Pós-Graduação em Ciência da Computação

RODRIGO ANTONIO ALVES DO NASCIMENTO

Design and Development of IDS for AVB/TSN

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
http://cin.ufpe.br/~posgraduacao

Recife
2019

RODRIGO ANTONIO ALVES DO NASCIMENTO

Design and Development of IDS for AVB/TSN

Thesis presented to the Graduate Program in Computer Science of the Centro de Informatica at the Universidade Federal de Pernambuco in complete fulfillment of the requirements for the degree of Master of Science (MSc) in Computer Science.

Área de Concentração: Redes de Computadores e Sistemas Distribuídos.

Supervisor: Divanilson Rodrigo de Sousa Campelo

Recife
2019

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

N244d Nascimento, Rodrigo Antonio Alves do
Design and development of IDS for AVB/TSN / Rodrigo Antonio Alves do Nascimento. – 2019.
97 f.: il., fig., tab.

Orientador: Divanilson Rodrigo de Sousa Campelo.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2019.
Inclui referências.

1. Redes de computadores. 2. Segurança. I. Campelo, Divanilson Rodrigo de Sousa (orientador). II. Título.

004.6 CDD (23. ed.) UFPE - CCEN 2020 - 51

Rodrigo Antonio do Nascimento

“Design and Development of IDS for AVB/TSN”

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Aprovado em: 29/05/2019.

BANCA EXAMINADORA

_____

Prof. Dr. Abel Guilhermino
Centro de Informática /UFPE

_____

Prof. Dr. Max Mauro Dias Santos
Departamento de Eletrônica /UTFPR-PG

_____

Prof. Dr. Divanilson Rodrigo de Sousa Campelo
Centro de Informática / UFPE
(**Orientador**)

Dedico este trabalho a minha família e minha esposa que foram porto seguro perante as dificuldades durante este percurso.

ACKNOWLEDGEMENTS

Esse trabalho não teria sido realizado sem a ajuda de inúmeras pessoas as quais gostaria de deixar aqui meu reconhecimento.

Primeiramente agradeço a Deus por sempre me guiar, ajudar e me dar forças frente aos questionamentos e dificuldades da vida, sem Ele nada disso seria possível.

Aos meus pais, Jorge e Jussara, por sempre terem investido em minha educação, colocando-a em primeiro lugar, apesar das dificuldades da vida. E também por terem me amado e incentivado durante toda minha vida. Amo vocês.

Agradeço também a minha esposa, Cecília, por todo seu carinho e incentivo durante os últimos 10 anos. Por sua compreensão durante meus mais de 2 anos de intercâmbio onde estivemos afastados, mas com a certeza de seria melhor para nosso futuro. Um obrigado também para meus sogros e cunhado por terem cuidado da minha princesa enquanto estive fisicamente ausente e por também me apoiarem.

Gostaria de agradecer ao Prof. Divanilson Campelo, que depositou sua confiança em mim para ser seu aluno, compartilhando seus conhecimentos e experiências de forma bastante enriquecedoura. Gostaria de agradecer em especial pela confiança em ter me indicado para a vaga de estágio na Alemanha, onde tive a oportunidade de trabalhar em um dos maiores grupos dedicados a Ethernet Automotiva do mundo.

Agraço aos meus colegas de trabalho no CESAR, Boga, Braga, Jika, Paulo e Thun Pin, por terem tornado o ambiente de trabalho descontraído, pelas risadas nos almoços e pelos projetos enriquecedores em que trabalhamos juntos.

Agradeço também aos meus colegas de trabalho na Alemanha, em especial a Fábio, Ibrahim, Karthik, Michael e Timo, por terem contribuído com este trabalho e por terem amenizado as dificuldades de estar longe de casa num momento de bastante pressão como o mestrado.

Agradeço aos meus amigos do Magos do Céu (Eles sabem quem são) por sempre lembrarem de mim e manterem o contato mesmo estando ausente em diversas ocasiões.

Um abraço mais que especial ao amigo Raul Aragão que perdi durante esse período. Sua falta será sempre sentida mas sua luz continuará nos alegrando para sempre.

Expresso também minha gratidão aos professores da minha banca avaliadora, Professor Abel (UFPE) e Professor Max Mauro (UTFPR) por aceitarem o convite e dedicarem tempo a leitura deste trabalho, contribuindo com críticas e sugestões para a melhoria desta pesquisa.

A todos, muito obrigado.

ABSTRACT

In the past few years, the number of attacks focusing on automotive systems have grown tremendously. In general, to combat such attacks, the use of a layered model of security is advised, where complementary security mechanisms can be applied to protect devices. Among them are Intrusion Detection System (IDS), which monitor systems to detect anomalous activities. With the emerging use of deterministic Ethernet brought by AVB/TSN into automotive Ethernet networks, newer attack surfaces are expected in cars. These are related to the set of new features to enable the parallel transmission of time-sensitive and best-effort media, management data, stream reservation, time synchronization and the connections among them. This thesis fills the gap left by previous works by: Summarizing exploitation vectors present on transport protocols (IEEE 1722/1733), management protocol (IEEE 1722.1) and time synchronization (IEEE 802.1AS); Discussing the software/hardware requirements of the IDS, which also incorporates IEEE 802.1Qci features, deployed on an automotive switch present in a central gateway; Sharing the challenges of designing and implementing an IDS for AVB/TSN; Introducing IDS concepts for other TSN protocols such as 802.1CB and 802.1Qbu. To conclude the thesis, the results of a practical testbed using automotive equipment and testing tools used to confirm the feasibility of such a system will be shown. This evaluation measures whether the IDS, as an additional processing layer in the switch, impacts on the normal switch performance.

Keywords: AVB. TSN. IDS. Security. Threat analysis.

RESUMO

Nos últimos anos, o número de aataques à sistemas automotivos aumentou tremendamente. Em geral, para combater tais ameaças, o uso de um modelo de segurança em camadas permite que uma série de mecanismos de segurança sejam aplicados para proteger os elementos de rede. Entre eles estão os Sistemas de Detecção de Intrusão (IDS), que monitoram sistemas afim de detectar atividades anômalas. Com o advento AVB/TSN (Audio Video Bridging/Time Sensitive Networking) para redes Ethernet automotivas, novas superfícies de ataques são esperadas em carros. Estas superfícieis estão relacionadas ao número de novas funcionalidades que permitem a transmissão paralela de dados sensíveis ao tempo e tráfego de melhor-esforço, dados de configurações, reserva de recursos, sincronização de relógios, e conexões entre dispositivos. Este trabalho visa preencher a lacuna deixada por trabalhos anteriores com as devidas contribuições: Apresentação dos vetores de ataques presentes nos protocolos de transporte (IEEE 1722/1733), protocolos de configuração (IEEE 1722.1) e protocolo de sincronização de relógios (IEEE 802.1AS); Discussão dos requisitos de software e hardware para o desenvolvimento de um IDS, incorporando os conceitos sugeridos pelo padrão IEEE 802.1Qci, visando um gateway automotivo como plataforma de integração; Introdução de detalhes adicionais que precisam ser levados em consideração à medida que outros protocolos ainda não finalizados ou pouco usados como o IEEE 802.1CB e o 802.1Qbu são introduzidos nos sistemas. A dissertação também apresenta resultados de um testbed utilizando hardware automotivo e ferramentas de testes serão apresentados de forma a avaliar a viabilidade do sistema IDS. Esta avaliação indica se o IDS, por ser uma camada de processamento adicional em um switch Ethernet automotivo, impacta de forma considerável no desempenho da operação normal deste switch.

Palavras-chaves: AVB. TSN. IDS. Segurança. Análise de ameaças.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

μC	microcontroller
μP	microprocessor
AAF	Address Access Format
ABS	Anti-Lock Braking System
ACF	AVTP Control Format
ACL	Access Control List
ACMP	AVDECC Connection Management Protocol
AD	Anomaly Detector
ADAS	Advanced Driver Assistance Systems
ADP	AVDECC Discovery Protocol
AECP	AVDECC Enumeration and Control Protocol
AEM	AVDECC Entity Model
AP	Authentication Protocol
API	Application Programming Interface
ATS	Asynchronous Traffic Shaping
AVB	Audio Video Bridging
AVDECC	Audio Video Discovery, Enumeration, Connection Management and Control Protocol
AVTP	Audio Video Transport Protocol
BCMA	Best Master Clock Algorithm
BLS	Burst Limiting Shaper
CAN	Controller Area Network
CBS	Credit Based Shaper
CCH	Common Control Header
CDT	Control Data Traffic
CGW	Central Gateway
CNC	Centralized Network Configuration
CRC	Cyclic Redundancy Check

CRF	Clock Reference Format
CS	Cyclic Shaper
CUC	Centralized User Configuration
CVF	Compressed Video Format
DHCP	Dynamic Host Configuration Protocol
DIDS	Distributed IDS
DoIP	Diagnostics Over IP
DoS	Denial of Service
DPI	Deep Packet Inspection
ECU	Electronic Control Unit
EMI	Electromagnetic Interference
ESC	Electronic Stability Control
FN	False Negative
FP	False Positive
FQTSS	Forward and Queuing of Time-Sensitive Streams
GPS	Global Positioning System
gPTP	Generalized Precision Time Protocol
HDCP	High-bandwidth Digital Content Protection
HIDS	Host Intrusion Detection System
HTTP	Hypertext Transfer Protocol
HW	Hardware
ICV	Integrity Check Value
IDS	Intrusion Detection System
IIA	Interface Independent Adaptation
IP	Internet Protocol
IPC	Inter-Protocol Communication
ISO	International Organization for Standards
IVN	In-Vehicle Networks
JSON	JavaScript Object Notation
LAN	Local Area Networks
LD	Link Delay

LIN	Local Interconnect Network
LLC	Logical Link Control
LVDS	Low-Voltage Differential Signaling
MAAP	MAC Address Acquisition Protocol
MAC	Media Access Control
MACsec	Media Access Control Security
MAD	Multiple Attribute Declaration
MAP	Multiple Attribute Propagation
MIB	Management Information Base
MITM	Man-In-the-Middle
MMRP	Multiple MAC Registration Protocol
MOST	Media Oriented Systems Transport
MRP	Multiple Registration Protocol
MSRP	Multiple Stream Registration Protocol
MTU	Maximum Transmission Unit
MVRP	Multiple VLAN Registration Protocol
NDP	Neighbor Discovery Protocol
NIDS	Network Intrusion Detection System
OBD	On-board Diagnostics
OEM	Original Equipment Manufacturer
OPEN	One Pair Ethernet
OSI	Open Systems Interconnection
PCAP	Packet Capture
PoC	Proof of Concept
PTP	Precision Time Protocol
QoS	Quality of Service
RFC	Request For Comments
RLM	Rate Limiting Mechanism
RTP	Real-time Transport Protocol
SAE	Society of Automotive Engineers
SEP	Security Engineering Process

SNMP	Simple Network Management Protocol
SOME/IP	Scalable Service-Oriented Middleware over IP
SRC	Sample Rate Conversion
SRP	Stream Reservation Protocol
TARA	Threat Assessment and Remediation Analysis
TAS	Time Aware Shaper
TCAM	Ternary Content-Addressable Memory
TCP	Transport Control Protocol
TLV	Type-Length-Value
TN	False Negative
TP	True Positive
TSN	Time Sensitive Networking
UDP	User Datagram Protocol
V&V	Verification and Validation
VLAN	Virtual Local Area Network
XML	Extensible Markup Language
XSS	Cross-Site Scripting
YANG	Yet Another Next Generation

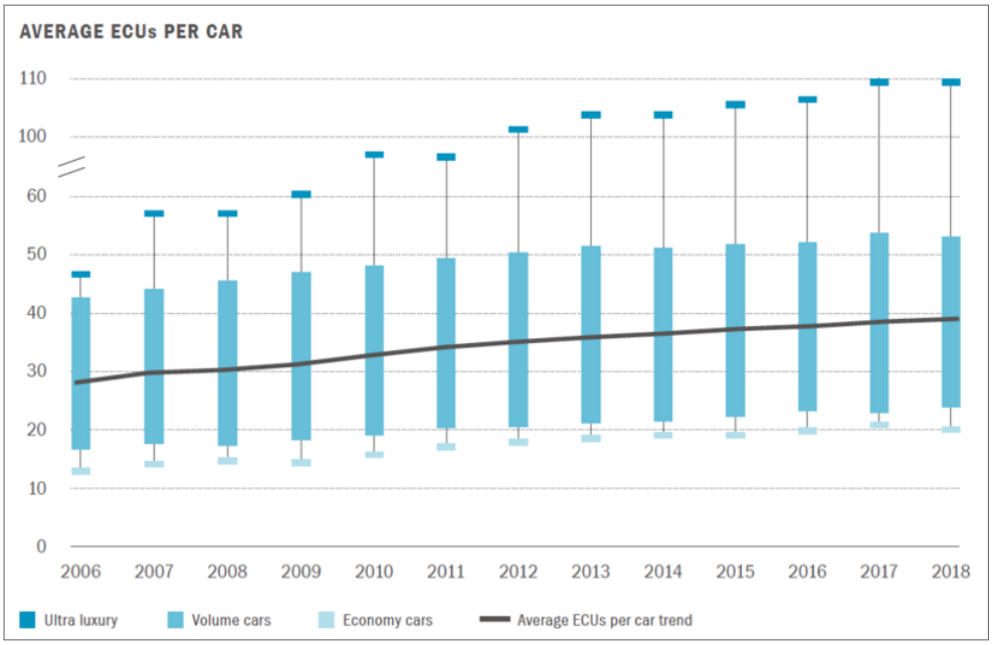
CONTENTS

1 INTRODUCTION

The progressive improvements on the performance and reliability of hardware components alongside with the possibilities brought by software technologies led to an rapid growth on the number of electronic systems inside vehicles. Such systems have been replacing on a gradual manner those that were exclusively mechanical or hydraulic. This paradigm change was consolidated by the addition of Electronic Control Unit (ECU), subsystems composed by a processing unit, like a microcontroller (μC), microprocessor (μP), among others and a set of sensors and actuators.

This growth is well represented on Figure 1, where it is possible to that the number of ECUs more than doubled in a short period of time. The composition of ECUs also changed over the years, now, they also got more powerful, containing microprocessors, CPUs, GPUs and other types of processing units.

Figure 1 – Evolution on the number of ECUs per car



Source: (BERGER, 2015)

The need of interconnecting multiple systems and sensors inside vehicles is essential to include functionalities like driver assistance and autonomous driving, increasingly indispensable to passengers comfort, safety and practicality of modern life. Many functionalities are distributed over diverse ECUs. For example, the software that controls the head-up display lights may be distributed over 8 ECUs on high end vehicles.

Furthermore, most of future feature's processing will not be performed on neighbor ECUs. For example, drive-by-wire systems, electronic control systems responsible for han-

dling vehicle's control devices, will need a closer and secure interaction with ECUs from different domains. (NAVET; SIMONOT-LION, 2013)

The communication among ECUs architecturally distributed along the car, has become more and more complex as we have added more systems on an In-Vehicle Networks (IVN). The interconnection of multiple ECUs on cars requires automotive networking protocols capable of providing not only the transmission of chunks small amounts safety-critical data but also the need for higher bandwidth availability. This requirement was the opportunity to introduce Ethernet on vehicles. The use of such a mature protocol brought initially many challenges to the automotive industry. First, the need to have a physical medium capable of supporting a more hostile environment with higher temperatures, electromagnetic interference. Efforts made by car manufacturers and semiconductor suppliers led to the creation of a new physical layer (PHY) called BroadR-Reach, later standardized as 100BASE-T1 (TUOHY et al., 2015).

Ethernet, is a well adopted technology in Local Area Networks (LAN) and other networks, offering many advantages to the automotive industry, such as: high bandwidth, required to meet the increasing demand on data transfer, up to 100 times bigger than the technologies currently used: Controller Area Network (CAN), Local Interconnect Network (LIN), Media Oriented Systems Transport (MOST) and FlexRay. Decrease of production costs is also an advantage, unlike the other technologies mentioned here, Ethernet is not specifically produced for the automotive industry's requirements and its high adoption allows a well structured standardization, improving tests, maintainability and further developments (NOLTE; HANSSON; BELLO, 2005).

Two aspects are shaping the next advances of in-vehicle networks, *security* and *real-time communication with determinism*. Security is needed because it has been demonstrated that the security of most of today's cars is flawed allowing vehicles to be targeted by malicious attackers (Koscher et al., 2010). Determinism on the other hand is a requirement demanded for building autonomous cars, where video streams sourced from multiple cameras need to be processed at the same time in a single powerful ECU.

Audio Video Bridging (AVB)Time Sensitive Networking (TSN) are a set of standards that adds a deterministic/real-time behavior to the Ethernet protocol, which in its usual version, does not offer it. Both acronyms refer to the same group of standards, just AVB being the predecessor of TSN. Nowadays, the term is basically used to refer to Audio/Video applications. This protocol set is summarized as precise timing, bounded latency, fault tolerant systems.

A major reason for the lack of security comes from the fact that, mainly due to cost restrictions, current ECUs have only enough processing power as it is required to perform its main tasks. However, to address security threats and attacks, the embedded in-vehicle nodes must be designed with defense mechanisms, which demand more capable hardware with stringent response times. Next-generation, autonomous cars will need to transmit

and process in real-time a large amount of information collected by advanced sensors and communication interfaces. Safety-critical control data must have a higher priority over any other data in the network and the delivery of safety-critical information must be guaranteed within a specific time frame.

In the automotive field, it is critical for the transmission to be secured against malicious attackers since the data transferred might put the safety of drivers at risk. In general, for enterprise networks, the use of a layered model of security is advised and many security mechanisms can be applied to protect devices. The problem is that in the automotive industry, almost in every case, there is no space for resource intensive mechanisms. One example of this is the use of Media Access Control Security (MACsec), (IEEE..., 2018), a protocol that provides secure communication for Ethernet traffic but has a big overhead brought by the need of performing cryptographic operations on packets at every hop in order to process the Virtual Local Area Network (VLAN) on the frame. Security architects face big demands when designing the security mechanisms for in-vehicle networks: use secure protocols and/or less intensive monitoring solutions?

Complex infrastructures of keys and certificates might not be the best solution for every use case due to added costs and demanding computation power, therefore, monitoring also is also an important alternative. Again this is specially the case for MACsec, which is the possible solution for securing Ethernet. MACsec relies on pairwise symmetric keys, one for each peer-to-peer communication. Therefore, decryption and encryption occur at every hop.

One type of such monitoring tools are the Intrusion Detection System (IDS). IDS are mainly categorized in two types. Host Intrusion Detection System (HIDS) and Network Intrusion Detection System (NIDS). The first aims to monitor the resources of a host such as memory, processes, files, etc. and the second aims at monitoring network traffic, both in order to detect intrusions on a system. A combination of both is also possible and when they actively act on the host/network they are called intrusion detection and prevention systems. Examples of actions can be killing processes or dropping packets.

1.1 CONTRIBUTIONS

The main contributions of thesis master thesis are:

- The overview of the current status of TSN standardization process;
- The overview of threat scenarios, which are based on three pillars:
  - Attack vectors found on time synchronization
    - There are many publications describing threats on Precision Time Protocol (PTP), but those make assumptions that are not valid for the Generalized

Precision Time Protocol (gPTP) (IEEE 802.1AS). Many of them also consider older versions of PTP (IEEE 1588-2002 v1). This work considers IEEE 1588-2008 v2 and also makes some regards to P1588 v3 which is still a work in progress.

  - Attack vectors found on the transport protocols such as the IEEE 1722 and IEEE 1733

  - Attack vectors involving management data such as from the protocol IEEE 1722.1

- Lessons learned on implementing an IDS for automotive systems that are AVB/TSN capable

- Proof-of-Concept using an automotive switch as the IDS host and an evaluation done with automotive testing tools conclude the thesis aiming to answer the following questions:

  - Does the IDS compromise the expected end-to-end latencies for deterministic networks?
  - Does the IDS compromise the expected timing accuracy between the master and slaves when clocks are synchronized using gPTP?
  - Considering that the chosen target board has a limited amount of processing power, how should the network look like and how should the traffic's periodicity be like?

Security for AVB/TSN is a new and not well explored topic. The decision was to make this thesis rather broad in the sense of going through all of the standards then going deep into one or some of the standards. Probably newer and uncovered points of exploitation are going to be found as AVB/TSN will become interesting to more researchers. Therefore, the goal for this thesis is to give a wide overview of all of the protocols that make AVB/TSN so that in the future other researchers can chose which one they should focus and dig deeper.

1.2 THESIS' STRUCTURE

This thesis is organized as follow. Chapter 2 introduces the set of electronic parts present in vehicles as well as in-vehicle networks connecting them. By the end of Chapter 2, Automotive Ethernet will have been explained, describing how IVN have evolved until the current days. Chapter 3 shows why improvements to the ordinary Ethernet protocol were needed and then the next Ethernet generation will be introduced. Chapter 4 raises awareness to security on automotive networks, and describes counter-measures. Still related to

security, Chapter 5 summarizes exploitation vectors derived from the new features introduced on the new Ethernet generation. Chapter 6 proposes a defense mechanism to detect the vulnerabilities previously explained. Chapter 7 shows the results of a practical evaluation of the proposed security mechanism. Chapter 8 concludes this thesis by pointing some recommendations for developing similar monitoring mechanisms and future work expected for this thesis.

2 IN-VEHICLE NETWORKS

ECUs has grown to be a considerable part of today's vehicles. Gradually, functions that were exclusively mechanical or hydraulic started being replaced by these electronic components. Newer functionalities that would not be feasible without automotive embedded systems, were also soon developed. As examples of simple functionalities we have the controlling of the air-fuel ratio inside the engine. This feature already existed as mechanical/pneumatic means but it was replaced by electronic components. On the other hand we have more complex functionalities like automated parking or lane keep assistant, features that would not be possible by non-electronic means.

Interconnecting multiple ECUs became necessary in order to create these complex functionalities like Anti-Lock Braking System (ABS), Electronic Stability Control (ESC) and Advanced Driver Assistance Systems (ADAS). Increasingly indispensable to passengers comfort, safety and practicality of modern life. Many functionalities are distributed over diverse ECUs. For example, the software that controls the head-up display lights may be distributed over 8 ECUs in high end vehicles. Furthermore, most of future feature's processing will not be performed on neighbor ECUs. For example, drive-by-wire systems, electronic control systems responsible for handling vehicle's control devices, will need a closer and secure interaction with ECUs from different domains. (NAVET; SIMONOT-LION, 2008)

The goal of this chapter is not to deeply explain all the protocols and its inner headers and bits, but rather introduce the evolution and requirements of in-vehicle networks. This discussion will follow a chronological sequence that will lead to the current status of IVN.

2.1 BUS SYSTEMS

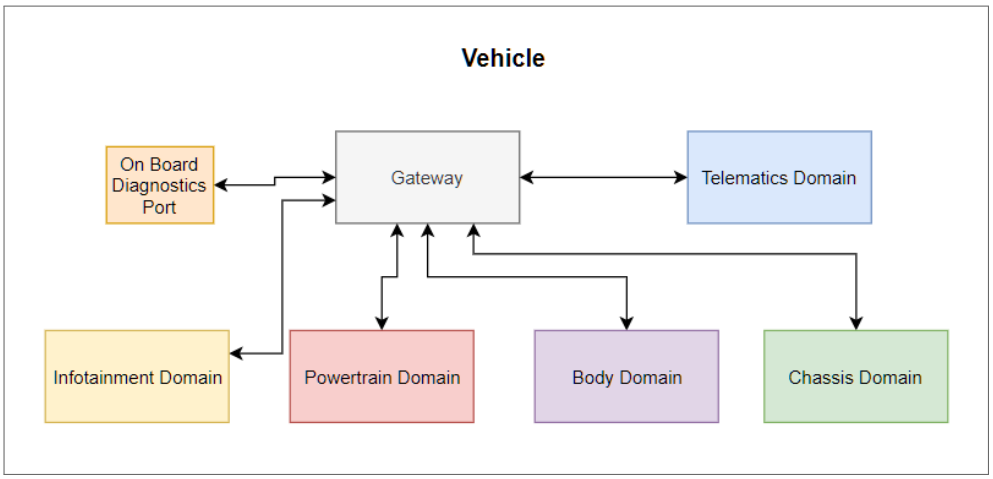
In the right beginning of interconnecting ECUs on cars, Point-to-Point links were used to exchange information. This strategy was quickly proven to be insufficient with the growth in the number of electronics in vehicles, it became inefficient. Historically, for each new task that was supposed to be electronically handled, a new ECU would be used and integrated to the network. In order to connect a number N of ECUs, N^2 connections would be necessary, with point-to-point links, what implies added weight and costs for manufacturers. The rise of automotive networks was marked by the establishment and evolution of the so called bus systems. Bus systems enabled the possibility of connecting multiple ECUs to a common bus, without needing to connect them with point-to-point links.

In the beginning of the 1980s, Robert Bosch GmbH developed the first of such bus network. The CAN is still the most widely used network in automotive systems. As

different functions started being introduced in cars, they were categorized into domains. With their own needs of Quality of Service (QoS).

Today, ECUs and applications belong to one or more of the following domains: Body, Powertrain, Chassis, Infotainment, and Telematics. As previously mentioned, each of these domains has different requirements, such as latency, jitter, bandwidth, reliability for transmission errors, network topology, among others. Different needs gave rise to different in-vehicle networks such as, the LIN, FlexRay, and MOST. These different requirements are divided into classes of traffic. Table 1 presents a summary of each class together with an in-vehicle network that can be well-suited for carrying traffic for this class.(Bosch, 2018)

Figure 2 – Vehicle Domains



The Powertrain domain is responsible for components that brings movement to cars like: transmission, engine, wheels, among others. The Chassis domain, responsible for powertrain's supporting components, such as brakes, steering and suspension. Body domain, which comprises lighting, window control, heating and air-conditioning. Telematics, which is the system responsible for the car communication with the outside world, including Global Positioning System (GPS), radio, internet access, and others. Infotainment is the domain responsible for entertainment systems like audio and video and also for interfacing with the telematics domain for displaying driver information. Nowadays, there seems to be a trend of consolidating Infotainment and Telematics into one single domain.

Figure 2 shows how these different domains could be disposed inside cars. All of the domains are normally connected to a central gateway. Each domain's box represent an interconnected set of ECUs and these connections are usually different protocols. Having the gateway as a central ECU means that it is able to translate messages coming from different protocols and make the domains interact.

Table 1 basically shows how different IVN technologies differ from each other in terms of transfer rates. However, the choice of one protocol to a domain/solution depends on many factors: Maximum number of ECUs on the network, how many speed grades are supported using the same protocol stack but different physical layers, bandwidth efficiency,

Table 1 – Traffic Classes for automotive data

Class A	
Transfer rates	Low data rates (Up to 10kbps)
Applications	Actuator and sensor network
Representative protocol	LIN
Class B	
Transfer rates	Average data rates (up to 125 kbps)
Applications	Complex mechanisms for error handliing, control unit networking in confort functions
Representative protocol	Low speed CAN
Class C	
Transfer rates	Medium data rates (up to 1 Mbps)
Applications	Soft real-time requirements, control unit networking in the drive and running gear functions
Representative protocol	High speed CAN
Class C+	
Transfer rates	High data rates (up to 1 Mbps)
Applications	Soft real-time requirements, control unit networking in the drive and running gear functions
Representative protocol	FlexRay
Class D	
Transfer rates	Extremely high data rates (bigger than 10 Mbps)
Applications	control unit networking in the telematics and multimedia functions
Representative protocol	MOST

Adapted from: (Bosch, 2018)

topologies possible, addressability, security, extendability, and many others. The legacy protocols mentioned so far were really limited in many of these aspects, giving space to other technologies.

2.2 AUTOMOTIVE ETHERNET

The complexity of the tasks handled by ECUs brought new challenges to automotive system architects. One of these challenges was the need for more bandwidth, i.e. a way to transmit bigger chunks of data. In this chapter, we first give a overview on the Ethernet protocol and then highlight a few milestones of the Automotive Ethernet growth, what were these and how Ethernet got its place inside cars. For a more detailed description of the historical aspects, please refer to (MATHEUS; KöNIGSEDER, 2017).

2.2.1 Ethernet

Developed in the early 1970s by Xerox in Palo Alto, Ethernet has become one of the most, if not the most, used protocol in LAN and other networks. This market dominance lays on several aspects like reduced production costs, small range of speeds from 10Mb/s up to 100 Gb/s, flexibility of topologies, seamless interoperability with other wired and wireless technologies and also the plug-and-play characteristic, requiring almost no configuration to get it running. (Sommer et al., 2010)

Ethernet was defined in a series of IEEE 802.3 standards which defined the physical and link layer of the International Organization for Standards (ISO)/Open Systems Interconnection (OSI) reference model. For the physical layer, the IEEE 802.3 work group standardize different technologies, like: 10BASE-T and 100BASE-TX. The first part of the acronym refers to the speed of the standard and the last part to the cabling specification. The link layer has two main responsibilities, they are:

- Media Access Control (MAC): Since Ethernet is based on a shared medium, different network adapters might want to send packets at the same time. Therefore, in order to avoid collisions, devices need to listen to the medium;
- Logical Link Control (LLC): Responsible for creating and managing logical links between devices in order to provide the seamless services, this is achieved by:
  - Data Encapsulation: Ethernet defines a frame format used by higher layers to use its services;
  - Error handling: Ethernet provides an unreliable data delivery service, therefore, an error handling mechanism, the Cyclic Redundancy Check (CRC), is available to detect transmission failures;
  - Addressing: Each network adapter on the network has an identification, which other adapters use to itself as the sender and specify the destination of a message.

2.2.2 First Generation

The first signal that showed the demand for more bandwidth was seen on software diagnostics. The On-board Diagnostics (OBD) interface, depicted in Figure 2, is an external port that allows the vehicle owner or repair technician attach diagnostic tools to communicate with ECUs. This port is normally connected to the Central Gateway (CGW). The CGW is responsible for interacting with other ECUs to check their current health status and provide this information back to the repair shop. After the diagnostic of some functionality that is not working properly, the firmware of one or more ECUs might need to be updated. Through the years, the number of tasks performed by ECUs grew considerably. This growth resulted directly in bigger firmware size. As a consequence, software updates started becoming too long to be conducted by CAN.

That is when Ethernet took place. The well-known and mature Ethernet protocol was first applied for diagnostics in cars by BMW in 2007. During test phase, it became known that Ethernet could not support the rough environment of the inner vehicle. High temperatures and Electromagnetic Interference (EMI) were disturbing the communication. As well as the proper Ethernet was interfering with radio signals. But since the cars would

be stopped in repair shops or at Original Equipment Manufacturer (OEM) reseller establishments, Ethernet was still chosen. At this time, the Ethernet 100BASE-TX was used, reducing the total diagnostics time from 10 hours (when it was performed over CAN bus) to 20 minutes (TUOHY et al., 2015). The higher bandwidth offered by Ethernet not only shortened the diagnostic time but also cheapened the production and maintenance costs.

2.2.3 Second Generation

Upon experiencing the benefits of using Ethernet, new use cases started taking place. This was the case for entertainment and ADAS, which deal with high resolution media data, also needing high bandwidth availability. For these types of system, a protocol named Low-Voltage Differential Signaling (LVDS) was initially used. The protocol was an attractive solution due to its high bandwidth availability, suitable for automotive camera data. Despite of being one of the few protocols in vehicles that is not automotive specific, which is considered to be a positive aspect, LVDS has an inefficient use of cables, which increase production costs.

Another use case was the transmission of geolocation data from the Head Unit to Rear Seat Entertainment systems. The problem now is that these features are going to be used with cars in movement. Therefore, an EMI resistant physical medium was required and thus pushing the efforts that lead us to the milestone that marks the third generation. (MATHEUS; KöNIGSEDER, 2017)

2.2.4 Third Generation

Thinking on the longer use for Ethernet, car manufacturers and semiconductor suppliers joined forces in order to develop a new physical medium capable of supporting this hostile environment. Proposed by Broadcom, the BroadR-Reach technology was initially proposed as a cheaper medium for Asian markets. Aware of the benefits and capabilities of such technology, BMW quickly backed the idea. BroadR-Reach delivers 100Mbit/s over an unshielded single twisted pair cable. This means less cabling and reduced costs for manufacturers when compared to the two shielded twisted pairs, used in 100BASE-TX.

Soon, other interested partners formed a collaboration group called One Pair Ethernet (OPEN) Alliance. Being a proprietary solution meant that every OEM was dependent on BMW and Broadcom. Under pressure from automotive OEMs and suppliers such as Bosch, the BroadR-Reach technology was standardized in 2015 by the IEEE as 100BASE-T1. (MATHEUS; KöNIGSEDER, 2017)

As the maturity of Ethernet in automotive grows, we see a rapidly expansion of its presence inside vehicles. For the next generations, it is expected that Ethernet will be used more than just being an additional protocol. In this sense, Ethernet will act as a backbone for the different domains, carrying traffic generated from sub-domains where CAN, FlexRay and other protocols are used.

3 DETERMINISTIC ETHERNET

Consumer needs for entertainment and ADAS systems led to challenges beyond bandwidth. For entertainment, the difficulty was the distribution of audio and video to multiple speakers and screens inside the vehicle. For ADAS, the challenge was the transmission of cameras' and sensors' streams to the ECUs that analyze these streams. Ethernet could only partially meet this distribution requirement because, beyond bandwidth availability, it is also needed to bring streams from multiple sources to the endpoint(s) precisely at the same time so they can be merged before processed.

The problem is that Ethernet, in its original form, does not provide determinism. Therefore, a set of solutions were made available to cover this need. Among them are the protocols AFDX, EtherCAT, PROFINET, TTEthernet and others (Decotignie, 2009). The main problem is that those technologies were developed for specific domains. Those proprietary solutions are basically based on three main pillars: time synchronization, real-time scheduling and stream reservation and configuration, they are:

- Time synchronization to support low-jitter and accurate synchronization of multiple streams;
- Resource reservation to ensure applications can send or receive time-sensitive streams;
- Queuing and forwarding specifications that ensure streams will be transmitted according to the reserved resources and meeting the needed latency.

A new extension was made available to enable the deterministic behavior on Ethernet: the Audio Video Bridging (AVB) standard (IEEE 802.1BA, 2011).

In a traditional Ethernet network it is acceptable for a switch to drop frames; even if VLANs are used to add priority to packets, the switch will still have to process the entire outgoing buffer before getting to the higher priority packets. Consequently, it is not possible to have frames sent at a specific time and be sure that they will arrive. These are the challenges AVB and TSN try to solve. AVB can be summarized as precise timing: bounded jitter and latency, and guaranteed bandwidth on a network. TSN is backward compatible with AVB but adds newer features regarding loss rates, fault tolerance, redundancy and further improvements in time sensitive scheduling and latency control. It is expected in TSN networks that both time aware and best-effort traffic, as in the usual Ethernet, will co-exist (BELLO, 2011). This makes Ethernet more robust and capable of being used in more complex use cases such as a backbone for the different automotive domains (NAVET; SIMONOT-LION, 2008).

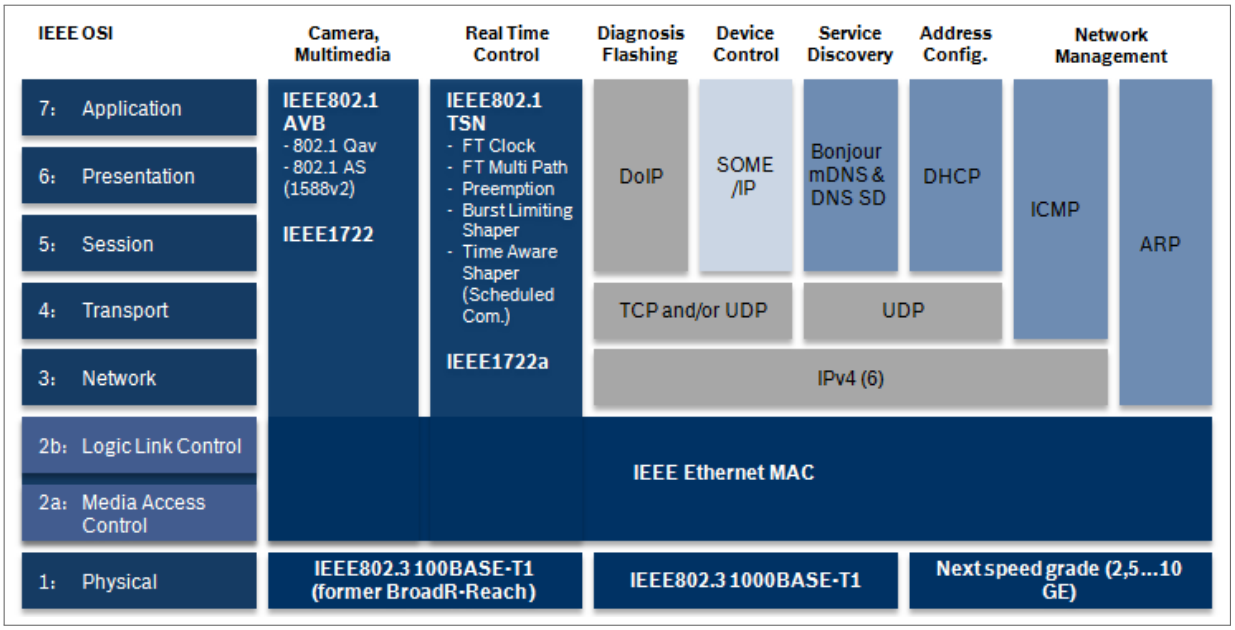
Despite being treated in different thesis sections AVB and TSN refer to the same concept. The names are only different due to a distinct time where they were released.

Before jumping into the inner specifics of AVB/TSN, it is worth going through a quick overview on the IEEE 802.1Q standard. As it will be clarified in the next two sub-sections, TSN is a generalization of AVB. Therefore, the sections in this chapter take a timeline approach where it should be easier to understand the evolution of the 802.1 standards.

From a historical point of view, IEEE first introduced the IEEE 802.1D standard, which is composed by all the basic switching mechanisms. This first standard was succeeded by the IEEE 802.1Q standard, commonly known for the VLAN specifications, which brought the concepts of streams and QoS. As amendments to the IEEE 802.1Q standard, the Stream Reservation Protocol and the Forwarding and Queuing were introduced. The IEEE 802.1BA standard was then published, using all the previously mentioned features. Finally, the second generation of AVB was renamed TSN by the addition of an even bigger set of building blocks which will be explained in the next sections.

It is useful to make a parallel between deterministic Ethernet protocols and other common protocols like Internet Protocol (IP), Transport Control Protocol (TCP), Hypertext Transfer Protocol (HTTP). AVB/TSN does not follow the usual layered approach where there is one protocol for each layer with specific responsibilities. From Figure 3 it is possible to see that AVB/TSN works almost entirely up to layer 2. Therefore, there is no network protocol like IP or transport protocol like TCP/User Datagram Protocol (UDP) involved in this. Most of the link layer, network, transport, protocols until application layer are all comprised until layer 2.

Figure 3 – AVB/TSN in conformance with the ISO/OSI layers



Source: Robert Bosch GmbH

3.1 IEEE 802.1Q

When dealing with layer two in the OSI model, Ethernet is the most common protocol. Ethernet, by definition, is a broadcast link. In this approach, multiple sending and receiving nodes can be connected to the same channel. For this reason, we need a Medium Access Control protocol to coordinate transmissions among nodes. This aspect has implications on the topology of the network, in the sense that normally it is possible to find domains having its own switched LAN and connected to other domains via a switched hierarchy. This switched structure has three drawbacks. The first is the lack of isolation between different domains, which has a negative impact on the LAN performance as well as in the security and privacy. It may not always be desired that traffic form one domain be seen by another. The second drawback is the inefficient use of switches. For any number of levels in the hierarchy, it would be necessary to have the same amount of switches, with enough ports to accommodate the number of nodes. The third and last drawback is the problem of moving nodes between domains or having nodes belonging to multiple domains. In this case, physical cabling would have to be reorganized and duplicated, respectively.

To tackle this issue, IEEE released the concept of VLANs. The IEEE 802.1Q standard proposes that switches be capable of having to have multiple virtual/logical LANs over a single physical LAN. This means that nodes within a VLAN communicate with each other as if they were connected individually with a switch, but in fact they share the same physical switch with nodes from other VLANs.

With the use of VLANs, all the aforementioned drawbacks can be overcome. Isolation is brought by the use of different VLANs for each port. The switch is more efficiently used, since the same equipment can be used for different workgroups. Finally, to move the group a device belong to, it is only necessary to change the VLAN of the port the device is connected. In order for the workgroups to have means to communicate with each other, it is necessary to make this path using IP addresses. Normally, devices that support VLANs have also a router inside it. A similar problem is to have the same VLAN present in two different switches. The solution to this problem is the so called VLAN trunking. This means that one port in each switch belongs to all VLANs.

To be identified belonging a particular VLAN, each Ethernet frame gets extended with a 802.1Q tag. The format of the Ethernet frame containing this tag can be seen in Figure **??**. The length and description of each field in this tag can be found in Table 2.

Figure 4 – VLAN Tag

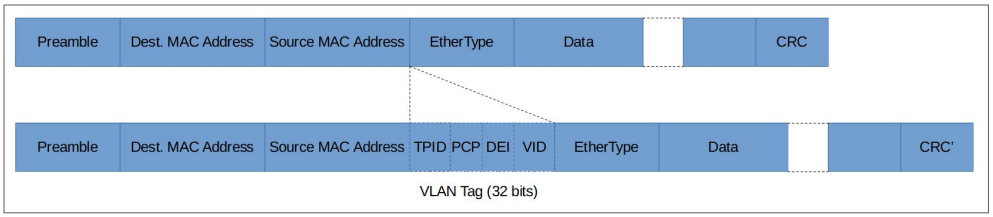


Table 2 – VLAN Fields

Name	Length	Description
Tag Protocol Identifier (TPID)	16 bits	Has a value of 0x8100 in order to identify a frame as tagged. This has the same length and position of the EtherType field on the Ethernet frame in order to differentiate tagged and untagged frames
Priority Code Point (PCP)	3 bits	Indicates the priority level of a frame: 000: best effort 001: background 010: excellent effort 011: critical application 100: video 101: voice 110: inter-network control 111: network control
Drop Eligible Indicator (DEI)	1 bit	Indicates if a packet can be dropped in case of congestion
VLAN Identifier (VID)	12 bits	Specifies an ID for the VLAN. The values 0x000 and 0xFFF are reserved and must not be used. All other values may be used as VLAN identifiers, thus allowing up to 4,094 VLANs.

In the context of AVB and TSN explained in the next sections, the IEEE 802.1Q standard plays an important role. This role regards traffic prioritization, classification and specification. Regarding traffic prioritization, this can be easily seen on the description for the PCP field in Table 2. Whereas frames belonging to different streams can be treated with different QoS. As it will be seen in the AVB/TSN sections, many standards are actually amendments to the IEEE 802.1Q standard. It is possible to identify the standards based on this by seeing two small letters following the "Q" as in IEEE 802.1 Q**at**.

Traffic classification means how priorities are mapped into classes. Each class has a specific traffic specification. Traffic specification is how the bandwidth for each traffic class should be characterized. This is described in terms of the Maximum Transmission Unit (MTU), Minimal Frame Interval and expected end-to-end latency over a maximum amount of hops.

Most part of the set of supporting protocols used by AVB/TSN are an improvement made on the IEEE 802.1Q standard that introduced the concept of VLANs and QoS. VLANs have multiple purposes such as defining a logical sub-network defined via the switches but one of the main usages of VLANs in AVB/TSN networks is to assign priority to frames and classify frames into different traffic classes. An AVB network normally has three traffic classes: class A for audio, B for video and BE for the default Ethernet's Best-Effort. For TSN, a new traffic class was introduced for Control Data Traffic (CDT). Different traffic classes have different traffic specifications implying different MTU, minimum frame interval and expected end-to-end latency. An overview of all traffic classes and specifications is found on Table 3

Table 3 – Traffic Specification for AVB and TSN

Class	**MTU**		**Min. Frame Interval**		**Exp. E2E Latency**	
	AVB	TSN	AVB	TSN	AVB	TSN
CDT	-	128 bytes	-	500 µs	-	100 ms (5 hops)
A	1500 bytes	256 bytes	125 µs	125 µs	2 ms 7 hops	2 ms 7 hops
B	1500 bytes	256 bytes	125 µs	250 µs	50 ms 7 hops	50 ms 7 hops
BE	1500 bytes	256 bytes	-		-	

The MTU, also called Maximum Frame Size, is the maximum frame size that a Talker (stream producer) will produce as part of a stream for a certain class. It does not include headers, only payload. The Minimum Frame Interval is the number of frames that the Talker will produce per class measurement interval.

3.2 AUDIO VIDEO BRIDGING (AVB)

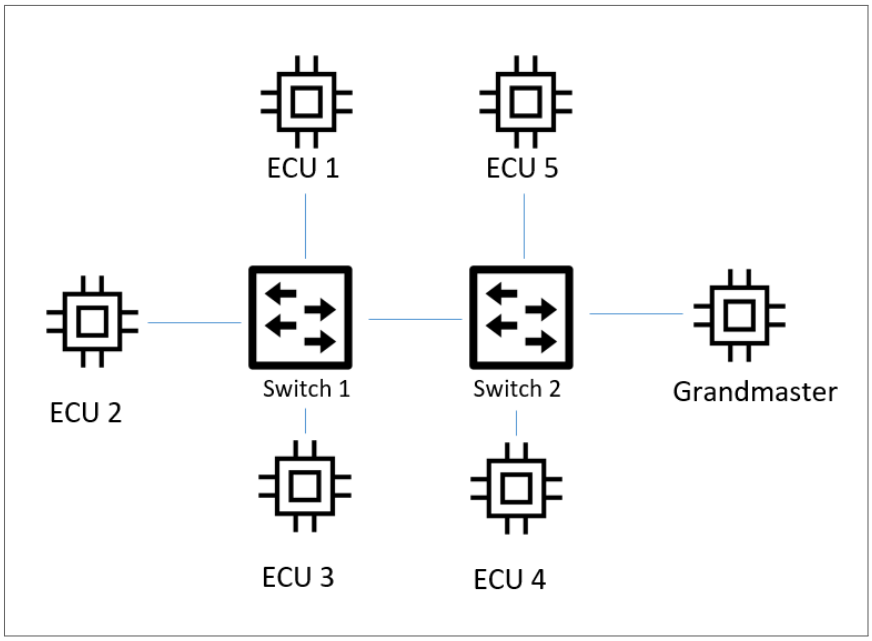
AVB started in 2005 and is a set of four IEEE standards that enable the transmission of media streams over Ethernet. They are: IEEE 802.1AS, a protocol that provides time synchronization using the precision clock synchronization hardware implementation of IEEE 1588, IEEE 802.1Qat, a protocol for stream reservation and IEEE 802.1Qav for forwarding and queuing. Finally IEEE 802.1BA, which is a profile that selects features, options, configurations, protocols and procedures of bridges, stations in LANs that are necessary to build networks that are capable of transporting time-sensitive audio and/or video data stream (Zinner et al., 2011).

An AVB network consists of end nodes that can act as either a talker, listener or both. That is, they act as a source or sink, respectively, of media streams on the network. The second kind of devices in the network are switches, also called bridges. Beyond offering normal switching functionality, they offer additional capabilities that allow time-sensitive data to be transmitted in the network. A depiction of these elements can be seen in Figure 5.

3.2.1 IEEE 802.1AS

The clocks in two different nodes in a network do not necessarily run at the precise same frequency. Therefore, they do not agree on a common time and they cannot perform actions at the same time. To fill the time synchronization needed on deterministic networks, AVB relies on the IEEE 802.1AS (IEEE 802.1AS, 2011). Also called gPTP, it is a specific profile of the IEEE 1588, the PTP. This is necessary, for example, when two end-nodes

Figure 5 – AVB Network



need to take actions at the same time, such as two listeners who are supposed to play the same video stream.

As just mentioned, the IEEE 802.1AS is a standard the specifies a profile using the PTP protocol. PTP alone defines only the protocol and mechanisms to transport time. PTP as in 1588 is specified for running over multiple transport protocols like UDP or IP. That is where the 802.1AS comes into play, defining the specifics of PTP for 802.1 as well as performance goals. Among the performance goals are timestamping accuracy in the range of 300ns, maximum number of nodes in the network, maximum distance between nodes, etc. It is expected, for example, that the synchronization accuracy to be less or equal than 1 μs for a path of up to seven hops. Over the years, multiple other profiles have been proposed like in IEEE PC37.238 (IEEE PC37.238, 2017) or ITU G.8265.1 (ITU G.8265.1, 2014). Another difference is that gPTP expects all switches between time-aware devices to be gPTP compliant, while PTP does not.

The process of synchronizing clocks within a network consists of two main steps. First, the Grandmaster Selection with the Best Master Clock Algorithm (BCMA) and subsequently, the distribution of the synchronization of the selected clock with gPTP. The first part in this process is independent of the physical medium, where the second part depends on which medium the messages are going to be exchanged. It is important to highlight that during the uptime of the network (when the network is active) these two activities continuously occur on the network. Normally, BCMA runs at each 125 ms. The first step is optional in many small and static networks, such as in automotive networks. In these networks, the grandmaster clock can be defined statically at the design phase.

3.2.2 Message Formats

In order to perform the two gPTP phases described previously, a special set of messages formats is exchanged among nodes. A frame carrying gPTP messages does not contain a VLAN tag. The destination address of all messages is the reserved multicast address 01:80:C2:00:00:0E and the EtherType is 0x88F7. All messages have a header, which is the IEEE 1588 header, a body and possibly one or more Type-Length-Value (TLV) options.

3.2.3 Media Independent

There are two message types that do not depend on the physical medium, they are:

- **Announce** messages carry information from a priority vector, which are information from the sending time-aware system. These parameters are used by nodes in order to identify which node is best fit for the role of grandmaster. The priority vector is a 14 byte long set of properties that indicate the clock's class, accuracy, priority and others. The 802.1AS standard also defines comparison rules for identifying the best clock master.
- **Signaling** messages are used to constantly update/signal all the nodes in the network about the intervals at which the time synchronization, delay calculations and announcements should happen.

3.3 MEDIA DEPENDENT

As stated in the introduction of this section, the distribution of the grandmaster time depends on which medium the messages are going to be transmitted. The reason is mainly because of the calculations required for measuring the propagation delay of the link in question. For this reason, it is possible to say that the time distribution is composed of the time propagation and the delay calculation. We focus only on the messages exchanged through the IEEE 802.3, Ethernet.

The propagation of the master time in the network starts from the grandmaster to its directly attached nodes. The synchronization time is then propagated by other bridges attached to the grandmaster, to its nodes, as traversing a tree. Each port on a bridge might either be characterized as master or slave and time distribution always flows from master to slave ports.

The grandmaster sends first a Sync message containing its current time. After some propagation delay time, the message is received by its first neighbor. The grandmaster then sends again a second message called Follow-Up message, which contains three information fields: The time when the Grandmaster sent the first Sync message, the delay between the Sync message of the grandmaster and Sync message of the considered device and the rate ratio. The third item, the rate ratio, is the ratio of the master clock speeds and the clock

speed of the current device. If the Sync master is the grandmaster itself, the sync time is 0 and the ratio is 1. After some processing time, also called residence time, the neighbor sends out a Sync message, to its neighbor, therefore going one level deep in the network. As before, a Follow_Up message follows the Sync message. When the sending device is not the grandmaster itself, the following question has to be answered before propagating the time deeper: What time in grandmaster time was the Sync message sent? In order to calculate this time, we add to the time of the Sync message the link delay and the processing time. Link delay is calculated regularly by two connected network ports as shown on Figure 7. The link delay and residence time must be indicated in grandmaster time, therefore, they are adjusted using the rate ratio value. This process is illustrated on Figure 8.

The link delay measurement depends on the medium in use. For Ethernet, a two step peer-to-peer delay calculation is used, for this procedure, the following messages are used: Pdelay_Req, Pdelay_Resp and Pdelay, Resp_Follow_Up. Image 7 illustrates the link delay discovery between two peers. The discovery starts with the initiator sending a Pdelay_Req at time t_1. The other peer will receive this message at time t_2 and responds with a Pdelay_Resp at time t_3 containing the time at which the request was received t_2. When the response is received, at time t_4, the responder will generate another message called Pdelay_Resp_Follow_Up, carrying the time t_3. Upon receiving all this messages, the initiator can compute the Link Delay (LD) as in the following equation:

$$LD = \frac{(t_4 - t_1) - (t_3 - t_2)}{2} \tag{3.1}$$

The neighborRateRatio is calculated by using the transmission and reception timestamps of either two successive Sync or Pdelay_Resp messages. The rateRatio is then calculated as the product of the most currently received rateRatio multiplied by the neighborRateRatio.

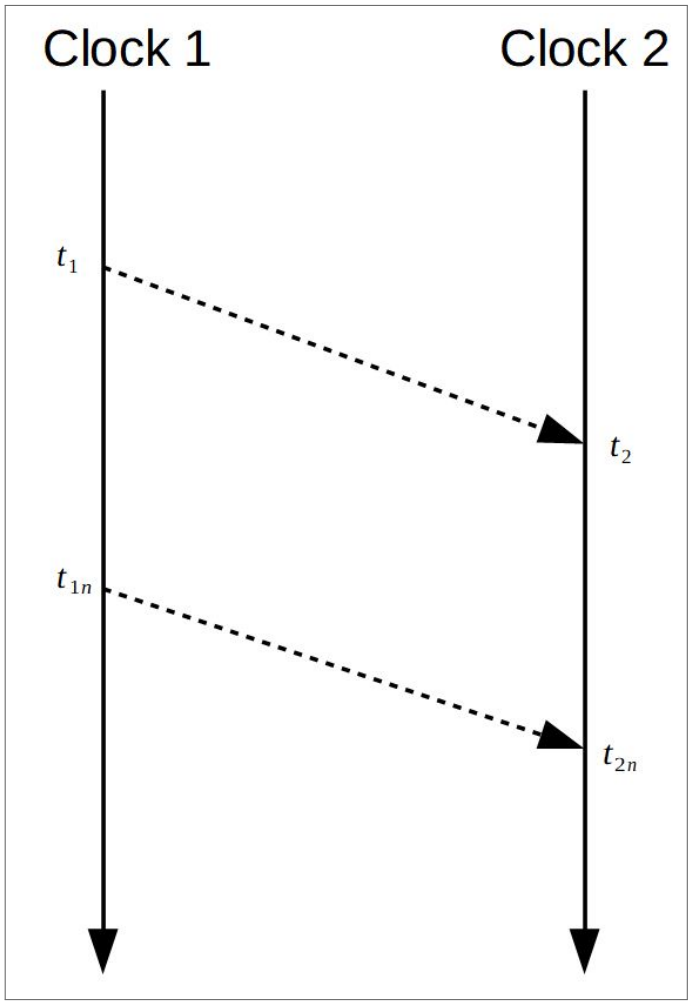
$$neighborRateRatio = \frac{(t_{(i-1)^1} - t_{i^1})}{(t_{(i-1))^2} - t_{i^2}} \tag{3.2}$$

$$rateRatio = rateRatio_{i-1} * neighborRateRatio_i \tag{3.3}$$

When considering PTP enabled devices, every device and port has a responsibility. These characteristics this can be:

- Master Port: Any port, P, of the time-aware system that is closer to the root than any other port of the gPTP communication path connected to P;
- Slave Port: The one port of the time-aware system that is closest to the root time-aware system. If the root is grandmaster-capable, the slave port is also closest to the grandmaster. The time-aware system does not transmit Sync or Announce messages on the slave port.

Figure 6 – Neighbor Rate Ratio



- Disabled Port: A port that does not participate in the synchronization process;
- Passive Port: A port whose role is not one of the previous ones;
- Ordinary clock: has only one PTP port and can be grandmaster-capable or slave-only;
- Boundary clock: has multiple PTP ports and can synchronize different network segments. These ports might be master or slaves;
- Transparent clocks: do not participate on the master/slave hierarchy;

  End-to-end transparent clock: residence times accumulated in the correctionField of Sync, Follow_Up, Delay_Req, or Delay_Resp messages;

  Peer-to-peer transparent clock: residence times plus egress-link delays accumulated in the correction field of Sync or Follow_Up messages;
- Management Node: A node that configures via a networking configuration protocol the gPTP settings of other nodes.

Figure 7 – Peer Delay Calculation

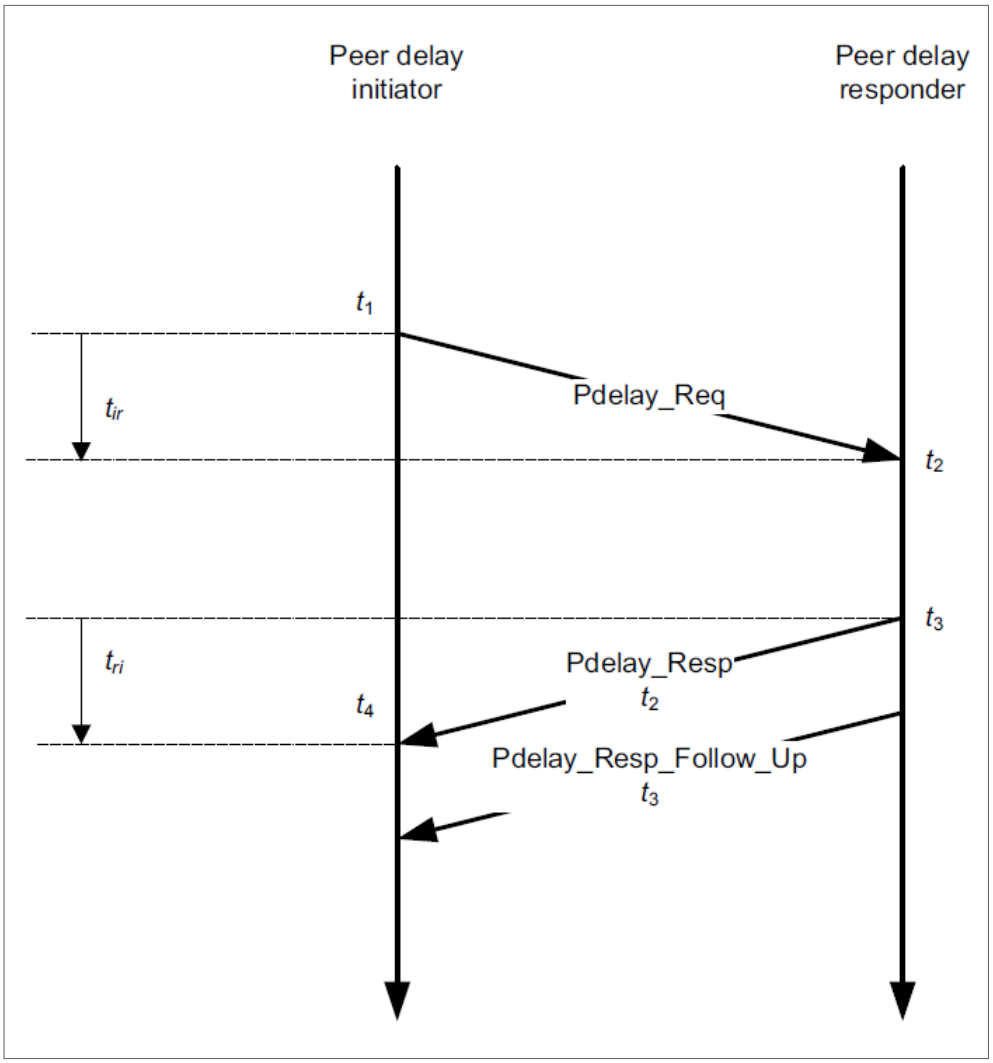
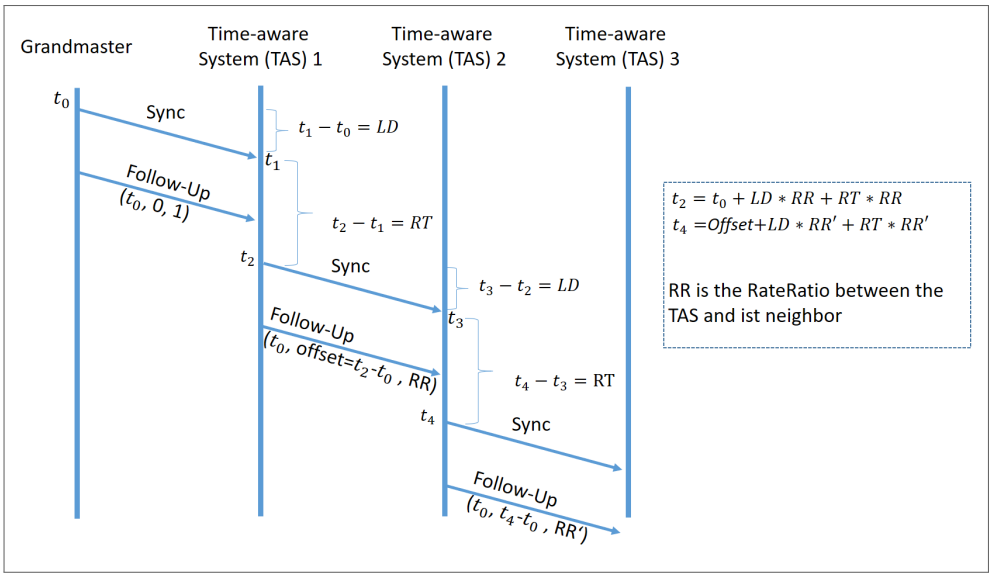


Figure 8 – Time Distribution for Synchronization



A time-aware end station corresponds to an IEEE 1588 ordinary clock, and a time-aware Bridge is a type of IEEE 1588 boundary clock. These roles of master/slave are also important when considering the communication paths of the network.

- Slaves must always be connected to master ports;
- No two masters can be connected;
- Masters might be connected to passive ports;
- Disabled ports are always connected to disabled ports.

They also infer on restrictions for the existence of messages on a given communication path. A boundary clock always delimits the end of a communication path. For example if a boundary clock sits in the middle of a grandmaster and a slave, the slave will never see messages directly from the grandmaster.

Participants in a network typically belong to a domain. Domains are portions of the network where all the clocks are synchronized to one specific Grandmaster. It is possible to have two domains, having two grandmasters and different nodes synced to one of these.

3.4 IEEE 802.1QAT

The (IEEE 802.1Qat, 2010) standard, commonly known as the Stream Reservation Protocol (SRP) defines how bandwidth is allocated in the network on the way from a talker to its listeners. These reservations help that no packets will be lost due to network congestion. SRP deals with the management of resource reservation for data streams requiring guaranteed QoS. It relies on three sub-protocols, the Multiple Stream Registration Protocol (MSRP), Multiple VLAN Registration Protocol (MVRP) and Multiple MAC Registration Protocol (MMRP). Those in turn are based on the Multiple Registration Protocol (MRP), an already existing standard, named IEEE 802.1Qak. They allow endpoints to register their willingness to talk or listen to specific streams and propagates this will to the network.

These resource requirements are propagated in the network in the format of attributes. By managing these attributes is how bandwidth is allocated or deallocated. Basically, one participant, talker or listener, must change (create or remove) a declaration of an attribute in order to cause the (creation or removal) registration of this attribute in other participants.

The management of attributes within a participant takes place in a component called Multiple Attribute Declaration (MAD). In turn, the Multiple Attribute Propagation (MAP) component enables the propagation of attributes registered on bridge ports across the network to other participants. The Application component talks to the MAD to make or remove attribute declarations and received from MAD indications when registrations

are made or removed. An Application might have a variable number of software interfaces that are used to make and revoke attribute declaration.

3.4.1 MMRP

By the use of MMRP, talker registrations are propagated throughout the network. This application is not in AVB as seen in 802.1BA. It is not a mandatory protocol for SRP.

3.4.2 MVRP

MVRP is used by switches and end stations to declared membership of a port in the VLAN in which a stream is being sourced. Talker and Listeners must use this protocol in order to join the correct VLAN for a specific stream, otherwise MSRP won't allow streams to be established across bridge ports that are untagged.

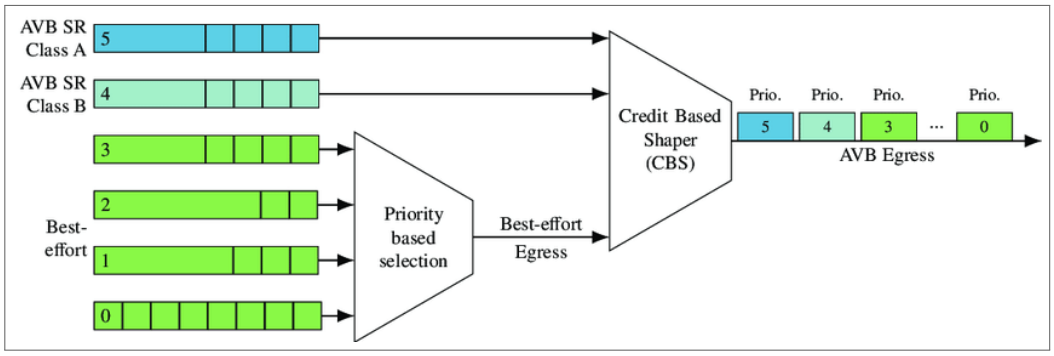
3.4.3 MSRP

MSRP is an MRP application protocol to manage attributes that state the intention of end stations to send or receive streams. As an attribute is declared by end stations, this information is then propagated to the bridge it is connected. At the bridge, their internal data structures such as forwarding, queuing and filtering state tables are updated. Basically talkers declare a Talker Advertise Attribute which triggers MSRP messages advertising as a broadcast, through this message, listeners are informed of data streams. Listeners might answer with the declaration of a Listener Ready Attribute meaning that it is interested and capable to receive the content that is going to be transmitted. When the listener ready messages, sent in a unicast fashion, is received by the source, all resources have been reserved along the path.

3.5 IEEE 802.1QAV

The third and last standard that is part of the AVB standard is the IEEE 802.1Qav (IEEE 802.1Qav, 2010) which goes by the name of Forward and Queuing of Time-Sensitive Streams (FQTSS). Well, once resources were properly reserved with SRP explained in Section 3.4, there should be a way to regulate how it must consumed, and that is the purpose of FQTSS. As it is emphasized in its name, there are two main features of this standard: Queuing and Forwarding, representing the two steps frames go through inside switches: first they are queued and then forwarded. Figure 9 shows how the frames egress the switch. According to its traffic class (Stated in the PCP field of the VLAN tag) the frame is placed in one of the multiple outbound queues and then, the Credit Based Shaper (CBS) define the frames that are sent first.

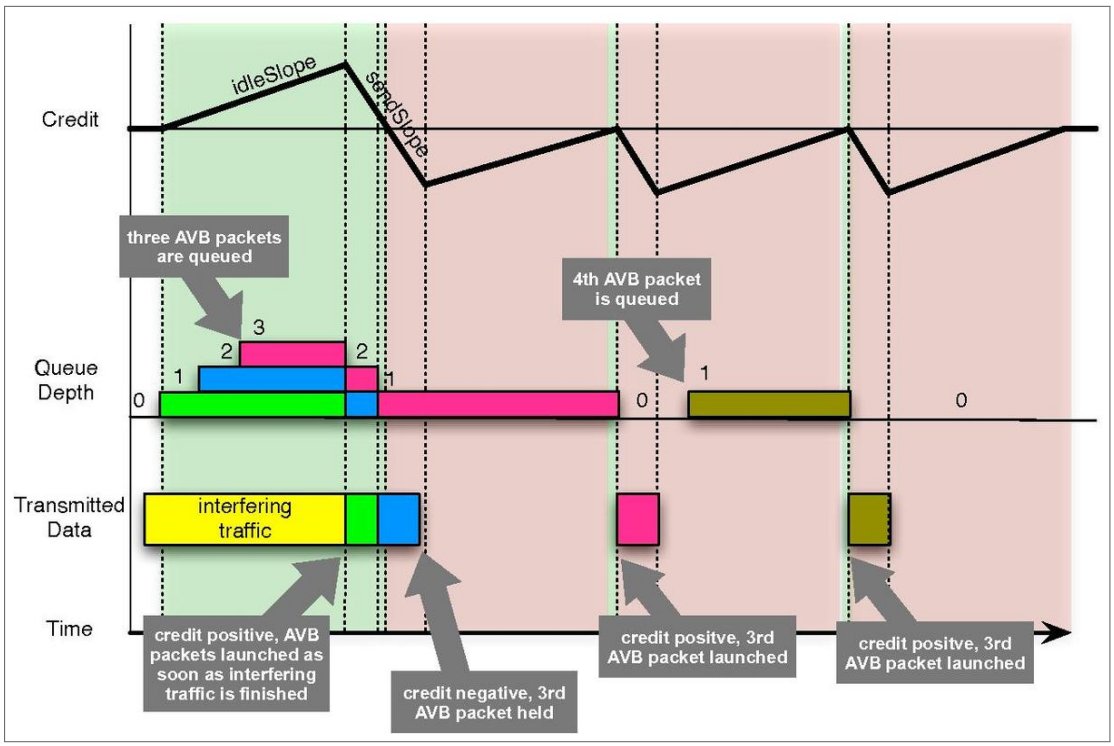
Figure 9 – Queuing Specification



3.5.1 Credit-Based Shaper

CBS has a simple working. It can be better understood by taking Figure 10 as an example. It is possible to observe that two queues are for classes A and B, while the other queues are used for other Best-Effort or Non-critical traffic. AVB queues have credits associated with each queue and frames are only sent when credit is positive. Credits are accumulated when lower priority frames are being sent and lost when frames of this queue are being sent. Figure 10 also shows how the process of sending packets affects the credits of a queue. It is possible to see that the credit of a queue is limited by a high credit and low credit. IdleSlope and sendSlope represent the rate at which credits are increased and decreased respectively.

Figure 10 – Credit Based Shaper



Source: (WIKIPEDIA, 2019)

3.6 TIME SENSITIVE NETWORKING (TSN)

In 2012, the task force AVB was renamed to Time-Sensitive Networking (TSN), introducing a bigger set of extensions. TSN aims to be a generalization of AVB, being compliant with different industries' needs. Markets like Professional Audio/Video (AV), Consumer Electronics, Automotive and Factory Control and Monitoring (Industry 4.0). Because of this generalization, the previous AVB standards were remodeled and new ones were proposed. The following sections highlights the differences in those.

3.6.1 gPTP-Revised

One goal of the new version, IEEE 802.1AS-Rev, is to provide faster recovery to failures on the grandmaster and thus enable high availability for the common global clock, a basic function for the TSN standards. It achieves this by the parallel usage of multiple (and synchronized) grandmaster clocks (a concept called "hot standbys") in conjunction with multiple clock synchronization paths.

As off 2015, the IEEE 802.1AS standard started being remodeled to be compliant with the new TSN requirements. By the time of this writing, the standard is in its 8th version but it is already possible to see a few changes, ideas, suggestions that we will probably see on its final version:

- More domains, compared to only one in the IEEE 802.1AS-2011 version;
- Replacement of reserved fields by some useful values, like the "Synchronization Uncertain" flag, so that devices know when it has stable synchronization along the complete path;
- Changes in the delay calculation (Christian Boiger, 2019);
- Master clock redundancy;
- Link Aggregation.

3.6.2 New traffic shapers

For TSN networks, newer shapers were introduced, the reason is that CBS reduces bunching by smoothing out the traffic flow to greatly reduce the possibility of dropped packets due to congestion but the average delay is increased. CBS is a good approach but in the presence of traffic with stringent requirements it might not be good enough. This is due to the fact the there is a lot of interference from lower priority traffic, which also makes the higher priority data sparse on the network. For not being a well fit for CDT traffic, new shapers started being introduced by TSN.

One of them is the Burst Limiting Shaper (BLS), which as the name suggests, tries to enable bursts of high priority traffic, minimizing the impact of other traffic classes.

BLS works as acting in the opposite way as CBS. Credits are accumulated when traffic of this class is being sent and decremented otherwise. Another shaper, the (802.1QBV, 2016), differs from the CBS on the fact that while CBS aims at transmitting all packets of the same traffic class at a bounded bandwidth. The Time Aware Shaper (TAS), as it is also called, makes enhancements to scheduled traffic by executing transmission algorithms on queues at possibly different periods.

Multiple other additional shapers, like the Asynchronous Traffic Shaping (ATS), standardized in IEEE 802.1Qcr and the Cyclic Queuing and Forwarding (IEEE 802.1Qch), also called Cyclic Shaper (CS), have also come into play. Due to this variety of options, researchers have been formally analysing them. (Thiele; Ernst, 2016)

3.6.3 IEEE 802.1Qcc

In TSN networks, Switches/Bridges and End nodes are still found. But, depending on the type of the system, it is also possible to find two additional elements in the network, the Centralized User Configuration (CUC) and Centralized Network Configuration (CNC) modules.

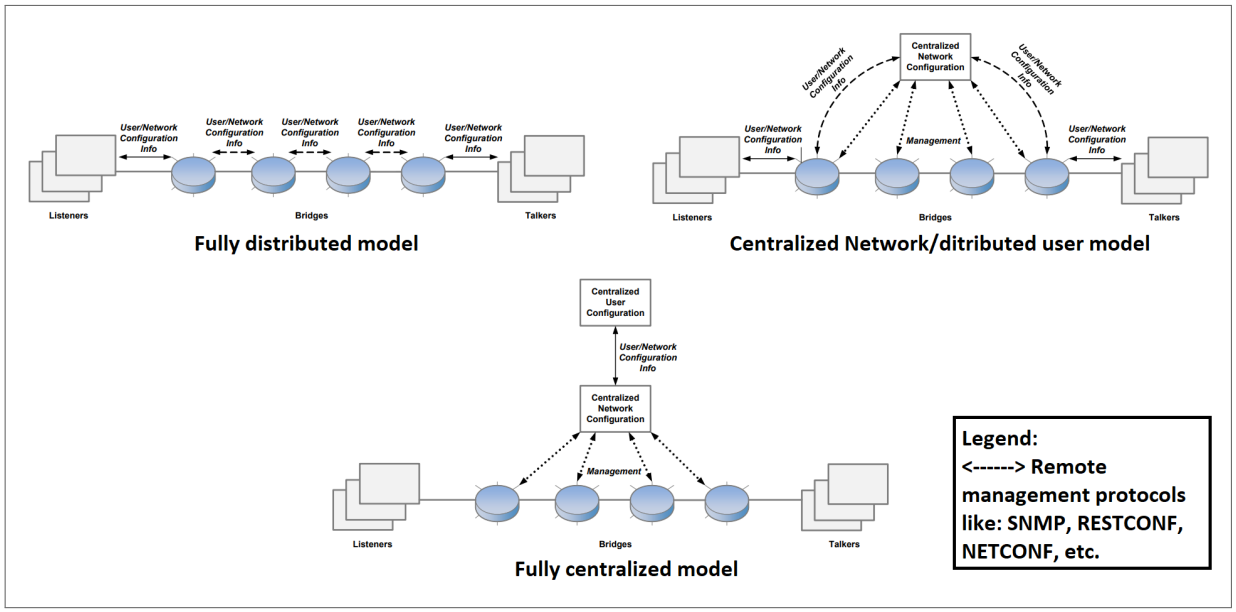
Depending on the domain, safety-critical real-time applications require dynamic reconfiguration of the devices to meet new operational requirements. Although this is always the case for the automotive domain, for factories this is quite common. Still, it might be needed to change ECU's properties to fix eventual problems. That is the main goal of the IEEE 802.1Qcc standard, allow systems to be reconfigured at runtime with minimal interruption. It aims to provide enhancements to the proposals made on the Qat. Among the improvements are the specification of enhanced managed objects for SRP and the proposal of different network configuration models. The three proposed configuration models are:

- **Fully distributed model** where the end-stations are responsible for transmitting configuration information to the network. This means that bridges must receive this messages and readjust its internal state without knowing anything from the rest of the network;

- **Centralized model** also has messages propagating from the end-stations but they are directly forwarded to a CNC component. The CNC has a complete view of the network and has higher computational capacity, being then responsible for calculating the new configuration for all the bridges;

- **Fully centralized** model introduces a new component called CUC component. This is responsible for discovering end-stations and enumerate its capabilities before providing this information to the CNC. After receiving this information from the CBS, the CNC will compute the suitable configuration and configure all the bridges.

Figure 11 – TSN Network models



Source: (IEEE 802.1Qcc, 2018)

The choice of one of these three models will depend on how complex and dynamic the network topology and its configuration is. As expected, the fully distributed is for use cases with less complexity and the fully centralized for those when the TSN stream requirements depend on inputs from the physical environment.

Most of the TSN standards defines a set of Management Information Base (MIB) andYet Another Next Generation (YANG) models. MIB defines a set of managed objects that allow the configuration of a set of parameters. YANG is a data modeling language that is used for describing the managed objects prior to accessing them. Data models encoded in this modeling language might be represented in Extensible Markup Language (XML) or JavaScript Object Notation (JSON) format. These parameters can be red or written by the use of YANG models.

The IEEE 802.1Qcc standard foresees the use of the network management protocol to exchange these configuration models. A specific protocol is not defined but options are the Simple Network Management Protocol (SNMP), NETCONF and RESTCONF. They are application layer protocols whose responsibility is to access and modify information on managed devices. Typically accessible devices on a network, like printers and switches, support this protocol. These devices can then have their internal behavior changed by manipulating these information.

The access of these objects whether the purpose (reading, changing, creating or deleting) must be strongly controlled in order to avoid malicious activities that could lead to dangerous attacks. Even a simple read of these parameters can indicate to the attacker which configurations are being used, how the topology looks like, what applications are being run. These information might be useful when deciding which attack would work

under those circumstances. (Pop et al., 2018) Among the manageable features are:

- Credit-based Shaper algorithm and its configuration;
- Frame preemption;
- Scheduled traffic;
- Frame Replication and Elimination for Reliability;
- Per-stream filtering and policing;
- Cyclic queuing and forwarding.

3.6.4 Seamless Redundancy

Failures on the network can happen when it is not expected. Losing a critical packet, that is carrying a safety-critical information, because a link is down, must never happen. Therefore, for such use cases, where losses are not tolerable, a redundancy method must be available. There are two types of redundancy: Seamless and Non-seamless, for the first, (IEEE 802.1CB, 2017) can be applied. This standard has options for duplication and elimination of frames to occur only on switches avoiding overload the constrained end-devices.

3.6.5 Per-Stream Filtering and Policing

Another primary feature to improve the reliability of the network are filtering and policing rules introduced by (IEEE 802.1Qci, 2017). It aims at protecting overusage of reserved bandwidth by limiting the sending of time-sensitive streams. (Rodney Cummings,) The standard conceptualize an operation where incoming streams are classified by a Stream Identification and goes through a set of filtering and metering instances. After passing or failing the filters, the frame can be allowed to be transmitted or not. Although initially thought for reliability purposes, the same information can be valuable for improving the security of the network. The same filtering database and counters maintained can be accessed and updated by security mechanisms to enhance the security of the network. (BARTON; HENRY, 2018)

3.6.6 Frame Preemption

The time-triggered behavior of 802.1Qbv solved one problem but created another. What happens when a lower-priority frame delays the transmission of a higher priority frame? If lower-priority frames requires more time for being sent than the size of the window reserved for it, delays in the next window will happen. This was the reason behind the proposal of Frame Preemption in (IEEE 802.1Qbu, 2016). Frame Preemption specifies how

the transmission of lower-priority frames can be interrupted, so that higher-priority frames can be sent at the right time. Although one frame can preempt another, the frame that triggered that a preemption of other frame cannot be preempted (multiple levels of preemption).

3.7 TRANSPORT PROTOCOLS

Time-sensitive media streams on AVB/TSN network is encapsulated in special transport protocols standardized to connect and interoperate media based devices on a network. This task is responsibility of three protocols: IEEE 1722, IEEE 1722.1, and IEEE 1733. 1722 is a layer 2 transport protocol, also called Audio Video Transport Protocol (AVTP). 1722.1 is the Audio Video Discovery, Enumeration, Connection Management and Control Protocol (AVDECC), which is a supporting protocol for the 1722. 1733 is a layer 3 transport protocol that leverages the Real-time Transport Protocol (RTP) and is an alternative to the 1722 protocols.

3.7.1 Audio Video Transport Protocol (AVTP)

Used since the time when only AVB existed, the IEEE 1722-2011 (IEEE 1722, 2011) was the first transport protocol for transporting time-sensitive media streams. The Layer-2 protocol allows timestamped frames of multiple media formats to be received by nodes such that they know exactly when to play. This makes sure that data is streamed in multiple nodes synchronously. During its evolution the AVTP has also been modified to support additional critical control applications.

3.7.1.1 Media Formats

AVTP is an extremely flexible protocol, which can be used to encapsulate dozens of different audio formats, video formats and other controlling data. For this reason, one Common Header Format and three additional header formats are used to support the different media types. Common Stream, Alternative and Control Header formats.

3.7.1.1.1 Audio and Video formats

Present since the early days of AVB, media formats to support the transmission of Audio and Video have been one of the most important of them. Address Access Format (AAF), Compressed Video Format (CVF) and IEC 61883/IIDC Format are a few examples of the multiple formats that support Audio and Video data. They enable the transmission of a variety of multimedia, which are used for entertainment applications, driver assistance systems and autonomous systems.

3.7.1.1.2 Clock Reference Format (CRF)

AVTP also defines a message type used to embed gPTP time and distribute to talkers and listeners. This allows, for example, the switching from sources to be streamed on a listener or streaming data from multiple talkers and know that the streams are all synchronized with each other. This is needed because the listener must be prepared to receive media in different sample rates. For example, upon receiving audio signals, a listener might receive a stereo audio from one source and a 5.1 audio from the other, with Clock Reference Format (CRF), sample rate conversion is not needed. This message type is not required for TSN but its use simplifies the maintenance of the system timing.

The CRF message is usually distributed from one controller with CRF capabilities to multiple talkers and also to the listener who will sync the multiple streams.

On an AVB network there are two different clocks to be considered: gPTP clocks and media clocks. Sample rate is how often a signal is measure at each second, like how many snapshots of the signal are taken. Used to allow listeners to receive audio from these multiple talker without the use of Sample Rate Conversion (SRC). SRC is needed because media, Audio/Video for example, is stored and sent at different sampling rates compared to the listener.

3.7.1.1.3 AVTP Control Format (ACF)

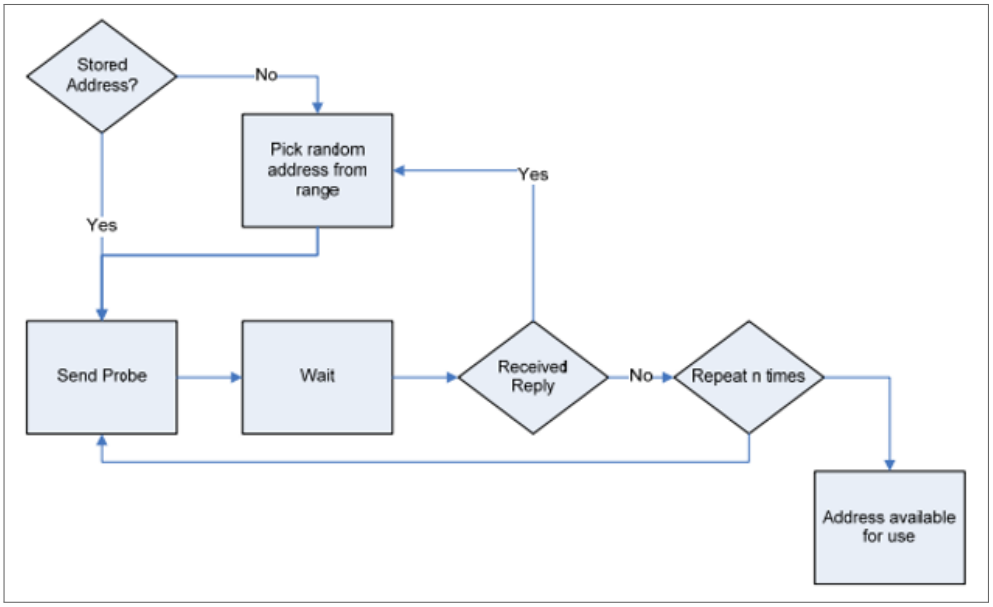
As introduced at the beginning of this chapter, TSN aims at extending the support for different use cases in different industries. This goal is well reflected in the AVTP Control Format (ACF), which provides a way of transmitting control messages of other protocols. Examples are the possibility of transmitting CAN, LIN, FlexRay and other legacy automotive protocols over Ethernet. This is a highly important use case for automotive when considering Ethernet as a backbone technology (STEINBACH, 2018). This way, it is possible to transmit such messages with the lowest latency possible. Even though it is possible to transport control data with TSN capabilities, this is not mandatory. ACF offers the possibility of transmitting messages as time-aware as well as non-time-aware. Therefore, two different headers are available, one for each type. The first uses the Common Stream Header, which carries the presentation time so that messages can be synchronously delivered. The second type uses a subset of the fields in the Common Header Format and the Common Control Header. This can be used to deliver diagnostic messages, such as updates, where low latency is desired, but synchronization is not required. The control messages are encapsulated by concatenating as many types of messages as needed. These are chained as multiple TLVs after the header, there is no limit and unrecognized messages types should be ignored by the Listener.

3.7.1.2 MAAP

The MAC Address Acquisition Protocol (MAAP) is the method used in the network in which nodes allocate themselves a unique multicast address. This protocol works similarly to how nodes can auto-generate addresses using the Neighbor Discovery Protocol (NDP). A range of multicast addresses was reserved to be allocated dynamically. Multicast addresses are used for nodes such that when streams need to be transmitted they can reach all the interested nodes.

Interested applications can use this mechanism to request a single or multiple addresses. For this, the application must select an address from the pool and send one or more addresses in MAAP_PROBE messages in order to determine whether the selected address(es) are available or not. After sending, the requesting node must listen to possible MAAP_DEFEND messages from other nodes that could indicate the address is already in use. This process repeats until the required amount of addresses has been found. Once the required amount of addresses was found, the node must send periodically MAAP_ANNOUNCE messages in order to inform other nodes of the unavailable address(es).

Figure 12 – MAAP Address Acquisition



3.7.2 Audio/Video Discovery, Enumeration, Connection Management and Control (AVDECC)

Serving as a supportive protocol for the IEEE 1722, is the (IEEE 1722.1, 2013) whose goal is to define device discovery, enumeration, connection management, stream setup and control. In more detail, here is a list of tasks performed by this protocol.

- Automatically discovers the addition and removal of devices on the network;

- Retrieves the entity model of the discovered pro audio devices;
- Connects and disconnects streams between the discovered devices;
- Obtains status information about the discovered devices and their connections;
- Controls the discovered devices;
- Remote modification of parameters.

In a network where 1722.1 is used as management protocol, end stations can have an additional role to being talker or listeners. The use of AVDECC is similar to the combination SNMP MIB explained in section 3.6.3. They can also have a controller role. All of the three sub-protocols described here use the Common Control Header (CCH). Each message adds its own custom fields to the CCH. All three protocols and its features are independent from the other. In the next sub-sections, a short overview of the responsibilities from each protocol will be presented and what are the added fields and its meanings.

3.7.2.1 AVDECC Discovery Protocol (ADP)

As the name suggests, AVDECC Discovery Protocol (ADP) is the message type that allows finding end stations on the network. Besides discovery, termination of service publication is also done by ADP. It is basically a protocol nodes use in order to inform the whole network about its availability and its capabilities. There are three message types that compose the protocol

- **Availability announcements** to advertise that a device just joined the network along with its capabilities;
- **Departure announcements** to advertise that a device is about to leave the network;
- **Discovery messages** to trigger availability announcements of one or multiple device.

3.7.2.2 AVDECC Enumeration and Control Protocol (AECP)

Enumeration is the process of fetching and listing the capabilities of an end station with the purpose of connection management. These capabilities might be simple things as volume control, bass levels, mute, etc. Control on the other hand is the process of actually modifying the enumerated capabilities in form of parameters.

The **Entity Model Format (AEM)** is the main message format of the AECP protocol. it is responsible to manage an entity's features. Each entity contains a set of descriptors, which are attributes that describe the capabilities or features of an entity. There are

in total 37 descriptor's types each of which with a variable amount of fields. Descriptors can have different types, such as Entity Descriptors, AVB descriptors, Audio Descriptors and so on. Among Entity descriptors are the Entity ID, Name, Firmware Version fields. Examples of AVB descriptors are the Clock Identity, Priority fields that are sent on the announcement and signaling messages from the BCMA algorithm.
Fields of a descriptor a disposed as elements in a vector but with variable offsets. Each descriptor has at least a type and an index field. The index field is used when an entity has more than one of this descriptor. For example, an entity that has two audio speakers might have two descriptors of the audio type with indexes 0 and 1. An imaginary descriptor and its fields can be seen on table 4

Table 4 – Example of a descriptor

Offset	Length	Name
0	2	descriptor_type
2	2	descriptor_index
4	8	field1
12	16	field2

For managing entities, a set of command and response message pairs is available. These messages allow other entities to perform operations like read/write values to a descriptor field. Every talker/listener must implement at least three commands:

- **Acquire entity** used by a controller to get long-term exclusive access to an entity;
- **Lock entity** used by a controller to get short-term exclusive access to an entity;
- **Entity available** used by one entity to check if another entity is still alive.

The other commands like read descriptor, write descriptor are optional. A complete list containing all possible messages as well as the command and response message structure is available in the IEEE 1722.1 (IEEE 1722.1, 2013) standard. Additionally, the following formats are also available.

- **Address Access Format (AAF)** allows controllers to Read from, Write to or Execute an action on an Entity's memory;
- **AV/C Format** allows to carry commands and responses compatible with the IEEE 1394 standard, also known as FireWire;
- The **High-bandwidth Digital Content Protection (HDCP)** Interface Independent Adaptation (IIA) is an authentication protocol.

3.7.2.3 AVDECC Connection Management Protocol (ACMP)

Connection management is the process of connecting and disconnecting streams between interested end stations. This message format is used to trigger the registration and de-registration processes of SRP explained in section 3.4. Beyond starting and stopping connections, AVDECC Connection Management Protocol (ACMP) also allows retrieving information about the connection status. This enables the controller to know how many devices are listening to a stream, how many devices a talker is transmitting to, the VLAN ID of the stream, among other properties.

3.7.3 Audio/Video Real Time Protocol

Similar to the AVTP introduced previously, the IEEE 1733 standard (1733, 2011) proposes how to achieve interoperability between time-aware systems across bridged and routed LANs. This protocol leverages the Real-time Transport Protocol (RTP) set of protocols and AVB/TSN protocols. It becomes even more relevant because it is compatible with IP for encapsulating data of critical applications. RTP alone as specified in Request For Comments (RFC) 3550 (SCHULZRINNE S. CASNER, 2003) does not specify any means of reserving bandwidth or how to guarantee QoS. Therefore, the IEEE 1733 standard specifies how this protocol can be used together with the AVB/TSN mechanisms, e.g. time synchronization, stream reservation and it introduces a new RTCP packet type, the AVB RTCP, a message specific for sending necessary AVB information. There are two relevant protocols for understanding the IEEE 1733 standard:

- RTP: Which is the encapsulation protocol for real-time data;
- RTCP: Which is a supporting protocol for RTP and is responsible for management of transport sessions.

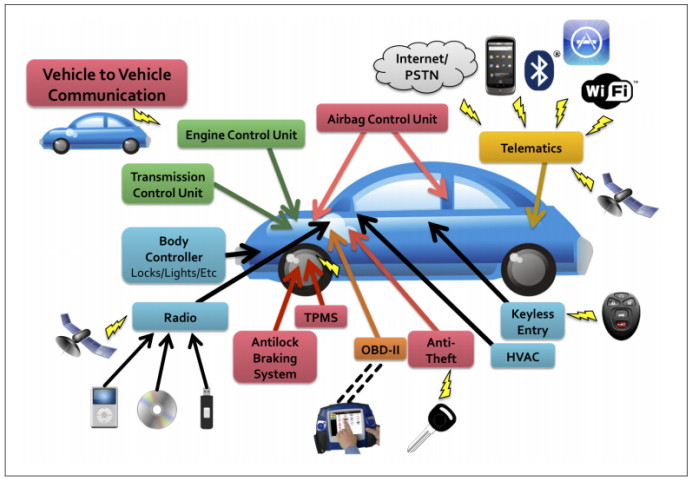
The new RCTP type introduced in the IEEE 1733 standard, works similarly to the Clock Reference Format (CRF), explained in Section 3.7.1.1.2, as a way to ensure media are going to be rendered at the same rate as he talker

4 AUTOMOTIVE SECURITY

4.1 INTRODUCTION

The new progress found in cars today is in a big part due to the replacement from mechanical functions to electronic ones. In this context, programmable computers play an enormous role. This added lots of new vulnerabilities to cars. In Figure 13, it is possible to see how diverse the attack vectors present in cars are today. Even though cars remained disconnected from the outside world for a long time, due to evolving market demands, this reality has changed. Reports have shown that cars are getting connected to external servers and that vulnerabilities regarding these remote connections are increasing and becoming the main entry-point for targets into vehicles. (Upstream Security, 2019)

Figure 13 – Automotive Attack Vectors



Source: (CHECKOWAY et al., 2011)

For each of these attack vectors, different defense mechanisms can be applied to protect against abnormal activities. Independently of whether this unexpected behavior was actively provoked, by a malicious attacker, due to a broken equipment.

This work focuses mainly on the security aspects of the communication on in-vehicle networks and its protocols, as the ones explained in Chapters 2 and 3. Switched Ethernet offers by default some security advantages then the bus based protocols. Technologies like CAN, where a shared bus is used, allow any malicious node to be physically attached and send messages to the network. The presence of switches connecting devices enforces a physical barrier between them. This allows switches to isolate devices at the port level in case something is detected.

Usually, the same defensive mechanisms used in enterprise networks, like the ones applied to protect organizations, can also be used for in-vehicle networks. A big difference from enterprise and automotive worlds are the outcomes from attacks, where the financial

impacts of attacks in the automotive industry tend to be way bigger than in the enterprise world. An example comes from the Fiat Chrysler Hack back in 2015 (WIRED, 2015). This case became the most known attack in the automotive history, leading to an estimate cost of 1.1 billion dollars due to the recall of more then 1.4 million vehicles, expenses with lawsuits and other related financial hits.

Security is a generic term that is actually composed of many specific individual requirements. To consider the communications of systems to be secure, it is required that these systems provide the following aspects: Confidentiality, availability, authentication, authorization, integrity, non-repudiation and others (SALINI; KANMANI, 2012). In the end, there is a close relationship between all of these individual points. For example, the integrity and authentication of a message are directly related to the availability of the system that will process the message. As a rule of thumb, in-depth security is advised by security experts. This means multiple layers of security on top of each other, but in the automotive industry this is not always possible. There are several problems in this case:

- More security mechanisms means more implementation and tests costs;
- The capabilities of the ECUs are limited. Normally the ECU will only have enough processing, memory, storage to handle the specific tasks it is supposed to handle;
- Applications cannot have their performance affected because they are dealing with critical information.

The number one approach to have secured communication channels is by the use of secured networking protocols. Here, it is again possible to apply a layered approach. There are multiple secured networking protocols, basically there is one for each protocol in the OSI model. Some possibilities are: MACsec, IPsec, TLS, HTTPS and SEND. Security protocols as these ones, are based on cryptographic primitives like encryption function, hash functions, etc. These, in turn, have their strength proved mathematically, with the degree of strength based on a security parameter which can be, for example, the size of a secret key in an encryption scheme. In general, the bigger the size of the key or the number of rounds the key(s) is/are handled, the more difficult it is to recover the plain text from an encrypted text. But bigger keys also means more space to store them and also means higher computational cycles in order to generate encrypted outputs. Going against the limited memory of ECUs. What security architects tries to find is a balance, being secure does not mean impossible to break. It is theoretically possible to break such a system, but it is unfeasible to do so by any known practical means.

The overhead brought from cryptographyc solutions have been studied in the literature. The authors from (Treytl; Hirschler, 2008) evaluated the security overhead from applying cryptography to secure PTP messages and they could an increase in the size of a message from almost 70% as well as a considerable increase in the computational (CPU) load for each node.

Table 5 – Layered Security Model

Target Device	Mechanism	Security Concept
Host μC	Secure External Communication	Cryptographic protocols
Switch Controller	Secure gateway	Firewall/IDS
Host μC/Switch Controller	Secure In-Vehicle Communication	Cryptographic protocols
Hardware	Secure Platform	Secure Boot/Flash through HSM

Table 5 shows how a layered security concept looks like. On the bottom you have the security features of the hardware. Some of the key points to be secured on a hardware level are storing critical information such as keys, securing the firmware, as these should boot the right software and also be updated securely. This can be provided by Hardware Security Modules (HSM) and Trusted Platform Modules (TPM). Going one step outside you see the application of security protocols and its primitives for securing communication channels. Another step further there are the secured gateways that are central devices and have a broader view of the network with direct contact with multiple domains. Finally, there are the security of modules that have connection with the outside world.

Of course these are not all security steps for creating secured systems. Security must be presence in all phases of the development lifecycle of products. From the development phase and the use of tools to check code for bad implementation aspects, to the usage of tools to protect the runtime behavior of the software, until the time to update the software with newer functionalities.

The main protection tools for Gateways are Firewalls and Intrusion Detection Systems (IDS), mechanisms that use Deep Packet Inspection (DPI) tools, packet filters, packet monitors. The difference between these security mechanisms might be confusing to understand. Sometimes, intersections between the coverage of these defense mechanisms can be found. Section 4.3 discusses more the differences between these often confused security engines.

4.2 SECURITY PROCESS

Despite having the Layered Security Model as a target for protecting devices, the process of securing them goes beyond it. The actual level of security needed for a project vary tremendously depending on specific details of each system. That is why each individual project should go through a Security Process, which is a series of phases in order to investigate the security requirements of it. Different models have been proposed to serve as a guide to securing products, the Verification and Validation (V&V) Security Engineering Process (SEP) is an example of one. Security has become such a complex topic that it is now a specific topic just as important as the Software Engineering Processes. (SALINI; KANMANI, 2012)

The V&V where it is possible to see verification phases on the left side and validation

phases on the right side. Different variations of such processes have been proposed in the literature, but some tasks are normally present in the process like the threat modeling at the beginning of the process in order to identify assets, potential vulnerabilities, the impact and likelihood of them.

Security Requirements which uses the output from the previous phase to derive security goals and functional requirements.

Security Architecture which is a more detailed and technical refinement of the security requirements, where Software and Hardware components are assigned to fulfill the generic requirements from the previous phase. This phase is where the actual security layers needed are commonly defined. Here it is possible to point out Cryptographic Accelerators, as a hardware component and cryptography libraries as software components.

Security Implementation is the time to bring the defined architecture to life. During this phase, secure coding practices guide the development of the different parts of the product. An example of best practice is static code analysis, where are tool that aim at detecting memory leaks, invalid memory accesses, null pointers, etc.

Security Validation After the security implementation, the validation part of the process starts. This assurance phase, which is a way to check if the implementation satisfies the requirements defined at the beginning of the process. This is normally seen as a testing phase where fuzzing tests, used to find uncovered vulnerabilities by sending malformed data in uncommon patterns to the testing target. Penetration tests are adopted to find more complex weaknesses in the system and works by combining automated tools with expert's knowledge to create custom attacks.

No matter how much effort it's put to defend a system or network, attacks are probably going to take place. Having **Incident Response** teams prepared to handle the outcome of unwanted actions is also a very important part of the security engineering process. A professional unit ready to read incident logs, and promptly trigger security patches to neutralise the threat quickly. Neutralising not only technically speaking but also taking possible legal actions to suppress brand damaging and business impacts.

Some of these tasks are also supported by standards and methodologies, like the EVITA standard for threat modeling, the Threat Assessment and Remediation Analysis (TARA) for, ISO 27000 (ISO-27000..., 2018) for information security management and others, Society of Automotive Engineers (SAE) J3061A (SAE-J3061..., 2016).

As it should be clear, security, must be present in the complete life-cycle of a product, From its development, to its deployment in production, continuous monitoring and upgrades, incidence response, and so on.

4.3 INTRUSION DETECTION AND PREVENTION SYSTEMS (IDPS)

Intrusion Detection Systems (IDS) is a defense mechanism that monitors different characteristics of a system in order to detect abnormal activities or behaviors. A similar type of

system, called Intrusion Detection and Prevention System (IDPS) or Intrusion Prevention System (IPS) only, is a similar type of system that differs from the first by the fact that it acts actively upon detecting an abnormal behavior while IDS normally logs and raises alarms when detecting something.

Sometimes IDSs are confused with Firewalls but, to set some boundaries, it is possible to say the following statements. Firewalls are standalone systems, comprised by a set of rules that are verified individually against every individual packet that arrives or exits a port in order to purely block the packet. An IDS can be seen as a full standing mechanism, which is part of a complete solution that also involves the presence of an operator. Operators receive logs from multiple IDSs and is able to detect distributed attacks, trigger updates to these agents. Complete solution means that the application might be deployed together with that not only checks individual packets but rather the history of packets in order to detect a combination of malicious packets. In addition to dropping packets, other actions might also be possible with IDS, such as alerting agents, (de)activating rules, logging. IDS are sometimes also called Next-Generation Firewalls.

In the subsection both types IDS and IPS are going to be treated as IDS. Two main types of IDS are going to be introduced: Network IDS (NIDS) and Host IDS (HIDS) as well as how these different types categorized according to its detection technique. A combination of both NIDS and HIDS is also possible and it is called Hybrid IDS.

4.3.1 Network IDS

The first thing that typically comes to mind when thinking about breaking into a system is doing so over the internet. Several cases of hacking have happened because attackers exploited open ports on servers or because the attacker was able to run Cross-Site Scripting (XSS), Application Programming Interface (API) end-points without authentication, the list is huge.

In order for these attacks to be effective, the attacker needs to use common networking protocols like HTTP, TCP, IP, with an special content or in specific sequences of messages. Network IDS is a type of IDS that monitors protocol messages over the network in order to detect such abnormal behaviors, therefore, it is needed.

There are different locations in the network where a NIDS can be deployed, they are: Switch (or other forwarding/routing device), End-Point, Middle-Box. The choice of the appropriate location depends on several factors such as whether the topology is a switched network or bus network. For example, an IDS for a Controller Area Network (CAN) Bus, might as well be a unique centralized device, since all messages are broadcasted. On the other hand, if the topology represents a switched network, depending on its size, a decentralized approach might be necessary, where multiple IDS instances work together. Example of Open-Source NIDS are (SNORT, 2019) and (SURICATA, 2019).

4.3.2 Host IDS

Many people think that the network is the only way to get into a system and damage it. This is not completely correct, indeed, networking attacks are the most common entry points to systems, but they are not the only one. An attacker might for example, exploit targets by using physical connection like USB, CD ports or even leverage hardware vulnerabilities by making changes in the electromagnetic field, voltage, clock. This latter type of attacks is called fault injection. One common thing is that for both networking attacks and physical attacks, the attacker will almost likely try to access confidential files, escalate privileges for specific users.

This is when Host IDS makes itself important, as a monitoring tool to watch the internal state of a system. Observing host events such as user space and kernel space processes, files, networking sockets, signals, pipes, Inter-Protocol Communication (IPC), terminals, threads, I/O, etc. allows the possibility to detect the presence of abnormal activities before the damage reach its full extent.

Just as in the case of Network the IDS, the placement of a HIDS plays a big role in the defense strategy. A HIDS can the placed in User Space, Kernel Space or even in virtualized systems, being located in virtual machines or hypervisors. Example of Open-Source HIDS is (OSSEC, 2019).

4.4 DETECTION TECHNIQUES

A further categorization of IDS is by the type of detection it uses to identify attacks. They are: Specification-based, Signature-based and Anomaly-based. Each approach has its own advantages and disadvantages, among them is the number of False Positive (FP) and False Negative (FN) raised. Anomaly-based detection tends to identify an activity as not normal when the activity is normal and also to identify it as normal when it is not, FP and FN respectively. Specification-based detection on the other hand tends to output more True Positive (TP) and False Negative (TN), which are the cases when an intrusion activity is correctly identified, in the first case and when a normal activity is truly classified as normal.

4.4.1 Specification-based detection

Also known as rule-based detection, this is the most simple and straightforward type of detection. It consists of a set of conditions that are considered expected to be seen in the system. The detection is then made by verifying whether the condition holds or not.

4.4.2 Signature-based detection

Signature-based detection takes the previously explained rules-based detection one step further. It works by defining a set of known malicious behaviors and the detection is made

by checking if the input activity matches one of these signatures, also known as patterns.

4.4.3 Anomaly-based detection

Anomaly-based IDSs work by defining profiles of what is considered a normal behavior and inconsistencies to it raises alerts. In the past statistical methods and more recently, machine learning, deep learning and other artificial intelligence techniques have been proposed to detect anomalies. In the automotive industry however, there is a preference for proven methods. This has a few reasons, like required computing power, energy consumption, number of false positives. AI-based detection is constantly being improved and must find its place in production security products really soon.

4.5 TESTING TECHNIQUES

One of the biggest challenges in developing an IDS actually validating if it operates properly. This is because, the IDS must be fed with realistic and complete input, in the sense of mirroring all possible conditions, to be able to see how it behaves (ANTONATOS; ANAGNOSTAKIS; MARKATOS, 2004). (ZARPELÃO et al., 2017) summarizes a few strategy approaches to creating these testing data, they are:

- **Hypothetical**: makes different assumptions about how real data will be present in production but without any guarantee about similarity;
- **Empirical**: exemplify input data recorded from real world scenarios;
- **Simulation**: show the data created by simulation tools;
- **Theoretical**: represent input data created with formal. methods.

Many times, finding empirical data that contains both normal and malicious activities is difficult. Therefore, merging empirical data with simulated, theoretical and hypothetical data is a commonly used strategy.

4.6 SUMMARY

This chapter gave a broad overview on the main security aspects of automotive systems. The main attack vectors were introduced, followed by the main defense mechanisms with a deeper focus on Intrusion Detection Systems. In this chapter, the main steps of developing a secured system were also shown. The V&V SEP shows the TARA as the first step in the process, therefore, a limited TARA will be conducted on Chapter 5.

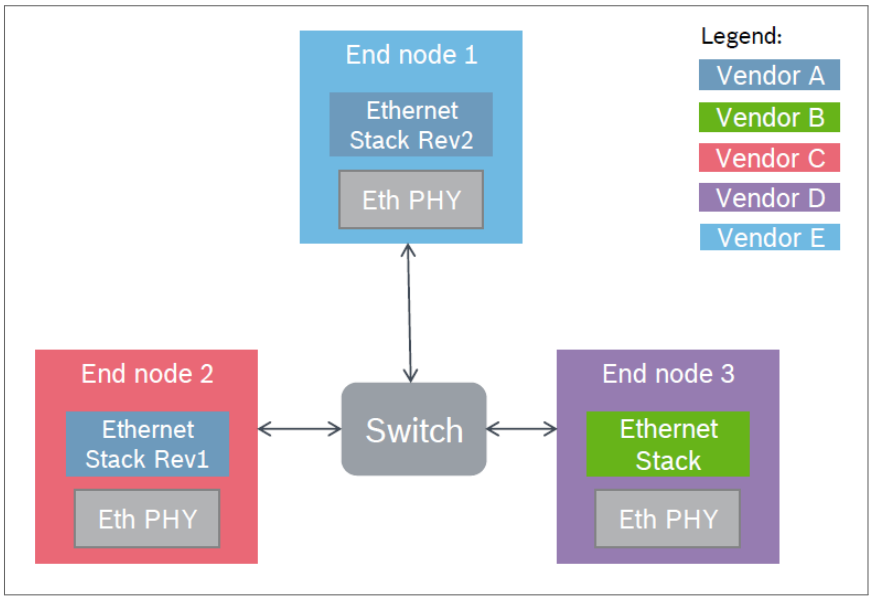
5 THREAT ANALYSIS

There are a number of security considerations regarding AVB/TSN that are not addressed in the standardization documents. This chapter shows possible vectors used to attack AVB/TSN devices.

As it should be clear from Chapter 3, TSN is a really powerful and big set of standards to add the deterministic behavior to the common Ethernet standard. However, there are a few characteristics from AVB/TSN that make TSN-based systems vulnerable to multiple attacks.

TSN standards work like building blocks, while some of them are mandatory, like gPTP, using others, like SRP will depend on your system's requirements. This flexible nature added to the fact that these building blocks might be supplied by different vendors, as illustrated in Figure 14, can cause some problems. In fact, interoperability is a good characteristic, using components from different vendors on a seamless manner is a really good feature. However, in the case of TSN this can be dangerous because TSN is a really recent standard, which means that the individual standards are not so mature as other protocols like TCP, IP and so on. TCP/IP are really old protocols which have gone through many iterations of improvements over the decades. Some of the TSN standards, on the other hand, are not even 1 year old by the time of this writing. The standards are still prone to contain errors or be miss-interpreted by stack developers. For this reason, such evaluation as shown on this chapter is really important.

Figure 14 – TSN Devices



For each protocol/standard, a list of threats are going to be listed. The items are enumerated in the format *"TS_PROTOCOL NUMBER: TITLE"*. TS is a short for "Threat

Scenario" and "PROTOCOL" is the the protocol in analysis, like gPTP, for example. The NUMBER is only for enumeration purposes, not reflecting any severity levels. A "TITLE" shortly describes the attack vector in question.

5.1 THREATS ON GPTP

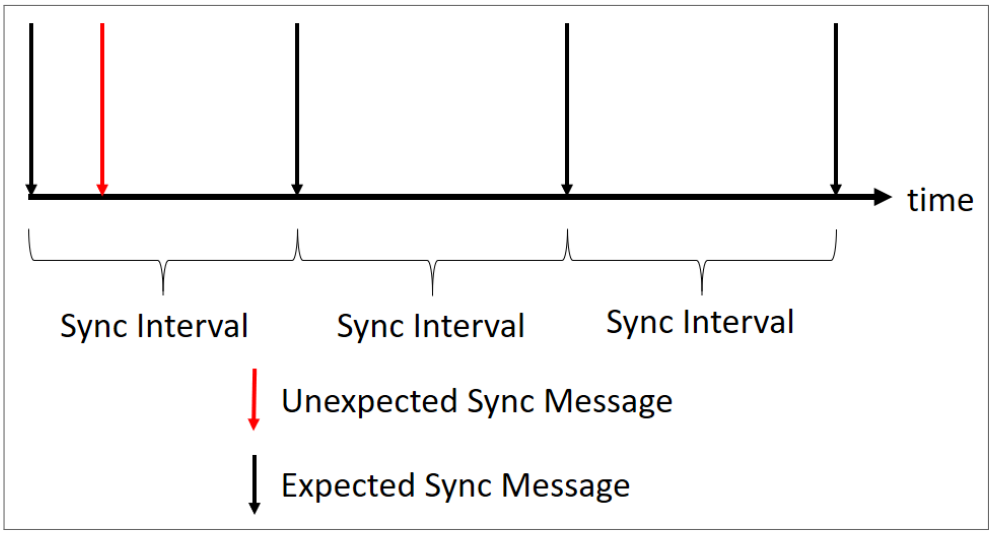
One of the main requirements that allow real time critical systems to work properly is time synchronization. The same importance of the mechanism however, was not given when designing security properties for it. After the publication of gPTP, a security extension, called Annex K, was introduced to try to cover security gaps. Despite proposing good solutions, the annex is not sufficient to cover all attack vectors of the protocol. Many publications describe uncovered gaps (TSANG; BEZNOSOV, 2006) (Ullmann; Vögeler, 2009) (O'Donoghue, 2016) and possible countermeasures. It is important to highlight that the threat scenarios described here do not cover all threat scenarios from the IEEE 1588-2008 (PTP) standard. This is due to the fact that PTP has additional features not used in gPTP. Examples are the threats regarding the management messages of PTP as well as by the use of different lower-level networking layers as UDP/IP. Additional and more recent literature covering the threats in this protocol can be found in (Itkin; Wool, 2017) (Moreira et al., 2015) (MIZRAHI, 2014).

The first two threats in this thesis surround gPTP are delay attacks. They can be achieved in different ways and are easily confusable with normal congestions on the network. Therefore extremely difficult to detect and prevent delay attacks.

TS_GPTP 1: Delay Response Spoofing In this attack the malicious node is able to forge DELAY_RESPONSE as it were the grandmaster. The attacker needs to replay a valid sync or announce message, which will cause nodes to send DELAY_REQUESTS and only then the attacker can send the forged DELAY_RESPONSE. As a consequence the attacker will produce negative impacts on the delay measurements for the target node, i.e. a clock not so accurate when compared to the grandmaster. DELAY_RESPONSE messages also have a sequenceId, the attacker needs to keep track of the real messages sent by the grandmaster.

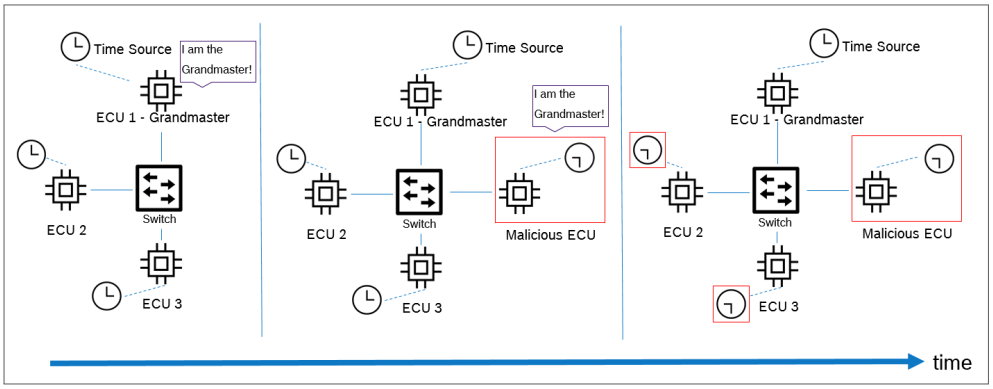
TS_GPTP 2: Follow-up Spoofing A similar scenario as TS_GPTP 1, but relying on Follow-Up messages. Sending spoofed follow-up messages on behalf of the grandmaster can also force the target node to recalculate its clock according to the presented value. Two things are important to note: First, that the attack is effective during short intervals; and second, the difference between one-step and two-step synchronization. This is because the real grandmaster will send legitimate messages, making the slave set the clock back to the true time. The gPTP-2011 always uses two-step synchronization, where, only the follow-up messages carry the timestamp at which the initial sync message was sent. In one-step synchronization, both messages carry this original timestamp value.

Figure 15 – gPTP Follow-up spoofing



TS_GPTP 3: Rogue Grandmaster When BCMA is used for determining the most suitable clock in the network, any node can participate in the procedure. A malicious node can advertise itself as having the best possible attributes and thus win the right of being the grandmaster. Two consequences can derive from this attack. First, the new malicious grandmaster dictates the pace at which the other clocks are synchronized. Second, the whole Master/Slave hierarchy can change, this process is illustrated on Figure 16.

Figure 16 – gPTP Rogue Grandmaster



TS_GPTP 4: Denial-of-Service Attacks Denial-of-Service (DoS) attacks are characterized by the target being overwhelmed with operations that will gradually exhaust its computing resources. Slowing it down or even crashing the system so that it is no longer responsive. In networks this type of attacks are achieved by sending multiple packets to the target. This flooding attack can be classified in a few categories:

- Spoofed: In this type of flooding attacks the attacker makes sure to use a spoofed address as a source address . This guarantees, to some degree, that the identity of the attacker remains unknown.

- Distributed: In this attack, the attacker performs a flooding attack by sending traffic from multiple sources instead of one. Leveraging this amount of traffic, the attacker is able to attack multiple victims at the same time and also make sure the effects are seen more quickly.

- Reflective: The attacker attempts to flood the victim not by directly sending frames. Instead, the attacker sends frames to third party entities that will altogether reply to the victim.

- Amplified: This type combines the spoofed and reflective properties. The attacker leverages queries that generate multiple responses. These queries however, have a spoofed source address and hence, reflects bigger number of responses to the target in return.

TS_GPTP 5: Man-In-the-Middle Attacks (MITM) In this scenario, a malicious node sits in the middle of a slave and its master. By monitoring the network traffic, the attacker is able to intercept, read, copy, modify, replay and drop messages to its destination. This fact combined with the non-usage of authentication and integrity mechanisms (what could be provided by MACsec, for example) might lead to different types of attacks. Modifying messages can be a good way to introduce wrong timestamps that will mislead the slave nodes into adjusting itself to an invalid clock value. Replay attacks are one example where a previously sent message is re-transmitted. Replaying a message can be used to trigger a previously performed action in a system. Dropping messages is also a possibility and can be used to avoid messages from reaching a destination.

TS_GPTP 6: Header Manipulation Many of the messages exchanged in the context of PTP/gPTP have fields with clearly defined values. It is possible, though, that the attacker fills these fields with invalid information in order to trigger unexpected behaviors. Some of the messages described in Section 3.2.1 can carry additional information in the format of TLV. The standard does not restrict the number of TLVs that can be chained in headers. This also opens space for malicious nodes to craft messages with an unlimited number of TLVs, which can trigger faults on the protocol design and implementation vulnerabilities.

TS_GPTP 7: Delay attacks A specific MITM attack is worthwhile highlighting. Without modifying any message as in TS_GPTP 1 and TS_GPTP 2 it is also possible to get slaves to set the wrong clock. A node just has to make sure it will delay the arrival of messages at the intended recipient. This will cause an increase in values used for offset and delay calculations. An additional consequence is forcing the slave to timeout while waiting.

TS_GPTP 8: Management messages Time-aware nodes keep a set of parameters that characterize their attributes on a gPTP enabled network. As introduced, this data set can be configurable by the usage of manageable objects, YANG models and the SNMP

protocol. These three last features can be used to misconfigure nodes in a way that will completely corrupt the operation of a node.

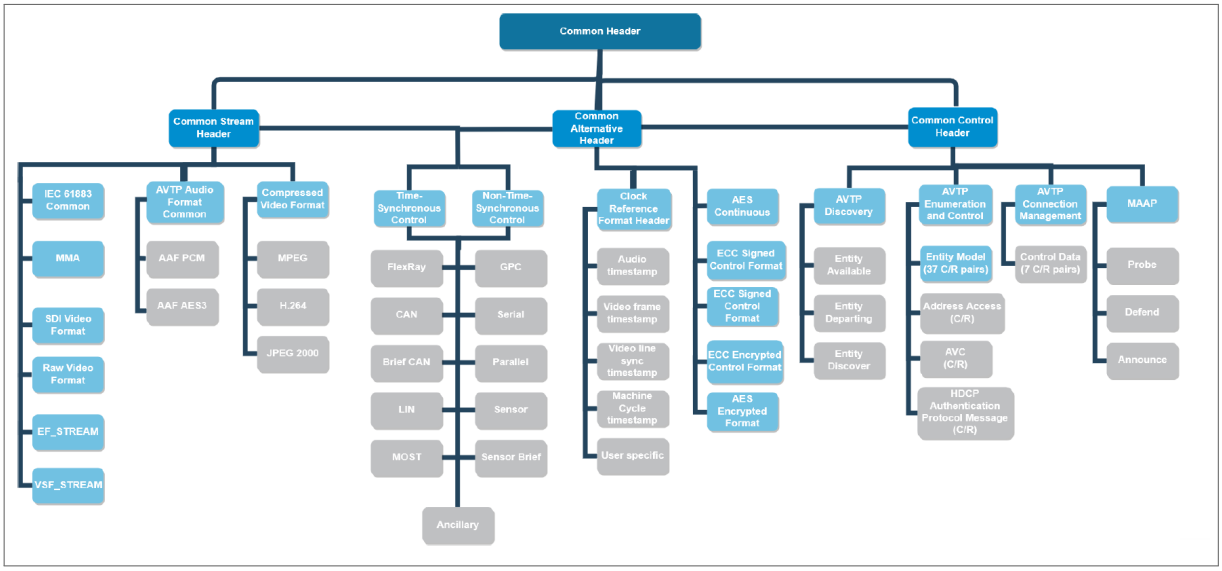
5.2 THREATS ON AVTP

As it should be clear from the explanation on Section 3.7.1, there are many media and control formats possible for AVTP messages. Different media formats might have different additional fields beyond the fields defined in one of the three common headers. Therefore, those additional fields might bring additional concerns to the network.

TS_AVTP 1 Header Manipulation

The AVTP protocol offers a high variety of media formats to be encapsulated in the network. Figure 17 illustrates how many different headers combinations are possible to be seen in the network. On the root there is the Common Header, then it comes three other headers: Stream, Control and Alternative headers. From each of these three, a multitude of media format's headers are derived (The tree's leaves). In turn, each media type can have its own subtypes also with specific headers. Due to time constraints, not all of these media formats are going to be deeply assessed individually. Therefore, this threat scenario is an alert for IDS designers to take a second look at each media format that is going to be supported on the network. Each one of them might introduce additional risks, either by some protocol fields, or by how the state machine of the protocol works.

Figure 17 – AVTP Media Formats



TS_AVTP 2 Complex Decoding DoS

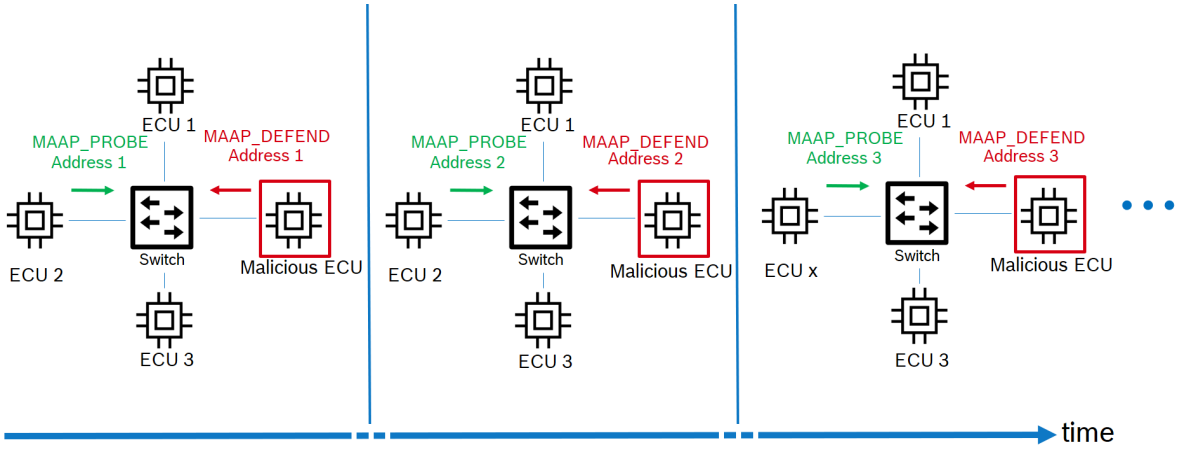
Some message formats, like the IEC 61883, introduce the concept of compression mechanisms to encode streams. This feature can be useful for networks but also dangerous, since it can cause overload due to complex uncompressing/decoding mechanisms. It is possible to cause DoS on Listeners trying to process this types of data. The attacker can

craft payloads that apparently do not present any sign of non-uniformity but will cause the receiver to be overloaded.

TS_AVTP 3 MAAP DoS

Entities might use the MAAP mechanism explained in Section 3.7.1.2 to assign multicast addresses for the transmission of streams. The operation of MAAP, based on Probe messages to verify if an address is free to be used, allows malicious nodes to send Defend messages stating the address is already in use. If a malicious node keeps replying probe requests with defend messages, a valid address will never be assigned to the talker. This exploitation is illustrated in Figure 18.

Figure 18 – MAAP DoS



TS_AVTP 4 Reconnaissance

The same MAAP mechanism from TS_AVTP 3 can also be used to map the nodes in the network. Malicious nodes are able to monitor probe, defend and announcement messages to verify which addresses are used in the network.

TS_AVTP 5 Hidden communication channel

Some fields of the AVTP header like the avtp_timestamp are ignored when its value cannot be confirmed as valid, as stated in the "time valid" field. Therefore, malicious nodes can set the validity flag bit to zero and use the respective value field to carry information, which by definition should be ignored by recipients. This is also the case for fields that require padding like the payload field of almost all of the ACF message types. In this case, a zero padding is expected and they are said to be ignored by receivers. Apparently, harmless message's fields could be used to hide data from one malicious node to the other inside the network.

TS_AVTP 6 Flooding attacks Data formats like the CVF can use fragmentation to send video streams that exceed the MTU. Fragmentation is a well-known method for performing Denial of Service (DoS) attacks where the attacker floods the recipient with multiple fragments without ever sending the last fragment.

TS_AVTP 7 TLV exhaustion

Data formats like ACF where multiple subtypes can be chained together as TLV messages require additional caution. Complex and long chains might overload the Listener causing DoS attacks.

TS_AVTP 8 Fake CRF talker

The CRF used to synchronize media streams coming from different sources can also be used to disturb the time synchronization. If a malicious node disseminates fake timing information.

5.3 THREATS ON AVDECC

TS_AVDECC 1 Entity Spoofing Entities use ADP announce messages to advertise the availability of an entity to the network. A malicious node might spoof fake entities into the network to confuse other nodes into thinking that more nodes have joined the network. By sending multiple spoofed messages, an attacker might cause nodes to overwrite valid entries on their Entities' table with invalid entries.

TS_AVDECC 2 Departure Spoofing

Fake departure messages (explained in Section 3.7.2.1) will cause the information of the entity under attack to be removed from the database of all of the entities in the network.

TS_AVDECC 3 Reconnaissance

Malicious nodes might monitor announcement messages to map other nodes availability and their capabilities. This allows nodes to better understand the environment and then produce crafted attacks based on what was discovered. Information such as manufacturer, model number, supported network protocols, system address, and system memory map can then be learned.

TS_AVDECC 4 Acquire/Lock spoof

Controllers might get long or short term exclusive access to entities by sending AVDECC Entity Model (AEM) acquire/lock commands. If a malicious controller does this before the valid controller, the latter will never get access to the entity. This is even worse if the "persistent" flag is set to true, causing the entity to deny immediately further requests by other controllers.

TS_AVDECC 5 Memory Manipulation

The AVDECC Enumeration and Control Protocol (AECP) AAF message type allow direct access to an Entity's memory/register. This format can be used to completely overwrite or extract the device's firmware if not protected.

TS_AVDECC 6 TLV Exhaustion

Some AECP message types like the AAF might carry a variable number of TLVs. Chaining a large number of TLVs can be used to hide advanced attacks. Processing a large number of TLVs might also compromise the Entity's execution and can be difficult for protection mechanisms to detect the masqueraded attacks.

TS_AVDECC 7 Flooding attack

The AECP High-bandwidth Digital Content Protection (HDCP) Interface Independent Adaptation (IIA) Authentication Protocol (AP) message type uses the concept of fragmentation to send content bigger than the 524 bytes, which is the limit of the protocol's length field, found in the header. Fragmentation is a well-known method for performing DoS attacks where the attacker floods the recipient with multiple fragments with different sequence_id, but without ever sending the last fragment.

Another possible flooding attack is using ADP entity discovery messages. If an entity discovery message is sent with entity id equals to 0, all available nodes will answer with an entity_available response. Fake messages can make nodes send multiple answer back to one target overloading it, as in the amplified attack described in TS_GPTP 4.

TS_AVDECC 8 Stream Spoofing

ACMP messages can be used to disarrange streams within the network. After knowing what is being streamed, attackers might disconnect streams from talker to listener. If the dropped stream is critical data, it can affect the safety of the system. Creation of unwanted streams is also possible.

TS_AVDECC 9 Man-In-The-Middle

By monitoring the network traffic, an attacker is able to intercept, read, copy, modify, replay and drop messages to its destination. This fact combined with the non-usage of authentication and integrity mechanisms (provided by MACsec, for example) might lead to different types of attacks.

Replay attacks are one example where a previously sent

message is re-transmitted. Replaying a message can be used to trigger a previously performed action in a system such as login, logout, object creation, deletion, update, and so on. Dropping messages is also a possibility and can be used to avoid messages reach a destination.

TS_AVDECC 10 Header Manipulation

An attacker sitting in the middle of a talker and listener can perform different attacks. Among those capabilities is editing packets. Nevertheless, this is also a capability that an attacker as a sender can have. Attackers might generate crafted packets with header's field containing inappropriate values. Causing implementations to behave in unexpected ways. As explained in Section 3.7.2.2, AEM offers an enormous variety of command messages to manipulate manageable objects of Entities. The headers of such commands can easily be exploited to carry designed values the will completely update an Entity's behavior.

5.4 THREATS ON CBS

TS_CBS 1 Shaper Compromising

As explained in Section 3.5.1, the CBS is responsible for guaranteeing that the bandwidth reserved by SRP (Section 3.4) is properly consumed. A malicious attacker can

compromise a device and use the operation mode of CBS to cause frames from other nodes to be delayed or dropped. In order for the attack to be successful, the attacker has to send multiple frames with the same priority (Same traffic class) of the targeted stream. The attacker might first make some reconnaissance of other nodes in the network by using one of the methods described in TS_AVDECC 3, as an example. After having found and controlled a talker, the attacker can start sending frames at its will. Bridges on the path will try to forward the frames but due to the congestion created, frames from other talkers are going to be delayed or if the queue grows too much they are going to be dropped.

6 PROPOSED MECHANISM

6.1 EXISTING MONITORING AND DETECTION TECHNIQUES

In enterprise networking, many IDS and other security solutions have been proposed from a variety of companies for protecting their network. Most of them have been proposed for monitoring the common TCP/IP stack. Solutions for automotive protocols also exist, but they are mainly focused on automotive consolidated protocols like CAN, LIN, FlexRay and MOST. Some companies have publicly said to offer security solutions also for the Automotive Ethernet stack, but they only talk about Scalable Service-Oriented Middleware over IP (SOME/IP) and Diagnostics Over IP (DoIP). An exception was the IDS developed by a company named EMBAS, which works using SNORT (SNORT, 2019) rules and supposedly also protects Ethernet AVB. Due to market competition, it is difficult to get access to detailed information on how these solution work. Providers try to keep their solutions confidential to keep some market advantage from its competitors. To the best of our knowledge, this is the first proposal for an IDS focusing on protecting the whole Ethernet AVB/TSN stack.

For multiple communication protocols, there is a variety of approaches to protect traffic. They can be generalized as the multi-pronged approach, as proposed by the IEEE Standards Association (SA). In this proposal, (IEEE SA, 2015) four main categories can be found:

- Integrated Security Mechanisms (Prong A);
- External Transport Security Mechanisms (Prong B);
- Architecture Guidance (Prong C);
- Monitoring and Management Guidance (Prong D);

The Prong A are security mechanisms where specific fields of the protocol in question are used to carry security information. One example is the encapsulation of an Integrity Check Value (ICV) within an additional option field or TLV, similar to what happens in IPsec. Prong B aims at security by using security mechanisms from protocols in lower layers such as IPsec and MACsec. Prong C is the mitigation of security threats by architectural modifications like using redundancy. Having redundant paths, for example, can make it more difficult for Man-In-the-Middle (MITM) attackers to drop or modify packets in the network. This is due to the fact that the attacker will not see the redundant packets sent. Consequently, the receiver is able to see if one packet is different from the other or if only one packet arrives. Prong D proposes security by the definition of an untrusted

data set, parameters, or messages that could be monitored in order to detect malicious activities.

6.2 BEHAVIOR AND ASPECTS SPECIFIC TO THE AUTOMOTIVE DOMAIN

The automotive industry is known for not using networking protocols with its complete set of features and to have restricted setups when compared to other networks. Examples of this is the usage of IPv4 without fragmentation or IPv6 without extension headers. When it comes to AVB/TSN networks, it is not different. Some characteristics that can be highlighted (TEENER; KIM, 2011):

- The network supports both automotive multimedia applications, critical control applications and best-effort traffic.
- Critical control applications send and receive messages that require fault-tolerant, low-latency communication.
- The number of endpoints is limited to approximately 32.
- The most time-sensitive applications use Class CDT which has a class measurement interval of 500μs, Class A streams have a 125μs class measurement interval. Class B have a 250μs class measurement interval. Best effort frames have the lowest priority.
- The number of hops in a path from talker to listener is limited to 7 for class A and B and 5 to Class CDT.
- Maximum end-to-end delay for Class CDT is 100ms, for Class A is 2ms and Class B 50ms.
- The network recovery time must be smaller than 100ms. This is measured from the detection of a failure event to the time that the network is ready to transmit data from applications.
- In general, a protocol like AVDECC is not mandatory for a minimal AVB/TSN network, but considering more dynamic networks with centralized configuration devices, as proposed in Section 3.6.3, it might be a suitable management protocol.

6.3 PROPOSED MONITORING AND DETECTION TECHNIQUES

In the enterprise context, NIDS have become known for providing many false positives. This is mainly due to the diversity of traffic and size of the networks (SOMMER; PAXSON, 2010). Automotive networks have grown hugely on the past few years, but in comparison to this previous context, automotive networks are considered pretty small and static. Normally, a high end car has around 90 ECUs. Newer devices are typically only introduced

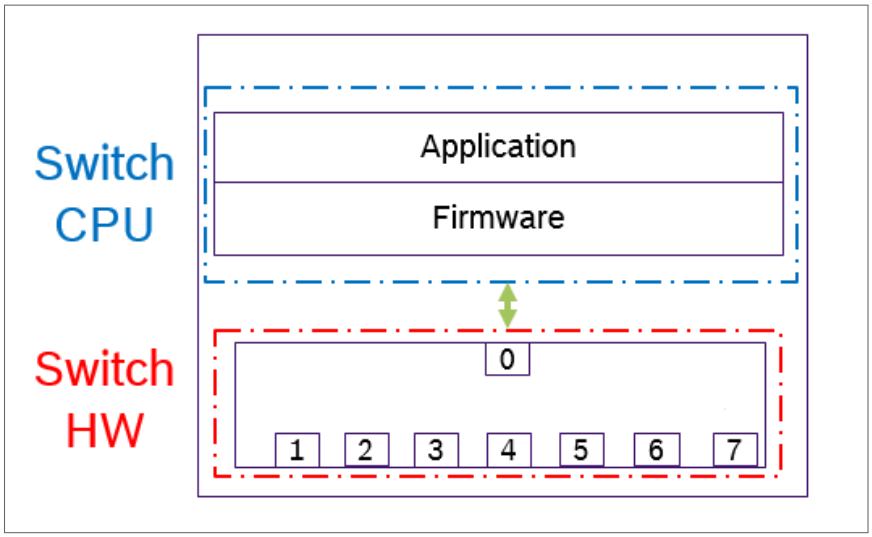
on a network when replacing a broken equipment and when a diagnostic session is running, which encourages the use of IDS systems on automotive.

An NIDS for monitoring AVB/TSN traffic has, in theory, good implications. Based on the deterministic behavior, where we know what traffic to await and when, it should be easier to detect differences on the expected and seen traffic shape. Combining determinism with the fact that in the automotive context we have a static network, we should be able to conceptualize a NIDS for AVB/TSN with good performance. As introduced previously for CBS and BLS, traffic is distributed to the network by shapers and normally controlled by credits. Credits are computed for each queue/traffic class as a function of the traffic specification, that is, reserved bandwidth, frame interval time, port transmission rate, among others. Knowing all of these parameters, it is possible to verify the conformance of configured and monitored traffic.

Since TSN is a set of standards that are being created to cope with different industries' use cases, it is expected that not all of them will necessarily be used together for automotive purposes. Consequently, it is necessary to define the subset of standards that are essential for automotive networks. These protocols must be investigated in order to list possible attack scenarios from the bad use of header's fields as well as threats from message types. It is really important to monitor data and management messages to detect intrusions, such as possible attempts to remotely read and modify the configuration of switches. An IDS for AVB/TSN should be able to verify the conformance of the configured and monitored traffic to find anomalies.

From the perspective of where such a monitoring solution should be deployed, a automotive switch present on an automotive central gateway will be considered as the target platform. A high-level schematic of the switch is showed on Figure 19. The IDS proposed is hosted in the Application layer of the Switch CPU.

Figure 19 – Automotive Switch



This choice is based on the fact that it interconnects many sub-domains in the auto-

motive Electrical/Electronic (E/E) Architecture and therefore has a broader view as the network as a whole. In this thesis a Proof of Concept (PoC) will be implemented after the Risk and Threat Analysis presented in 5. In the PoC, a set of proposed monitoring techniques will be evaluated using automotive Ethernet switches' features, in order to measure the overhead brought by the monitoring solution. An example of such feature are the Ternary Content-Addressable Memory (TCAM) rules that enable custom policing of packets beyond layer two. It is important to measure the impact on the deterministic behavior of the network and the minimum and end-to-end latency required by safety standards.

From the IDS types explained in Section 4.3, even though signature-based IDS are vulnerable to not recognize 0-day attacks, they have also the advantage of not producing false-negatives or false-positives. False-positives in handling automotive critical data can be dangerous because if critical data is confused with an attack and it is blocked then it can be harmful for passengers.

This types of detection systems are well explored in the literature and also have a rule-based filtering and a stateful packet filter. But, as they are normally designed for normal network which are very dynamic, the data structures responsible for keeping track of connections, are populated after the packet successfully passes the filter rules. In our case, since all connections are already known, the table keeps history of the connection has a fixed size.

As it should be clear by now, TSN is not one protocol but a set of them. Creating an IDS to watch a ABN/TSN capable network must then incorporate mechanisms to monitor every one of its building blocks. Therefore, the IDS must have five main pillars:

- Monitoring time synchronization traffic;
- Monitoring Stream Reservation traffic;
- Monitoring Transported data;
- Monitoring if traffic shape is compliant with reserved resources;
- Monitoring management data.

For the purposes of this thesis, however, SRP is not going to be analysed, since, so far, it is rarely used in automotive networks. Using the threats described in Chapter 5, some software requirements were derived. Such as fields, parameters, counters that should be monitored in order to detect and prevent such threats. In this section, only a few examples of different detection technique's categories covered by the implementation are going to be described.

6.4 STATELESS PHASE

This section will introduce a set of detection rules that are applied individually for each incoming packet without taking the network history into consideration, i.e. previously seen packets (stateless). The stateless phase is categorized in two types of rules according to their derivation method. The first one is derived from the protocol's definition and the second one is extracted from the network knowledge.

6.4.1 Protocol Specific Rules

These types of rules are conditions derived from the specification of each individual message type contained in each protocol's standard. For this reason, this type of detection method is also called "Specification-based" detection. This is a straightforward task but it is also challenging due to the number of different header formats and the commonalities among them. Grouping the common checks in order to reduce software size and complexity is what makes the task demanding. Examples of this type of rules are direct verifications that can be used to check most of the fields' values. It checks fields like the Version Number, for attacks that use older versions or length field, for messages that are indicating to have a size diverging from the actual size that was captured and so many others.

6.4.2 Trusted Knowledge Rules

The level of network-awareness present in in-vehicle networks is also used to derive an important set of rules. Knowing which devices are present in the network, their addresses, their supported protocols, and other properties, allowing the creation of multiple whitelists and blacklists. Such lists work as access control and they can prevent unauthorized devices from creating malicious communications in the network. In the proposed system, the trusted knowledge is represented as a set of multidimensional tables. An example of 1-Dimensional table is the list of authorized MAC addresses. The higher dimensional tables represent different types of relationships. Examples of 2-Dimensional tables are the communication pair between Source and Destination MAC addresses or the link between a MAC address and its gPTP clock identity.

6.5 STATEFUL PHASE

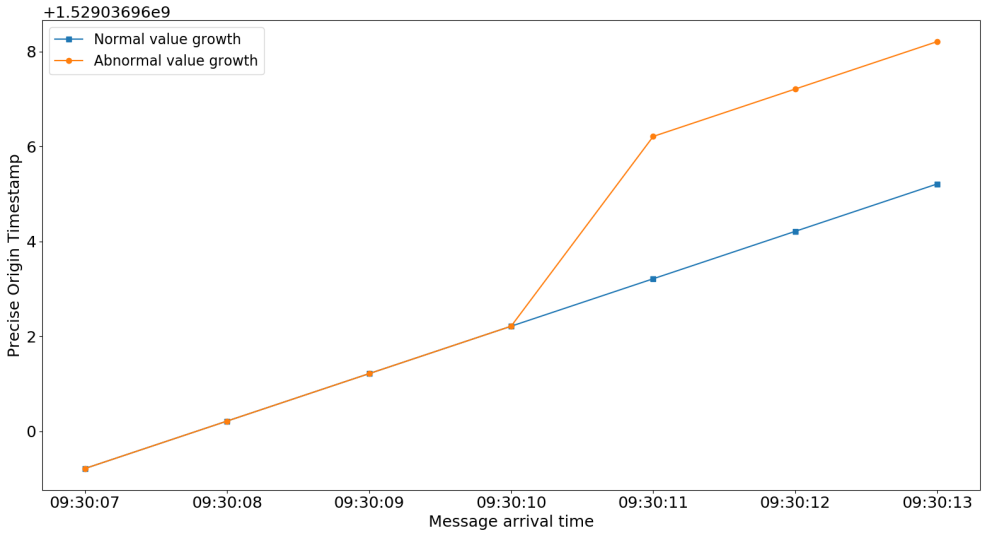
In contrast to the stateless phase, the stateful phase consists of a set of rules that takes into consideration properties that were saved from previously seen packets.

6.5.1 Threshold violation detection

This type of detection considers acceptable range of values for some fields to detect abnormal patterns. Values like the "Precise Origin Timestamp" of Follow-up Messages, "Request Receipt Timestamp" of Path Delay Request messages, or the "Response Origin Timestamp" of Path Delay Response Follow-up messages should grow almost linearly. Figure **??** illustrates the scenario of a normal (blue line with squares) and abnormal (orange line with dots) "Precise Origin Timestamp" growth.

In this example the during a normal gPTP time synchronization the time propagated by the Grandmaster grows in small and similar steps but suddenly, in the abnormal scenario, the value has a bigger step. This could indicate that a previous message was not delivered but also modified as a malicious activity. Monitoring two consecutive messages and verifying if the difference between is not bigger than an upper limit can be enough for the detection of such behavior.

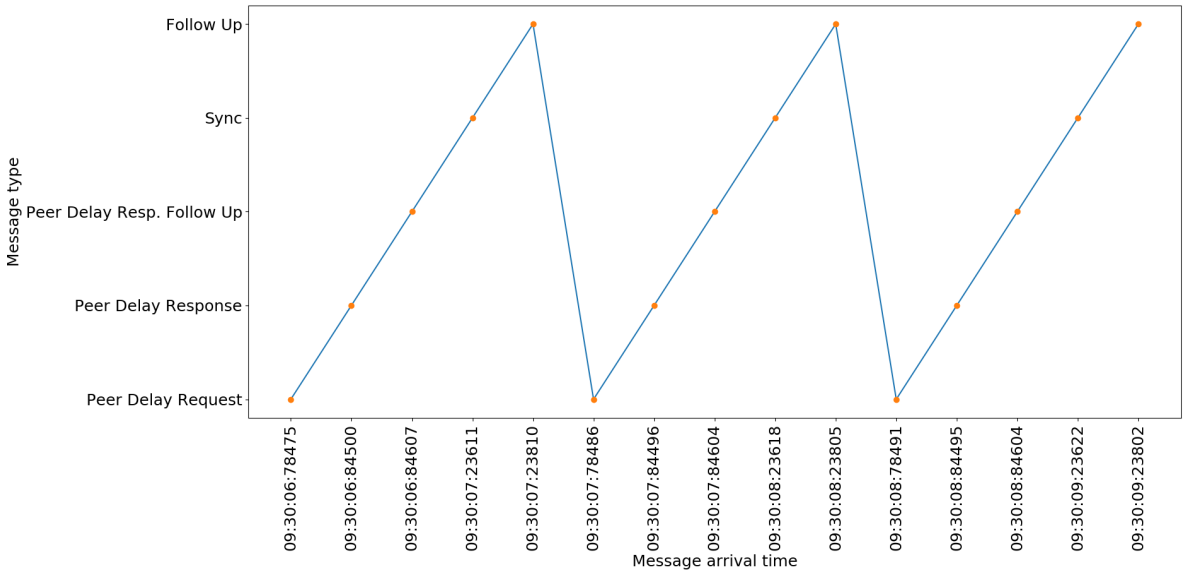
Figure 20 – gPTP:Follow-Up Precise Origin Timestamp



6.5.2 Frequency change detection

Some routines in the AVB/TSN protocol stack have a periodic behavior. Good examples of this type of behavior are the synchronization and delay measurement state machines in the gPTP protocol. The network grandmaster is periodically transmitting sync/follow-up messages to other nodes and the delay calculations between neighbors also happen in a periodic manner. This type of behavior is illustrated in figure 21, which contains the arrival timestamps of different message types. By calculating the difference between two successive packets it is possible to check for possible delay attacks.

Figure 21 – gPTP Message Frequency



6.5.3 Connection state monitoring

A characteristic of multiple protocols is a sequence of specific messages to achieve a specific connection state. The TCP protocol, for example, has famous the 3-way handshake as a connection establishment procedure. Having such a sequence of messages exchanged among peers allows monitoring systems to watch the sequence, timeouts, specific flags in some headers for malicious activities. This is also possible for different procedures in the AVB/TSN protocol stack. From the gPTP introduction in Section 3.2.1 and also in Figure 21 there is always a Synchronization message before a Follow-up message, they should have the same Sequence Number field and should come from the same device. Keeping track of this called connection state, as in a state-machine also improves the effectiveness of the IDS.

6.5.4 Shapers Software Requirements

A good way to assure good behavior of the traffic shaping is to apply the mechanisms proposed in the IEEE 802.1Qci. As stated in (BARTON; HENRY, 2018), the current standard does not specifically instruct how to integrate the Per-stream filtering and policing into the network but a good way is to use updatable rules into centralized devices. The setup proposed in the next section contains pretty much similar ideas as the ones detailed in the document. For this, two different kinds of buckets are used: normal buckets and leaky buckets. The first type of bucket is used to keep overall statics during the up-time of the IDS, counting the total number of packets, packets of each message type, etc. On the other hand, leaky buckets are used in different cases to detect the temporal behavior of some attacks. Leaky buckets are counters incremented for a certain event but decremented or reset after a certain period of time.

The predictability of some shapers like the TAS and the CS enables is also a feature that needs to be taken advantage. It allows the IDS to be configured to expect messages at specific points in time, coming from specific source and also with specific contents.

6.6 ANOMALY RESPONSE

Beyond detection requirements, the IDS must also meet some requirements regarding its reaction after an anomaly is detected. The IDS shall be able to combine the following actions if a condition is met.

- Drop: block and log the datagram;
- Block: simply block the datagram without the need to log;
- (De)activate: (de)activate a new rule based on some condition met;
- Log: Simply log the datagram and the condition met.

Dropping frames may not suffice at some point in order to prevent an ongoing attack. If an entity with a specific address misbehaves for a certain time, then it needs to be blocked. Obviously, this does not take any safety concerns into account, therefore, in order to apply this response on a real scenario, a safety evaluation must be previously performed.

It is important that it may be possible to combine these actions, e.g. to log datagram and to activate a new rule for future datagrams. For the next requirements, act might be replaced by any combination of the previously described actions.

Table 6 – Anomaly response requirement 1

SR_IDPS 1	The IDS shall offer the option to act all packets that: • Contradict the trusted knowledge • For which a timeout exceeds • If the maximum amount of memory has been allocated
Rational	Needed to be able to react on possible DoS attacks

Table 7 – Anomaly response requirement 2

SR_IDPS 2	The IDS shall offer the option to act on all traffic to and from a certain source if: • It contradicts the trusted knowledge a certain number of times per time unit, e.g 10 times in 1 minute.
Rational	Needed to be able to react on time spreaded attacks

Table 8 – Anomaly response requirement 3

SR_IDPS 3	The IDS shall offer the option to act on all traffic to and from a certain source if: • The average counters are exceeded by a certain source address for the n-th time • The leaky bucket counters exceed the bucket for the n-th time
Rational	Needed for monitoring regular usage of the network

Table 9 – Anomaly response requirement 4

SR_IDPS 4	The IDS shall offer the option to act all packets • Total number of received packets • Total number of analyzed packets • Total number of logged packets • Total number of dropped packets • Total number of times a rule was (de)activated
Rational	Allows additional log interpretation in the future to detect 0-day patterns or detection mistakes to trigger updates

6.7 IMPLEMENTATION SETUP

The security mechanism proposed in this thesis aims at being one of the first implementations to prove the concept described in (BARTON; HENRY, 2018). The referred work suggests that switches should have MAC filtering, security rules, Access Control List (ACL), etc. These rules are needed to ensure that the devices are protected to any misconfiguration that could lead endpoints to failures. The Proof-of-Concept developed in this thesis takes real automotive elements and assigns concrete rules to the generic ones detailed before.

For the role of a bridge, the Automotive Ethernet Switch whose manufacturer has been omitted due to NDA restrictions. The switch employs different features to enable the creation of secure automotive Ethernet solutions.

It is equipped with multiple Ethernet ports, among them Automotive Ethernet 100BASE-T1, 1000BASE-T1 port (1 Gb/s). two Ports are normal 100 Mb Ethernet ports (100BASE-TX), which allow the connection of ordinary PCs to configure the Switch. The Switch CPU, as highlighted in Figure 19 is an ARM Cortex M7 with 250 MHz of frequency. Between the Switch HW and CPU there is a 1 Gb/s internal Ethernet link.

Ethernet frames received by any switch port can be forwarded to the CPU. This can be done by the use of TCAMs. TCAM is a high-speed memory type that allows a complete search in only one clock. TCAMs make the process of Deep Packet Inspection (DPI) customizable. The switch has 48 or 96 byte TCAMs that can match an equivalent

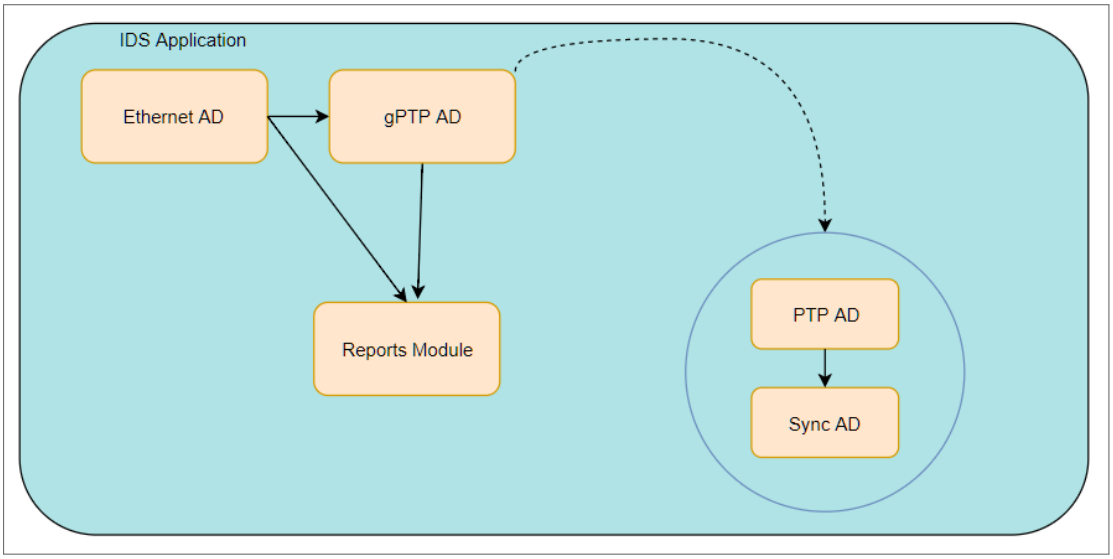
number of bytes, bit by bit, on every received packet. Upon having received a packet that matches a certain rule, a number of actions can be executed.

Every frame received by the switch is checked against different ACL. Such lists contain allowed communication mappings between two addresses for example, as well as other message content. Thus, it is possible to have a trusted knowledge for the contents in every field of the packet header. This model also supports Rate Limiting Mechanism (RLM) which were initially designed to guarantee QoS to high priority traffic. Nevertheless, RLM can be also used as a defense mechanism to limit or completely block data coming from one port that is trying to flood the network

6.8 IDS COMPONENTS

This section describes the internal IDS software components and state-machines, showing the processing flow throughout which all incoming packets are analyzed. The IDS is composed by single Anomaly Detector (AD) Components, where each is responsible for one specific protocol. Figure 22 illustrates the flow of a gPTP Follow-Up message as it is received by the switch.

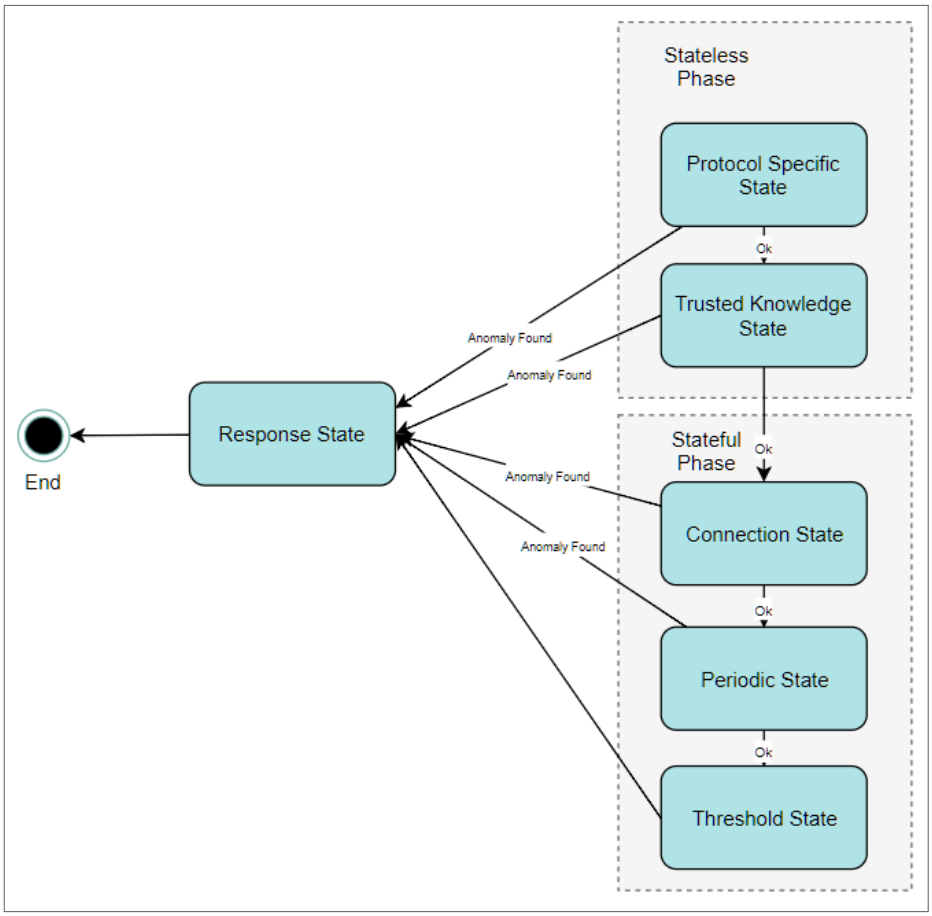
Figure 22 – IDS Architecture



Basically, each protocol has its own ADs, like the Ethernet and gPTP ADs, but these ADs themselves have smaller components for each message type. 22 does not shows all of the inner ADs of gPTP but highlights that the gPTP ADs contains the PTP ADs, which contains rules for the PTP Common Header (This module is shared by all other gPTP ADs like Sync, Peer Delay Request, etc) and the Follow-Up AD, which contains only rules for Follow-Up specific headers and TLVs.

Additionally, Figure 23 shows the state machine representing a single AD where the stateless and stateful phases as described in sections 6.4 and 6.5 respectively. From Figure 23 it is possible to see that if an anomaly is found during the evaluation of each state,

Figure 23 – Generic State-Machine



the processing stops and the response state is initialized, in order to take an appropriate action.

7 RESULTS

In this chapter, the performance of the proposed IDS is presented as well as the setup and tools used for obtaining the metrics. In total, four main test cases were conducted to evaluate the proposed IDS mechanism.

7.1 RESULTS

Testing the IDS is one of the most challenging parts, because it requires the generation of testing data in such a way that it tests every corner case. For AVB/TSN, this is really challenging because there is a need to create packets from specific nodes at specific times, creating a sequence of actions the will represent the state machines of the protocols. For testing the implementation of the IDS, we used two approaches, the offline approach and the online approach. The Offline approach is a way to create communication without using real devices and real connections. In this case, the network traces simulate the communication. On the other hand, in the Online approach the network is recreated in a realistic manner.

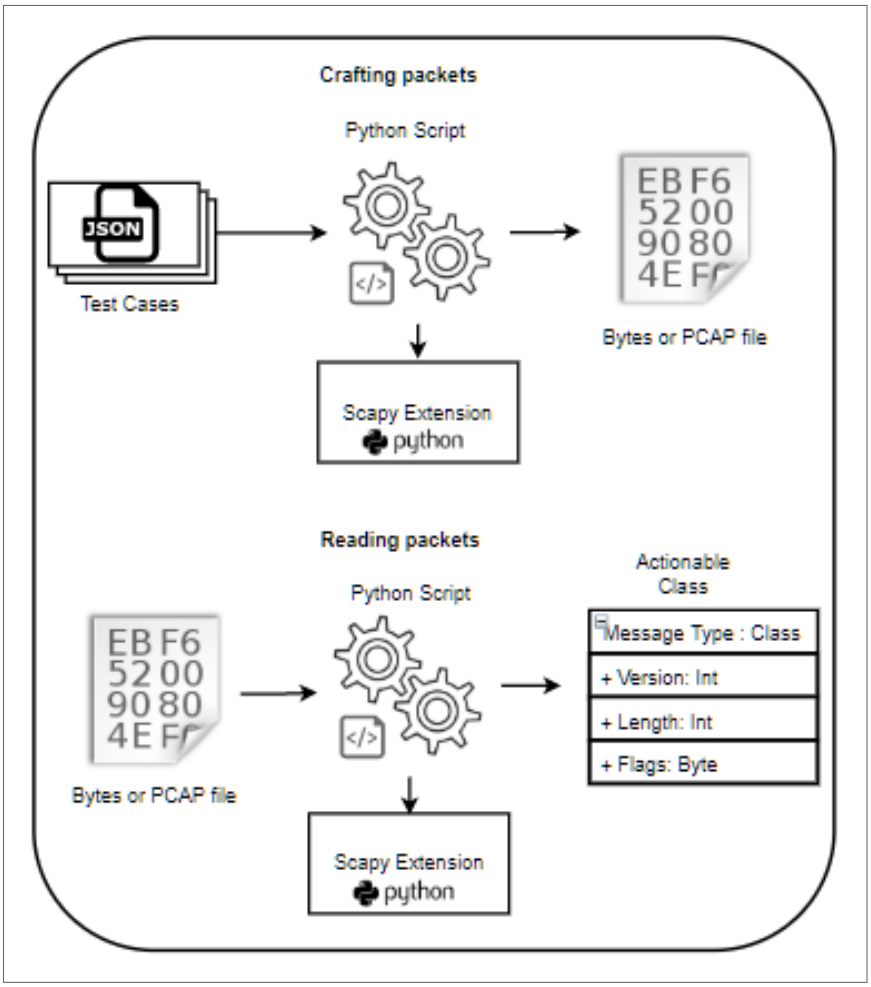
For the offline approach, an AVB/TSN packet generator was created using a Python library called Scapy (SCAPY, 2019). Scapy enables developers and testers to, among other functionalities, forge and dissect network packets in order to test network devices. The library is open-source and supports a wide range of protocols like Ethernet, IPv4/IPv6, TCP/UDP, HTTP, and so on. However, it does not support the creation of gPTP, AVTP or AVDECC frames and therefore, the different message types' structure of these protocols was created using the Scapy's building blocks.

This Scapy's extension allows the crafting of single packets and complete network communications. It reads a JSON file describing each field of each message header and generates either a PCAP file or sends the packet(s) to a chosen network interface. This testing phase worked as a way to check if all foreseen conditions would be correctly detected. So, the network traces contained packets which individually represented different conditions to be tested, for example, one packet with unknown protocol version, a sequence of packets representing an incorrect gPTP communications were generated. Additionally, the packet generator can also be used to read PCAP files parsing packet bytes to meaningfully understandable objects. Both features are illustrate on Figure 24.

Once all corrections were made and unreliable packets were being detected, the online phase began. This second testing phase was mainly distinguishable from the previous one by the use of real automotive equipment: The Automotive Switch, present in various production ECUs, and the Automotive Test Hardware Spirent C50.

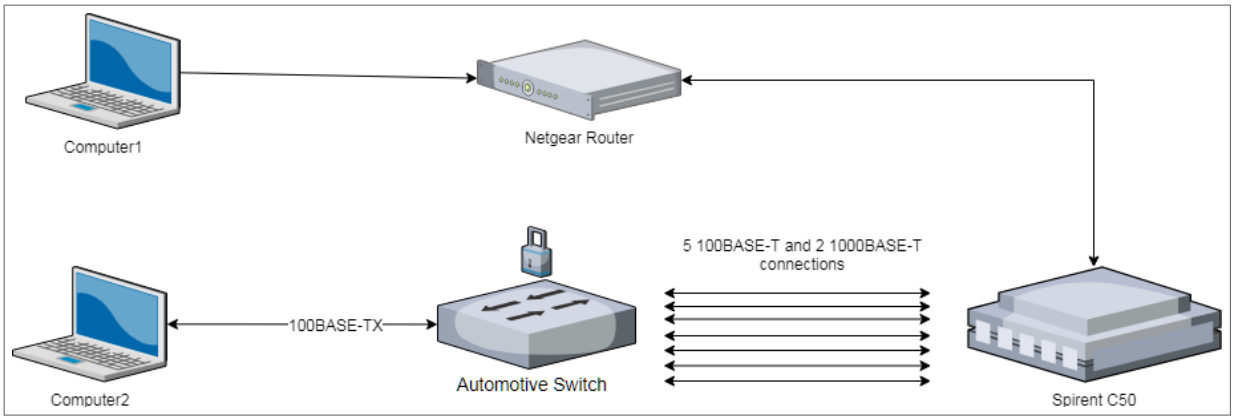
This Spirent equipment enables automotive developers from OEMs, Tier-1 suppliers

Figure 24 – Extension features



and other providers to perform different kinds of tests. Among these tests are: device and protocol testing, conformance and certification testing, security and vulnerability testing, etc. The equipment allows the simulation of up to 16 ports with different physical layers like 10/100/1000BASE-T.

Figure 25 – Test Setup



For the tests described here, four 100 Mb/s ports and two 1000 Mb/s ports of the

automotive switch were used. They were connected to the C50 equipment as in Figure 25, where each port can represent a time-aware ECU which also sources and/or sinks data. In this case, the developed Scapy extension was once again used so that its output was used to configure the traffic generator of the C50 equipment.

Getting frames inside the internal switch CPU to be processed by the IDS is the action triggered as the result of a TCAM rule match. In this case, six TCAM rules were created, each containing the MAC address of one of the C50's ports. Therefore, the automotive switch offers two ways to have this:

1. Having a packet copied from the switch ports;

2. Having packets forwarded from the switch ports.

Figure 26 – Test Scenarios

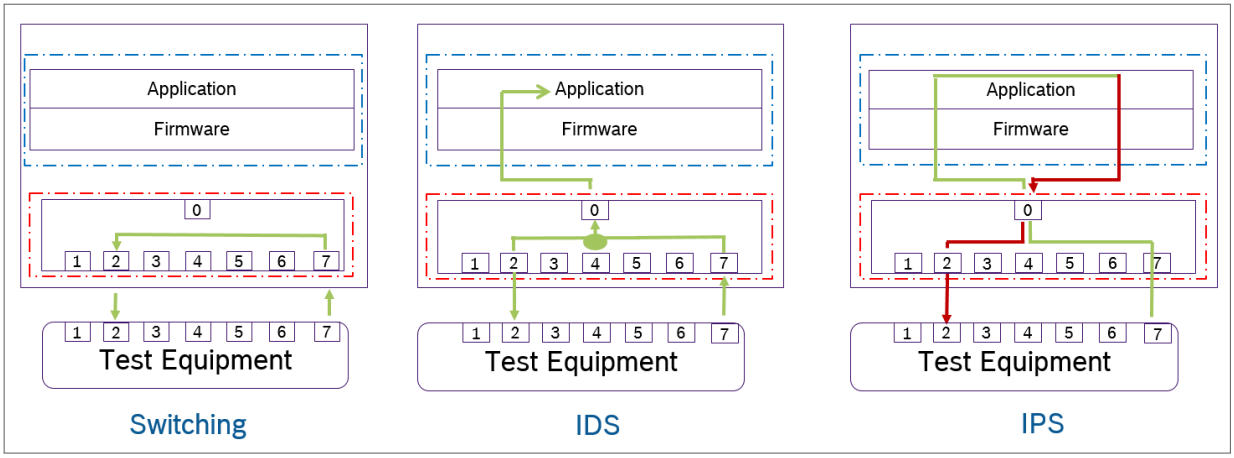


Figure 26 shows the three main test scenarios tested on this thesis. The first case (Switching) is the normal switch operation. In the second case (IDS) a copy of a packet is received by the switch and even before copying a packet is already forwarded towards the destination. In the third case a packet will only be forwarded to the destination after the application processes it. Thus, the difference is that the third case allows a packet to be dropped before being forwarded to the destination while the second one does not. This makes the third case suitable for a Firewall or an IDS with Prevention (IPS) features while the former fits the needs of an IDS that only monitors the network. Once the packet is in the CPU the IDS Application can interact with the switch firmware to get the packet as a buffer.

Knowing the path traversed by the packet from the switch hardware until it reaches the IDS Application allows the possibility to draw different scenarios to test the overhead of using the IDS. The tested scenarios are:

1. The switch working without moving any frames to the CPU (normal switch operation).

2. The difference between the forwarding and copying approaches.

3. The overhead added only by the IDS when compared to just receiving a packet in the CPU and directly releasing it.

These scenarios will allow the understanding of the CPU's impact on the normal switch operation and also to truly measure the impact of the IDS itself when the switch is already in use.

Back to Figure 25, computer1 has three purposes: it configures the C50 equipment with the correct parameters, creates the test scenarios and starts the tests. Computer2 is connected to the 100BASE-TX port of the automotive switch in order to receive report messages coming from the IDS. Every time a packet is received and analyzed by the IDS, it sends a report message to Computer2 specifying if any threat condition was met. Reports for the whole run-time of the IDS are also periodically sent.

7.2 TEST CASE 1: RFC 2544

The RFC 2544 (BRADNER, 1999) describes a methodology on how to benchmark networking devices and aims at measuring general network metrics like throughput, latency, jitter, and packet-loss rate, among others. For Test Case 1, six devices were used and they were talking in pairs send bidirectional data one to another. Two pairs of 100BASE-T1 capable devices and one pair of 1000BASE-T1 devices. The devices sent packets with size of 64, 128, 256, 512, 1024 and 1518 bytes of AVTP packets, representing media transfers inside the vehicle. The traffic load rate, i.e. how much of the total available bandwidth should be used, was incremented in steps of 10% of the maximum link throughput, i.e. 10 Mb/s for the 100BASE-T1 ports and 100 Mb/s steps for the 1000BASE-T1 ports. Then, it is possible to see in the following charts the individual results for each of the aforementioned properties. Each test combination had the duration of 30 seconds and an interval of 15 seconds was introduced between them. Table 10 summarizes the different scenarios where the aforementioned metrics were collected. There were three main types of scenarios, the first is called "Without IDS" this is the normal switch operation and for this case metrics about the 100 Mb/s and 1 Gb/s links were collected. The second and third scenario refer to the switch with the internal CPU enabled but the difference is that on the second scenario packets are copied or forwarded over 100Mb/s or 1Gb/s links to the switch firmware only and on the third scenario packets are copied or forwarded over 100Mb/s or 1Gb/s passing by the Firmware and going through the complete IDS Application.

	Copy		Forward			
	100 Mb/s	1Gb/s	100 Mb/s	1 Gb/s	100 Mb/s	1 Gb/s
Without IDS	Not Applicable		Not Applicable		✓	✓
Firmware	✓	✓	✓	✓	Not Applicable	
IDS	✓	✓	✓	✓		

Table 10 – Different setups tested

7.2.1 Latency and Jitter

There are multiple definitions of latency which differ depending on the type of device you are testing. Basically, latency is the difference between two timestamps of specific bits of a frame. These two bits are normally the first or the last bit to ingress the switch and the first or last bit to egress from the switch. Figure 27 shows two distinct methods of measuring latency. As it can be seen, the upper part of the figure shows the first timestamp being recorded at the last bit of the incoming frame and the second timestamp at the first bit to leave the switch. For that reason, this approach is called Last-In First-Out (LIFO). The lower part of the figure shows another approach called First-In First-Out (FIFO), that is, first timestamp at first bit ingressing and second timestamp at first timestamp egressing. Different combinations for specific purposes are possible, like Last-In Last-Out (LILO) or First-In Last-Out (FILO) (BRADNER, 1991) (NEWMAN, 2011).

The chosen method for measuring the latency in this experiment was the LIFO, which used for "Store-and-Forward" devices, that is, it measures the amount of time the switch keeps the complete frame inside it before forwarding.

Another important result that helps understanding the latency measured is jitter, which is defined as the variation in latency across multiple frames. From the results in Figure 29 and Figure 28, we can see that the results might suggest a significant impact on the latency and jitter of the Forwarding scenarios in comparison to the other scenarios. When comparing the normal operation, i.e. without IDS with the Copy approach, there are almost no differences in the results. Since for the Forwarding scenario the path of the packet inside the switch is considerably changed, it was expected that a difference would occur.

7.2.2 Loss Rate

Besides the impact of the forwarding approach already discussed for latency and jitter, the same impact was seen when measuring the loss rate. Figure 30 and 31 summarize the frame loss rates for different throughput rates. The lines represent the different packet sizes in bytes. The lines with circles are the from the experiment where the packets where forwarded until the IDS Application. The lines with triangles represent the test where packets were forwarded only until the firmware. The difference between the images is the

Figure 27 – Latency measurement comparison

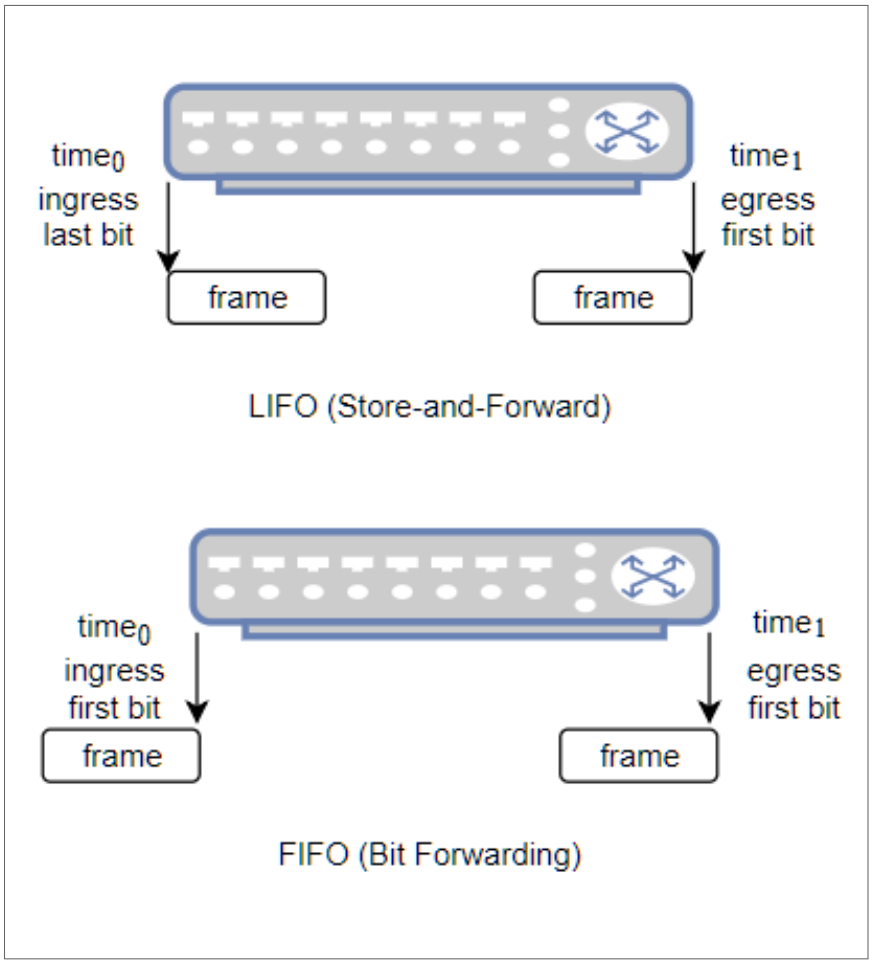


Figure 28 – Average Latency by Test Type

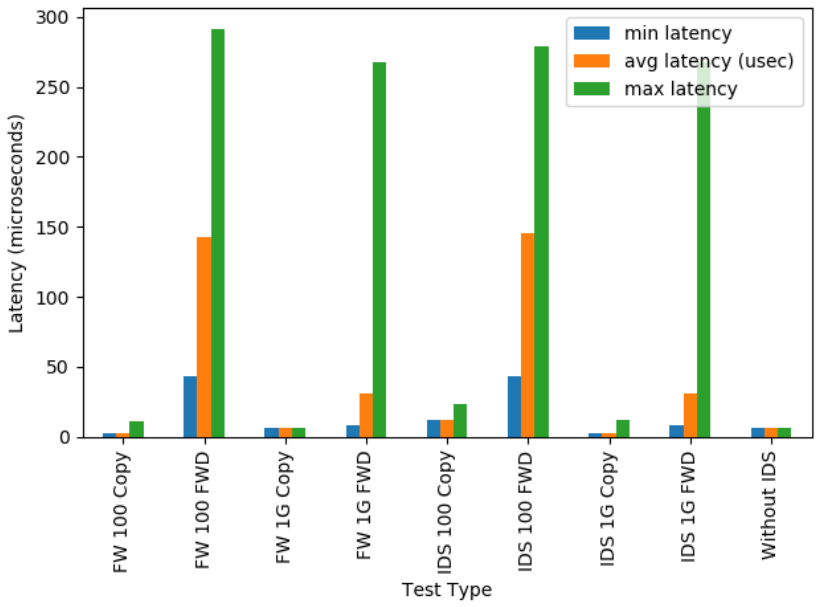
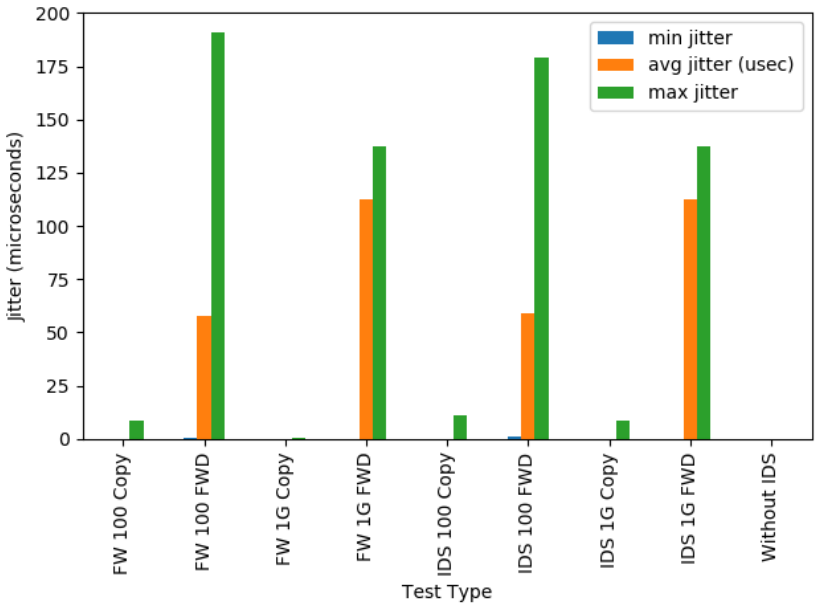


Figure 30 represents the results of a test running over a 1 Gigabit/s link and Figure 31 shows the same test over a 100 Mb/s link.

Figure 29 – Average Jitter by Test Type



In general, the bigger packet size is, the smaller loss rate is. This comes from the fact that with bigger packets, less packets per second are sent as it will be confirmed in section 7.4. Another interpretation is that the IDS Application alone does not add a bigger overhead to the switch as, in many cases, the lines for the same packet size in the two different scenarios are overlapping and on a few cases the difference is minimal. The loss rates in the scenarios where packets where copied as well as when the CPU was disabled were 0% for all packet sizes.

Figure 30 – Loss Rate: IDS vs Firmware 1Gb/s link

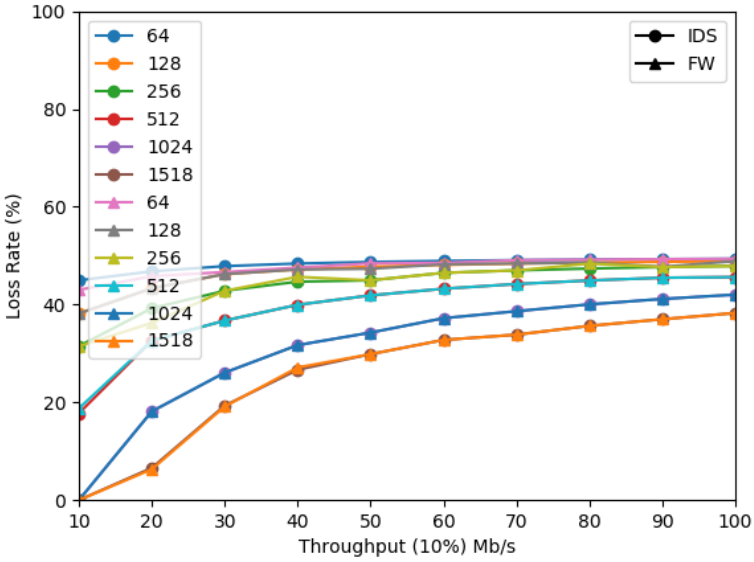
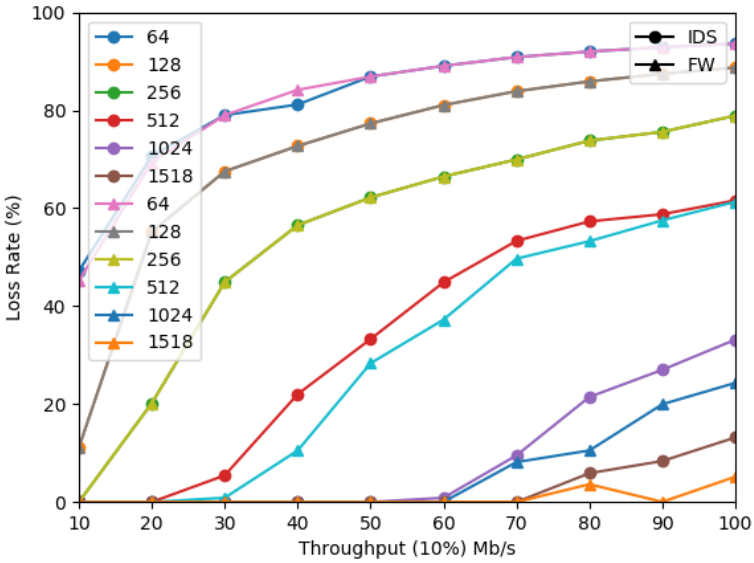


Figure 31 – Loss Rate: IDS vs Firmware 100Mb/s links



7.3 TEST CASE 2: TIME SYNCHRONIZATION WITH GPTP

The second test case was to run gPTP over 10 minutes in the different scenarios explained before to see the differences between them. It was a personal choice to run the test for 10 minutes because it produced a good amount of traffic as input for the IDS. Since the results from previous test cases did not show almost any impact from the IDS Application in relation to the switch firmware, the following tests only considered the scenarios with IDS. The test consisted again of the same six devices as the last test case.

In this setup, one 100BASE-T1 device was chosen as the Grandmaster and the others were all slaves. Neither signaling nor announcement messages were used and the synchronization period was set to 100ms. In the results shown in Figure 32, it is possible to see the mean, minimum, maximum and average path delay between the master and slaves. Figure 33 shows the mean offset, i.e. the difference between the master clock and each slave clock. The chart shows the negative and positive peak offsets during the course of the experiment as well as the current offset at the end of the experiment. As already observed in the previous test case, there is a notorious impact when packets are forwarded to the CPU when compared to the copy approach.

These results show that both the path delay and the clock offset in the Forwarding scenarios have a bigger variance when compared to the scenario without IDS and with IDS but copying packets from the ports, which is practically none. Since the IDS also keeps track of such values in order to find anomalies, this test is also important because it provides a range of good values to configure the IDS minimizing false negatives. Even having these metrics impacted by the use of the IDS, the current offset between master and slaves is still as expected, being less than 1 microsecond.

Figure 32 – gPTP Mean Path Delays

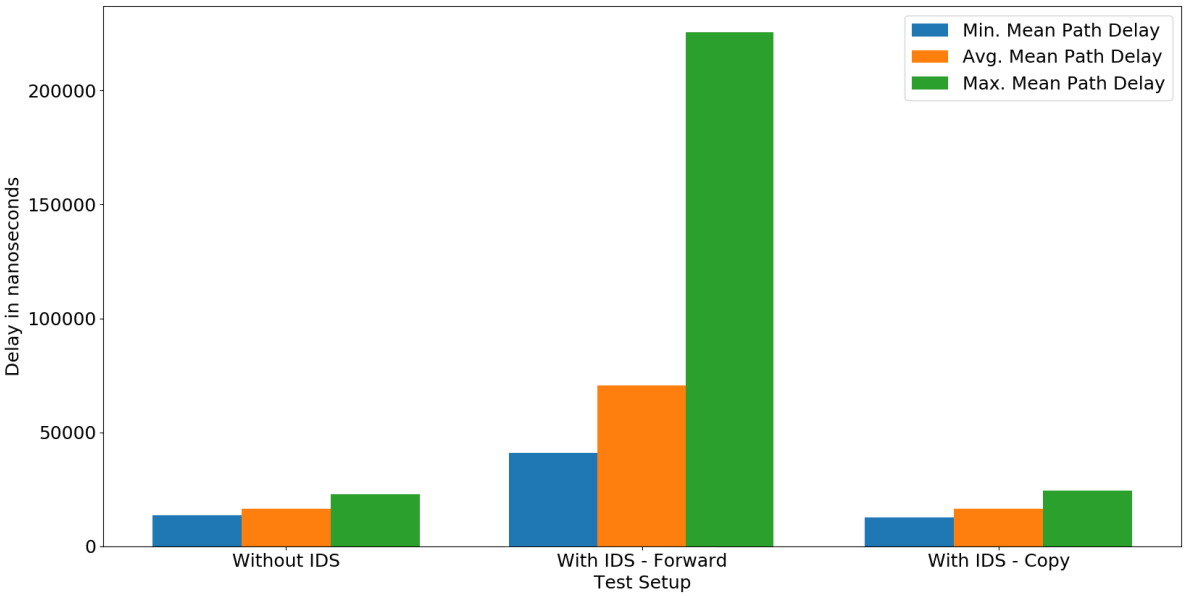
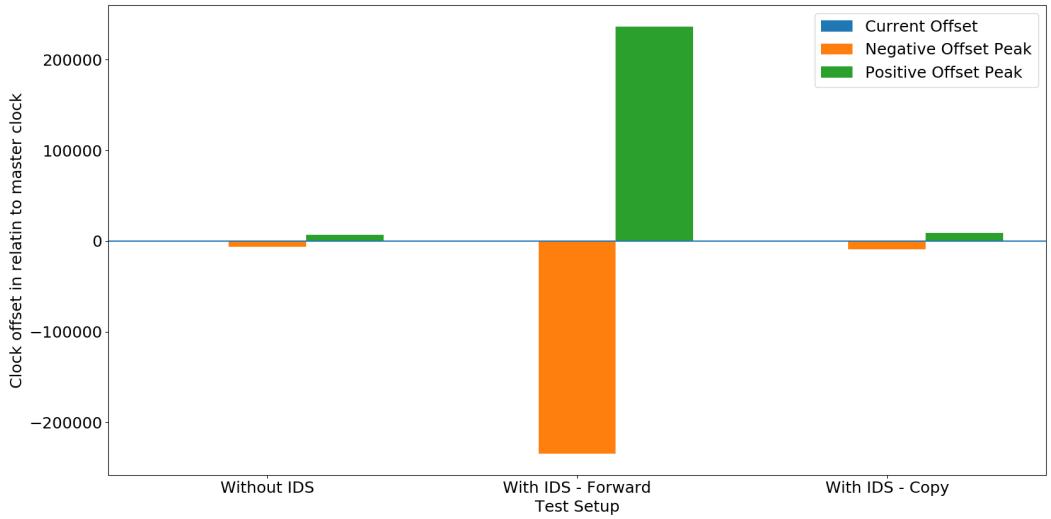


Figure 33 – gPTP Mean Offsets



7.4 TEST CASE 3: IDS CAPACITY

From the introduction to IDS in chapter 4, it was explained that there are two main types of detection techniques for IDS: statistical and content matching detection. When it comes to evaluating an IDS, metrics like attack detection rate (accuracy), false negative rate (FNR), false positive rate (FPR) are normally collected. FNR and FPR reflect to a good extent how well the detection algorithm is, because after the inspection is complete, the likelihood of the packet being malicious will be given as output. For content matching detection, like the one proposed in this master thesis, such metrics are not really relevant

since the IDS is going to detect based on the rules configured.

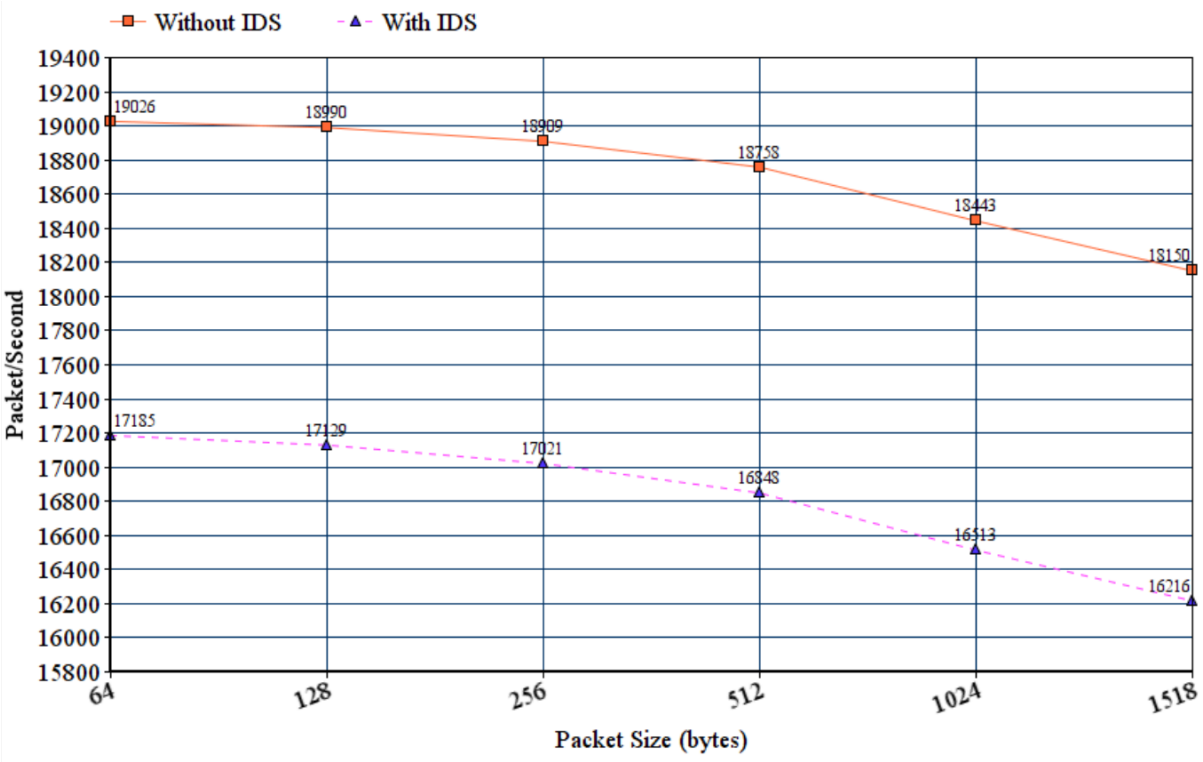
So, to evaluate this IDS, the maximum number of packets processed per second was measured. For this, only packets without any anomaly were sent; this makes sure the packet will flow through the complete state machine of the IDS and the result will better represent the maximum capacity of the CPU.

In total, 20.000 packets with sizes of 64, 128, 256, 512, 1024 and 1518 were sent in one second from only one port. For each trial, a 32 bit counter was kept in the IDS application and for every received packet the counter was incremented. After 5 seconds without receiving any packet (to make sure no remaining test packets would arrive), a report packet was sent with the number of packets processed by the switch and the counter was reset. The parsing capability of the testing tool introduced in Figure 24 allowed to read the packets and get the throughput for each scenario.

The throughput, as expected, decreases as the packet size increases and comparing the scenarios with and without IDS, there is a difference of 2.840 packets between them. The individual results can be seen on Figure 34.

It is very important to have these results because they allow to security architects to properly scale the IDS settings to the network needs. On AVB/TSN networks, there are configurations that will imply on a higher or lower number of messages exchanged by nodes. Examples of such configurations are the intervals at which nodes will recalculate their path delay to its neighbors and synchronize their clocks, another implication is the choice between one-step and two-step synchronization. More frequent intervals as well as two-step synchronization will lead to more messages being exchanged in one second. If more messages are exchanged in one second than the CPU is able to handle, malicious packets might flow undetected.

Figure 34 – CPU Capacity



7.5 TEST CASE 4: IDS DETECTION TIME

The third test case consists on measuring how long it takes to process a frame. For the processing time measurement, individual ACMP, ADP, AECP, gPTP, AVTP messages were sent. Since when an anomaly is found, the packet stops being processed, only packets without anomalies were sent. These frames did not raise any alerts in order to make sure the frame would go through every possible check and thus measuring the longer processing time for that protocol type. Six different timestamps were recorded for every received packet: Before the IDS started and after it finished, before the Ethernet Anomaly Detector started and after it finished and also before the Protocol Anomaly Detector started and after it finished. These timestamps were sent in UDP Packets to Computer1 where Wireshark was used to capture packets from the Ethernet interface. Again the Python script was used to read the individual Packet Capture (PCAP) files and calculate the difference between the three pairs of timestamps. The mean processing time of these three code sections can be seen in Figure 35. The average processing time within the whole test-case was 176 microseconds.

The result shows that a great proportion of the processing time is not by the detection itself, which means that tasks like parsing the packets, creating reports is consuming too much time and must be optimized in the future. The chart also shows a considerable difference in the processing time of Ethernet AD in AVTP/AVDECC messages in comparison to gPTP messages. This is due to the presence of VLAN tags in the first one

which leads to additional rules to be verified. As expected, message types with big header sizes also implied in longer Protocol Detection Time. Announce messages, which contain a big header plus a TLV have a longer detection time when compared to Sync messages, which have a minimal header. To better understand the impact of the number of rules evaluated (Directly related to header size) two additional results were added: the first, in Figure 35, shows the Processing Time by rule/conditions evaluated and the Second is the Processing Time byte. Those can be seen in Figure 36

Figure 35 – Processing time by message type

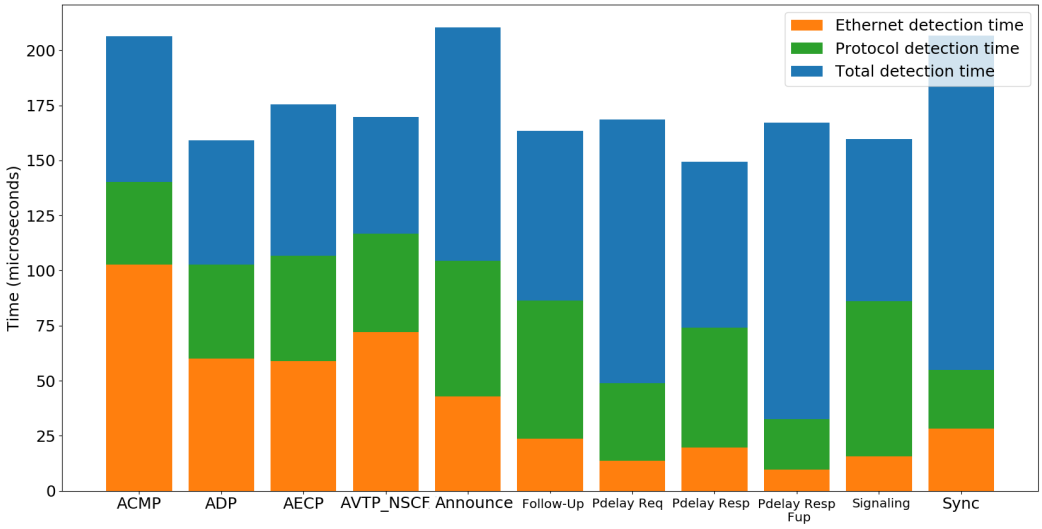
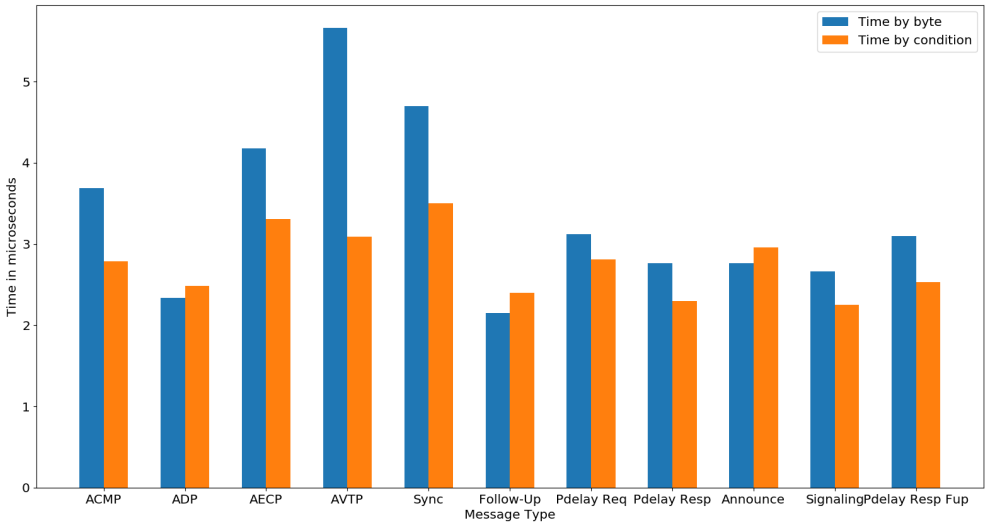


Figure 36 – Processing time by byte and condition



8 CONCLUSION

In this final chapter, we start wrapping up, in a section entitled recommendations, by discussing a few lessons learned throughout the completion of this thesis. Right after, a section called future work considers how this work could be further developed. Finally, a conclusion section highlights the main take-aways of this work.

8.1 RECOMMENDATIONS

In this work, because of the lack of 802.1Qci in the equipment available, its features were also implemented in software. It is expected though, that the next generation of automotive switches will have this standard implemented as a hardware feature. Having it in Hardware (HW) will mean faster processing time, but a few considerations should be made. It would be nice for security implementations to have interfaces exposed to the software side. By having such, security mechanisms like the IDS proposed in this thesis, could access which port, which stream, and timestamps at the time when the reserved resources were misused. This would enable our IDS to capture this information multiple times during the run-time of the switch and conclude whether an intrusion is happening or not.

The choice of only using an IDS only, to protect the network, was by the lack of MACsec support by the available equipment. Otherwise, it is completely advised to use IDS together with cryptographyc protocols. MACsec would solve multiple problems like preventing man-in-the-middle attacks, etc. But the IDS has different capabilities, like monitoring which type of media are valid on the network, which identities are valid for certain nodes, which devices are allowed to communicate with each other, detecting unusual amount of messages on the network to prevent DoS.

Another possibility is to have multiple instances of the IDS distributed in the network. A Distributed IDS (DIDS) would allow for a central agent to receive logs from the other distributed agents and confirm the presence of malicious nodes. Of course this should be also focus of intensive studies, because adding new agents to the network and a communication protocol between them can introduce new attack vectors to be exploited.

For the current lack of SRP usage in automotive networks, as well as in the near future, was SRP left aside. Nevertheless, this protocol can have serious security implications, therefore, all the offerings and requests should be monitored.It is important that the IDS must be deployed in multiple central points of the network, otherwise it is possible that messages won't be seen by it.

Knowing the complete packet structure for each protocol is also needed so the IDS is able to deeply inspect every received frame. But some of the standards, like the IEEE

1722.1-2013 lack a detailed explanation of the value of some fields on specific cases. For example, during the description of the ACMP, we are able to see 14 different message types, but it is not possible to clearly identify if all fields are used for all of these message types. For example, the fast connect flag is also used for fast disconnect procedures? During getState command messages, can it be assumed that Stream ID, Stream Destination MAC and Stream VLAN ID are 0 and filled only in response messages? Also, are getState command messages issued only by controllers? Can two Stream IDs be transmitted to the same destination MAC Address These types of questions are not clearly addressed in the standard and such level of detail is important for an IDS to know what is normal and what is not.

During the IDS implementation, many decisions were made to fill some gaps in the standards. Therefore, a good recommendation for improvements and configuration of the IDS is to take a look at real implementations and real traffic captures to be able to see, for example, default values used for some fields. This is advised in order to improve the correctness of the IDS and provide less false-positives.

8.2 FUTURE WORK

There are multiple additional challenges brought by the introduction of deterministic networks in in-vehicle communications. Still in the field of security, a possible future work is to evaluate how an IDS for AVB/TSN can be deployed in other locations of the architecture. Two examples of this is an IDS running on one single end node, like an ECU as well as on multiple ECUs, i.e. distributed IDS.

The presence of MACsec and the proposed IDS together to protect the network is also a possible future work. This would infer in additional changes to the IDS that should be capable of decrypting messages in order to inspect all the fields. This feature would mean that the device should be able to handle multiple keys, one for each link. Having to decrypt frames to inspect would also introduce additional processing time that reduces the performance of the system. Two possible future work could be:

- The adaptation of this proposal for networks already protected with MACsec.
- Impacts of MACsec in the performance, how much more computing power should nodes have in order to smooth the performance impacts. Financial implications of such.

Some of the TSN standards, like the new version of the time synchronization protocol (IEEE 802.1AS-Rev) and the IEEE 802.1CB Frame replication and elimination, expect that the same frame is going to be sent twice in the network. For such non-fault tolerant use cases the IDS needs to be positioned in strategic parts of the network such that it can detect the duplicate frames at distinct points and check for malicious activities.

As previously mentioned, dropping or blocking frames are possible actions to anomaly detection, however this does not take any safety requirements into concern. Thus, another possible future work is regarding safety evaluation on IDS anomaly response. Such work should research safety assessment of actions before dropping packets in order to mitigate potential danger to passengers.

As recommended in the previous section, taking a look at real implementations, and traffic captures is highly advised. This improves the correctness of the IDS and can also help on having realistic workloads for testing the detection mechanism. Since different implementations might have taken different decisions to fill the lack of details in the standards, it is possible that different fields have different default values. For this reason, having an automated tool to read network traffic and identify default values in use would be extremely helpful.

8.3 CONCLUSION

Automotive in-vehicle networks are still evolving and the obscurity placed from the OEMs has until some extension protected cars from being targeted sooner. This is related to the use of automotive specific protocols, hardware and so on, but now we see a bigger convergence as protocols used in the IT world are gaining space in the automotive industry. As more protocols like Ethernet join vehicles, hackers with IT expertise are also going to have more knowledge about automotive technologies and this will increase the number of attacks. This number will also be increased by newer simulation tools, hardware devices, etc.

In this thesis the security of some chosen AVB/TSN protocols were analysed. With that, the attack vectors, how attacks can be performed and its results were described. The identified threats were them used to derived requirements for designing an Intrusion Detection System (IDS). Such monitoring mechanism has be shown of very importance on protecting vehicles from being targeted. For this reason, this work presents an innovative Proof-of-Concept implementation showing how it could be used. The implementation adopts real automotive tooling like switches, development stacks and testing devices.

In order to support the implementation and not less important, a series of testing cases were performed. This testing phase included the development of a packet generation tool that can be used to generate packets of different protocols' message types. It also enables reading network capture files, useful for creating automated tasks that helps understand by extracting insights from unstructured data. With the help of a Spirent's testing device, a real networking scenario was reproduced to evaluate the IDS. The combination of both the packet generator and the Spirent equipment allowed to perform a kind of fuzz testing. The results show the actual potential of the IDS.

While it might not look harmful or a really elaborated attack, crafting messages can really lead to destructive consequences. Just by the time of this writing, Microsoft pub-

lished a vulnerability found on a Windows Dynamic Host Configuration Protocol (DHCP) Client the leads to the attacker to be able to perform Remote Code Execution on the target. (CVE-2019-0547, 2019) affects Windows 10, which is the latest Operating System from Microsoft by the time of this writing, and works by corrupting the targets memory upon sending modified DHCP responses.

By conducting such test cases, it is possible to differentiate what you think it is on your network and what is actually in use. Networks are configured by humans, and even experts make mistakes. It is highly possible that something is missconfigured or should not be set up in the network. Many people associate threats only with fancy combination of malicious activities or 0-day attacks, but even small details should not be ignored. Persistency even on the less likely threat will get attackers in control of the network. Therefore, even vectors that are easy or less risky should not be underestimated. Another fact is that many people also underrate the scenario where the attacker is already in the network. Monitoring tools such as the IDS proposed can detect attackers who are inside of the network and prevent them from being even more harmful.

REFERENCES

1733, I. Ieee standard for layer 3 transport protocol for time-sensitive applications in local area networks. *IEEE Std 1733-2011*, p. 1–21, April 2011.

802.1QBV, I. Ieee standard for local and metropolitan area networks – bridges and bridged networks - amendment 25: Enhancements for scheduled traffic. *IEEE Std 802.1Qbv-2015 (Amendment to IEEE Std 802.1Q— as amended by IEEE Std 802.1Qca-2015, IEEE Std 802.1Qcd-2015, and IEEE Std 802.1Q—/Cor 1-2015)*, p. 1–57, March 2016.

ANTONATOS, S.; ANAGNOSTAKIS, K. G.; MARKATOS, E. P. Generating realistic workloads for network intrusion detection systems. *ACM SIGSOFT Software Engineering Notes*, v. 29, n. 1, p. 207, 2004. ISSN 01635948.

BARTON, R.; HENRY, J. Management of IEEE 802.1Qci Security Policies for Time Sensitive Networks (TSN). p. 1–6, 2018.

BELLO, L. L. The case for ethernet in automotive communications. *SIGBED Rev.*, ACM, New York, NY, USA, v. 8, n. 4, p. 7–15, Dec. 2011. ISSN 1551-3688. Available at: <http://doi.acm.org/10.1145/2095256.2095257>.

BERGER, R. *Roland Berger*. 2015. Available at: <https://www.greencarcongress.com/2015/07/20150729-berger.html>.

Bosch, R. *BOSCH Automotive Handbook*. [S.l.]: Robert Bosch GmbH, 2018. (Bosch Handbooks). ISBN 9780768095678.

BRADNER, J. M. S. *Benchmarking Methodology for Network Interconnect Devices)*. [S.l.], 1999. 1-56 p. Available at: <https://tools.ietf.org/html/rfc2544>.

BRADNER, S. *Benchmarking Terminology for Network Interconnection Devices*. [S.l.], 1991. Available at: <https://www.rfc-editor.org/rfc/rfc1242.txt>.

CHECKOWAY, S.; MCCOY, D.; ANDERSON, D.; KANTOR, B.; SHACHAM, H.; SAVAGE, S.; KOSCHER, K.; CZESKIS, A.; ROESNER, F.; KOHNO, T. Comprehensive experimental analyses of automotive attack surfaces. In: WAGNER, D. (Ed.). *Proceedings USENIX Security 2011*. USENIX, 2011. Available at: <https://stevecheckoway.github.io/papers/car2011>.

Christian Boiger. *Pdelay – IEEE P802.1AS-Rev vs IEEE P1588-Rev*. 2019. <http://www.ieee802.org/1/files/public/docs2017/as-boiger-pdelay-802-1AS-vs-1588-0717-v01.pdf>. Online; 19/04/2019.

CVE-2019-0547. 2019. Available from MITRE, CVE-ID CVE-2019-0547. Available at: <https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-0547>.

Decotignie, J. The many faces of industrial ethernet [past and present]. *IEEE Industrial Electronics Magazine*, v. 3, n. 1, p. 8–19, March 2009. ISSN 1932-4529.

IEEE 1722. Ieee standard for layer 2 transport protocol for time sensitive applications in a bridged local area network. *IEEE Std 1722-2011*, p. 1–65, May 2011.

IEEE 1722.1. Ieee standard for device discovery, connection management, and control protocol for ieee 1722(tm) based devices. *IEEE Std 1722.1-2013*, p. 1–366, Oct 2013.

IEEE 802.1AS. Ieee standard for local and metropolitan area networks - timing and synchronization for time-sensitive applications in bridged local area networks. *IEEE Std 802.1AS-2011*, p. 1–292, March 2011.

IEEE 802.1BA. Ieee standard for local and metropolitan area networks–audio video bridging (avb) systems. *IEEE Std 802.1BA-2011*, p. 1–45, Sep. 2011.

IEEE 802.1CB. Ieee standard for local and metropolitan area networks–frame replication and elimination for reliability. *IEEE Std 802.1CB-2017*, p. 1–102, Oct 2017.

IEEE 802.1Qat. Ieee standard for local and metropolitan area networks–virtual bridged local area networks amendment 14: Stream reservation protocol (srp). *IEEE Std 802.1Qat-2010 (Revision of IEEE Std 802.1Q-2005)*, p. 1–119, Sep. 2010.

IEEE 802.1Qav. Ieee standard for local and metropolitan area networks– virtual bridged local area networks amendment 12: Forwarding and queuing enhancements for time-sensitive streams. *IEEE Std 802.1Qav-2009 (Amendment to IEEE Std 802.1Q-2005)*, p. 1–72, Jan 2010.

IEEE 802.1Qbu. Ieee standard for local and metropolitan area networks – bridges and bridged networks – amendment 26: Frame preemption. *IEEE Std 802.1Qbu-2016 (Amendment to IEEE Std 802.1Q-2014)*, p. 1–52, Aug 2016.

IEEE 802.1Qcc. Ieee standard for local and metropolitan area networks–bridges and bridged networks – amendment 31: Stream reservation protocol (srp) enhancements and performance improvements. *IEEE Std 802.1Qcc-2018 (Amendment to IEEE Std 802.1Q-2018 as amended by IEEE Std 802.1Qcp-2018)*, p. 1–208, Oct 2018.

IEEE 802.1Qci. Ieee standard for local and metropolitan area networks–bridges and bridged networks–amendment 28: Per-stream filtering and policing. *IEEE Std 802.1Qci-2017 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qca-2015, IEEE Std 802.1Qcd-2015, IEEE Std 802.1Q-2014/Cor 1-2015, IEEE Std 802.1Qbv-2015, IEEE Std 802.1Qbu-2016, and IEEE Std 802.1Qbz-2016)*, p. 1–65, Sep. 2017.

IEEE PC37.238. Ieee draft standard profile for use of ieee 1588 precision time protocol in power system applications. *IEEE PC37.238/D21.2, January 2017*, p. 1–40, Jan 2017.

IEEE SA. *Subcommittee S: Security SC standing document v5*. 2015. Available at: <https://ieee-sa.centraldesktop.com/1588/file/39406935>.

IEEE Standard for Local and metropolitan area networks-Media Access Control (MAC) Security. *IEEE Std 802.1AE-2018 (Revision of IEEE Std 802.1AE-2006)*, p. 1–239, Dec 2018.

ISO-27000 Information technology – Security techniques – Information security management systems – Overview and vocabular. Geneva, CH, 2018. Available at: <https://www.iso.org/standard/73906.html>.

Itkin, E.; Wool, A. A security analysis and revised security extension for the precision time protocol. *IEEE Transactions on Dependable and Secure Computing*, p. 1–1, 2017. ISSN 1545-5971.

ITU G.8265.1. G.8265.1 : Precision time protocol telecom profile for frequency synchronization. *ITU G.8265.1, July 2014*, Jul 2014.

Koscher, K.; Czeskis, A.; Roesner, F.; Patel, S.; Kohno, T.; Checkoway, S.; McCoy, D.; Kantor, B.; Anderson, D.; Shacham, H.; Savage, S. Experimental security analysis of a modern automobile. In: *2010 IEEE Symposium on Security and Privacy*. [S.l.: s.n.], 2010. p. 447–462. ISSN 2375-1207.

MATHEUS, K.; KöNIGSEDER, T. *Automotive Ethernet*. 2. ed. [S.l.]: Cambridge University Press, 2017.

MIZRAHI, T. Security requirements of time protocols in packet switched networks. *RFC*, v. 7384, p. 1–36, 2014.

Moreira, N.; Lázaro, J.; Jimenez, J.; Idirin, M.; Astarloa, A. Security mechanisms to protect ieee 1588 synchronization: State of the art and trends. In: *2015 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS)*. [S.l.: s.n.], 2015. p. 115–120. ISSN 1949-0305.

NAVET, N.; SIMONOT-LION, F. *Automotive Embedded Systems Handbook*. 1st. ed. Boca Raton, FL, USA: CRC Press, Inc., 2008. ISBN 084938026X, 9780849380266.

NAVET, N.; SIMONOT-LION, F. In-vehicle communication networks - a historical perspective and review. *Industrial Communication Technology Handbook, Second Edition*, v. 96, 01 2013.

NEWMAN, D. The Role of Latency and Jitter in Network Performance Assessment. 2011. Available at: <https://support-kb.spirent.com/resources/sites/SPIRENT/content/live/FAQS/12000/FAQ12288/en_US/NetworkTest_Latency-and-Jitter_Whitepaper.pdf>.

NOLTE, T.; HANSSON, H.; BELLO, L. L. Automotive communications-past, current and future. In: . [S.l.: s.n.], 2005.

O'Donoghue, K. Emerging solutions for time protocol security. In: *2016 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS)*. [S.l.: s.n.], 2016. p. 1–6. ISSN 1949-0313.

OSSEC. *OSSEC*. 2019. Available at: <https://www.ossec.net>.

Pop, P.; Raagaard, M. L.; Gutierrez, M.; Steiner, W. Enabling fog computing for industrial automation through time-sensitive networking (tsn). *IEEE Communications Standards Magazine*, v. 2, n. 2, p. 55–61, JUNE 2018. ISSN 2471-2825.

Rodney Cummings. *Ingress policing*. <https://avnu.org/wp-content/uploads/2014/05/AVnu-AAA2C_Ingress-Policing_Rodney-Cummings.pdf>. Online; 06/04/2019.

SAE-J3061 Cybersecurity Guidebook for Cyber-Physical Vehicle Systems. [S.l.], 2016. Available at: <https://www.sae.org/standards/content/j3061_201601/>.

SALINI, P.; KANMANI, S. Survey and analysis on security requirements engineering. *Computers & Electrical Engineering*, v. 38, n. 6, p. 1785 – 1797, 2012. ISSN 0045-7906. Available at: <http://www.sciencedirect.com/science/article/pii/S0045790612001644>.

SCAPY. *Scapy*. 2019. Available at: <https://scapy.net/>.

SCHULZRINNE S. CASNER, R. F. V. J. H. *RTP: A Transport Protocol for Real-Time Applications*. [S.l.], 2003. Available at: <https://tools.ietf.org/html/rfc3550>.

SNORT. *Snort*. 2019. Available at: <https://www.snort.org/>.

Sommer, J.; Gunreben, S.; Feller, F.; Kohn, M.; Mifdaoui, A.; Sass, D.; Scharf, J. Ethernet – a survey on its fields of application. *IEEE Communications Surveys Tutorials*, v. 12, n. 2, p. 263–284, Second 2010. ISSN 1553-877X.

SOMMER, R.; PAXSON, V. Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. *2010 IEEE Symposium on Security and Privacy*, 2010. ISSN 10816011.

STEINBACH, T. *Ethernet-basierte Fahrzeugnetzwerkarchitekturen für zukünftige Echtzeitsysteme im Automobil*. Wiesbaden: Springer Vieweg, 2018. ISBN 978-3-658-23499-7.

SURICATA. *Suricata*. 2019. Available at: <https://suricata-ids.org/>.

TEENER, M. J.; KIM, Y. Requirements for Automotive AVB System Profiles. p. 23, 2011. Available at: <https://avnu.org/wp-content/uploads/2014/05/Contributed-Automotive-Whitepaper{_}April-2011.pdf>.

Thiele, D.; Ernst, R. Formal worst-case timing analysis of ethernet tsn's burst-limiting shaper. In: *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*. [S.l.: s.n.], 2016. p. 187–192. ISSN 1558-1101.

Treytl, A.; Hirschler, B. Practical application of 1588 security. In: *2008 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*. [S.l.: s.n.], 2008. p. 37–43. ISSN 1949-0305.

TSANG, J.; BEZNOSOV, K. A security analysis of the precise time protocol (short paper). In: . [S.l.: s.n.], 2006. v. 4307, p. 50–59.

TUOHY, S.; GLAVIN, M.; HUGHES, C.; JONES, E.; TRIVEDI, M.; KILMARTIN, L. Intra-Vehicle Networks: A Review. *IEEE Transactions on Intelligent Transportation Systems*, IEEE, v. 16, n. 2, p. 534–545, apr 2015. ISSN 1524-9050.

Ullmann, M.; Vögeler, M. Delay attacks — implication on ntp and ptp time synchronization. In: *2009 International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*. [S.l.: s.n.], 2009. p. 1–6. ISSN 1949-0305.

Upstream Security. *Global Automotive Cybersecurity Report*. 2019. <https://www.upstream.auto/upstream-security-global-automotive-cybersecurity-report-2019/>. Online; 08/04/2019.

WIKIPEDIA. *Credit Based Shaper*. 2019. Available at: <https://en.wikipedia.org/wiki/Credit-based_fair_queuing>.

WIRED. *HACKERS REMOTELY KILL A JEEP ON THE HIGHWAY*. 2015. Available at: <https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/>.

ZARPELÃO, B. B.; MIANI, R. S.; KAWAKANI, C. T.; ALVARENGA, S. C. de. A survey of intrusion detection in Internet of Things. *Journal of Network and Computer Applications*, Elsevier Ltd, v. 84, n. February, p. 25–37, 2017. ISSN 10958592. Available at: <http://dx.doi.org/10.1016/j.jnca.2017.02.009>.

Zinner, H.; Noebauer, J.; Gallner, T.; Seitz, J.; Waas, T. Application and realization of gateways between conventional automotive and ip/ethernet-based networks. In: *2011 48th ACM/EDAC/IEEE Design Automation Conference (DAC)*. [S.l.: s.n.], 2011. p. 1–6. ISSN 85-644924.