACsec Activation Time

The first part of the test is related to the MACsec working principle, which ensures secure communication between nodes. In the automotive field, it is crucial to establish a secure communication channel as quickly as possible so that ECUs can become operational without delay.

After the correct configuration of the Intrepid Control Systems tools and the Ethernet Camera, the following step was to start the packet transmission in the experimental network.

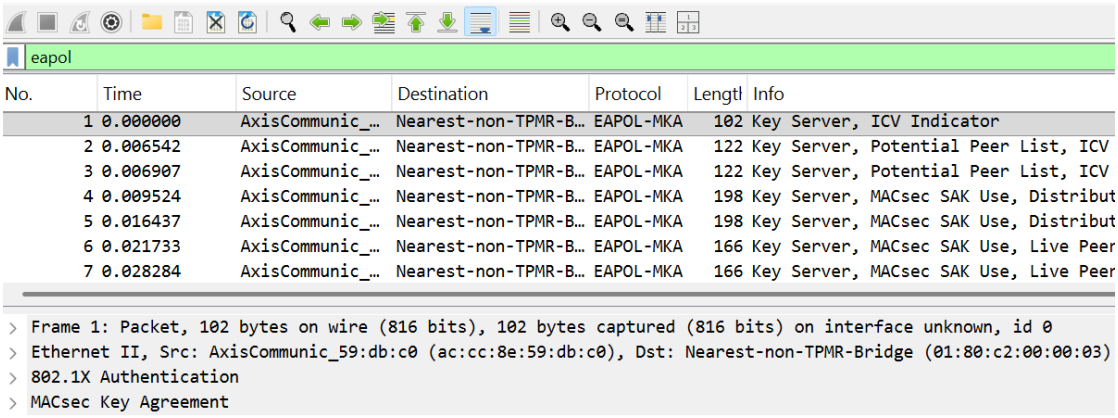


Figure 7.1: Initial Frames

The expected behavior of the setup is to create a secure channel, which is initiated by sending requests to recognize other nodes using the same security

protocol.

In Figure 7.1, the initial packets responsible for establishing secure communication are shown.

As expected, these requests are of the type Extensible Authentication Protocol over LAN (EAPoL), whose function is to identify destination nodes belonging to the same Connectivity Association (CA) for key exchange.

Subsequently, additional EAPoL frames are exchanged with other active nodes. This occurs because the experimental setup includes a PC connected to other units that the system recognizes as potential nodes to be added. After this exchange, the first MACsec frame is transmitted, as shown in Figure 7.2.

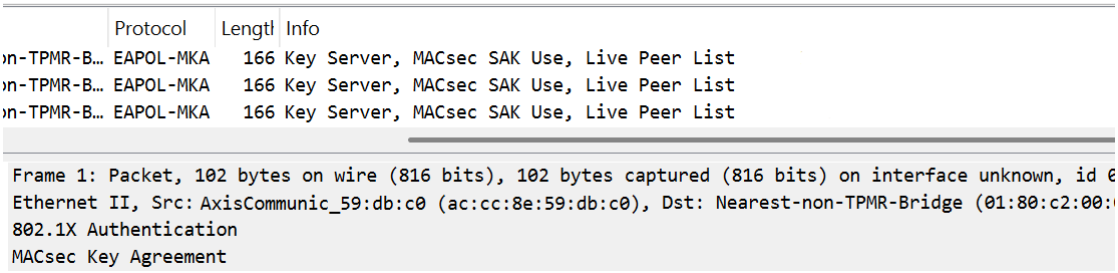


Figure 7.2: Initial frame about Secure Channel

By analyzing the `.pcap` file, it was possible to analyze the timestamps of the frames from the first key exchange to the last. The result obtained in the first capture are summarized in Table 7.1.

Table 7.1: Initial frames

Number	Time [hh:mm:ss]
1	18 : 34 : 43.782949
2	18 : 34 : 43.789491
3	18 : 34 : 43.789856
4	18 : 34 : 43.792473
5	18 : 34 : 43.799386
6	18 : 34 : 43.804682
7	18 : 34 : 43.811233

This means that the total activation time to ensure the connection between the nodes can be calculated as the difference between the first and the last frame. In the first test, the total activation time is:

$$\Delta t = 43.811233 - 43.782949 = 0.028284s \implies Activation\ Time \approx 28\ ms \quad (7.1)$$

This test was repeated multiple times, and the results are illustrated in Figure 7.3.

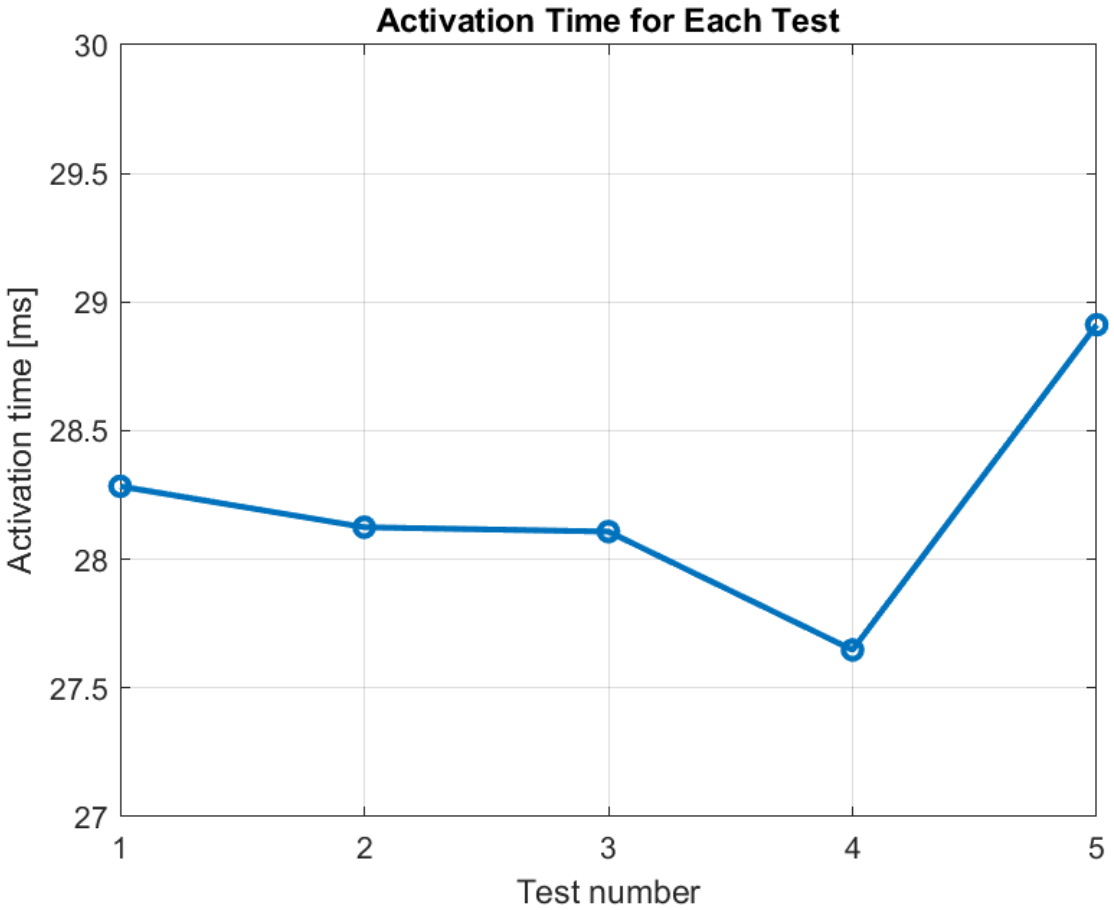


Figure 7.3: Activation Time, tested 5 times

The activation time of MACsec in the setup environment is about equal to 28 ms, an acceptable value for the automotive environment. This value can also be lower by optimizing the Key Server that is inside the SFP module.

7.2 Packets Latency

The main object of the latency analysis is the Ethernet camera used as an ECU. The camera transmits different packets over the network depending on the selected image quality.

By analyzing the image stream on the network using Vehicle Spy, it was possible to study the transmission behavior of the video stream. The camera transmits a 1280×720 pixel video by splitting each image into multiple Ethernet packets. Each frame is transmitted as a burst of approximately 30 Ethernet packets of 1514 bytes each, corresponding to the maximum Ethernet frame size (MTU).

Line	Time (abs/rel)	Tx	Er	Descri...	Source	Src Port	Destination	Dst Port	EtherType	Protocol	VLAN	Len	Seq#	Ack#	Window	Flags	Network
					192.168.0.90		192.168.0.80										
7222	0 ns			Ethern...	192.168.0.90	80	192.168.0.80	59869	IPv4	TCP		360	819915311	968853256	950	ACK,PSH	Ethernet
7223	29.9C⁻ms			Ethern...	192.168.0.90	80	192.168.0.80	59869	IPv4	TCP		1514	819915617	968853256	950	ACK	Ethernet

Figure 7.4: First Video frame

The first frame of the video stream can be identified by analyzing the packet details, as shown in Figure 7.5. It is possible to observe several parameters describing the image data, such as the total length of the frame and the image format, which in this case is JPEG/JFIF. Moreover, the beginning of the image can be identified through the Start Of Image (SOI) marker, represented by the hexadecimal value 0xFFD8, which is typical for the JFIF image format.

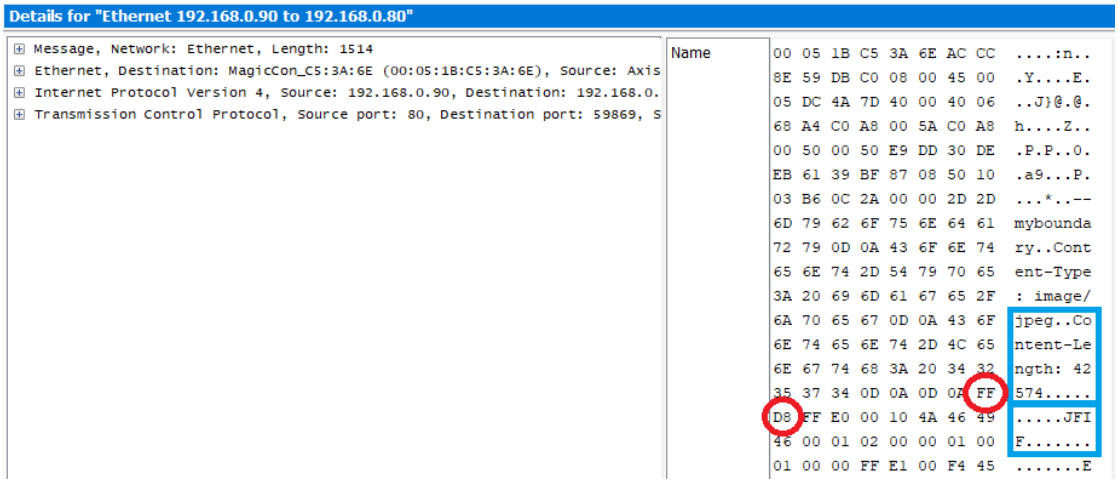


Figure 7.5: Details of the first frame of the video

Since the transmission timestamp generated by the camera is not available, the absolute end-to-end latency cannot be directly measured. For this reason, the latency has been estimated using the packet inter-arrival time measured at the receiver, so in this case the PC, which represents the time difference between two consecutive packets belonging to the same video stream. Even if this method does not provide the exact transmission latency, it allows to estimate the delay perceived at the receiver and to compare different transmission conditions.

By capturing the traffic, it was possible to measure the time difference between consecutive packets and compare the behavior of the video stream with MACsec enabled and disabled.

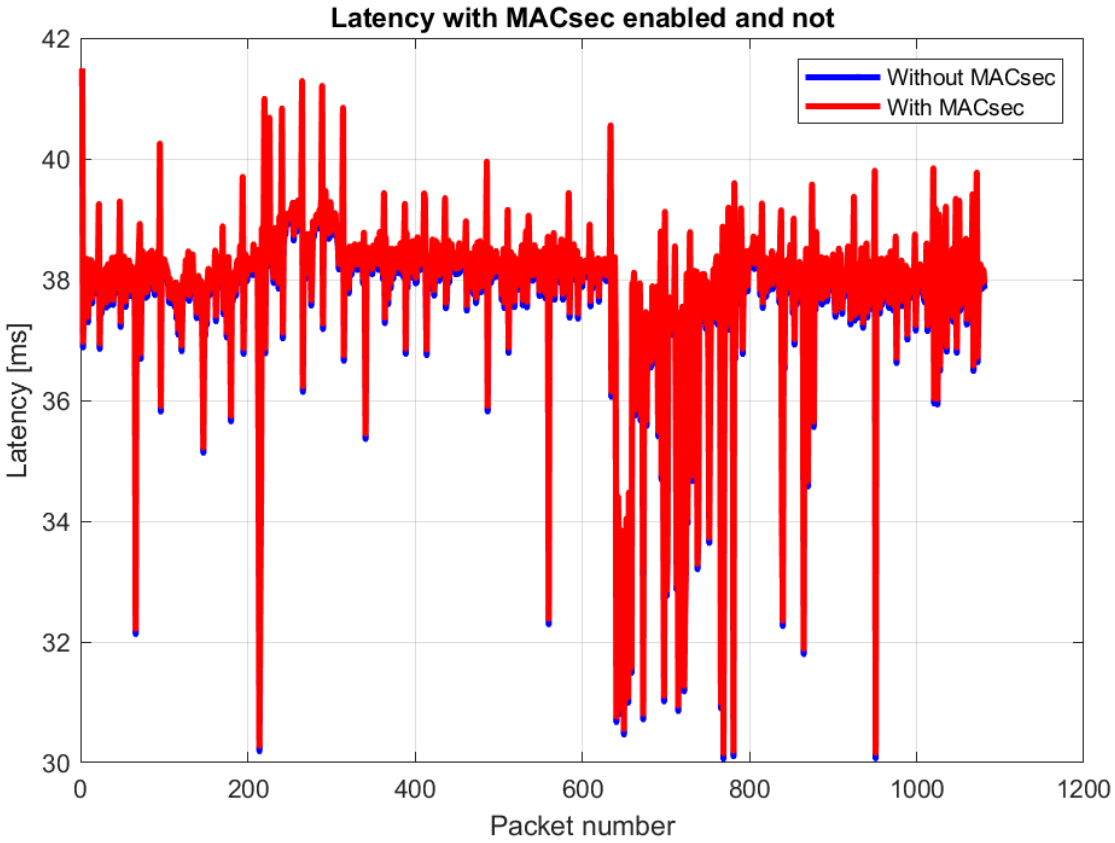


Figure 7.6: Estimated packet latency with MACsec enabled (red) and MACsec disabled (blue)

As shown in Figure 7.6, the average transmission latency of the packets corresponds to approximately one complete video frame every 38 ms. This value

represents the time required by the camera to transmit a full image over the Ethernet network.

In order to observe in a better way the difference between the two configurations, a zoomed view of the packet range between 450 and 550 is shown in Figure 7.7.

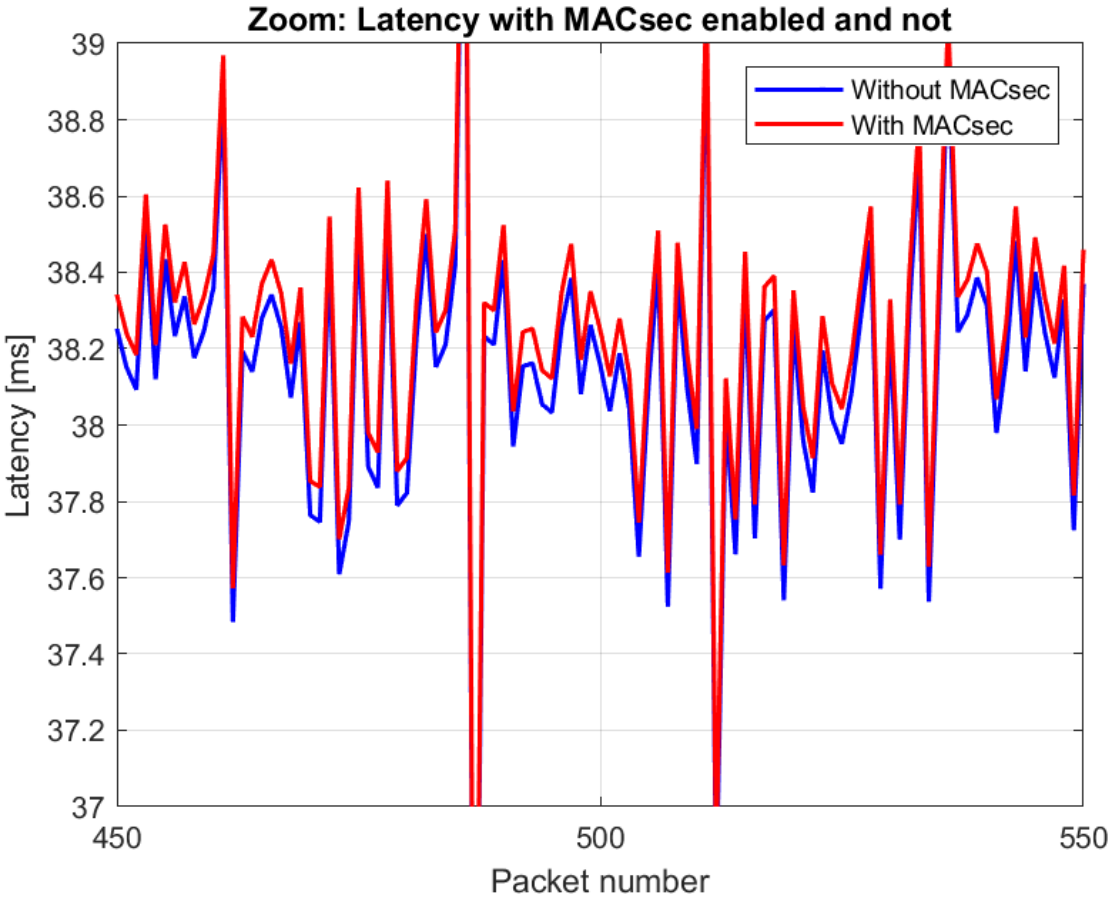


Figure 7.7: Zoom of the latency values between packet 450 and 550

In both figures, it can be observed that the red line, which represents the packets transmitted with MACsec encryption enabled, shows a slightly higher latency value compared to the case without MACsec. By analyzing the measurements, the additional latency introduced by MACsec is approximately 80–90 μs per packet.

This increase is primarily due to the overhead introduced by MACsec, which adds additional bytes to the Ethernet frame, specifically the SecTAG and ICV fields. These additional bytes slightly increase the frame size and introduce a small processing delay during encryption and decryption.

Overall, the results of this test show that MACsec has a minimal impact on Ethernet packet transmission, even for a high quality video stream. The additional latency introduced by encryption is orders of magnitude smaller than the packet transmission time itself. For this reason, this added latency is negligible, particularly in typical automotive scenarios where cameras are used for video streaming or ADAS systems.

7.3 Overhead Introduced

With MACsec enabled on the Ethernet link between the Camera and the PC, additional fields are inserted into the Ethernet frame according to the IEEE 802.1AE standard. In particular, every frame encrypted by MACsec becomes longer due to the addition of the SecTAG and the ICV, resulting in an increase of 32 bytes per frame.

As done previously, the test was carried out in two different scenarios: with MACsec enabled and without MACsec, in order to evaluate the frame size increase and its effects.

As shown in Figure 7.4, in the case of a video stream without MACsec protection, most frames have a length of 1514 bytes, which corresponds to the standard maximum transmission unit (MTU). After enabling MACsec, the frame length increased to 1546 bytes, consistent with the addition of 32 bytes for the SecTAG and the ICV.

Using these values, the overhead introduced by MACsec per frame can be calculated as:

$$O_{frame} = \frac{L_{MACsec} - L_{noMACsec}}{L_{noMACsec}} \tag{7.2}$$

In this case, $L_{MACsec} = 1546$ bytes and $L_{noMACsec} = 1514$ bytes, so the maximum overhead percentage is:

$$O_{frame} = \frac{1546 - 1514}{1514} \times 100 \approx 2.1 \tag{7.3}$$

For frames near the MTU value, this overhead is not very significant. However, for smaller frames, the relative impact can be more important.

Regarding the video stream, each video image is transmitted using multiple

Ethernet frames that form a burst; in our case, approximately 30 frames per image. Assuming each image is transmitted using 30 frames, the total overhead for a single image is:

$$O_{burst} = N \times 32 = 30 \times 32 = 960 \text{ bytes} \tag{7.4}$$

This means that using MACsec for a video stream adds 960 bytes per image compared to a non-protected transmission, which can increase the required average bitrate for the link.

The average bitrate represents the amount of data transmitted per unit of time, expressed in bit/s or Mbit/s. In automotive applications, this is crucial to ensure that the link can handle all the traffic.

The additional bitrate due to MACsec can be calculated as:

$$\Delta B = N_f \times 32 \times 8 \tag{7.5}$$

where 32 is the MACsec overhead in bytes, 8 is the conversion factor from bytes to bits, and N_f is the number of Ethernet frames transmitted per second. In our case, the camera transmits at 30 frame per second (fps), and each image is divided into approximately 30 Ethernet packets. Therefore, the number of transmitted Ethernet frames per second is:

$$N_f = 30 \times 30 = 900 \ frames/s \tag{7.6}$$

The additional bitrate introduced by MACsec becomes:

$$\Delta B = 900 \times 32 \times 8 = 230400 \text{ bit/s (or 230.4 kbps)} \tag{7.7}$$

Thanks to these measurements, it was possible to observe that enabling MACsec does not change the number of frames needed to transmit a single video image. Instead, it consistently increases the size of each protected frame, leading to a higher traffic load on the link. On a 1 Gbps link, this increase is generally negligible and easily handled by the link. On a 100 Mbps link, however, the relative impact becomes more relevant.

In our case, MACsec introduces a consistent and predictable increase in Ethernet frame size, of about 32 bytes per frame, corresponding to approximately 2.1%

for a 1514-byte frame. While this reduces transmission efficiency and increases the required bitrate, the impact remains limited and fully compatible with the requirements of modern Automotive Ethernet networks.

Chapter 8

Conclusions and Further Developments

With the enormous demand in the automotive sector and the continuous evolution of technologies inside modern vehicles, manufacturers are constantly developing new solutions to improve safety, security, and infotainment systems, aiming to simplify and enhance the experience of the end user. As a consequence, the communication network inside a vehicle has significantly evolved, especially in the last two decades.

The increasing number of Electronic Control Units (ECUs) and connected services has led to a massive growth in the amount of data exchanged within the vehicle, creating the need for more efficient and secure communication infrastructures.

The presence of many ECUs allows a significant improvement in vehicle functionalities, but at the same time it increases the complexity of designing and managing the in-vehicle network while fulfilling strict performance and safety requirements.

To address these challenges, automotive companies are progressively moving from traditional communication systems, such as CAN, to more advanced technologies. Automotive Ethernet represents one of the most promising solutions, since it enables high bandwidth communication and supports advanced applications such as Advanced Driver Assistance Systems (ADAS), infotainment systems, cameras and Vehicle-to-Everything (V2X) communication.

However, these functionalities generate a large amount of data, some of which are safety-critical or sensitive. If not protected properly, this information could be accessed or manipulated by malicious actors. For this reason, cybersecurity has

become a fundamental topic of Automotive Ethernet networks.

This thesis provided an overview of Automotive Ethernet, focusing on its architecture and working principles, and then analyzed MACsec as a suitable cybersecurity solution at the link layer. Beyond the state-of-the-art analysis, this work explained the creation of an experimental setup using professional Automotive Ethernet tools in order to evaluate MACsec in a realistic scenario.

Through the experimental measurements performed using tools and software provided by Intrepid Control Systems, this study has proven that the integration of MACsec in an Automotive Ethernet network preserves the main advantages of the technology, such as high bandwidth and low latency, while adding encryption and data integrity protection.

Therefore, this work contributes to reducing the gap between theoretical studies and practical implementation, confirming that securing Automotive Ethernet with MACsec can represent a valid solution for next-generation vehicles.

Furthermore, this thesis can be considered a starting point for further developments related to the integration of MACsec as a cybersecurity protocol in in-vehicle networks. Future studies could investigate in more detail the behavior of secure gateways handling MACsec-encrypted frames, especially in complex network topologies. Another important aspect that deserves further analysis is the economic impact of introducing MACsec inside a vehicle, including hardware requirements and overall system cost.

Bibliography

[1] Rinat Khoussainov and Ahmed Patel. «LAN security: problems and solutions for Ethernet networks». In: *Computer Standards Interfaces* 22.3 (2000), pp. 191–202. ISSN: 0920-5489. DOI: https://doi.org/10.1016/S0920-5489(00)00047-7. URL: https://www.sciencedirect.com/science/article/pii/S0920548900000477 (cit. on p. 2).

[2] EVITA Consortium. *EVITA: E-safety Vehicle Intrusion Protected Applications*. Project website. Co-funded within the European Union's Seventh Framework Programme (FP7) to design, verify and prototype secure architectures for automotive on-board networks, protecting components and sensitive data against tampering and compromise. 2011. URL: https://www.evita-project.org/index.html (cit. on p. 3).

[3] SeVeCom Consortium. *SeVeCom – Secure Vehicle Communication*. Project website. EU-funded research project focusing on security and privacy mechanisms for vehicular and inter-vehicular communications, part of the European Commission IST programme. 2009. URL: https://www.sevecom.eu (cit. on p. 3).

[4] Moritz Schaufelberger, Johannes Röhrl, Diego Pereda, and Ralf Demo. *PERADEMO: A Demonstrator for Performance Analysis of Real-Time Ethernet Protocols*. Tech. rep. University of Ulm, Institute of Distributed Systems (IUI), 2011. URL: https://www.uni-ulm.de/fileadmin/website_uni_ulm/iui.inst.100/institut/verz-ma-ehedem/schaub/2011-MC2R-perademo.pdf (cit. on p. 3).

[5] PRESERVE Consortium. *PRESERVE Project – Privacy and Security for Automotive Systems*. Project website. 2025. URL: https://www.preserve-project.eu/www.preserve-project.eu/index.html (cit. on p. 3).

[6] Omer Khan, Muhammad Mubashir, and Jawaid Iqbal. «Comprehensive Review of CAN Bus Security: Vulnerabilities, Cryptographic and IDS Approaches, and Countermeasures». In: *Journal of Engineering Technology and Applied Physics* 7 (Mar. 2025), pp. 19–26. DOI: 10.33093/jetap.2025.7.1.4 (cit. on p. 3).

[7] Junko Takahashi, Yosuke Aragane, Toshiyuki Miyazawa, Hitoshi Fuji, Hirofumi Yamashita, Keita Hayakawa, Shintarou Ukai, and Hiroshi Hayakawa. «Automotive Attacks and Countermeasures on LIN-Bus». In: *Journal of Information Processing* 25 (Feb. 2017), pp. 220–228. DOI: 10.2197/ipsjjip.25.220 (cit. on p. 3).

[8] Pal-Stefan Murvay and Bogdan Groza. «Practical Security Exploits of the FlexRay In-Vehicle Communication Protocol». In: Jan. 2019, pp. 172–187. ISBN: 978-3-030-12142-6. DOI: 10.1007/978-3-030-12143-3_15 (cit. on p. 3).

[9] Timo Kiravuo, Mikko Särelä, and Jukka Manner. «A survey of ethernet LAN security». In: *Communications Surveys Tutorials, IEEE* 15 (Jan. 2013), pp. 1477–1491. DOI: 10.1109/SURV.2012.121112.00190 (cit. on p. 3).

[10] Aida Ben Chehida Douss, Ryma Abassi, and Damien Sauveron. «State-of-the-art survey of in-vehicle protocols and automotive Ethernet security and vulnerabilities». In: *Mathematical Biosciences and Engineering* 20 (Jan. 2023), pp. 17057–17095. DOI: 10.3934/mbe.2023761 (cit. on p. 3).

[11] Berardino Carnevale, Francesco Falaschi, Luca Crocetti, Harman Hunjan, Samson Bisase, and Luca Fanucci. «An implementation of the 802.1AE MAC Security Standard for in-car networks». In: Dec. 2015, pp. 24–28. DOI: 10.1109/WF-IoT.2015.7389021 (cit. on p. 4).

[12] Berardino Carnevale, Luca Fanucci, Samson Bisase, and Harman Hunjan. «MACsec-Based Security for Automotive Ethernet Backbones». In: *Journal of Circuits, Systems and Computers* 27.05 (2018), p. 1850082. DOI: 10.1142/S0218126618500822 (cit. on p. 4).

[13] Andreas Theissler. «Detecting anomalies in multivariate time series from automotive systems». PhD thesis. Dec. 2013 (cit. on p. 4).

[14] Cristoph Hammerschmidt. «Vehicle E/E-Architecture: Reduce to the Max». In: 2019. URL: https://www.eenewseurope.com/en/vehicle-e-e-architecture-reduce-to-the-max/ (cit. on p. 6).

[15] Hailo. *AI ADAS and AD (Autonomous Driving) | Automotive Applications*. 2025. URL: https://hailo.ai/applications/automotive/adas-and-ad/ (cit. on p. 6).

[16] Erik Devaney. *What Is Vehicle-to-Everything (V2X) Technology?* Built In. 2023. URL: https://builtin.com/articles/v2x-vehicle-to-everything (cit. on p. 7).

[17] NXP Semiconductors. *Driving Automotive Design to 5nm Technology*. 2021. URL: https://www.nxp.com/company/about-nxp/smarter-world-blog/BL-AUTOMOTIVE-DESIGN-TO-5NM-TECHNOLOGY (cit. on p. 8).

[18] Lucia Lo Bello, Gaetano Patti, and Luca Leonardi. «A Perspective on Ethernet in Automotive Communications—Current Status and Future Trends». In: *Applied Sciences* 13.3 (2023). ISSN: 2076-3417. DOI: 10.3390/app13031278. URL: https://www.mdpi.com/2076-3417/13/3/1278 (cit. on pp. 9, 17, 52, 53).

[19] Nicolas Navet and Francoise Simonot-Lion. *Automotive Embedded Systems Handbook*. 1st. USA: CRC Press, Inc., 2008. ISBN: 084938026X (cit. on pp. 9, 10).

[20] Ixia. *Automotive Ethernet: An Overview*. White Paper 915-3510-01 Rev. A. Calabasas, CA: Ixia, May 2014. URL: https://support.ixiacom.com/sites/default/files/resources/whitepaper/ixia-automotive-ethernet-primer-whitepaper_1.pdf (cit. on pp. 9, 10, 12).

[21] Thomas Nolte, Hans Hansson, and Lucia Lo Bello. «Automotive communications-past, current and future». In: Jan. 2005. DOI: 10.1109/ETFA.2005.1612631 (cit. on pp. 11, 12).

[22] SRM Technologies. *Vehicle Electronics Architecture – Past, Present, and Future*. https://www.srmtech.com/knowledge-base/blogs/vehicle-electronics-architecture-past-present-and-future/. Mar. 2025 (cit. on pp. 12–14).

[23] Peter Aberl, Stefan Haas, and Arun Vemuri. *How a Zone Architecture Paves the Way to a Fully Software-Defined Vehicle*. Tech. rep. SPRY345C. Texas Instruments, Oct. 2024. URL: https://www.ti.com/lit/pdf/SPRY345 (cit. on pp. 12, 14, 15).

[24] Promwad. *Zonal Architecture in Modern Vehicle Electronics: The Shift from Domain-Based to Zonal ECUs*. Promwad website. Apr. 2025. URL: https://promwad.com/news/zonal-architecture-vehicle-electronics (cit. on p. 14).

[25] Broadcom. *Securing Software-Defined Vehicles with Zonal E/E Architectures*. Broadcom Engineering Blog. June 2025. URL: https://www.broadcom.com/blog/securing-software-defined-vehicles-with-zonal-e-e-architectures (cit. on p. 15).

[26] Colt Correa, John Simon, and Martin Gubow. *Automotive Ethernet: The Definitive Guide*. 2nd. Intrepid Control Systems, 2022, p. 1032. ISBN: 9780578274003 (cit. on pp. 17–20, 29, 32, 35–38, 46, 49).

[27] Tim Wilmshurst. «CHAPTER 20 - Connectivity and networks». In: *Designing Embedded Systems with PIC Microcontrollers (Second Edition)*. Ed. by Tim Wilmshurst. Second Edition. Boston: Newnes, 2010, pp. 575–590. ISBN: 978-1-85617-750-4. DOI: https://doi.org/10.1016/B978-1-85617-750-4.10025-3. URL: https://www.sciencedirect.com/science/article/pii/B9781856177504100253 (cit. on p. 18).

[28] CanEasy GmbH. *Was ist LIN? – LIN-Bus Grundlagen*. 2025. URL: https://www.caneasy.de/caneasyhelp/was_ist_lin.htm (cit. on p. 18).

[29] Ha Thanh, Banh Thanh, and Bang Khac. «Most (Media Oriented Systems Transport): A Comprehensive Analysis of the Automotive Multimedia Networking Protocol». In: *International Journal of Advances in Engineering and Management* 7 (Apr. 2025), pp. 647–651. DOI: 10.35629/5252-0704647651 (cit. on p. 19).

[30] GuardKnox Cyber Technologies Ltd. *Automotive Ethernet*. 2024. URL: https://www.guardknox.com/automotive-ethernet/ (cit. on p. 19).

[31] Vector Informatik GmbH. *FlexRay E: Communication Architecture (Learning Module)*. Apr. 2018. URL: https://elearning.vector.com/mod/page/view.php?id=379 (cit. on p. 20).

[32] Donovan Porter. «100BASE-T1 Ethernet: the evolution of automotive networking». In: *Texas Instruments, Techn. Ber* 2 (2018) (cit. on p. 23).

[33] Keysight Technologies. *Why use 10BASE-T1S instead of CAN and its variants*. 2024. URL: https://www.keysight.com/blogs/en/tech/2024/02/8/how-is-10base-t1s-different-from-can (cit. on p. 28).

[34] Granite River Labs (GRL Team). *Introduction to 10BASE-T1S Automotive Ethernet Test Standards*. Dec. 2023. URL: https://www.graniteriverlabs.com/en-us/technical-blog/automotive-ethernet-10-base-t1s (cit. on p. 29).

[35] Inc. Teledyne LeCroy. *What Is 10Base-T1S Automotive Ethernet?* Tech. rep. Teledyne LeCroy, Mar. 2023. URL: https://cdn.teledynelecroy.com/files/appnotes/10base-t1s-automotive-ethernet.pdf (cit. on p. 30).

[36] Rita Mitra, Glenn Brown, Melanie Huffman, and Hongyi Zhu. *Telecommunications and Networking*. University of Texas at San Antonio / Pressbooks, 2024. URL: https://utsa.pressbooks.pub/networking (cit. on p. 30).

[37] Wikipedia contributors. *MAC address — Wikipedia, The Free Encyclopedia*. 2025. URL: https://en.wikipedia.org/w/index.php?title=MAC_address&oldid=1312045712 (cit. on p. 31).

[38] Vijay K. Garg and Yih-Chen Wang. «6 - Communication Network Architecture». In: *The Electrical Engineering Handbook*. Ed. by WAI-KAI CHEN. Burlington: Academic Press, 2005, pp. 989–1003. ISBN: 978-0-12-170960-0. DOI: https://doi.org/10.1016/B978-012170960-0/50074-8. URL: https://www.sciencedirect.com/science/article/pii/B9780121709600500748 (cit. on pp. 32, 36).

[39] Gary Macknofsky. *Benefits of VLANs and Q in Q*. Tech. rep. EXFO, 2006. URL: https://www.exfo.com/contentassets/8eff0ee8c70d48b28c94860a24058b70/exfo_tnote032_benefits-vlans_en.pdf?utm_source=chatgpt.com (cit. on p. 34).

[40] Vector Informatik GmbH. *Ethernet Frame*. 2021. URL: https://elearning.vector.com/mod/page/view.php?id=157 (cit. on pp. 34, 35).

[41] Kirsten Matheus and Thomas Königseder. *Automotive Ethernet*. 2nd ed. Cambridge University Press, 2017 (cit. on pp. 35, 44, 45, 52, 53).

[42] J.F. Kurose and K. Ross. *Computer Networking: A Top-Down Approach, Global Edition*. Pearson Education, 2021. ISBN: 9781292405513 (cit. on p. 36).

[43] Ignas Anfalovas. *What Is a Subnet Mask? 2024 Updated Guide*. July 2025. URL: https://www.ipxo.com/blog/what-is-subnet-mask (cit. on p. 37).

[44] Olabenjo Babatunde and Omar Al-Debagy. «A Comparative Review Of Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6)». In: *International Journal of Computer Trends and Technology* 13 (July 2014). DOI: 10.14445/22312803/IJCTT-V13P103 (cit. on pp. 37–39).

[45] Michel Bakni and Sandra Hanbo. «A Survey on Internet Protocol version 4 (IPv4)». In: *WikiJournal of Science* 5 (Jan. 2022), p. 2. DOI: 10.15347/WJS/2022.002 (cit. on p. 37).

[46] Henry Paul and Kinn Bakon. «A STUDY ON IPv4 and IPv6: THE IMPORTANCE OF THEIR CO-EXISTENCE». In: *International Journal of Information Systems and Engineering* 4 (Nov. 2016), pp. 97–106. DOI: 10.24924/ijise/2016.11/v4.iss2/97.106 (cit. on p. 39).

[47] K.R. Fall and W.R. Stevens. *TCP/IP Illustrated: The Protocols, Volume 1*. Addison-Wesley Professional Computing Series. Pearson Education, 2011. ISBN: 9780132808187. URL: https://books.google.it/books?id=a23OAn5i8R0C (cit. on p. 39).

[48] FP Senekal and Johannes Vorster. «Network mapping and usage determination». In: *Military Information Communications Simposium South Africa 2007, Pretoria* (July 2007) (cit. on p. 40).

[49] «TCP and UDP». In: *HCNA Networking Study Guide*. Ed. by Ltd. Huawei Technologies Co. Singapore: Springer Singapore, 2016, pp. 167–178. ISBN: 978-981-10-1554-0. DOI: 10.1007/978-981-10-1554-0_7. URL: https://doi.org/10.1007/978-981-10-1554-0_7 (cit. on pp. 40, 43).

[50] Matija Mikac. «NETWORKING CASE STUDY IN STEM EDUCATION - TRANSPORT LAYER PROTOCOL (TCP AND UDP) LABS». In: July 2020, pp. 2328–2337. DOI: 10.21125/edulearn.2020.0715 (cit. on pp. 41, 42).

[51] A.S. Tanenbaum, D.J. Wetherall, S. Gaito, and D. Maggiorini. *Reti di calcolatori*. Pearson, 2011. ISBN: 9788871926407. URL: https://books.google.it/books?id=vXmJZwEACAAJ (cit. on p. 43).

[52] Embitel Technologies. *How SOME/IP Enables Service-Oriented Architecture in ECU Network*. URL: https://www.embitel.com/blog/embedded-blog/how-some-ip-enables-service-oriented-architecture-in-ecu-network (cit. on p. 44).

[53] Inc. The MathWorks. *What Is Service-Oriented Architecture (SOA)?* 2025. URL: https://www.mathworks.com/discovery/soa.html (cit. on p. 47).

[54] *Road vehicles — Diagnostic communication over Internet Protocol (DoIP) — Part 2: Transport protocol and network layer services*. Published June 2025. Geneva, Switzerland, 2025 (cit. on p. 47).

[55] Embitel Technologies. *What are the important security aspects of DoIP-based in-vehicle network and related best practices*. 2025. URL: https://www.embitel.com/blog/embedded-blog/what-are-the-important-security-aspects-of-doip-based-in-vehicle-network-and-related-best-practices (cit. on p. 48).

[56] Farooq Ansari. *DoIP: Remote Diagnosis Protocol for Smart Cars*. Apr. 2023. URL: https://www.keysight.com/blogs/en/tech/nwvs/2023/04/20/doip-remote-diagnosis-protocol-for-smart-cars (cit. on p. 48).

[57] Radhika Krishnan. *Time Sensitive Networks — Audio Video Bridging*. Mar. 2023. URL: https://nestdigital.com/blog/time-sensitive-networks-audio-video-bridging (cit. on p. 50).

[58] Hyung-Taek Lim, Daniel Herrscher, Martin Johannes Waltl, and Firas Chaari. «Performance analysis of the IEEE 802.1 ethernet audio/video bridging standard». In: *International ICST Conference on Simulation Tools and Techniques*. 2012. URL: https://api.semanticscholar.org/CorpusID:14913994 (cit. on p. 52).

[59] Colt Correa. *MACsec & MACsec Key Agreement (MKA) by Colt Correa Author - Automotive Ethernet - Definitive Guide*. Intrepid Control Systems. June 2025. URL: https://www.youtube.com/watch?v=bAJWiLsrffw (cit. on pp. 56, 62, 63).

[60] Tobias Belitz. *The Art of Networking (Series 6): MACsec and CANsec – Layer 2 Security for High Performance Networks*. Dec. 2021. URL: https://www.renesas.com/en/blogs/art-networking-series-6-macsec-and-cansec-layer-2-security-high-performance-networks?srsltid=AfmBOoq7h5Nh0ofCDUUwr07li_02dnMBB6F9XiKxqeBUMcthTkWWOgXA (cit. on p. 57).

[61] Tim Lackorzynski, Sebastian Rehms, Tao Li, Stefan Köpsell, and Hermann Härtig. *Secure and Efficient Tunneling of MACsec for Modern Industrial Use Cases*. 2022. arXiv: 2205.12748 [cs.CR]. URL: https://arxiv.org/abs/2205.12748 (cit. on p. 58).

[62] AUTOSAR. *Explanation of MACsec and MKA Protocols Implementation — AP_EXP_MACsec*. Tech. rep. R23-11/AP/AP_EXP_MACsec. AUTOSAR, 2023. URL: https://www.autosar.org/fileadmin/standards/R23-11/AP/AUTOSAR_AP_EXP_MACsec.pdf (cit. on p. 60).

[63] Comcores. *Automotive Ethernet Security Using MACsec*. Jan. 2025. URL: https://www.design-reuse.com/article/61570-automotive-ethernet-security-using-macsec/ (cit. on p. 66).

[64] Comcores. *MACsec for Deterministic Ethernet applications*. Sept. 2022. URL: https://www.design-reuse.com/article/61371-macsec-for-deterministic-ethernet-applications/ (cit. on pp. 69, 70).

[65] Elektrobit. *Secure Automotive Ethernet for automated driving—Multi-level security architecture*. Multi-level security architecture for Automotive Ethernet to protect in-vehicle networks against malicious attacks, enabling automated driving. 2025. URL: https://www.elektrobit.com/blog/automotive-ethernet-automated-driving-multi-level-security/ (cit. on p. 71).

[66] Lars Völker. *Why is Network Security in Vehicles so hard?* Tech. rep. Automotive Networks 2018 Conference, Nov. 2018. URL: https://automotive-

network - security . com / papers / 2018 - 11 - 06 - AutomotiveNetworks - DrLarsVoelker_v1.0.1_final.pdf (cit. on pp. 71, 72).

[67] Inc. Intrepid Control Systems. *RAD-Gigastar: 1000BASE-T1 Active Tap, Gateway, Media Converter, Vehicle Interface & Data Logger*. Tech. rep. Intrepid Control Systems, 2025. URL: https://guide.intrepidcs.com/brochures/icsusa/RAD_Gigastar.pdf (cit. on pp. 76, 77).

[68] Inc. Marvell Semiconductor. *Marvell Unveils Industry's First Automotive Gigabit Ethernet PHY with MACsec Security*. Oct. 2020. URL: https://www.prnewswire.com/news-releases/marvell-unveils-industrys-1st-automotive-gigabit-ethernet-phy-with-macsec-security-301147467.html (cit. on p. 78).

[69] Infineon Technologies AG. *88Q222xM Automotive Ethernet PHY – BRIGHTLANE™ 1000BASE-T1*. 2025. URL: https://www.infineon.com/products/ethernet/automotive-phy/88q222xm#about (cit. on p. 78).

[70] Inc. Intrepid Control Systems. *88Q2221M 1000BASE-T1 with TC10 & MACsec SFP Module*. Tech. rep. Product brochure rev. 20251112; Automotive Ethernet Small Form Pluggable module supporting 100/1000BASE-T1, TC10 sleep/wake and MACsec compliance. Intrepid Control Systems, Nov. 2025. URL: https://guide.intrepidcs.com/brochures/icsusa/SFP_88Q2221M_1000BASE-T1.pdf (cit. on p. 78).

[71] OPEN Alliance SIG. *TC10 Wake-up and Sleep Specification for Automotive Ethernet*. Tech. rep. Version 2.0, Sleep/Wake-up Specification for Automotive Ethernet (Feb 2017). Available at opensig.org. OPEN Alliance SIG, Feb. 2017. URL: https://opensig.org/wp-content/uploads/2024/01/TC10-Wake-up-and-Sleep-Specification-for-Automotive-Ethernet_11-2017.pdf (cit. on p. 78).

[72] Phytools. *Rosenberger H-MTD Connecting Cables*. Online product page. Accessory for Rosenberger H-MTD high-speed modular twisted-pair Ethernet connectors, female-to-female cable (STP). 2025. URL: https://phytools.com/products/rosenberger-h-mtd-connecting-cables (cit. on p. 80).