UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

EDILSON AUGUSTO SILVA JUNIOR

LOW-COST EXPERIMENTAL VALIDATION OF CRYPTOGRAPHIC PROTOCOLS FOR SAFETY-CRITICAL AUTOMOTIVE COMMUNICATION

Recife

2017

EDILSON AUGUSTO SILVA JUNIOR

LOW-COST EXPERIMENTAL VALIDATION OF CRYPTOGRAPHIC PROTOCOLS FOR SAFETY-CRITICAL AUTOMOTIVE COMMUNICATION

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

Área de Concentração: Redes de computadores e sistemas distribuídos

Orientador (a): Divanilson Rodrigo de Sousa Campelo

Recife

2017

Catalogação na fonte
Bibliotecária Nataly Soares Leite Moro, CRB4-1722

S586l Silva Junior, Edilson Augusto
Low-cost experimental validation of cryptographic protocols for safety-critical automotive communication / Edilson Augusto Silva Junior – 2017.
91 f.: il., fig., tab.

Orientador: Divanilson Rodrigo de Sousa Campelo.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2017.
Inclui referências e apêndice.

1. Redes de computadores e sistemas distribuídos. 2. Criptografia. 3. Sistemas embarcados. 4. Redes automotivas. 5. Redes intra-veiculares. 6. Segurança. I. Campelo, Divanilson Rodrigo de Sousa (orientador). II. Título

004.6 CDD (23. ed.) UFPE - CCEN 2024 – 25

EDILSON AUGUSTO SILVA JUNIOR

LOW-COST EXPERIMENTAL VALIDATION OF CRYPTOGRAPHIC PROTOCOLS FOR SAFETY-CRITICAL AUTOMOTIVE COMMUNICATION

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: 09/03/2017.

BANCA EXAMINADORA

_____
Prof. Dr. Abel Guilhermino da Silva Filho (Examinador Interno)
Centro de Informática / UFPE

_____
Prof. Dr. Iguatemi Eduardo da Fonseca (Examinador Externo)
Centro de Informática / UFPB

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

I dedicate this thesis to all my family, friends and professors who gave me the necessary support to get here.

ACKNOWLEDGEMENTS

I would like to thank God and also my family who gave me support through this time. My advisor Divanilson Campelo, for the patience and guidance. Paulo Araújo Filho, Fábio Leite, José Ivson Soares, Caio Araújo e Andrea Nogueira for helping during the development of the project. Finally, CAPES and FACEPE for the financial support.

ABSTRACT

The automobile, previously seen as an isolated system, is now immersed in a connected environment where the data exchanged inside the car and with the outside world has little or no security measures against unwanted attackers. In this new scenario, automotive security must be one of the most important architectural attributes of any car. This fact raises some challenges for OEMs, since they must keep producing economically attractive cars, but with ever-increasing enticing functionalities. This trade-off has a direct impact on the possibilities for the hardware utilized to execute such functionalities, and hence its available processing power. Since security measures frequently involve the use of encryption methods, the required hardware performance tends to escalate; when safety-critical applications are concerned, the hardware requirements are even more severe due to maximum latency limitations. Advances in technology, especially in dedicated processing modules for specific tasks and the increase in raw processing power of processors, along with a redesigned architecture for the exchange of in-vehicle data communication, turn feasible the implementation of functionalities in cars that previously suffered from performance limitations. Ethernet arises as a high-bandwidth, scalable and future-proof in-vehicle network technology and is the main component supporting this newly redesigned architecture. This work aims to investigate the performance implications introduced by the use of cryptographic protocols in the exchange of information among electronic modules in safety-critical automotive systems. Confidentiality, authenticity, and integrity are explored over different secret key sizes, and different payload data sizes on the layer 2 of the network. Hardware encryption accelerator-enabled micro-controller units are utilized to accelerate cryptography-related calculations. The latency of the data exchange is compared with the case where only a Cyclic Redundancy Check (CRC32) is used, which has been the most commonly used method for integrity verification over the last years in the automotive domain, as well as with the case with no verification. Low-cost processing modules and an ordinary switch are utilized to experimentally demonstrate that the typical maximum latency values for safety-critical applications can be respected even after the application of cryptography to protect data communication among electronic modules.

Keywords: computer networks and distributed systems; cryptography; automotive networks; ethernet; security; safety-critical.

RESUMO

O automóvel, antes um sistema isolado, está agora imerso em um ambiente conectado onde os dados trocados dentro do carro e com o mundo externo têm poucas ou nenhuma medidas de segurança. Neste novo cenário, a segurança automotiva precisa ser um dos atributos arquiteturais mais importantes. Este fato traz desafios aos fabricantes, já que eles devem continuar produzindo carros atrativos economicamente, porém com funcionalidades empolgantes. Isto tem um impacto direto nas possibilidades existentes para o hardware utilizado para tais funcionalidades, e, portanto, seu máximo poder de processamento alcançável. Como as medidas de segurança geralmente envolvem o uso de métodos de encriptação, o desempenho requerido tende a crescer; quanto às aplicações *safety-critica*l, os requerimentos de hardware são ainda mais severos devido às limitações de latência máxima. Avanços na tecnologia, especialmente nos módulos de processamento dedicados e o aumento em poder bruto de processamento, juntamente com uma arquitetura redesenhada para a troca de dados intra-veiculares, tornam factível a implementação de funcionalidades em carros que anteriormente sofriam limitações de desempenho. A Ethernet surge como uma tecnologia de rede intra-veicular de grande largura de banda, escalável e à prova do futuro, além de ser o principal componente base para esta redesenhada nova arquitetura. Este trabalho tem como objetivo investigar as implicações no desempenho, devido ao uso de protocolos de criptografia durante a troca de informações entre módulos eletrônicos em sistemas automotivos *safety-critical*. Confidencialidade, autenticidade e integridade são explorados através da utilização de diferentes tamanhos de chaves secretas e de dados de carga útil na camada 2 da rede. Microcontroladores com hardware acelerador de encriptação são utilizados para acelerar cálculos criptográficos. A latência da troca de dados é comparada com o caso em que apenas o método Verificação de Redundância Cíclica (CRC32) é utilizado, o qual tem sido o método de verificação de integridade mais comumente utilizado no domínio automotivo até os últimos anos, e também com o caso em que nenhuma verificação é feita. Módulos de processamento de baixo custo e um switch comum são utilizados para demonstrar experimentalmente que os valores comuns de latência máxima para aplicações safety-critical podem ser respeitados mesmo depois da aplicação da criptografia, para que se proteja a comunicação dos dados entre módulos eletrônicos.

Palavras-chave: redes de computadores e sistemas distribuídos; criptografia; redes automotivas; ethernet; segurança; sistemas críticos.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

ADAS	Advanced Driver Assistance Systems
AE	Authenticated Encryption
AES	Advanced Encryption Standard
AVB	Audio Video Bridging
CAN	Controller Area Network
CBC	Cipher Block Chaining Mode
CCM	CRC and Cryptographic Modules
CCM	Counter with Cipher Block Chaining-Message Authentication Code
CRC	Cyclic Redundancy Check
CRC32	Cyclic Redundancy Check - 32 bits
DES	Data Encryption Standard
ECB	Electronic Code Book Mode
ECUs	Electronic Control Units
FIPS	Federal Information Processing Standards
FPGA	Field Programmable Gate Array
GPUs	Graphical Processing Units
HMAC	Hash-based Message Authentication Code
IP	Internet Protocol
LIDAR	Light Detection and Ranging
LIN	Local Interconnect Network
MAC	Media Access Controller
MAC	Message Authentication Code
MCU	Micro-controller Unit
MOST	Media Oriented Systems Transport
MTU	Maximum Transfer Unit

NDK	Network Developer's Kit
OWD	One-Way Delay
PHY	Network Physical Layer Interface
PTP	Precision Time Protocol
QoS	Quality of Service
RADAR	Radio Detection and Ranging
RSUs	Roadside Units
RTOS	Real Time Operating System
RTT	Round-Trip Time
SHA	Secure Hash Algorithm
TCP	Transmission Control Protocol
TDMA	Time Division Multiple Access
TI	Texas Instruments
TSN	Time Sensitive Networking
uDMA	Micro Direct Memory Access

LIST OF SYMBOLS

μ	Mi

CONTENTS

1 INTRODUCTION

1.1 PROBLEM STATEMENT

Connectivity brings to its users a vast amount of information, which aids them during everyday tasks. This connectivity has been becoming more prevalent inside the automobiles as well, making new functionalities possible. It offers personalization options, and even makes driving itself easier, for example, with the Advanced Driver Assistance Systems (ADAS) (NAVET; SIMONOT-LION, 2013). The current automotive network standards, such as Controller Area Network (CAN), Local Interconnect Network (LIN), Media Oriented Systems Transport (MOST), and Flexray, do not have the required bandwidth and scalability to accommodate such an amount of information simultaneously (BELLO, 2011). Most vehicles have more than one of the types of automotive networks mentioned, consequently creating different domains. The use of gateways to create inter-domain connections only introduces software that adds unnecessary complexity (DAOUD et al., 2006).

Ethernet is the likely candidate towards a low-cost, unified and homogeneous architecture (STAEHLE et al., 2013), (TUOHY et al., 2015), (HANK et al., 2013), because of its technological maturity that is well established in other fields. Further on, it makes possible a paradigm shift to a centralized version, which lowers the complexity and may execute software independently from specific hardware, through the abstraction and virtualization of components (CHAKRABORTY et al., 2012).

However, the traffic of such critical and sensitive information requires protection against unwanted intruders. A poorly implemented security results in the lack of privacy and protection of the data exchanged by the car, consequently compromising the safety of the passengers, meaning their physical well-being. Therefore, the security aspects need to be planned and developed with extreme caution from the early conception stages of the vehicle (STAEHLE et al., 2013). The study presented by this work focuses on the security of the information that travels inside the car. It will not cover the efficiency of the car in protecting its passengers when suffering a collision, for example.

It has already been demonstrated that the security of most of the current cars is very flawed and was not even a concern until recent years. A great part of this is due to the low processing power of the available Electronic Control Units (ECUs) up to that point (KOSCHER et al., 2010), (CHECKOWAY et al., 2011), (CAMEK; BUCKL; KNOLL, 2013). Beyond that, the automobile has

been considered for a very long time, a system that is isolated from other external systems, but the increasing connectivity is rapidly changing this fact. As demonstrated in Miller and Valasek (2015), cars can be remotely controlled without having ever been touched. This method required an extensive study of the car intended to be exploited, which was carried out for approximately two years and involved disassembling the entire car. Once the research was completed for a single model, every car from the same manufacturer with a similar system was now vulnerable. The system was invaded and taken over through a subsystem that indirectly had access to the control of the primary functions of the cars. That is only an example of how individual security measures are rendered useless if a system is not planned as a whole and of how the driver may become completely vulnerable.

It is common knowledge in the security community that one should always use the best security available, without harming the intended functionality of the system. There is no such thing as a secure system; there is only a system that has not yet been compromised. With that in mind, and since automobiles deal with lives every day, everywhere security is applicable, it should be applied.

Some of these automotive systems, such as engine control, are safety-critical and have strict hard real-time requirements, therefore not affording delays. They are rigorous latency end-to-end delay requirements, so missing a safety-critical deadline could compromise the safety of passengers.

The above mentioned safety-critical systems rely upon network architectures that were historically based on a CAN bus system (TUOHY et al., 2015). However, the advances brought by 100BASE-T1, 100Mbit/s Ethernet over a single twisted pair for automotive applications, suggest a new automotive network architecture, that will be mainly based on a switched Ethernet network.

The use of cryptography (STALLINGS, 2006) is one of the most frequently used security methods applied. As described in Chakraborty et al. (2016), its use leads to additional processing of data and longer messages, having a significant impact on the scalability and performance control of the system. However, this behavior goes directly against the requirements of safety-critical systems, which demands strong maximum latency values.

A great deal is known about security, and there is a variety of previous work on the impact of cryptography on the performance of systems in general, as well as several others on automotive safety-critical requirements. However, to the best of our knowledge, little is known about the performance toll when applying cryptographic schemes in the automotive environment when

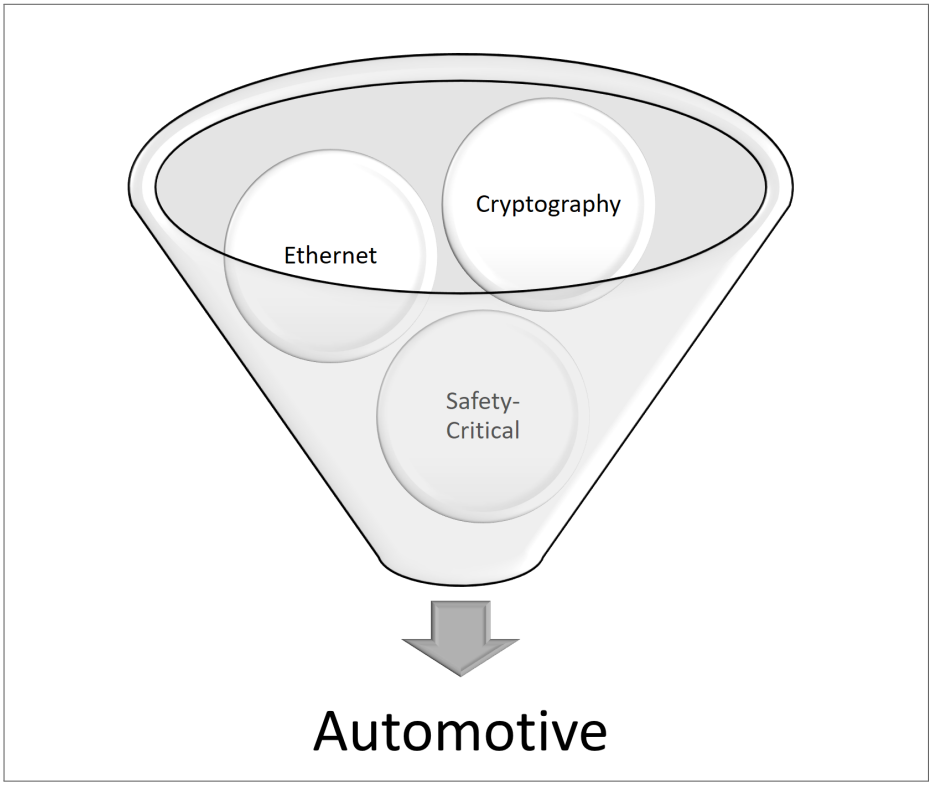
safety-critical systems are concerned.

Based on this context, the central research question investigated by this dissertation is:

Research question *Is it possible to apply cryptography to accomplish authenticated encryption of exchanged control data while still respecting the maximum latency of a safety-critical automotive system?*

This work will focus on combining the topics exhibited in figure 1, which are the following: Ethernet, Cryptography, and Safety-Critical. As the picture suggests, for each one of them to cooperate amongst themselves, some restrictions, limitations, or adaptations need to be taken into account, to adequate them to the automotive field.

Figure 1 – Researched topics.



Source: The author (2017)

1.2 OBJECTIVES

1.2.1 General Objectives

Research objective *The contribution of this work is the evaluation, from a latency performance standpoint, of the suitability of cryptographic methods for safety-critical control*

data of an automotive system.

1.2.2 Specific Objectives

- *Objective 1:* Propose a low-cost prototype to validate the usage of cryptographic algorithms for a safety-critical automotive system.
- *Objective 2:* Propose experiments to test different scenarios while still respecting the maximum allowed latency for safety-critical automotive systems.
- *Objective 3:* Compare different cryptographic algorithms regarding the latency performance.
- *Objective 4:* Validate the results obtained against the desired behavior by evaluating different metrics for the obtained latency values. Such metrics are the mean, worst case, standard deviation, median, and statistical tests.
- *Objective 5:* Evaluate the deterministic behavior of the proposed architecture regarding the chosen hardware and software platform.

1.3 RESEARCH METHODOLOGY

1. *Basic Knowledge:* Careful study of various Intra-vehicular networks, along with the requirements and peculiarities of the automotive field. Here a focus towards switched Ethernet networks was favored. Other different network types that were previously employed (some of them still are) by vehicles were also studied, be it internally or to communicate with the external world.
2. *Literature Review:* Based on scientific studies and also the current trend of the automotive market, the state-of-the-art of the Ethernet technology inside cars was researched to find possible challenges and opportunities in the field.
3. *Strategy Elaboration:* After isolating the target problem to be studied, a strategy with possible options to solve or improve the chosen scenario was elaborated.
4. *Devising the Architecture:* Here, along with designing the architecture's desired functionality, the necessary features were selected based on the literature review. After de-

termining these features, hardware devices were researched and compared against each other. The objective was to find a low-cost solution that supported the necessary technologies without further unnecessary resources, which would only increase the cost of the devices.

5. *Prototype Assembly:* In this case, involved the purchase of the selected hardware from the previous step and the configuration of the associated devices so that the development of the firmware could begin.

6. *Firmware Development:* The firmware development also included the familiarization with the environment, especially the Real Time Operating System (RTOS) and the Cryptographic Module, by taking part in online workshops from the manufacturer and resolving implementation-related problems.

7. *Experiments:* After the core functionality was implemented, experiments were elaborated to evaluate different scenarios.

8. *Evaluation of the Results:* Elaboration of the metrics to compare the results against the desired behavior and validate the experiment.

1.4 OUT OF SCOPE

As the proposed approach is part of a broader context, a set of related aspects will be left out of its scope. Thus, the following topics are not directly addressed in this work: no process for establishing a connection is involved, only the exchange of messages. Since the connection links are planned using offline scheduling, meaning they are decided in the development phase, a process for establishing a connection is not required for the current scenario. The management of secret keys is also not investigated. Further on, no qualitative analysis for finding the best cryptographic method, regarding security strength, for such a system is performed.

1.5 STRUCTURE OF THE DISSERTATION

The work presented is organized in the following manner: **Chapter 2** explains the necessary fundamental knowledge that gives support for the proposed assumptions. Further on, a review of the related work in the area is laid out, presenting the corresponding state-of-

the-art. **Chapter 3** offers an architecture for securing latency-constrained communication in automotive systems and exhibits a design of the prototyped experiment. An explanation is then carried out, over the results aimed to be achieved and the metrics utilized to validate the results. **Chapter 4** exposes and discusses the results acquired. Finally, **Chapter 5** concludes the work and discusses possible future studies.

2 BACKGROUND AND RELATED WORK

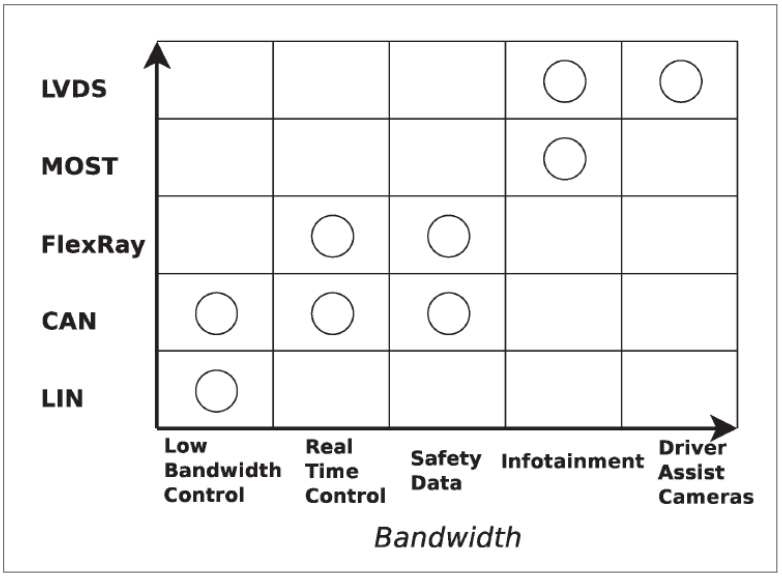
2.1 INTRODUCTION

The following chapter will begin by laying out the fundamental knowledge for the area of research, the area of Automotive Networks, also discussing the state-of-the-art of this field, by commenting on some relevant related work.

2.2 HETEROGENEOUS AUTOMOTIVE NETWORK ARCHITECTURES

Automotive Networks, also known as Intra-vehicular or In-vehicle Networks, are the terms used by professionals and academics in this field to describe the networks that exchange data inside the vehicle. Vehicular or Inter-vehicle Networks, on the other hand, are the terms used to address the networks that handle the traffic external to the car, which communicates with the outside world, meaning other vehicles, portable devices from the driver, or Roadside Units (RSUs) for example. This work is focused on Intra-vehicular Networks and, therefore, will not address the other case.

Figure 2 – Different traffic types, regarding the bandwidth required, are often related to the network utilized



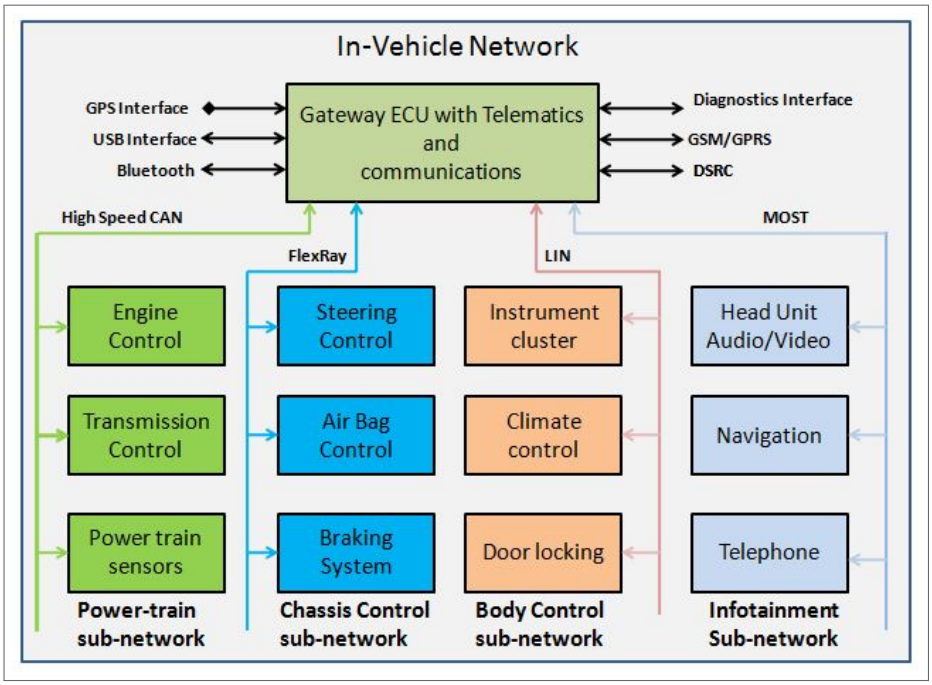
Source: Tuohy et al. (2015)

For years now, the Automotive Industry follows the world's trend that demands an increasing amount of technology. This technology may come in the form of new functions and capabilities for the car, or by providing relevant information to the driver, for example (NAVET;

SIMONOT-LION, 2013). In today's average car, this translates into a set of ECUs for a group of these said functions or even for each of them (CHAKRABORTY et al., 2012). In this manner, the number of ECUs continuously grows to offer more to the consumer. Since embedded electronics represent a substantial amount of the overall price and increase complexity, the manufacturers become very restrained by this fact (BUCKL et al., 2012).

At a certain point, it was acknowledged that the inter-connectivity between those units produces valuable information and makes even more advanced functionalities possible (BELLO, 2011). For this inter-connectivity to be achieved, different kinds of automotive networks were created throughout the years, such as CAN, LIN, MOST and Flexray (NAVET; SIMONOT-LION, 2013), (TUOHY et al., 2013), (TUOHY et al., 2015). However, most cars have more than one type of the mentioned networks at the same time, and therefore different domains are created (STAEHLE et al., 2013), (DAOUD et al., 2006). The networks that are shown in figure 2 are application dependent, and none of them is suitable for all of them as it can be seen from the chart.

Figure 3 – Example of a gateway for unifying the heterogeneous automotive (In-Vehicle) domains



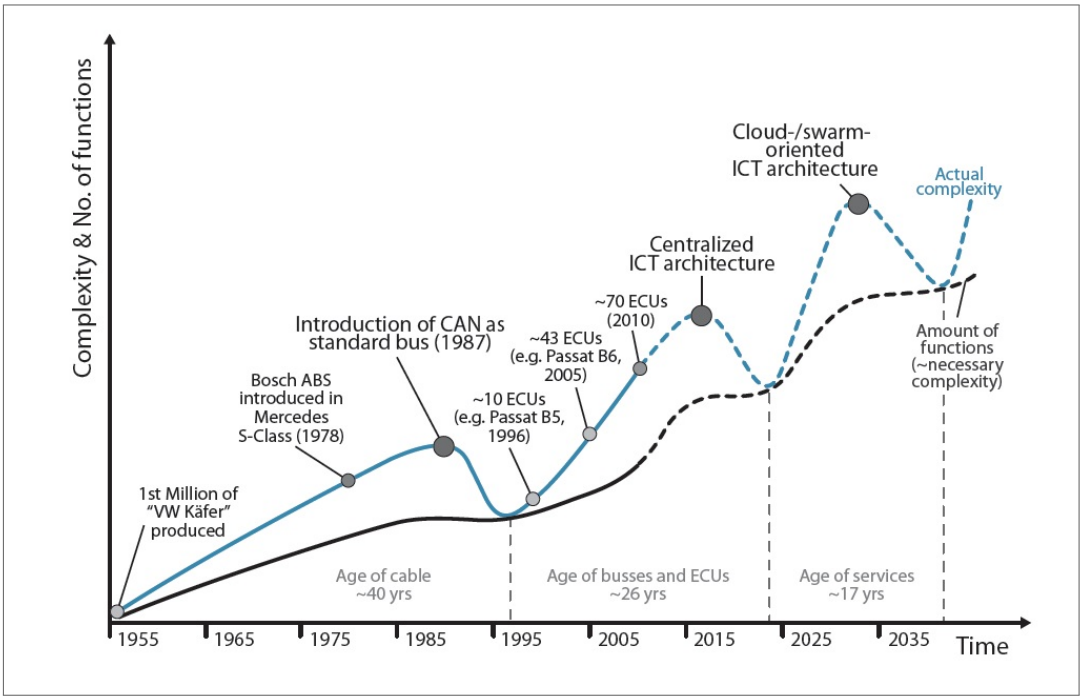
Source: Sordo (2016)

For this inter-connectivity to be system-wide, the use of gateways, such as in figure 3, and unnecessary complex software are introduced (DAOUD et al., 2006). Sometimes, systems that work completely fine on their own may need adaptations when working alongside with others. This implicates in longer developing times and presents flaws, especially security flaws, which

would not be present in the first place when using the standalone solutions (CAMEK; BUCKL; KNOLL, 2013), (BUCKL et al., 2012).

Further on, the current well-established automotive networks have limited bandwidth and cannot cope with a unified approach due to the amount of data that needs to be sent, and the scalability is also a significant issue (HANK et al., 2013). It has already been demonstrated that the security implemented nowadays on cars is fragile and wasn't a real concern for the industry until recently, mainly because of the limited processing power of the ECUs (MILLER; VALASEK, 2015), (KOSCHER et al., 2010), (CHECKOWAY et al., 2011), (CAMEK; BUCKL; KNOLL, 2013).

Figure 4 – Evolution of the complexity and number of functions in cars across the years



Source: Buckl et al. (2012)

The price and developing times continue to increase, as they are directly related to the number of ECUs. However, at the same time, the total required processing power for a unified approach has already been reached, despite being currently divided into several less capable units, as it can be seen in figure 4 (BUCKL et al., 2012).

2.3 ETHERNET BACKBONE

Ethernet (SPURGEON, 2000), which is a popular network communication technology, started making its way into the automotive world and is one of the primary keys for turning this ap-

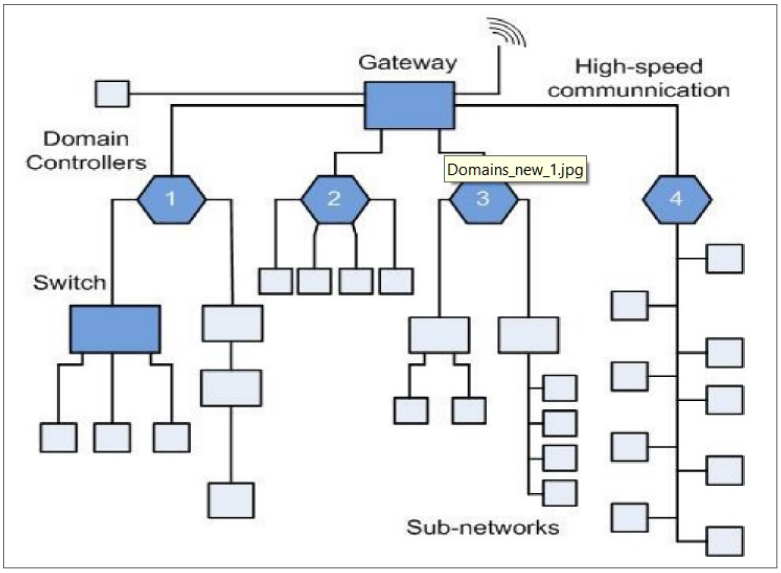
proach into reality. It will allow a higher bandwidth, at the same time reducing the cable harness, which contrarily to what one may think, currently accounts for a substantial percentage of the total weight of the car.

Despite being very competitive, the Automotive Industry is also conservative sometimes and only employs technologies that are thoroughly tested, as they deal with lives every day. The safety of the passengers is of the utmost importance. The financial aspect plays an important role, as well. Researching and testing new technologies can be very expensive, and in case they do not have a future, all investments are lost. Being conservative in this branch, the companies tend to wait for others to try a particular research possibility first. BMW was the first company to investigate the use of Ethernet in cars. A car has to be capable of working reliably under different harsh circumstances, such as temperature, weather, as well as electromagnetic emissions and interference, for example (HANK et al., 2013). These were some of the most critical aspects that delayed the usability of Ethernet in this scenario. That is why Ethernet was initially employed in scenarios where the car does not move, and which took a significant time with the then-available data transfer speeds (only for diagnostics and to update ECUs) (BELLO, 2011).

This network technology has matured continuously through time outside of the automotive sphere. So, hardware and software components from other domains should be re-used as much as possible, while adding the proper restrictions for vehicles. The Ethernet technology used in cars is mostly the same as the standard one, except for the physical layer. That means that most of the implementation issues related to the software side were already dealt with (HANK et al., 2013).

To enable advanced functionalities such as the ones of an autonomous car, for example, a complete redesign of how information is managed, processed, and transported is necessary. The growing complexity of adding standalone ECUs for each new functionality is not sustainable in the long term. A unified approach has the Ethernet network as a central homogeneous data highway where every subsystem can communicate to the other, and the administration can be centralized. That is called the Ethernet backbone, shown in figure 6. The hierarchical nature of Ethernet turns the vehicle easily scalable, as shown in figure 5, with new functionalities being added in a Plug and Play fashion. This architecture facilitates the development and enables features that were not previously possible, with the bandwidth and scalability of Ethernet being essential to accommodate so much information at the same time. Other than that, each device can now be easily reachable through a network address if needed (CHAKRABORTY et al., 2012),

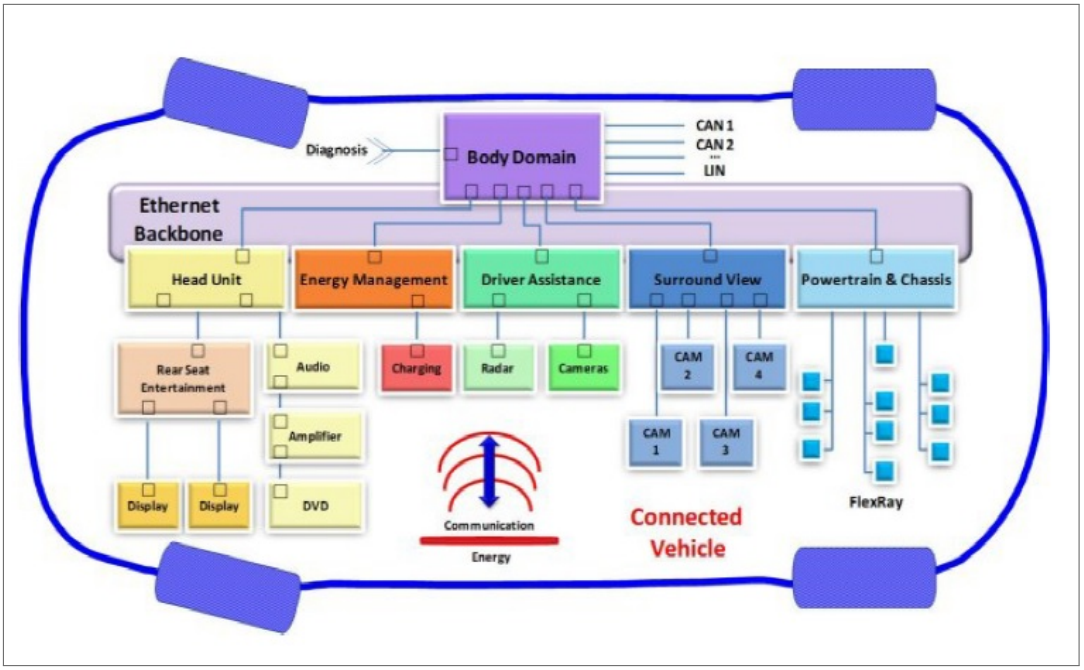
Figure 5 – Hierarchical architecture using Ethernet



Source: Hank et al. (2013)

(HANK et al., 2013), (STAEHLE et al., 2013).

Figure 6 – Centralizing Ethernet Backbone



Source: Hank et al. (2013)

For the feasibility of such an interconnected approach, as mentioned earlier, the distribution of hardware processing power needs to be reorganized. The centralization of the network also permits a centralization of the processing of information. The main unit requires now far more raw processing capabilities. On the other hand, the new functionalities, which, for example, would be implemented by adding more ECUs, sensors, and actuators, would have

their computational demands executed by the central unit. Only some pre-processing would be done to handle the data from sensors, and actuators, making the addition of new capabilities considerably cheaper. With that in mind, the use of dedicated hardware, which is usually more expensive, is now possible. Multi-core processing units, reconfigurable hardware such as a Field Programmable Gate Array (FPGA), also Graphical Processing Units (GPUs), high resolution cameras, sensors like Radio Detection and Ranging (RADAR), Light Detection and Ranging (LIDAR) are examples of devices that are now a possibility when using such architecture and whose functionalities could be very advantageous for an intelligent car (CHAKRABORTY et al., 2012), (KAINZ; BUCKL; KNOLL, 2010).

2.4 DETERMINISTIC ETHERNET COMMUNICATION

As mentioned before, several fronts have the objective of bringing determinism to Ethernet. Among them are IEEE 802.1Q, Audio Video Bridging (AVB) Ethernet, and TTEthernet. (TUOHY et al., 2015) does an ample review of these technologies, showing several related works to support the content.

IEEE 802.1Q relies on the tagging of packets, according to their priority, to ensure a Quality of Service (QoS) by adding an extra field to the Ethernet header. Tuohy et al. (2015) shows that it has been used in several works in the automotive field. Lim, Weckemann and Herrscher (2011) is one of them, and it does a study on the performance of an in-car switched Ethernet network without prioritization, based on Internet Protocol (IP), laying out the necessary automotive requirements and service constraints, showing that without any QoS mechanism is not possible to guarantee those conditions. However, Lee and Park (2013) shows that by limiting the Maximum Transfer Unit (MTU) of the messages, the hard real-time requirements may be achieved without any modification to the network stack or protocols.

AVB (KOFTINOFF, 2016) was initially developed for transmitting synchronous audio and video. It runs on network layer 2, so it does not use IP, and consists of four IEEE standards that encapsulate several functionalities, such as the one that provides QoS on the IEEE 802.1Q and several other more. It, however, demands that specialized switches and network nodes are utilized. Even with the added cost of specialized hardware, the automotive industry is showing a growing interest in the standard, and proposed a new edition of the standard, also known now as Time Sensitive Networking (TSN), whose improvements includes, for example, frame preemption (TUOHY et al., 2015). Preemption, in the case of a RTOS, allows stopping a lower

priority task in favor of a higher priority one, preventing that it misses its deadline. That is very common in a RTOS, but the majority of the networks still do not support it. 802.1Qbu - Frame Preemption, is the IEEE standard (IEEE, 2015) that is responsible for this functionality in the TSN networks.

Lim, Herrscher and Chaari (2012), later reinforced by Tuohy et al. (2015), did a comparison between the 802.1Q and the AVB, which resulted in the former outperforming the latter, for the transmission of control data. However, the situation changes when extra traffic is introduced inside the network. So, the author concludes that more work is necessary to ensure that the AVB meets control data real-time latency requirements.

Steffen et al. (2008) shows that an IP-based network may use traffic shaping mechanisms, along with proper network dimensioning, to guarantee real-time constraints. It suggests Precision Time Protocol (PTP), could be utilized to synchronize the ECUs inside the network. PTP, or IEEE 1588, is the base standard of IEEE 802.1AS, which by its turn, is one of the standards of AVB.

2.5 NETWORK SECURITY

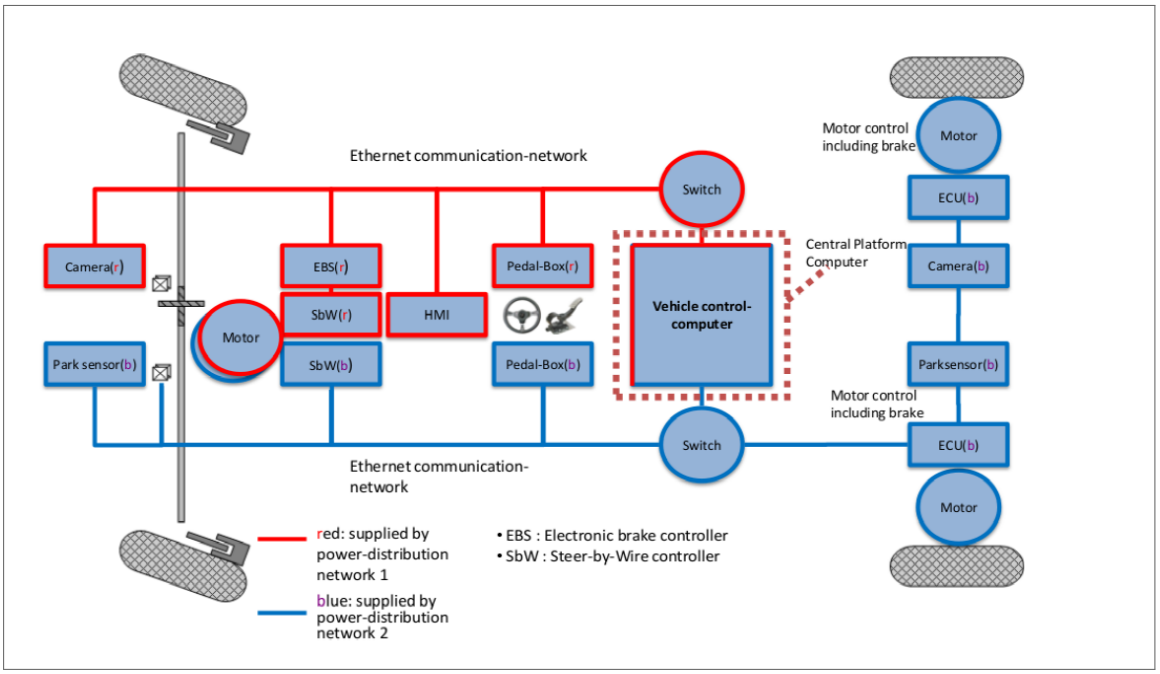
Differentiating these two terms is essential: safety and security. In the automotive world, there is a difference between safety and security. Safety is related to the well-being of the passengers, while security involves protecting sensitive information against unknown attackers.

Safety-critical is a relatively new term heavily utilized by several authors, such as TUOHY et al., STEFFEN et al., which is used to describe processes that require a hard real-time operation and therefore, determinism. "Safety" inside the term, is to emphasize that if a deadline were to be missed, the safety of the passengers could potentially be put in jeopardy.

When security is the subject, cryptography is the most common approach, for example, by encrypting the data to prevent eavesdropping or tampering (HUMAYED et al., 2017).

A unified security measure reduces complexity and, as stated earlier, may impede the appearance of problems resulting from the adaptations necessary for several standalone measures to work together. Physical isolation and hardware redundancy of critical components, as shown in figure 7, improve both security and safety, by ensuring a fail-safe operation (CAMEK; BUCKL; KNOLL, 2013). However, this work focuses on the performance regarding the latency while making use of cryptographic methods.

Figure 7 – Physical isolation and redundancy of critical components by using two distinct networks for ensuring a fail-safe operation



Source: Camek, Buckl and Knoll (2013)

2.5.1 Cyclic Redundancy Check (CRC32)

The Cyclic Redundancy Check (CRC) is a data verification method to check for integrity errors in the information sent, which is also employed by the IEEE 802.3 Ethernet. The 32-bit variation produces a 4 bytes code (32 bits), resulting from the rest of repeated polynomial divisions (PETERSON; BROWN, 1961). Miller and Valasek (2015) and Humayed et al. (2017) show that it is a popular method utilized as a security measure, especially in the automotive field. Even though it is better than not using any protection, it is easily broken.

2.5.2 Cryptography

One approach for securing a network communication using cryptography is through AE mechanisms, which simultaneously provide confidentiality, integrity, and authenticity assurances on the data. AE can be constructed by combining an encryption scheme with a Message Authentication Code (MAC). In this research, as explained later on section 3.2.3, the AES and the HMAC are combined to achieve authenticated encryption.

2.5.2.1 AES Overview

The AES specifies a cryptographic algorithm, approved by the Federal Information Processing Standards (FIPS), that consists of a symmetric block cipher that encrypts and decrypts data in 128 bits blocks (STANDARD, 1977). It is considered a substitution-permutation network, due to its sequential round operation consisting of 10, 12 or 14 iteration rounds, depending on the key size (HAMALAINEN et al., 2006).

The algorithm supports 128, 192, and 256 bits keys in a way that the higher the key size, the greater the number of iteration rounds and the slower the algorithm. On the other hand, security is improved when using larger key sizes.

2.5.2.2 HMAC Overview

While the AES algorithm provides confidentiality only, integrity and authenticity are provided by HMAC. It performs keyed-hash operations, using an input key and existing hash functions that convert plaintext into a Message Digest (MD) with a fixed length, such as MD5 and SHA-1 (WANG et al., 2004). Newer variations include different versions of the Secure Hash Algorithm (SHA), such as SHA-2 or SHA-3.

The key point of HMAC is choosing the associated hash function, since the security of the message authentication mechanism depends on the cryptographic properties of the hash function (KRAWCZYK; BELLARE; CANETTI, 1997), along with the size of the key and whether or not it is random, i.e., how "random" it is.

The input data and a key are provided to the hash function, which calculates a 20 bytes MAC, in the case of the SHA-1, that will be sent along with the data. The receiver will then recalculate the MAC and check if it matches with the code received so that it can be validated.

2.5.2.3 Performance Impact and the One-Way Delay

Considering a simple communication link or a switched Ethernet infrastructure, applying cryptographic schemes imply an increase in the number of clock cycles needed for data processing, error verification, cryptographic validation, and dispatch, on both sending and receiving communication sides. That has a direct impact on the scalability and performance control of the system (CHAKRABORTY et al., 2016).

The Round-Trip Time (RTT) measures the time between the sending of a frame and the receiving of the answer. It takes into consideration the duration of both transmissions, as well as the processing time on the receiving nodes. Since cryptography increases the processing time, consequently, the RTT is also increased. Therefore, despite the need for securing the data, applying cryptographic schemes must be done with caution, so the application can still achieve the latency requirements.

However, it is important to differentiate the RTT and the OWD, as described in Abdou, Matrawy and Oorschot (2015). Both of these measures include several aspects that contribute to the total delay. Among these aspects are the transmission and propagation delays, the congestion, including the queuing along the way, the number of network hops, which is the number of routers/switches along the path. Additionally, there are the QoS, and the processing times in each node. The latter is regarding the time for error verification and determination of the output link. As can be seen from all these variables, for most scenarios, one cannot just measure the RTT and divide it by two to obtain the OWD. The forward and backward ways are rarely ever the same on networks with many node hops and without isolation or determinism.

2.6 RELATED WORK

The automotive industry is very competitive, and the companies are always "racing" against each other to release first the best technologies and innovations. Because of that, a significant part of the knowledge in this area is confidential and proprietary. Also, the documentation of the standards related to this area is often very costly. These are factors that hinder research in this field.

The intra-vehicular networks have already been intensely studied, as summarized in the review made by Tuohy et al. (2013) and updated on Tuohy et al. (2015). This study exposes the leading standards and protocols used, the types of traffic, and the introduction of Ethernet in the automotive environment. Navet and Simonot-Lion (2013) provides a historical perspective of the in-vehicle networks, dating from the 90s until nowadays. Navet and Simonot-Lion (2013) also details the AUTOSAR standard and its history, which was a partnership created to standardize the software architecture of automotive ECUs.

Tuohy et al. (2013) also addresses the migration from a heterogeneous architecture to a future top-down construction bringing the concept of a unified approach based on an automotive Ethernet backbone. It highlights, however, the necessity of modifying the regular Ethernet to

support deterministic delivery of safety-critical traffic and some proposed approaches to overcome this issue, such as the time-synchronization and time-triggered real-time synchronization strategies provided by AVB and TTEthernet.

Following this same path, the work presented in Steffen et al. (2008) defends the automotive Ethernet communication backbone and how it supports the Internet Protocol (IP). The vision of an all-IP in-car communication network entails reducing gateways into a simple router of IP packets as well as in a variety of new applications that can easily be integrated into the car due to the standardized interfaces and the modular building blocks. This research also analyzes real-time communication with IP and proposes 2.5 ms as the most substantial requirement concerning the end-to-end delay between two ECUs for control data.

Table 1 – Timing requirements for the different classes of network traffic

Traffic class	Max End-to-End Delay	Service Rate
Control Data	2.5ms	10 - 100ms
Safety Data (Video)	45ms	0.05 - 1ms
Infotainment Data	150ms	~1ms

Source: Tuohy et al. (2015)

The 2.5 ms delay value for safety-critical control data, proposed by Steffen et al. (2008), is reinforced by Tuohy et al. (2013), shown in table 1, and identifies the maximum total time required to transmit a packet from the source domain node to the receiving node. For safety-critical control data applications, missing this deadline is not affordable.

Stähle, Huang and Knoll (2014) and Buckl et al. (2012) propose an electric car evaluation platform, called eCar, over which a steer-by-wire system is designed and prototyped. It is a safety-critical system based on a switched Ethernet network and constitutes a latency-constrained communication.

On the other hand, Kleberger, Olovsson and Jonsson (2011) surveys the research concerning the security of the connected car and Studnia et al. (2013) highlights modern cars' communication interfaces, such as USB, Bluetooth, WiFi or even 3G, as entry points for cyber-attacks, since they may expose the internal network to the outside world. Studnia et al. (2013) also discusses security threats and protection mechanisms in embedded automotive networks and presents the security solutions currently being devised to address these problems. One of these solutions is the encryption of communications.

Koftinoff (2016) and Holle and Lothspeich (2016) show that the introduction of authentication and optional AES encryption is on currently being developed for the AVB/TSN standard.

However, this requires specialized hardware.

The 802.1AE - Media Access Control (MAC) Security standard, or MACsec (ROMANOW, 2006), provides connectionless confidentiality, integrity, and authenticity on network layer 2. Its employment, however, does require support from the underlying operating system.

Freeman and Miller (1999) does a study of the impact caused by cryptography on performance-critical systems, showing that already then, the performance of processors was allowing the use of cryptography on distributed networks for performance-critical applications, even though no dedicated hardware module for this purpose was utilized. It demonstrated various cryptographic controls, such as signature, encryption algorithms, hashing, and used different processors, with different speeds and scenarios, as part of the experiment to validate the work.

No reference was found to relate the harm caused by applying protection mechanisms and security solutions, such as communication encryption, to automotive latency constrained systems, or even to the automotive domain.

2.7 CONSIDERATIONS

This chapter prepared the fundamental knowledge for understanding the current state-of-art of the automotive Ethernet in cars. It was pointed out that the automotive industry is a very private and competitive field, and several new changes are occurring that are shifting the paradigm of in-vehicle network architectures.

This new redesigned approach is allowing new functionalities and applications, while also decreasing the trend of rising complexity that has been verified in recent years. Along with it, is the awareness that the increasing connectivity opens a previously isolated system to the world.

In the next chapter, the methodology used to validate the security protocols for the safety-critical control data environment is explained. The experiments based on a switched network architecture are detailed.

3 METHODOLOGY

3.1 INTRODUCTION

The current chapter describes the methodology that was chosen to validate the proposed assumption that the maximum latency for safety-critical systems may be respected despite the utilization of cryptography. The architecture proposed will be described concerning its hardware, software, and design, so that the experiment may be explained later in this chapter.

3.2 ARCHITECTURE

3.2.1 Hardware

The network nodes were chosen for being a low-cost option, which is an essential requirement in the automotive field, but at the same time offer the necessary core features without further resources that are not required for the architecture proposed. Doing otherwise would only increase the cost. In summary, the features desired were the support for real-time processing, support for Ethernet, and a dedicated cryptographic hardware accelerator module.

The chosen development board was the Tiva TM4C Series Crypto Connected Launchpad from Texas Instruments (TI), described in Texas Instruments (2014). Additionally to the features mentioned before, it has also support for CAN, which is still broadly used in cars, and IEEE 1588 PTPv2, even though they were not utilized in this work. The IEEE 1588 standard may be used to synchronize the clocks from the network nodes and is now part of the IEEE 802.1AS: Timing and Synchronization for Time-Sensitive Applications, which by its turn is one of the IEEE standards that are part of the AVB as described in Tuohy et al. (2015). However, to fully support IEEE 802.1AS, some further features are necessary that are not present on the board in question. As a consequence, it does not support AVB. Further information may be found in Koftinoff (2016), which is a webpage developed by Jeff Koftinoff, who has been involved in the development of the IEEE AVB Standards and has taken part in the implementation of these standards for devices in the Pro Audio market.

TM4C129E's supports the required real-time processing. That means that when a particular section of code is executed, a deterministic behavior is to be expected, i.e., every time the same code runs, it will have the same duration and will behave the same manner each of

the times. The support for Ethernet comes in the form of a controller consisting of a fully integrated MAC and Network Physical Layer Interface (PHY), which is highly configurable and conforms to the IEEE 802.3 and IEEE 1588 PTPv2 specifications.

On the cryptography side, there are dedicated hardware modules that contain an AES accelerator module and a SHA/MD5 module and a Data Encryption Standard (DES) module. Along with the CRC module, they combined form the CRC and Cryptographic Modules (CCM), also described in Texas Instruments (2014). The AES module provides hardware-accelerated data encryption and decryption operations based on a binary key. It offers different operation modes, such as the Electronic Code Book Mode (ECB) and the Cipher Block Chaining Mode (CBC). Various sizes of keys may also be used, such as 128, 192, or 256 bits. The SHA/MD5 module provides hardware-accelerated hash functions and can run, for example, HMAC operations, using either one of the following variations: SHA-MD5, SHA1, SHA224 or SHA256. Further modes can also be found on the datasheet.

For communicating the network nodes, a regular commercial household switch was used, since the focus of the research is instead on the amount of extra work necessary to implement the cryptographic schemes. Besides that, the architecture proposed utilizes a dedicated network for safety-critical systems, isolated from lower priority messages. The network will not be congested unless a design flaw is present. A correctly dimensioned network for such an application will have no collisions or congestion present, as covered in section 2.4.

3.2.2 Software

3.2.2.1 TI-RTOS

Following the unified approach mentioned before, a common run-time software is utilized on all the nodes in the form of a RTOS, which is designed to be lightweight and built for meeting deadlines for hard real-time applications. To perform this task, the RTOS is required to be deterministic. As mentioned in the previous section, that means the execution of a given code that does not depend on external responses or resources has to be executed predictably every single time.

In this work, TI-RTOS(TEXAS INSTRUMENTS, 2016b) from TI, which is free to use with the development board chosen, was used. In the work done by Buckl et al. (2012) , for example, FreeRTOS (LTD., 2016) is utilized for being open-source and frequently employed in

this area. Since it supports a broader range of hardware devices, it is more portable. Leading innovative companies, such as Tesla, also have the FreeRTOS as one of the components of the architecture of the CAN/Ethernet Gateway of their cars, as described in Lab (2016). TI-RTOS is free to use, but only with TI hardware and currently also open-source. However, the primary reason to choose TI-RTOS over FreeRTOS, which is also supported, are the drivers for the cryptographic module. It is possible to adapt the software so that the FreeRTOS can be utilized. However, since this work's objective is only to validate the latency performance while using cryptography for safety-critical systems, the additional work was not justifiable.

The TI-RTOS has a modular nature, through which the developer may choose which components of the OS to enable, further optimizing the memory footprint, having available several different services. To exemplify, let us take the use of dynamic memory allocation or the IP layer of the network stack. If they are not necessary, the developer may remove them from the building process in a simple configuration file or, in the case of TI-RTOS, by just using a graphical user interface to select only the necessary boxes.

Instrumentation tools that evaluate the code during the execution are also available. By monitoring resource consumption and ensuring that only the appropriate amount is set in the configuration file (not more, not less), these now available resources may be designated to other threads or functions. There is a trade-off when dealing with applications that are so critical regarding resources as the automotive ones. As an example, the memory of the system needs to be shared among various tasks. However, it is not as much as in the standard fully capable and resourceful hardware that people are used to operating due to cost restrictions. Each component of the system has configurations that dictate if they will be enabled or not and if so, the corresponding amount of resources and functionalities. If too much memory is made available for the network stack and the heap, for example, there may not be enough memory left for the tasks' stack and so on. Therefore, careful decisions need to made on a system-wide level according to the available resources.

Priorities, for example, are a vital feature, and even the number of available priority levels can be configured. Since the RTOS handles how they are scheduled, depending on how they were set up, the developer must be careful not to set a higher priority for some less critical function because a lower priority one will stall no higher priority task. That turns easier the co-existence of both critical and non-critical code as long as they are correctly configured.

The instrumentation tools of the developing environment allow for real-time debugging and logging without adding significant overhead to the execution time. They are integrated

into the developing environment by the manufacturer, and they communicate directly with the Micro-controller Unit (MCU) through a USB debugging cable. These features were utilized in this work to measure the latency values of the experiments. The time is measured by acquiring the number of corresponding clock cycles and later converting the value to seconds.

By using a RTOS, when compared to no OS at all, the developer can focus on the desired application desired, and thus the developing time decreases. Besides that, using standard software across the devices facilitates the integration of different subsystems and functionalities in the system as a whole. Task scheduling, priorities, and timers, for example, are handled by the RTOS and the addition of new functionalities, and also the scalability becomes much more straightforward to be implemented. Even power saving features may be enabled if desired. The use of configuration files, such as the one mentioned above, allows the developer to focus on the functionality he wants to accomplish, while also hindering deep code manipulation. The software provided has been thoroughly tested by the development platform's team, which also holds high accountability. Assurances and guarantees need to be offered by them so that the developer can trust the tool. This approach transfers a significant part of the responsibility for the software platform instead of the developer. As mentioned in Macher, Armengaud and Kreiner (2014), the AUTOSAR industry-standard relies heavily on this approach.

3.2.3 Proposed Architecture

Advances in technology, along with the redesigned architecture, give freedom for using more powerful hardware, turning feasible the addition of cryptography on low-latency applications. Through the utilization of a dedicated hardware module for this type of processing, the time necessary for accomplishing such demanding calculations is now reduced.

It is proposed a switched Ethernet architecture consisting of TM4C129E boards as network nodes connected through a conventional switch, as explained before. One node represents the central processing unit of one of the subsystems of a car, and the other node represents the processing unit responsible for some local activity or task, such as controlling the wheels of an axle-less electric car.

The architecture is based on a periodic time frame scheduling in which each task would send and receive messages in their reserved time slot, similar to Time Division Multiple Access (TDMA). A protocol, such as IEEE 1588 PTP, which is supported by the development boards, could be used to complement the network to synchronize the clocks of processing modules.

That is necessary to ensure that the corresponding reserved time slots are triggered at the same moment across different devices.

No process for establishing a connection is involved, only the exchange of messages. Since the connection links are planned using offline scheduling, meaning they are decided in the development phase, the connection establishment process is not required for this scenario. The management of secret keys is also not investigated.

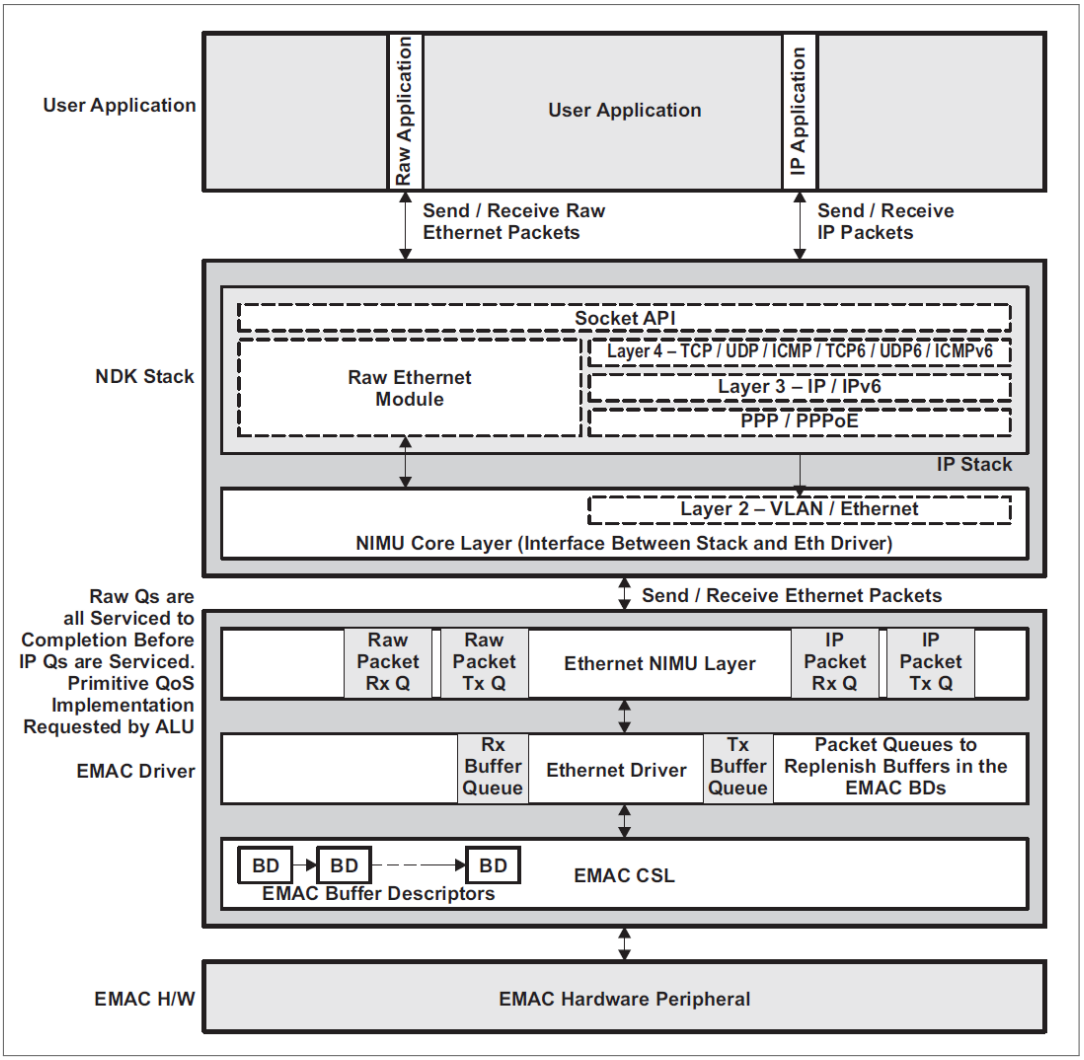
The main feature of the architecture is to securely exchange data between the processing units while respecting a maximum end-to-end delay of 2.5ms (described in section 2.6) so that it may be employed in safety-critical systems for control data. The network communication is secured by applying cryptographic schemes for simultaneously providing confidentiality, integrity, and authenticity assurances over the data. However, applying cryptography inherently affects the processing performance and the end-to-end delay, even if using a dedicated cryptography acceleration hardware module.

With the objective of minimizing the performance harm due to the introduction cryptography, the network stack is configured to use only Raw Ethernet Frames, which do not implement IP and uses network layer two. One may presume the proposed architecture is contrasting with references that strongly defend an IP-based automotive design as shown in Steffen et al. (2008), but it is a complementary feature. Since the overhead of cryptography is added, the performance using IP would be very tight to implement and each single processing unit would most likely be able to perform a single task or fewer tasks, which is not good financially speaking.

As described in Texas Instruments (2016a), "a Raw Ethernet packet can be defined as an Ethernet packet whose Protocol type (offset 12 in the Ethernet header) doesn't match any of the well known standard protocol types like IP (0x800), IPv6 (0x86DD), VLAN (0x8100), PPPoE Control (0x8863), PPPoE Data (0x8864). The Raw Ethernet Module interfaces with the application and the stack to provide the APIs required in configuring a Raw Ethernet socket, and in sending and receiving packets using it."

Figure 8 points out the placement of the dedicated module for Raw Ethernet inside the NDK Stack. As it can be seen, the figure shows the trajectory taken by a packet that is sent using the Raw Ethernet variation, as well as the trajectory of IP packets. On the first case, the Socket API, shown in the upper part of the NDK Stack, interfaces with the user application and also with the Raw Ethernet Module. It does not go through the IP Stack, whose components are labeled as Layer 4 - TCP/ UDP/ ICMP/ TCP6/ UDP6/ ICMPv6, Layer 3 - IP/ IPv6 and

Figure 8 – Raw Ethernet Module inside the NDK Stack



Source: Texas Instruments (2016a)

PPP/ PPPoE.

The performance is further enhanced in this work by using a no-copy (also called zero copy) variation of the Raw Ethernet frames. By opting for such variation, no copy of the data buffer is done so any additional overhead due to memory allocation due to the copying of the frames is eliminated, as described in Texas Instruments (2016a), and so the buffers are received directly. Other than that, options available for latency minimization of the RTOS were also enabled in the configuration file.

As stated in the datasheet of the TM4C129E board, there is also the possibility to enable the Micro Direct Memory Access (uDMA) Controller, which offloads data transfers from the main processor (Cortex™-M4F) and can perform transfers directly between memory and peripherals. One of those peripherals is the cryptography module. For larger data transfers,

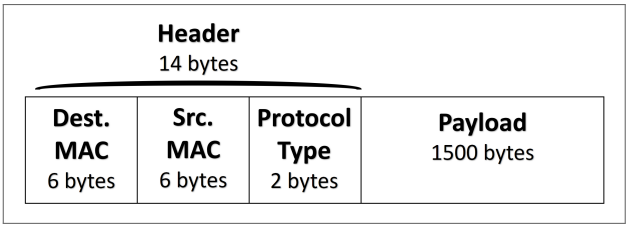
the use of memory can be more efficient by enabling this feature, resulting in faster speeds and consequently lower execution times. However, for smaller data sizes, the time required to activate the interruptions and transfer the control over to the uDMA Controller takes longer than the case when this functionality is not enabled. And since control data for safety-critical systems is usually small, this feature was not enabled.

In that way, cryptographic schemes are applied over network layer two, which is the layer that Raw Ethernet frames use. However, this choice considers that the devices in question are in a safety-critical control data setting. It does not mean the developer has to decide between only using Raw Ethernet frames or only using IP. The IP, in the case of this work, would not be utilized, so it could be removed from the stack as an opportunity to reduce the memory footprint. If required, by correctly using different threads and priority scheduling, all types of data may share the same processing module and co-exist in the same network. Still, as described in section 2.4, preemption is not yet a common functionality for networks. This feature is already available in a typical RTOS, but the network counterparts also need to implement such a feature; otherwise, it will be difficult to guarantee the latency values of control data when extra traffic load with mixed priority levels is introduced. That is the reason this work only uses an isolated network for the safety-critical control data for the time being, and, since it is isolated, there is no need to implement the QoS of the 802.1Q for example.

Usually, the Transmission Control Protocol (TCP)/IP is used to guarantee that the information is correctly received and that the traffic flow is respected. However, when working at the layer 2 level, there is no such feature. Instead, in this work, this is ensured by the use of HMAC and, as also stated in Steffen et al. (2008), by correctly dimensioning the network and using traffic shaping mechanisms, and in the case of safety-critical sections as mentioned in Steinbach, Korf and Schmidt (2011), the use of offline scheduling. That increases the scalability of the system and assures that the proper priorities (which here are handled by the RTOS, but are still configured in an offline fashion giving a time reserved slot for each task to transmit its data) of the tasks are respected.

The Ethernet frame used in this work has two main sections that are shown in figure 9. The first one, composed of 14 bytes, which is fixed in size, is the header. From these 14 bytes, 6 bytes are for the destination MAC (here not the Message Authentication Code) address of the device to which the frame is being sent, 6 bytes for the source MAC address from the device that is sending the frame, and 2 bytes are for the Protocol Type. The second part of the frame is the payload, which can be up to 1500 bytes.

Figure 9 – Raw Ethernet Frame - Header composition: Destination MAC, Source MAC, Protocol Type



Source: The author (2017)

To establish a secure channel, AE is applied over the frame just described. The approach Encrypt-then-MAC was conducted, which according to Krawczyk (2001) and Stallings (2006) has advantages over other common methods for this purpose, such as the MAC-then-Encrypt and the MAC-and-Encrypt. The chosen approach, as the name says, consists of first encrypting the data and, after this step, running the algorithm to obtain the MAC so it may be concatenated to the frame. Even though the MAC calculated may seem exposed because no encryption was performed over it, this allows the system to avoid processing the data in case the MAC verification fails. Any attack that changes the plaintext would also need a matching MAC, whereas by doing the other way around the MAC may give information about the plaintext for example.

The confidentiality (encryption) here is provided by the AES algorithm, by encrypting the payload section of the frame. For providing integrity and authenticity, a MAC is calculated by the HMAC algorithm, using as input the header plus the already encrypted payload. This MAC is then concatenated with the encrypted payload, finishing the composition of the frame. The AES and HMAC are the most popular protocols on the literature for the mentioned objective.

The choice for the specified algorithms for AE aims to provide a decent level of security amongst the algorithms that the hardware supports while covering the required aspects concerning confidentiality, authenticity, and integrity. With that said, there are several possible combinations, which include those three elements and this work's objective is meant to serve as a reference from a performance standpoint related to safety-critical control messages, rather than to point out the best security measure to be chosen. For example, as described in Holle and Lothspeich (2016), the MACsec uses the AES algorithm in Galois/Counter mode, which is also supported by the board chosen for this work. As mentioned in section 2.6, the MACsec does, however, require a supporting OS, whereas the method suggested here, could be implemented even without an OS, which is called a baremetal solution.

It is widespread knowledge amongst the security community that one should always use the

strongest security available while still being compatible with the desired objective. This fact may also depend on the supported protocols available on each hardware platform as well. For example, there is already algorithms based on SHA-3, for instance, which this board in question does not support. So the developer or project manager should always evaluate and research the available options since technology evolves so fast and also security exploits may be found at any moment for any given algorithm. The security strength is also partially related, in most algorithms, to the size of the secret key utilized, as it is for example on the HMAC (BELLARE; CANETTI; KRAWCZYK, 1996). So most of the times, the same algorithm with a larger secret key, that is random, is still safe, whereas by using a smaller key size it is entirely vulnerable. So always use the combination of best algorithm available and the largest key size that fits the desired application.

Having that in mind, it can be verified in the datasheet of the development board, that performance-wise the amount of clock cycles required by each of the different algorithms of the AES for example is practically the same. The difference is more significant when the size of the security keys is changed (128, 192 or 256 bits in the case of the AES). What this means is, that even if you have chosen for example the AES ECB mode, the performance for the same secret key size for another mode, such as the CBC mode, for instance, will be the same. The only exception is the Counter with Cipher Block Chaining-Message Authentication Code (CCM), which requires approximately double the number of clock cycles as the other algorithms. That is because it operates in the manner of the encrypt-then-MAC (STALLINGS, 2006), using a single key for both encryption and MAC algorithm. So it is equivalent to the two operations combined and therefore the double of clock cycles.

The performance information for the HMAC algorithms on the datasheet of the board (TEXAS INSTRUMENTS, 2014) is a different case. The four underlying algorithms supported by the HMAC are MD5, SHA-1, SHA-224, and SHA-256. According to it, the SHA-1 requires 21 more clock cycles than the other three, which have the same clock cycle performance. All four of them are tested in this work. Using only hash operations is a weak approach security-wise when compared to the HMAC (STALLINGS, 2006).

The strength of the HMAC depends as well on the size of the given key. All of the variations used for testing received a 512-bit key. Each of them produces a MAC based on this key. However, each of them produces a MAC of different size.

3.2.3.1 Use case: Steer-by-wire

The architecture proposed can be applied to any automotive safety-critical system which needs to secure its data through AE. It is a generic secure communication platform that can serve various purposes. A use case is a steer-by-wire system for an electric car, similar to the one proposed by Buckl et al. (2012), except that the messages are secured through AE.

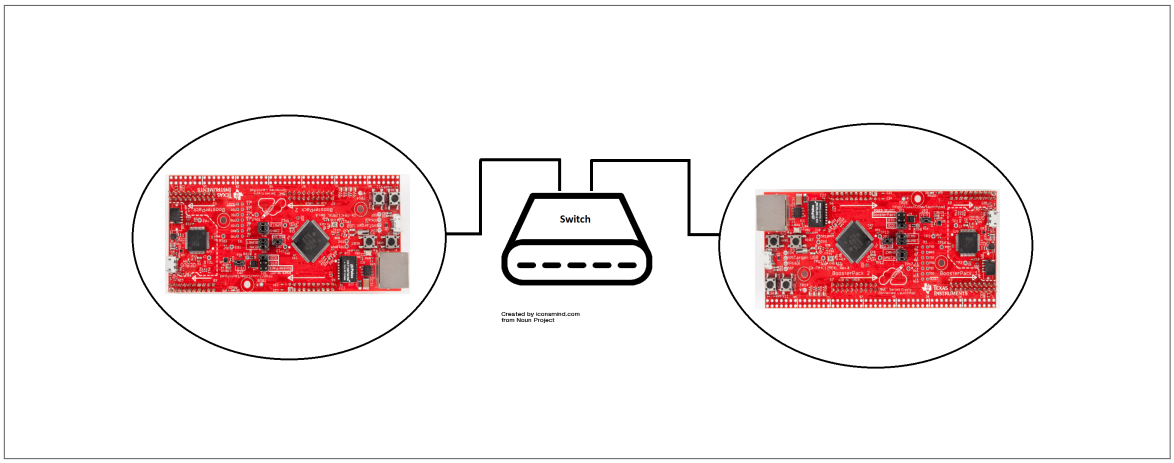
The use-case could be a subsystem of an electric car's architecture in which the car has three TM4C129E micro-controller units. One of the boards would be the main processing unit for this subsystem and the other two as controllers for the front and back vehicle's wheels. A switch connects all three boards. All three microcontrollers would be running threads in a RTOS. One thread would handle the communication, for example, encrypting the data and sending it and later receiving the data and decrypting it, exactly as described before. Since the system relies on a carefully planned offline scheduling and dimensioning of the network the messages will only be sent or received inside the window of time that is reserved and schedule for this. Beyond that, the payload would be used for commands from the main processing unit of the subsystem to the wheels' processing units, so the actuators are triggered. Also, the returning payload may be utilized for the sensor feedback from the wheels' processing units to the main processing unit of the subsystem. Parallel threads could be used to communicate with other devices, such as the central system of the car, but when doing so, a different network has to be used, since this one is isolated and reserved for safety-critical data. As mentioned before, when support for preemption on the network side is available, and the security is robust enough, the safety-critical data may co-exist with other types.

The synchronization of the clocks of the devices may be done by using the IEEE 1588 PTP for example so that the wheels are synchronized. That is crucial so that the reaction of wheels is not uncoordinated.

3.3 DESIGN OF THE EXPERIMENT

This section describes how the experiment was performed and what is the expected behavior which validates the necessary requirements. As mentioned earlier, the main objective for safety-critical control messages is a deterministic behavior that has guaranteed maximum end-to-end latency values of up to 2.5ms. The challenge is to respect the mentioned time value while still implementing cryptography to secure the exchange of messages. Contrarily to several other

Figure 10 – Architecture of the experiment consisting of two Tiva C Series connected by a switch using Ethernet



Source: The author (2017)

scenarios, the worst case time values play the most important role here. Since safety-critical means that the expected deadlines need to be met every single time, it will not matter for example if the mean value for the latency respects the deadline, but some messages surpass it, even if only by a minimal amount.

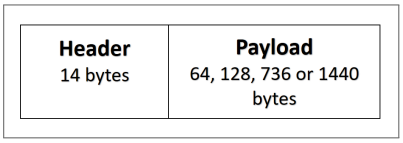
The cases using cryptography will also have their latency performance results compared to two other cases. The first case is using a CRC32 to verify the integrity of the data, a method that has often been employed by the automotive industry until recently due to the reasons already exposed here, since it does not require too much processing power. The hardware accelerator module from the chosen development board also supports CRC, so its calculations were also done taking advantage of this feature, rather than using the main processing unit. The next case would be the plain standard frame with no verification or cryptography at all.

Another point to be investigated when latency is concerned is the size of the messages. As mentioned on section 2.4, in the work done by Lee and Park (2013), one of the measures taken to guarantee the deadlines for safety-critical messages, even though the network was not deterministic on its own, was limiting the MTU. Control messages for safety-critical systems are often short because they are transmitted faster and also because they are usually only simple commands for actions to be taken or simply values measured by the sensors. So, for this objective, the action of limiting the MTU value did not interfere or harm the functionality desired. Taking that into account, four types of payload sizes were chosen to observe the behavior and the impact they cause. Two smaller sizes with respectively 64 and 128 bytes of payload data have been selected since they come closer to the size of a message with the

purpose of being used for safety-critical control. The other two sizes were chosen to evaluate the limits of the Ethernet frame. Since the payload has 1500 bytes as its maximum value, the determined values were around 50% and 100% of the maximum value, respectively with 736 and 1440 bytes. As it will be explained later, the size of the HMAC that will be concatenated to this payload goes up to 32 bytes, and so, some margin for it was reserved.

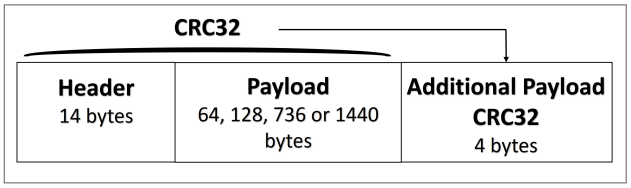
By observing the size of the payload data, the reader may notice that they are all multiples of 16. The reason for this choice is a consequence of how the dedicated cryptography hardware accelerator operates. When processing the data, the module operates 16 bytes (128 bits or 4 words in this platform) at a time. If this value is not a multiple of 16, it is required that the value is padded with zeros. This requirement goes along with the fact mentioned on section 2.5.2.1 regarding the AES algorithm.

Figure 11 – Frame using no verification - No additional bytes utilized



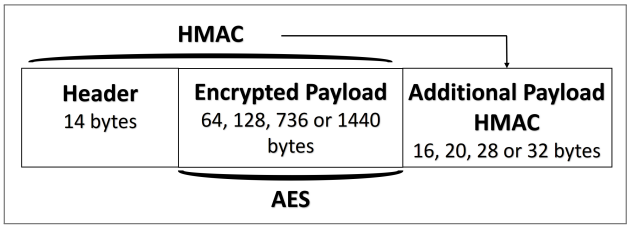
Source: The author (2017)

Figure 12 – Frame using CRC32 - Additional bytes are necessary for the CRC32



Source: The author (2017)

Figure 13 – Frame with Cryptography - Additional bytes are necessary for the HMAC

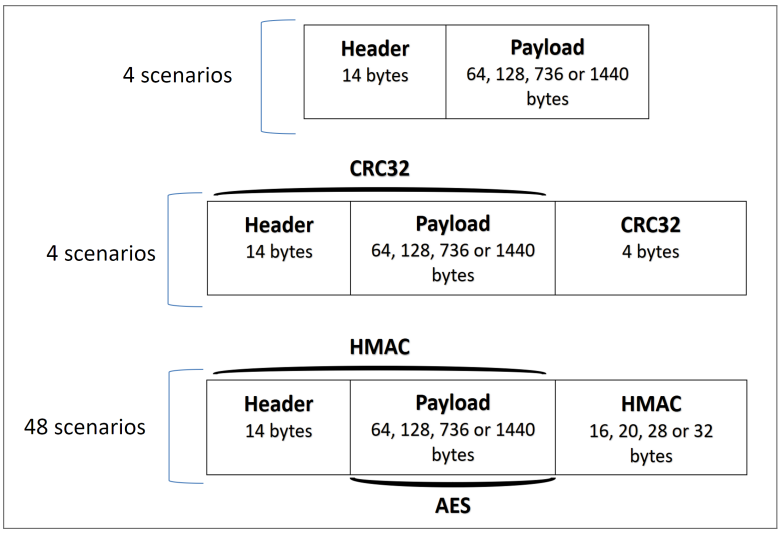


Source: The author (2017)

Figure 11 describes the format of the frame with no verification, and figure 12 by its turn describes the frame concatenated with the CRC32 value used for verifying the integrity of the frame. When varying the size of the payload, each of these two cases results in four

possible scenarios using CRC32, plus four more when no verification is performed as the figures mentioned above suggest.

Figure 14 – 56 Possible Scenarios for the experiment

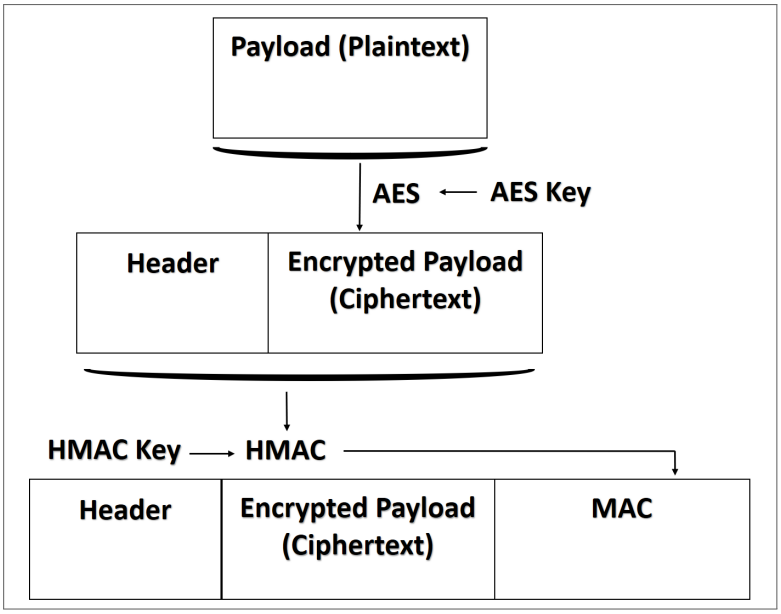


Source: The author (2017)

As for figure 13, it depicts the structure of a frame which has been processed by the cryptography algorithms and is now ready to be sent. Two algorithms are always run for this case. One for the confidentiality and one for the integrity/authenticity. Table 2 shows the possibilities for each of the aspects investigated on the frame when using cryptography with the mentioned algorithms. As is can be seen, this yields 48 possible combinations. So along with the four combinations using CRC32 and the other four with a plain frame, it results in a total of 56 possible scenarios which were tested, as shown in figure 14.

The entire process of forming the frame with cryptography is depicted in figure 15. As mentioned earlier, first the content of the payload is processed by the AES algorithm. This process receives the payload data and the respective key, whose size was chosen among the three possibilities from the central column of the table. The processed data is returned, and there is no alteration to its size. The HMAC algorithm, on the other hand, will always produce a MAC, which depending on the method chosen will have either 16, 20, 28 or 32 bytes as the third column of the table shows. The additional space could have been used for sending more payload data, increasing the efficiency of each frame, or it could have not been used at all, resulting in a smaller frame that is faster to be transmitted, and so, that has to be taken into account as well. In the experiments conducted, the payload size is always fixed, so when no cryptography (or CRC32) is present, no additional data was concatenated.

Figure 15 – Process of forming the frame using cryptography



Source: The author (2017)

Table 2 – Possible combinations with cryptography utilized to validate the experiment

Payload size	AES	HMAC
64 bytes	128 bits	SHA-MD5 (16 bytes)
128 bytes	192 bits	SHA-1 (20 bytes)
736 bytes	256 bits	SHA-224 (28 bytes)
1440 bytes		SHA-256 (32 bytes)

Source: The author (2017)

The aim of the experiment is to investigate the performance harm caused by the overhead introduced by cryptographic methods to an automotive safety-critical system. It follows an experimental design using two TM4C129E boards communicating through a common switch, such as depicted in figure 10. The target to be achieved is a OWD (as called by Abdou, Matrawy and Oorschot (2015)) lower than 2.5 ms, which corresponds to the adopted maximum end-to-end delay value when using safety-critical control data. That corresponds to the process of the frame formation regarding the steps required to apply AE, the transmission, and propagation of the frame, and the receiving time, including verification of the validity of the cryptography and error checking. It is important to separate here the complete RTT from the OWD, because for a lot of cases, one cannot calculate the OWD by measuring the RTT and dividing it by two, as described in section 2.5.2.3.

The evaluation of the synchronization of clocks was already investigated in other studies, as described in sections 2.4 and 2.6. Therefore, it is not necessary to employ the PTP initially,

replicating its functionality for the experiment as the focus of the present work is the overhead caused by cryptography. Naturally, following the offline scheduling demands, the necessary execution window for the clock synchronization needs to be taken into account and have its time slot reserved as well. But since the use of the PTP is a distinct component, whose performance is not directly affected or related to the introduction of cryptography, their latency requirements may be evaluated separately.

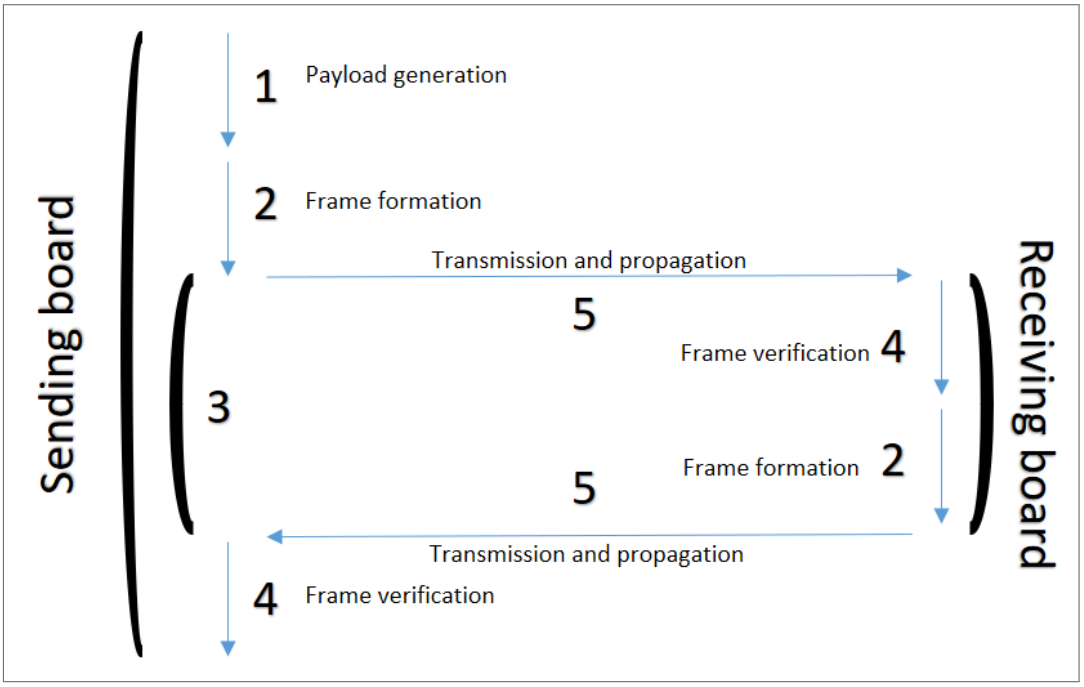
The number of periodic tasks tested was one. Using only one task is not a problem if correctly dividing the time frame and respecting the time necessary for each task, just as explained with the PTP. Once the experiments yield promising results, meaning that the 2.5ms time constraint still has enough available time left for taking into account the additional time required for the synchronization of the clocks of the devices involved, then a future step could be the evaluation of the fully functioning system, including the PTP functionality. Otherwise, there is no point in implementing the synchronization if the cryptography calculations alone exceed the required latency constraints of a safety-critical control data system.

The firmware of both boards consists of two main sections. One of them receives a frame, verifies the HMAC, and decrypts the payload data using AES. The second section prepares the data to be sent by encrypting the payload using AES, generates the corresponding HMAC and sends the message.

One of the boards is denoted the sending board, which produces the initial message, and the other the receiving board. The payload content generated on the sending board was random, only to prove the functionality of the cryptographic algorithms and the network latency values. Other than the initial creation of the payload content at the sending board, both boards execute the same exact code, but with their orders inverted. The sending board first prepares the frame and sends, so that later it receives it back and run the verification. The receiving board by its turn first receives the frame and runs the verification so that after it prepares the response frame to send it to the first board. The receiving board uses the same payload created by the sending board after receiving it and running the verification, and therefore the creation of the payload content itself is only done by the sending board every iteration. So no additional processing is done by the receiving board to create the payload content. There is also no decision making, other than the ones related to the cryptography and the communication. This work focus only where the cryptography has an impact, so any further decision making, such as processing the data from sensors and choosing the corresponding actuation, is not included and should not be taken into account, because they are not part of the end-to-end

delay calculation as described on section 2.5.2.3.

Figure 16 – Path taken for each iteration of the experiment. The numbers represent the parts of the code in each iteration. The flow of the code follows the arrows. The number 3 is equivalent to the parts 5, 4, 2, 5 combined.



Source: The author (2017)

If for example, we would attribute numbers to designate the parts of the code consisting of one iteration of message exchange, which runs on each of the boards, the sending board has four main portions, plus the transmission and propagation, as shown in figure 16. Part 1 would be the creation of the random payload data. Part 2 would be the preparation of the frame for sending, which includes the encryption of the payload data and the calculation of the HMAC to be concatenated. Part 3 would be from the moment the frame is sent and transmitted to the receiving board, passing through the switch, until the moment it returns and is received by the initial board, after being processed and sent by the receiving board. Then, part 4 consists of checking the HMAC of the frame which just arrived and decrypting it, which would then lead to the beginning of a new iteration. So from the perspective of the sending board, the sequence would be 1, 2, 3, 4. Moving on to the standpoint of the receiving board, it already starts waiting for the frame from the sending board. After the frame is received, starts the same code executed by section 4. After it is completed, the same code as the part 2 is then executed, which ends when the frame is sent back to the first board. So the sequence would be 4, 2.

This sequence just explained is important for understanding how the measurement of time

was organized. As it can be seen on figure 16, the part denominated as 3 is when the sending board sends the frame to the receiving board and waits for the answer. It is comprised of the transmission and propagation towards the receiving board (5), plus the whole processing at the receiving board ($4 + 2$) and then the transmission and propagation returning to the initial board. If we designate the number 5 as the one-way transmission and propagation, then part 3 can be translated as 5, 4, 2, 5. Since the path taken regarding network hops is the same, and it is isolated, also, the size of the frame sent is the same in both ways and the measurement of the time is done on the boards, then both ways of transmission are considered the same, each one designated by 5.

The values measured on the sending board were the following: one corresponding to the numbers 1 through 4 and also the measurement of part 3. On the receiving board, the value measured was from the moment the message is received, until it leaves the board again, which corresponds to the sections 4 and 2. Having measured the part 3 on the sending board and parts 4 and 2 on the receiving board, it is possible to isolate the transmission and propagation(5) by subtracting the parts 4 and 2 from 3, finally dividing it by two for the transmission and propagation of one way. Since the offline scheduling for the frames has a reserved time window for every frame in this isolated network, then there is no queuing time.

Having all the sections isolated, the one-way delay for the forward path corresponds to the numbers 1, 2, 5, 4. The way back by its turn, corresponds to the numbers 2, 5, 4, since there is no payload generation and the same content received is used as the payload of the response. The frame sent back, when not encrypted, only changes the content of the header, by exchanging the source and the destination values.

To evaluate the overhead introduced by cryptography, the experiment is conducted first applying the combinations of authenticated encryption schemes, later on applying only the CRC32 and finally without applying any method (a plain frame), as described on section 3.2.3.

Each run of the experiment consists of sending one frame and receiving another frame back from one MCU to the other for 120 times. After this is done, the program is blocked for 10 seconds and then proceeds to the next run of the experiment. For the purpose of acquiring statistical significance, the experiment is replicated 30 times, so each of the scenarios totals 3600 messages exchanged for each way of the transmission, i.e. 7200 messages total. That is repeated for each of the 56 scenarios mentioned earlier, which result in 403,200 messages.

From the results, it is calculated for each section the mean, the minimum value, the

maximum value (worst case), the standard deviation and the median value. Also, a statistical analysis is done with a combined approach using a non-parametric test. More details will be covered in the next chapter.

3.4 CONSIDERATIONS

This chapter described the architecture for secure communication that was used to validate the experiments which are necessary to determine the behavior of the application and compare it against the requirements of a safety-critical control data.

On the next chapter, the results of the experiments are displayed and discussed. Also, a statistical analysis is performed to reinforce the findings, and also, a comparison against the expected deterministic behavior is conducted.

4 ANALYSIS OF THE RESULTS

4.1 INTRODUCTION

This chapter will display the results obtained from the experiments described on Chapter 3. First, the results will be compared against each other and discussed with the aid of graphs from different scenarios and different grouping to give different points of view. For the full detailed latency values, along with the values for the corresponding mean, minimum (best case), maximum (worst case), standard deviation and median of each of the scenarios and measurement sections are available for reference on the Appendix 5.

Later, a statistical analysis will have its results displayed for the four scenarios, which are the cases where the best security algorithms from them ones tested are applied for each of the four payload sizes. Lastly, a brief discussion will be held to comment on how the expected deterministic behavior of the application compares against the obtained results.

4.2 RESULTS MEASURED

As mentioned earlier, the primary goal to be attained is the estimation of the impact that is suffered regarding the latency performance for a safety-critical control data system setting, after cryptographic protocols for authenticated encryption are utilized. The desired maximum end-to-end latency of 2.5 ms for such an environment was already discussed in the previous sections. The OWD is measured according to what is displayed on figure 16, which corresponds to the sections 1, 2, 5, 4 of said picture, and it is expected that the resulting latency value is smaller than 2.5 ms. A scenario where the complete iteration of the application is measured will also be discussed. For such path, which corresponds the entire sequence described on figure 16, the latency to be outperformed is twice the end-to-end delay since it corresponds to two times the full path in each direction. As discussed on section 3.3, it is expected that the OWD forwards is greater than half of the latency value for the combined forwards and returning paths, and that is why they are both measured.

The experiments are conducted in three main categories, being them the following:

1. Applying Authenticated Encryption

2. CRC32 integrity verification (also using the dedicated CCM)

3. Plain frame (no cryptography or integrity verification)

As previously exposed, item number 1 has 48 different scenarios, as already displayed on table 2. The table is repeated here, for a better visualization, as table 3. Both the items 2 and 3, vary only in payload size, resulting in 4 possibilities for each of the cases, that is, a total of 56 cases along with the item 1.

Table 3 – Possible combinations with cryptography utilized to validate the experiment

Payload size	AES	HMAC
64 bytes	128 bits	SHA-MD5 (16 bytes)
128 bytes	192 bits	SHA-1 (20 bytes)
736 bytes	256 bits	SHA-224 (28 bytes)
1440 bytes		SHA-256 (32 bytes)

Source: The author (2017)

The item number 2, CRC32, also uses a dedicated hardware module to accelerate its calculations. Each run of the experiment consists of sending one frame and receiving another frame back from one MCU to the other for 120 times. For the purpose of acquiring statistical significance, the experiment is replicated 30 times, so each of the scenarios totals 3,600 messages exchanged for each way of the transmission, i.e. 7,200 messages total. That is repeated for each of the 56 scenarios mentioned earlier, which result in 403,200 messages exchanged.

When not using the cryptographic schemes, in every round the data is received, and a response is built and sent back to the other board. When using it, in every round the data is verified by the HMAC algorithm and decrypted, and a response message is then built by using the same steps through the use of AES and HMAC. The message is then sent back to the original unit where the verification is carried out again, so the next round can begin.

Following the structure of table 3, the labels for the scenarios used on the graphs and on the spreadsheet present on the Appendix section follows the format "X/ Y/ Z". Where "X" is the size in bytes of the payload used for that scenario, which does not include the later concatenated MAC, "Y" is the size of the AES secret key used and "Z" is the algorithm variation used by the HMAC. Figures 13, 12 and 11, describe the frame structure of items 1, 2 and 3, respectively.

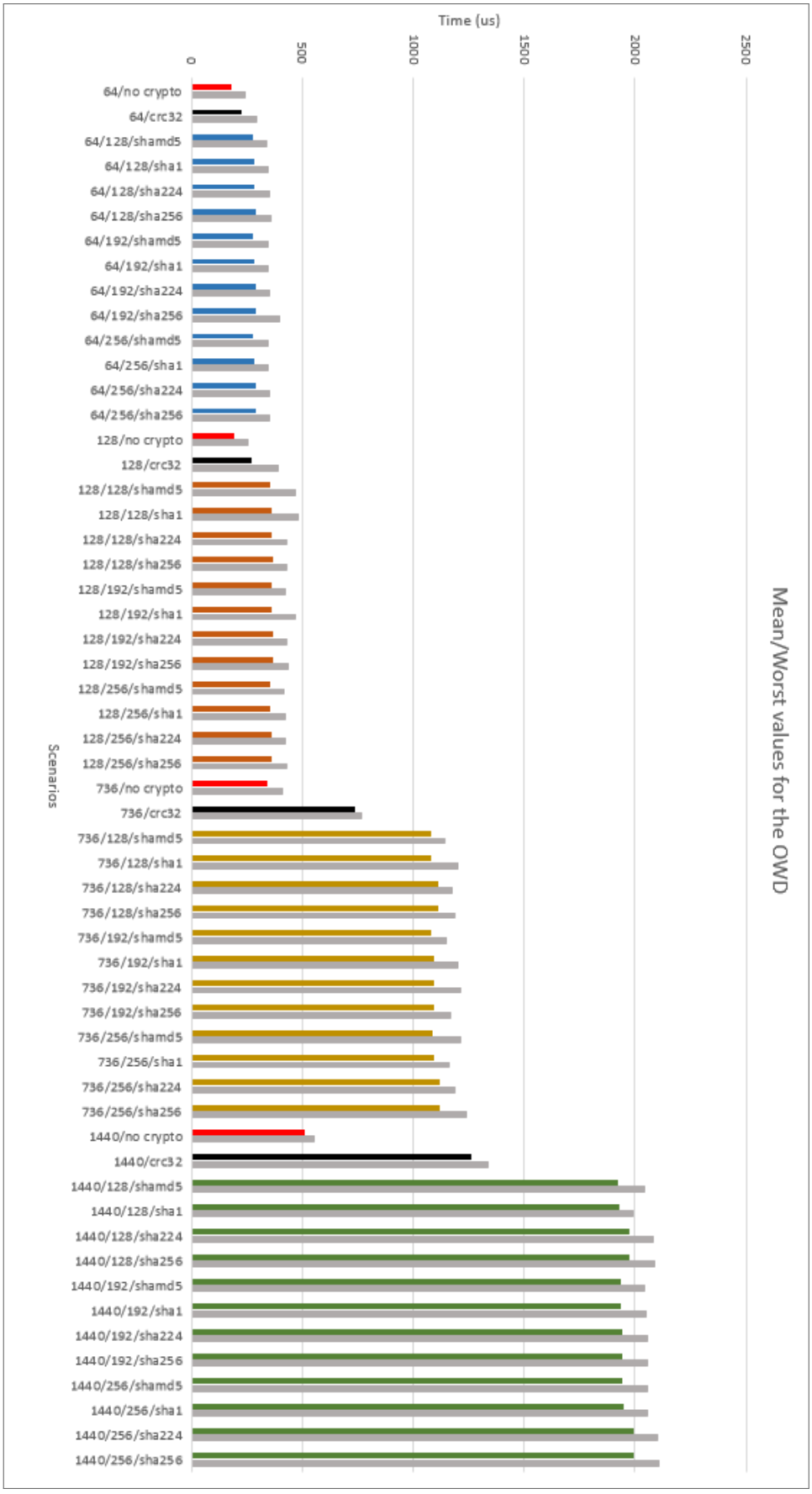
4.2.1 One-Way Delay Case

This section discusses the OWD case, which has its results for the mean and worst case values (the right side value next to each scenario) for each of the scenarios plotted on figure 17. As mentioned before, the maximum latency value of 2.5 ms is the goal to be reached. The graph shows us the scenarios in increasing payload size order.

The labels denominated "no crypto" refer to the case when the plain frame is transmitted with no cryptography or verification. As expected, those scenarios always have lower latency values than any other case using the same payload size, followed by the scenarios where the CRC32 is used. All of the scenarios were able to fulfill the objective of having latency results shorter than 2.5 ms, even the on the scenarios where the payload size of 1440 bytes was utilized, which was a quite surprising result since the uDMA was not enabled.

The results found in Daoud et al. (2006) of 463 μs could be met by the 64 bytes payload size, with a maximum latency value (worst case) of 395 μs of all the scenarios which had this payload size, which was the scenario 64 bytes/ 192 bits/ sha256. The worst mean latency value for this payload size was 289 μs, which was the scenario corresponding to 64 bytes/ 256 bits/ sha256.

Figure 17 – Mean/Worst Case values for OWD for all Scenarios



Source: The author (2017)

When using 128 bytes payload size, the result was very close to the one with 64 bytes, with the worst case of 481 μs (128 bytes/ 128 bits/ sha1) from all the scenarios with 128 bytes payload. The worst mean from these scenarios was 367 μs (128 bytes/ 192 bits/ sha256). The results there were achieved with mixed traffic (traffic with different priorities sharing the same network), whereas here, the traffic was isolated, but on the other hand, the AE was applied.

Summing up, the latencies for the case described in this section (OWD) varied according to table 4. This variation is across all the scenarios which are applying AE with the same payload size.

Table 4 – OWD Mean, Minimum and Maximum latency values variation among the different scenarios using AE. Time values are in microseconds (μs)

Payload size	Mean	Minimum	Maximum
64 bytes	[277, 289] μs	[256, 274] μs	[340, 395] μs
128 bytes	[351, 367] μs	[333, 351] μs	[418, 481] μs
736 bytes	[1077, 1121] μs	[1056, 1102] μs	[1140, 1240] μs
1440 bytes	[1923, 1993] μs	[1907, 1979] μs	[1991, 2110] μs

Source: The author (2017)

Taking a wide glance at the graph, it stands out that the characteristic which has more significance over the latency performance is clearly the payload size since the scenarios with the same payload size are all approximately at the same latency range.

Since this application is designed for a safety-critical objective, the most significant value to be evaluated is the worst case (maximum) value, since no deadline may be missed. Even if the mean were far lower than the critical value allowed, but the maximum value would cross that limit, then the system could not be employed for such a purpose. From table 4, it can be seen that the worst of the maximum values was 2110 μs, which still has a margin to the critical maximum value permitted of 2.5 ms.

Returning to the graph, it can be noted that in general, the maximum (worst case) values were relatively close to the respective mean values, especially for the larger payload sizes. As it can be seen from appendix 5 on the Appendix, the standard deviation was small, and the mean values were always very close to the minimum values (best case).

More about this difference between the minimum (best case), mean and maximum (worst case) values will be discussed in sections 4.3 and 4.4. The evaluation grouped by different payload sizes, AES type and HMAC type will be carried out in the next sections.

4.2.2 Complete Iteration Case

The case for the complete iteration corresponds to one full iteration of the application designed, which was explained on section 3.3, and represents the RTT, that is, the path taken from the sending board, all the way through to the receiving board and then receiving back the generated response. That represents the complete path shown in figure 16.

The graph corresponding to the results for this case is showcased on figure 18. Since the described trajectory for the frame goes from one end of the network and backward, the maximum allowed latency delay is equivalent to two times the end-to-end delay, which results in 5 ms. The structure of the graph follows the same one displayed in the last section.

By looking at the graph, as expected, the latency values for each of the scenarios were not equal to double the amount of the OWD case, giving an even larger margin from the maximum value permitted for this case. The behavior of the latency across the different scenarios continues to follow the same one presented in the OWD case. All scenarios were able to fulfill the requirement of being lower than the maximum allowed critical latency value of 5 ms.

The equivalent of table 4 for this case is displayed as table 5. What stands out, is that the scenarios for the three smaller payload sizes were not only able to achieve latencies lower than the required for the round-trip but also lower than the maximum OWD permitted. Both of the cases, this one, and the last section, show, up to a certain point, predictable behavior.

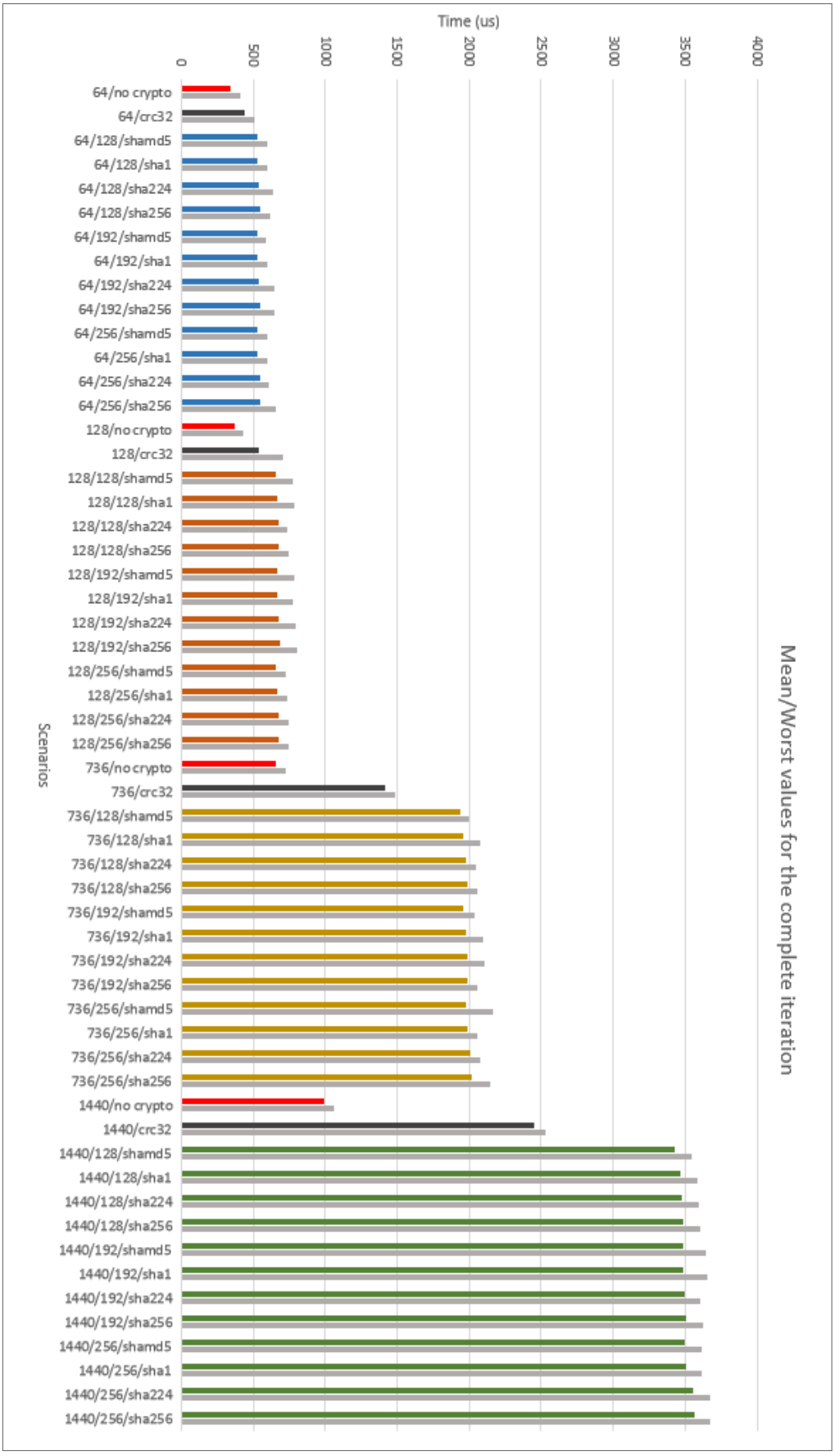
As in the previous section, more aspects about the results presented will be discussed in the next sections.

Table 5 – Complete Iteration Mean, Minimum and Maximum latency values variation among the different scenarios using AE. Time values are in microseconds (μs)

Payload size	Mean	Minimum	Maximum
64 bytes	[526, 549] μs	[481, 521] μs	[591, 655] μs
128 bytes	[657, 686] μs	[625, 653] μs	[728, 804] μs
736 bytes	[1938, 2021] μs	[1918, 1984] μs	[2001, 2164] μs
1440 bytes	[3423, 3561] μs	[3395, 3533] μs	[3546, 3678] μs

Source: The author (2017)

Figure 18 – Mean/Worst Case values for Complete Iteration for all Scenarios



Source: The author (2017)

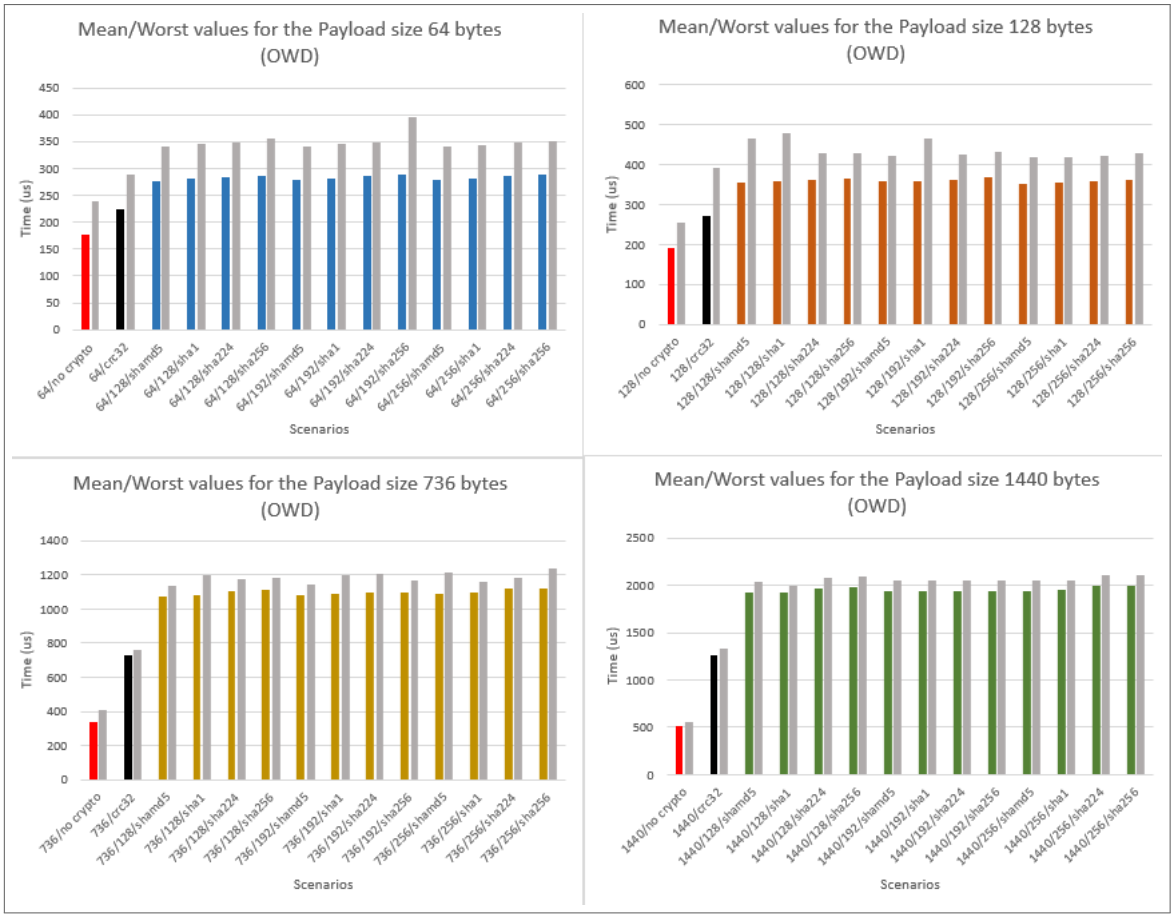
4.2.3 Payload Size

Here the discussion will be focused on how the size of the payload affects the performance of the latency. Some of the aspects were already discussed in the two previous sections, but there are still some details to be evaluated.

As reported before, the payload is the aspect which has the greatest impact on the latency performance, and as discussed on section 3.3, limiting the MTU is one of the methods utilized to guarantee latencies on non-deterministic networks.

Using a large payload size increases the efficiency of the Ethernet frame since more bytes of useful payload data can be transmitted. Limiting the MTU goes directly against frame efficiency. The efficiency of the frame is calculated by dividing the amount of useful data by the size of the full frame, including the header, and in this case, when AE is applied, the size of the MAC is also non-useful data. As an example, for the case with 1440 bytes payload size and applying the HMAC SHA-256, the full frame size is 14 (header) + 1440 (payload) + 32 (HMAC) = 1486 bytes, which results in approximately a 97% efficiency. The same case for example with the 64 bytes of payload, results in a 58% efficiency. For the case of safety-critical control data, this is not a problem, however, for others, such as multimedia, the more efficient, the better.

Figure 19 – Mean/Worst Case values for OWD grouped by payload size

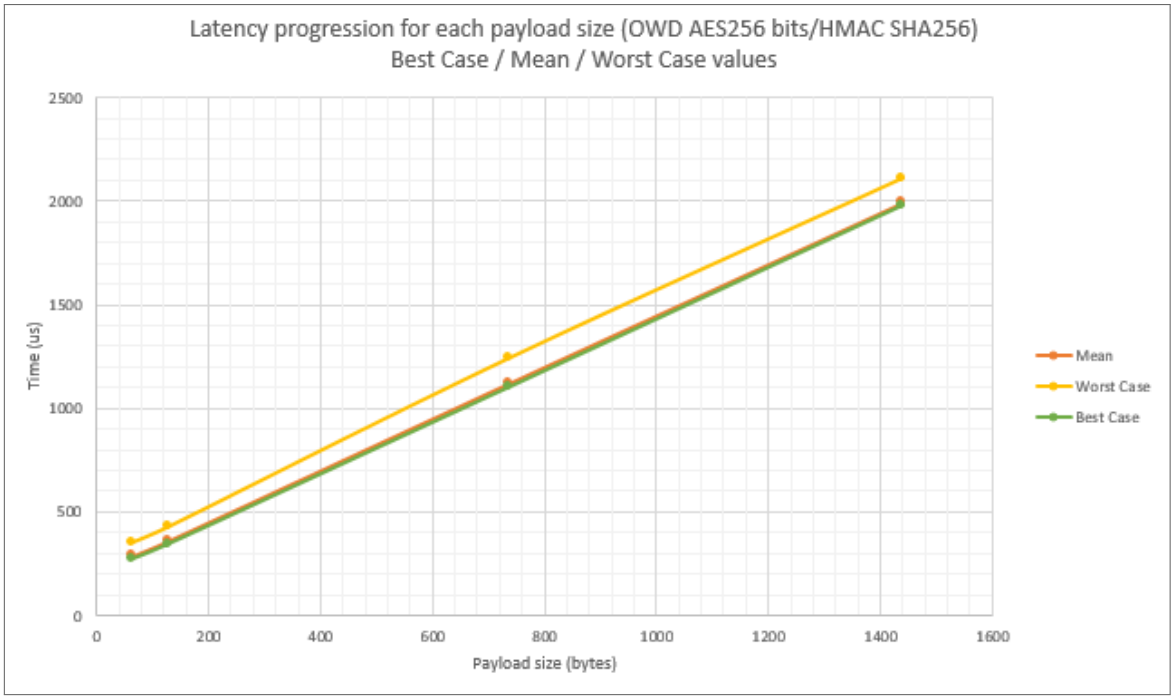


Source: The author (2017)

Figure 19 groups the scenarios by payload sizes for the OWD case for a better perspective of the mean and maximum values.

From figure 20, the progression of the latency values as a function of the payload size demonstrates a linear behavior. From that picture, it can also be inferred, by extrapolating the lines, that it would be necessary a payload size greater than 1600 bytes to surpass the critical maximum allowed latency of 2.5 ms. The graph shows three lines, being the lowest one the minimum values to the respective payload size for the OWD of the scenarios with AES 256 bits and HMAC SHA256. The middle line is the mean for each of the corresponding scenarios and the highest one plots the maximum values.

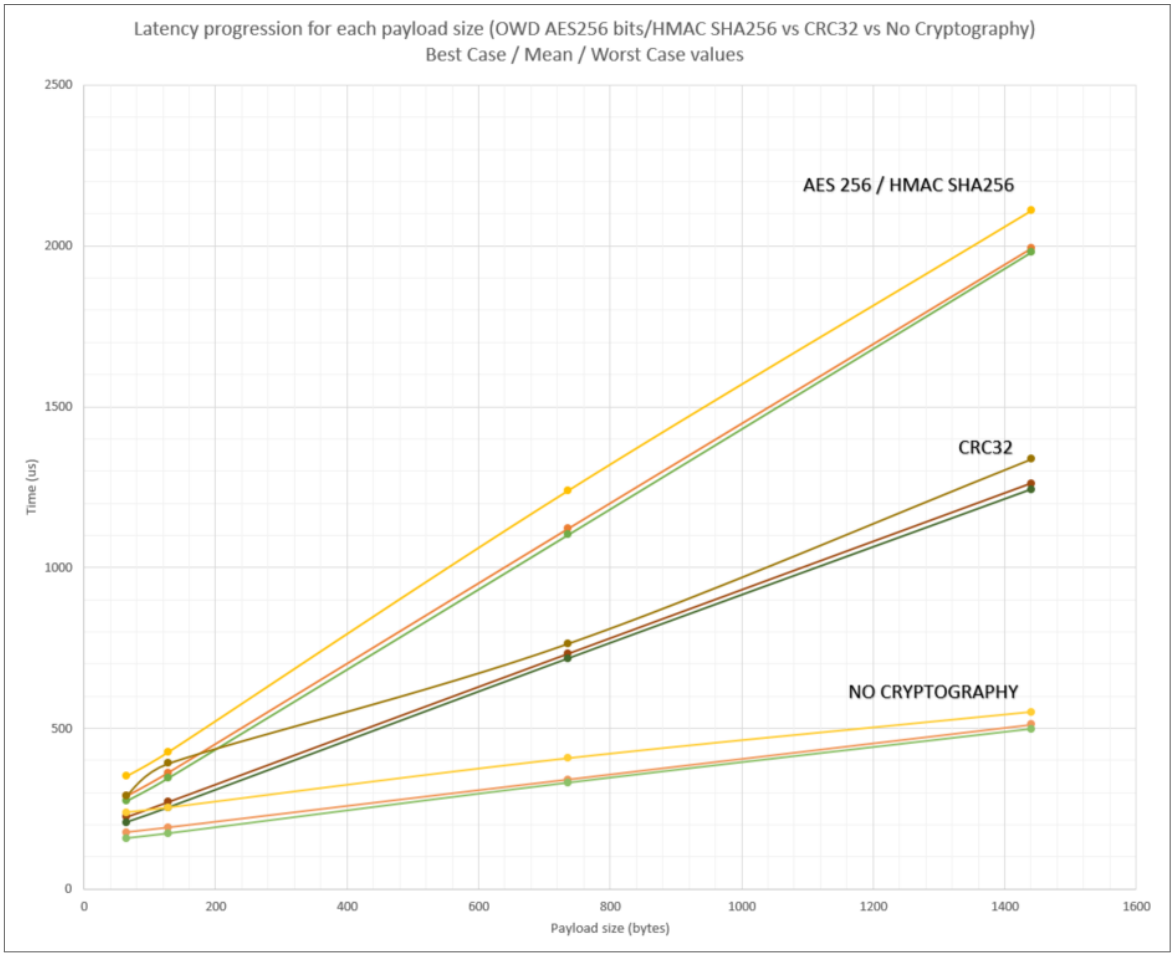
Figure 20 – Progression for Best/Mean/Worst Case values with Cryptography for OWD payloads



Source: The author (2017)

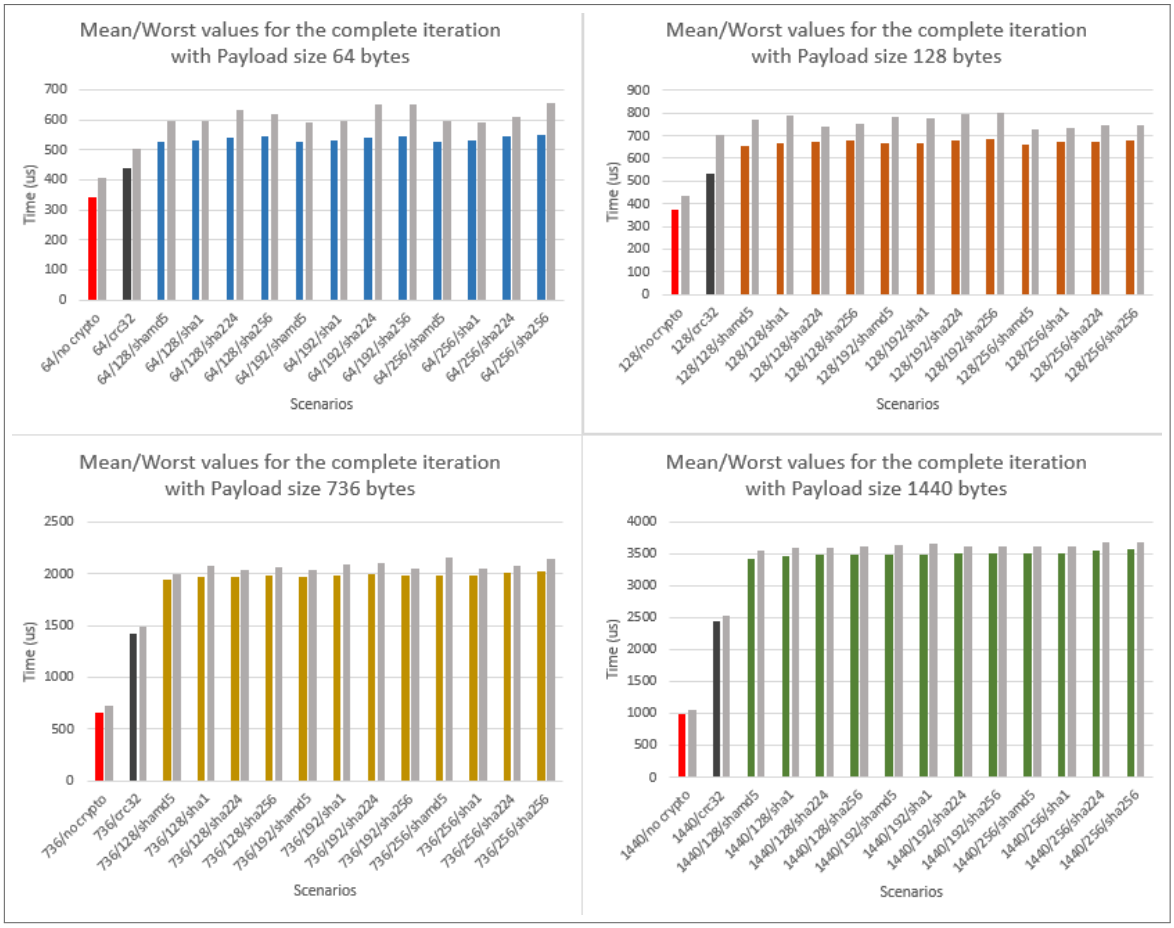
The bigger the payload is, the more the scenarios with cryptography distance themselves from the CRC32 variations, as well as the CRC32 ones also distances themselves from the plain frame variations. That can be seen on figure 21, where the progression of the latency values as a function of the payload is also plotted for the scenarios with CRC32 applied and for the plain frame with no cryptography. It can be seen how the slope of the lines is different between the three cases. It is peculiar though, how the plot of the maximum values for the CRC32 case behave, having a peak on the value for the 128 bytes scenario. Other than this case, all other have a linear behavior. Figure 22 groups the scenarios by payload sizes for the complete iteration case displaying the mean and maximum values, just like figure 19, showing a similar behavior.

Figure 21 – Progression for Best/Mean/Worst Case values for OWD payloads



Source: The author (2017)

Figure 22 – Mean/Worst Case values for Complete Iteration grouped by payload size



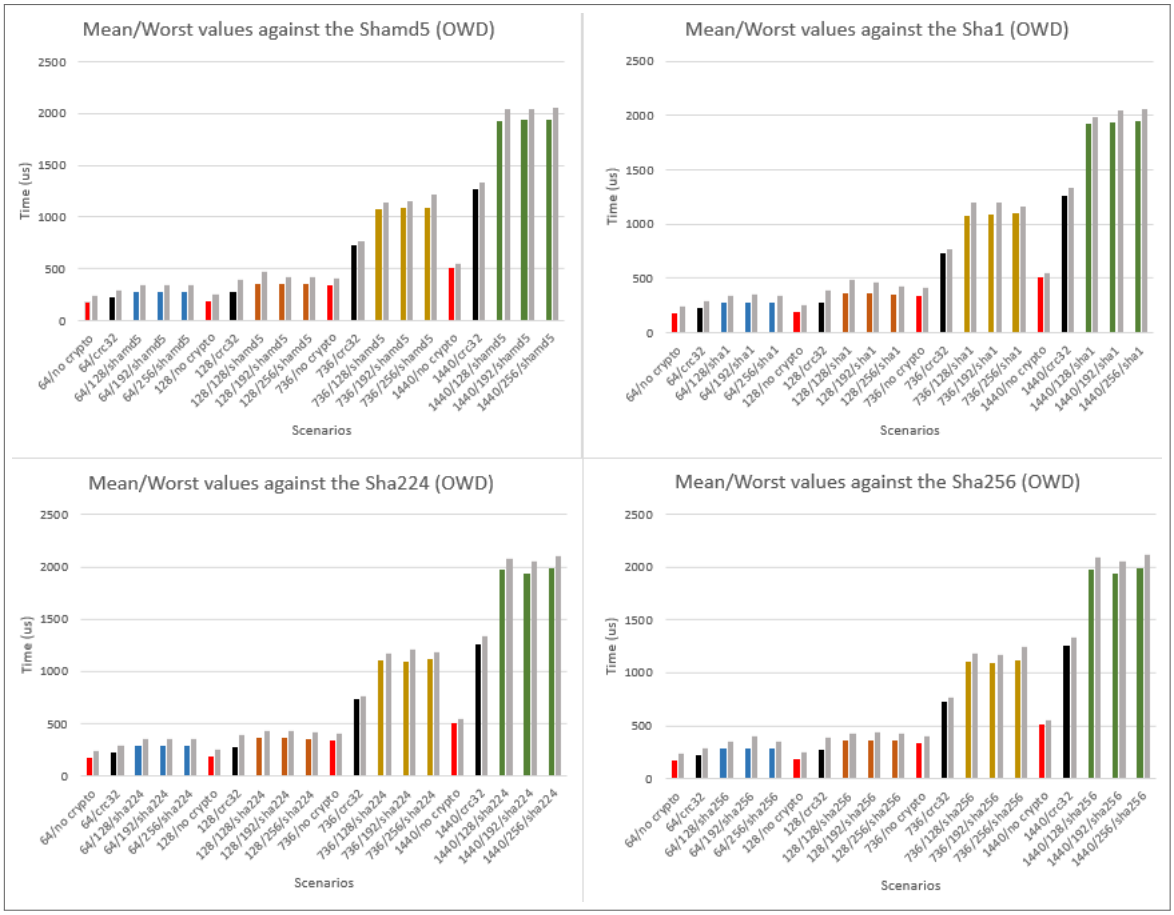
Source: The author (2017)

4.2.4 AES and HMAC Algorithms

This section will discuss how the size of the secret key used as input affects the performance of the AES algorithm. As explained on section 3.2.3, the datasheet of the development board has a chart displaying the cycle count for the AES algorithms and the size of the key is the determinant factor for the number of clock cycles spent. Even so, the difference is not that significant depending on the application.

Figures 23 and 24, even though they are grouped by the HMAC utilized, they allow one to see the impact the size of the AES keys has, along with the graphs that group the scenarios by payload size. Those graphs show that for the application this work is directed to, the performance is virtually the same.

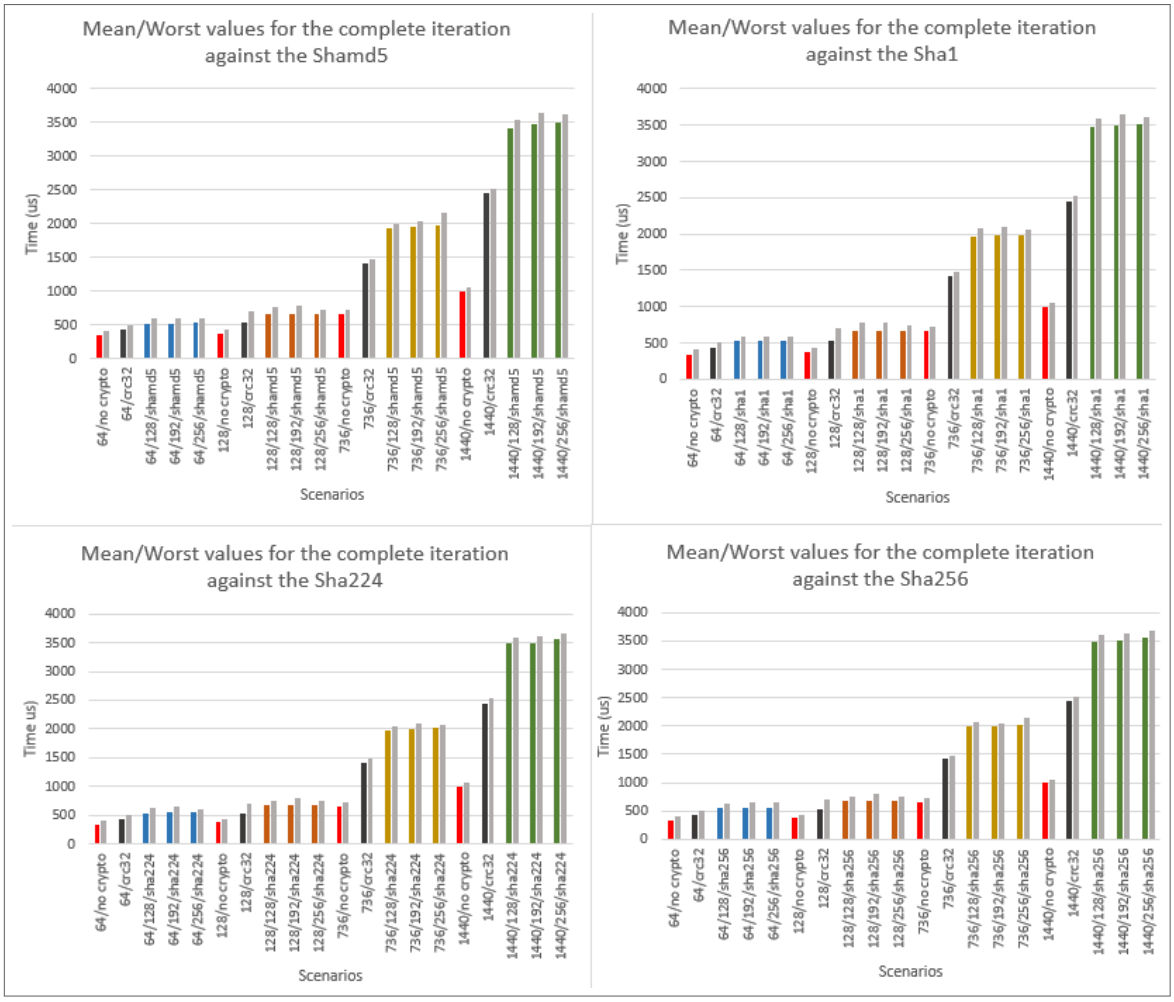
Figure 23 – Mean/Worst Case values for OWD grouped by HMAC type



Source: The author (2017)

From the spreadsheets in the Appendix, it can be seen that in general by looking at the bigger picture, that is, with a macro view, the mean of the results tend to grow, the larger the key is. However, this is not always the case, and especially for the maximum values, there are scenarios with bigger keys which are faster than others with smaller key sizes. More about this will be discussed on section 4.4.

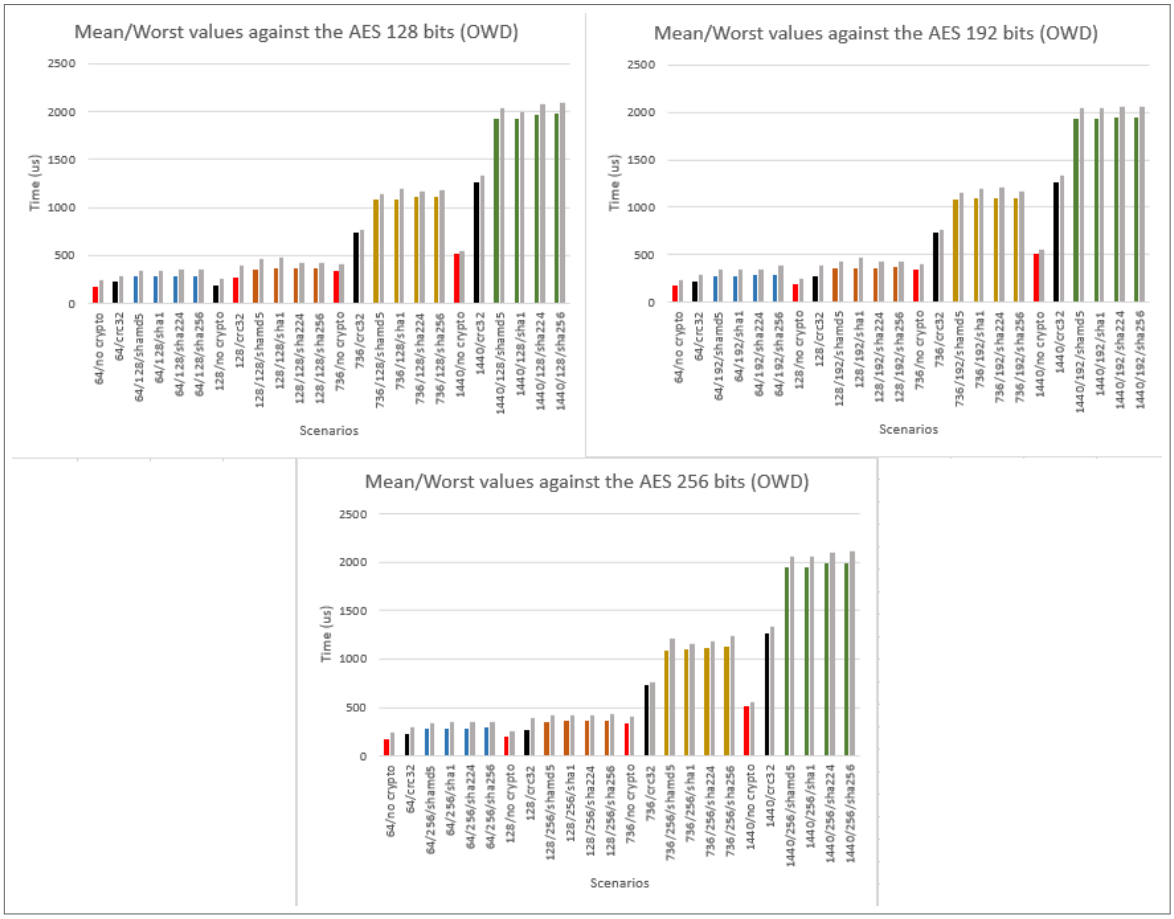
Figure 24 – Mean/Worst Case values for complete Iteration grouped by HMAC type



Source: The author (2017)

Figures 25 and 26 shows us that same behaviour of the AES algorithms is repeated for the different HMAC scenarios. Even though the datasheet shows that the SHA-1 algorithm requires more clock cycles than the other HMAC algorithms, the result experimentally obtained showed that in general, as it can be seen on the spreadsheet in the Appendix, the performance from faster to slower, followed the sequence MD5, SHA-1, SHA-224, SHA-256. As shown in table 3, this is probably due to the size of the MAC produced. The memory operations probably require more time to be executed, and also the transmission needs to send more bytes of data, due to the concatenation of additional information to the end of the frame.

Figure 25 – Mean/Worst Case values for OWD grouped by AES key size

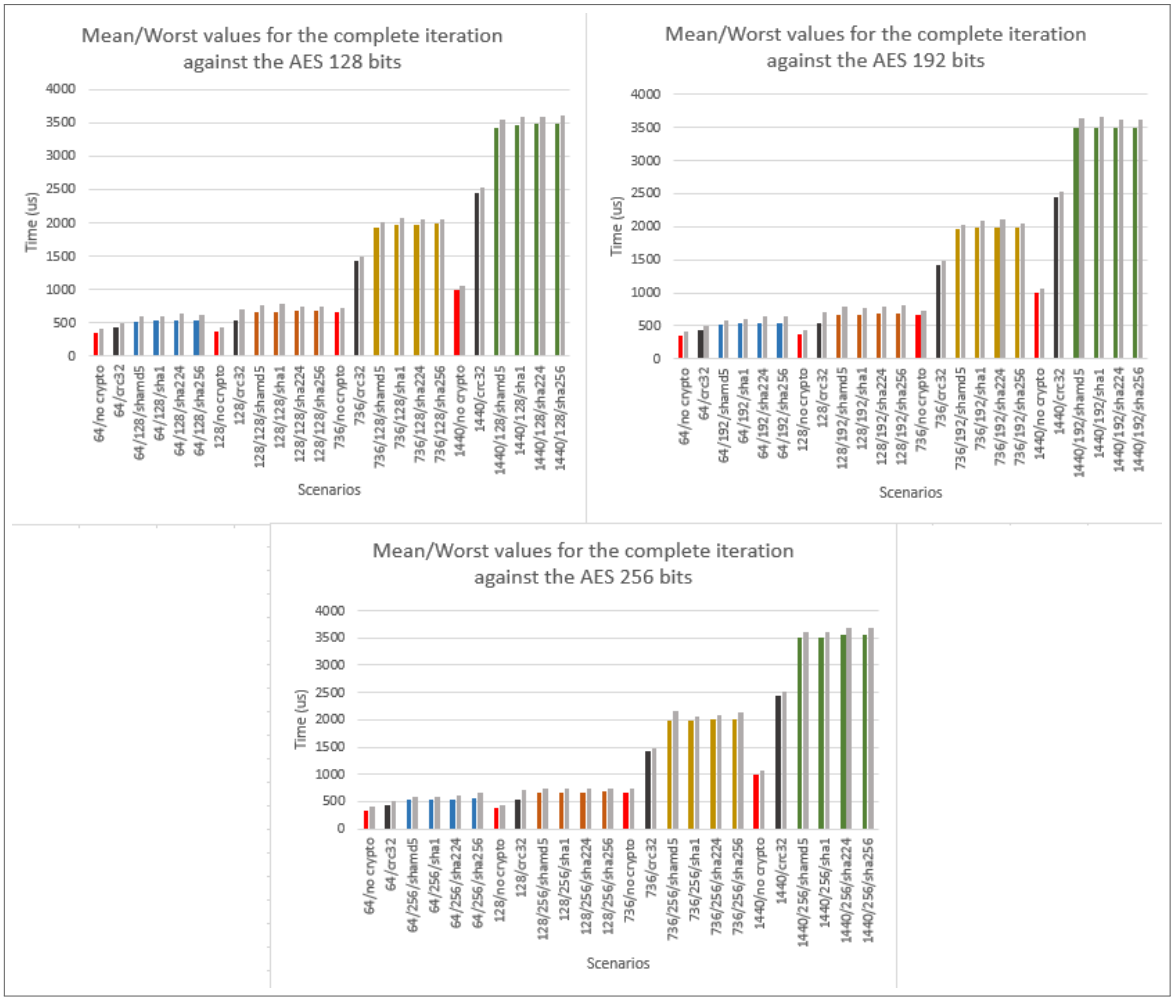


Source: The author (2017)

From the results, it becomes clear that the developer or project manager should always use the best security available since they are not significantly different from each other from a performance standpoint. All of them were able to complete the desired tasks and accomplish the objective in a shorter amount of time than the maximum latency allowed for safety-critical control messages. Also, the testing of several different methods helped to consolidate the behavior of the devices in longer duration tests.

With those results in mind, the best combination of the algorithms tested here would be the use of the AES with a 256 bits key size, along with the HMAC using the SHA-256 variation. However, as commented before, other algorithms could be employed for the AES for example, such as the CBC or other which the developer judges better for their application because they would produce similar latencies.

Figure 26 – Mean/Worst Case values for complete Iteration grouped by AES key size



Source: The author (2017)

4.3 STATISTICAL ANALYSIS

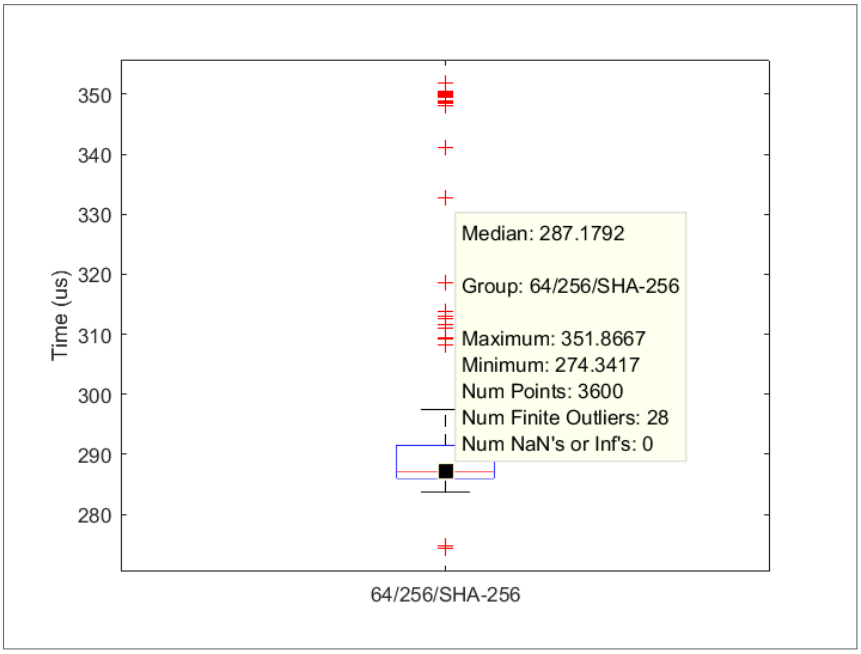
A statistical analysis is a study performed over sample values to infer the parameters of the whole population. Since only samples are obtained, rather than the entire population, it is not possible to be 100% sure that none of the values of the population overcome a defined threshold. However, safety-critical systems are concerned about the worst-case scenario, to ensure that deadlines are not missed. For ensuring that the latency boundaries are statistically lower than a threshold value, a combined approach is utilized. First, a statistical test is performed to demonstrate that the median of the population is statistically lower and distant from the permitted deadline. Secondly, a boxplot is utilized to also statistically prove that the population of results is concentrated around the value of the median, adding to the assumption that the threshold is not probable to be surpassed.

The results obtained from the experiments did not display a normal distribution, and because of that, a non-parametric statistical hypothesis test was first conducted to statistically demonstrate that the median of the population of possible latency results is statistically lower than a threshold value, from the sample obtained from the experiments.

The Wilcoxon Unilateral Statistical Test for a single sample was utilized, and the chosen scenario was the one with the best security level from the ones tested, which would be AES with a 256 bits key size and the HMAC SHA-256, for each of the payload sizes. A confidence level of 99% was chosen since safety-critical systems are concerned. The confidence level represents how likely to be contained within a specified confidence interval, a parameter of the population is.

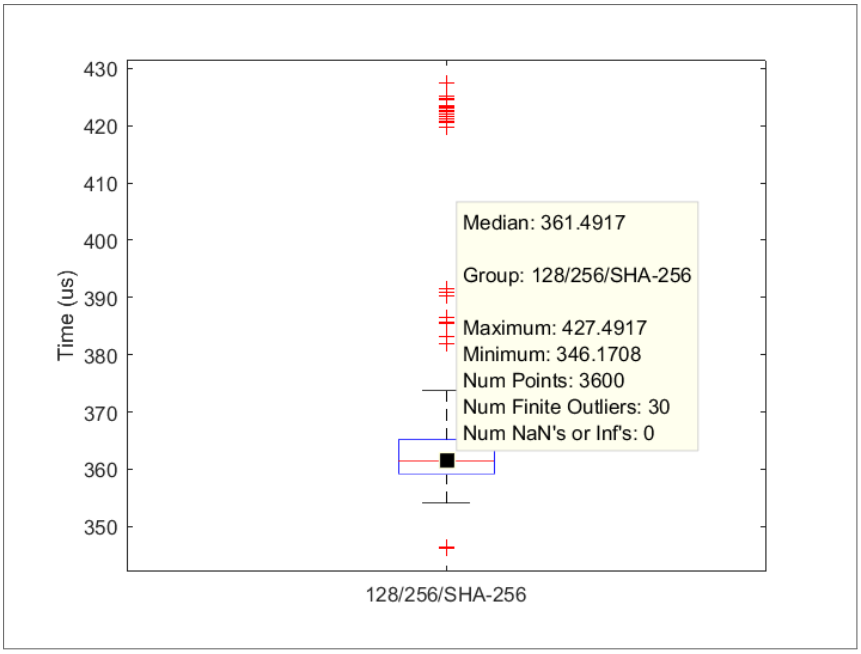
The Wilcoxon ranked-sum test (MONTGOMERY; RUNGER, 2010) is also called the Wilcoxon-Mann-Whitney test. The single sample variation of this non-parametric test will evaluate the null hypothesis on whether or not, a randomly selected value from a sample is less or greater than the parameter of a population from which there is no information regarding its distribution. In this case, the null hypothesis to be rejected is that the median of the population is greater than a chosen value, so we need to reject this hypothesis.

Figure 27 – Boxplot for the OWD of the scenario 64 bytes payload / 256 bits AES / HMAC SHA-256



Source: The author (2017)

Figure 28 – Boxplot for the OWD of the scenario 128 bytes payload / 256 bits AES / HMAC SHA-256



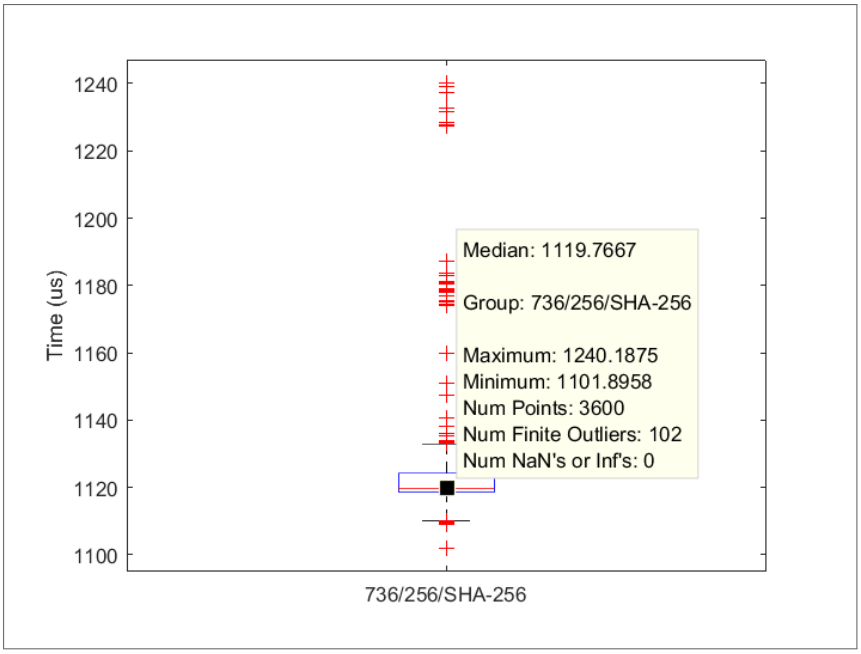
Source: The author (2017)

Considering the scenario with 1440 bytes of payload, for example, and we want to demonstrate that the critical latency value of 2.5 ms is not surpassed. We then choose a value for the median which is lower and distant from that value, because using a median with this exact value would mean that the results are centered around it.

Here are the results for the Wilcoxon Unilateral Statistical Test with a 99% confidence level. The null hypothesis for each case was rejected for each of the respective following latency median values:

- **64 bytes / AES256 / SHA256:** 289 μs
- **128 bytes / AES256 / SHA256:** 363 μs
- **736 bytes / AES256 / SHA256:** 1121 μs
- **1440 bytes / AES256 / SHA256:** 1993 μs

Figure 29 – Boxplot for the OWD of the scenario 736 bytes payload / 256 bits AES / HMAC SHA-256

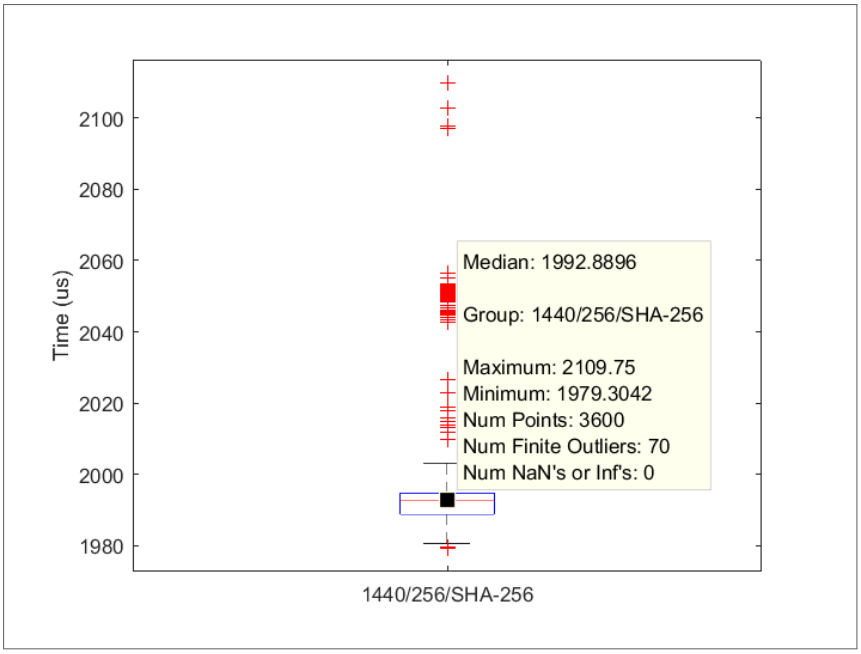


Source: The author (2017)

Once we have determined that the sample is statistically lower than the chosen median, we proceed to cover the upper boundary of the latencies ranges by doing a boxplot. Figures 27, 28, 29 and 30 show the boxplot of the scenarios which were tested by the Wilcoxon Hypothesis Test. From those, the maximum values and outliers can be determined. For example, for the 1440 bytes scenario, the maximum value was 2109.75 μs. So with the combined statistical demonstration that the median is not greater than a chosen threshold and distant from the critical value, and also that the maximum value is lower than the critical value, we demonstrate that with a confidence level of 99%, the latency will not be greater than the critical value.

It can also be seen that the height of the boxes is tiny, showing that the majority of the latency values is concentrated around a determined range, and the small number of outliers compared to the whole amount of messages also contributes for consolidating this fact, which was also reinforced the statistical test. All the medians were much closer to the minimum values than they were from the maximum ones. The median in each of the boxplots is the thin line crossing the box.

Figure 30 – Boxplot for the OWD of the scenario 1440 bytes payload / 256 bits AES / HMAC SHA-256



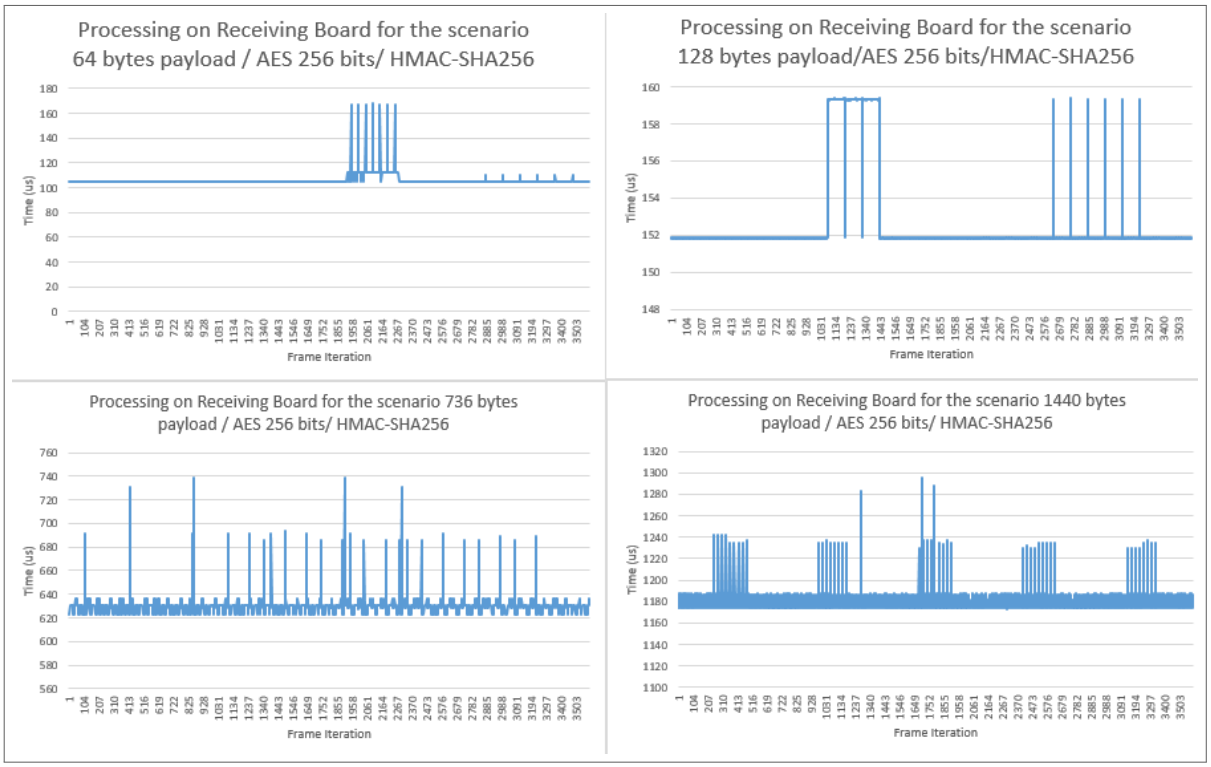
Source: The author (2017)

4.4 DETERMINISTIC BEHAVIOUR AND CONSIDERATIONS

This section will evaluate the deterministic behavior of the application. Since the network transmission did not implement any protocol with that purpose, the focus will be on the behavior of the TI-RTOS, but the transmission will also be commented briefly.

The deterministic behavior inside a RTOS implies that every time a particular code is executed, the same amount of clock cycles is observed. However, this behavior was not experimentally verified. As commented before, the network utilized did not implement any deterministic variation, so such behavior wouldn't be unexpected at the network side, as it can be seen on figure 33.

Figure 31 – Dispersion Graph of the processing time of the receiving board for the scenarios 256 bits AES / HMAC SHA-256

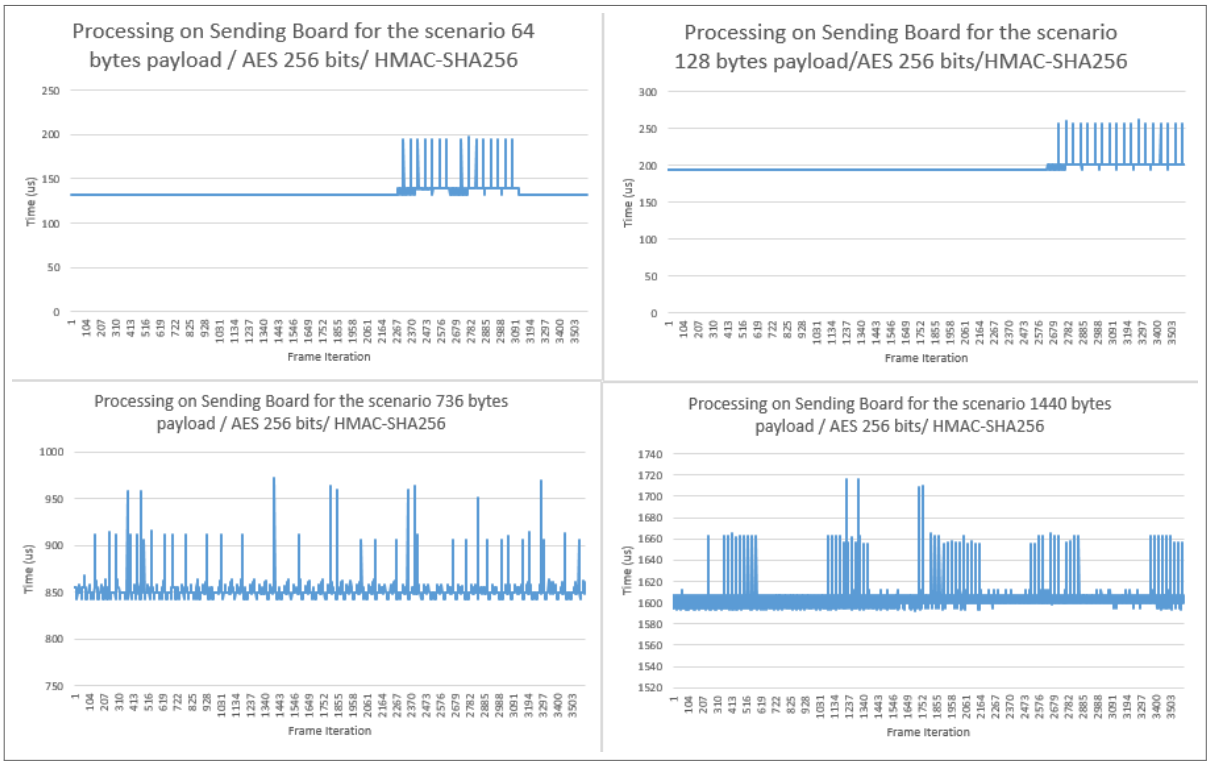


Source: The author (2017)

The code implementation for the TI-RTOS user application did not involve any user defined dynamic memory allocation since it is not recommended for real-time applications with safety-critical requirements. However, the network API function provided, for sending and receiving the frames using the no-copy variation, does this allocation within itself. As described in Texas Instruments (2016a), the deallocation is performed as soon as the packet is sent to the sending function and in the case of the receiving function, the user must do it as soon as the manipulation of the data buffer is done. Everything was done according to the example obtained. But as it can be verified on figure 31 and on figure 32, which are graphs of the execution time on respectively the receiving and sending board, there are peaks of latency. If one looks carefully, it can be noticed the larger the payload size, more often the peaks occur. That shows that probably the memory deallocation is not being done properly and is only forced when there is no space left, and that is why the larger the payload used, the more frequent the peaks become.

It might be possible though, that some additional setting on the TI-RTOS configuration file may need to be enabled. However, it does not seem to be the case. No parallel threads were running, other than simple timers and the system only executes the user application.

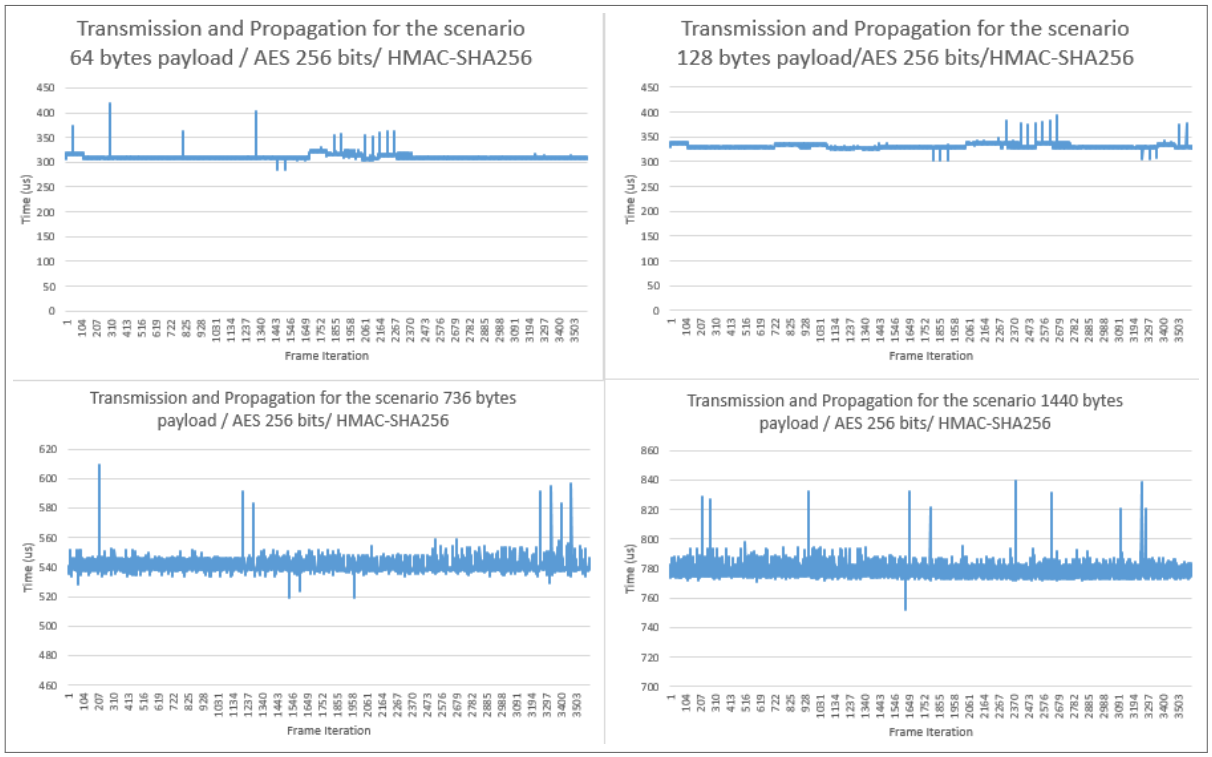
Figure 32 – Dispersion Graph of the processing time of the sending board for the scenarios 256 bits AES / HMAC SHA-256



Source: The author (2017)

To know that this additional time for memory allocation is happening is important to correctly dimension the network scheduling without any unexpected behaviors. To a certain degree, even though there were scenario results not always showing a sequential growth according to what would be expected, there was a certain predictability to the behavior. That behavior may be accounted for by analyzing the statistical results, to ensure that the majority of the samples are far from the safety-critical latency value, then by establishing a safety margin over the worst case scenario. This safety margin may be an amount determined by the company policies or the project manager for example since the exact worst case scenario value should not be used because what is obtained by the test is the worst case of the sample values and not from the whole population of values. This safety margin may be, as an example, 25% over the worst case value or higher/lower, depending on the desired confidence level. By doing so, a proper time frame for the each task in question can be established.

Figure 33 – Dispersion Graph of the transmission and propagation time for both forwards and backwards for the scenarios 256 bits AES / HMAC SHA-256



Source: The author (2017)

As an example, let us take the worst case latency value for the scenario with 64 bytes of payload size. For the scenario where the AES 256 bits and HMAC SHA-256 were utilized, the worst case value for the OWD was 351 μs. However, as it can be seen from table 4, from all the scenarios, the worst case value ranged from 340 μs until 395 μs. If a margin of 25% was utilized for example over the 351 μs, then values of up to 436 μs would not be a problem.

The results for the larger payload sizes, like 736 bytes and 1440 bytes, could be potentially improved by enabling the uDMA controller to speed up the data transfers inside the cryptography module. As explained on section 3.2.3, for smaller data sizes, enabling this feature takes longer to complete the tasks, and since the focus were safety-critical control messages, this feature was not enabled. The larger payload sizes were used just for a demonstration of how the latency performance behaves in those situations.

Also, by using a network protocol which implements QoS, may also improve the results obtained. However, they already performed very well, showing a predictable behavior when one considers the range of values it operates. For the application of safety-critical control messages, only a clock synchronization protocol, such as the PTP which the board supports, would be enough to complement the functionality, for example, to synchronize the wheels of

the steer-by-wire use case.

The results obtained by this work are to show that the currently available hardware when used correctly to plan the architecture, making use of dedicated processing modules for a particular task, along with the centralized architecture, provide a flexible future-proof environment that already supports the latency requirements lower than the currently adopted ones. This way, the requirements for future applications which may surface are assured. Further on, the developer may implement the desired level of security without any specialized network hardware or nodes for doing so, and therefore, the system as a whole is lower in cost.

5 CONCLUSION

The study conducted in the present work faces a very competitive environment, which is the automotive field. The advent of Ethernet inside vehicles brought with itself the possibility for several new applications. The redesigned centralized architecture allows the use of devices which were not common to the area, reducing at the same time the necessary complexity that has been building up in the last years. This work is the result of the concern that is being brought forth by several studies and the media, towards the security aspects of the cars. The security in vehicles protects our privacy, but is also closely related to the safety of the passengers, since a security leak may allow an intruder to take control of the car.

This work demonstrated experimentally that by correctly dimensioning the network, along with enforcing performance oriented architectural and implemental decisions, the unpredictability range of the latency values could be accounted for, and thus, the behavior becomes predictable inside an isolated network for secure safety-critical control data. By using cryptographic protocols to ensure integrity, confidentiality, and authenticity, authenticated encryption is achieved, and the use of a dedicated hardware accelerator allows for the deadlines to be met comfortably. Not once the maximum end-to-end delay allowed was surpassed, and predictability was established by planning beforehand for the expected variation of the latency values. The realization of a statistical test helps to reinforce the delimitation of the boundaries of the latency values.

The result is a generic communication platform for the secure exchange of safety-critical control data, which can be adapted for the desired application. That is all performed using low-cost equipment. With the resulting maximum latency values being significantly lower than the currently adopted values, it results in a future-proof architecture that is flexible enough to support the maximum latency requirements of possible future applications.

Future work possibilities would include the expansion of this platform to implement the synchronization of devices, through the use of IEEE 1588 PTP for example, and possibly the use of QoS. Other possibilities are the creation of a working prototype of the use case, to evaluate the real-life performance of such a system. Another option would also include the portability of the code to a hardware/software platform that supports more devices using the FreeRTOS, for example. Further on, another front would be the qualitative evaluation of the security level achieved and how it could be improved.

REFERENCES

ABDOU, A.; MATRAWY, A.; OORSCHOT, P. C. V. Accurate one-way delay estimation with reduced client trustworthiness. *IEEE Communications Letters*, IEEE, v. 19, n. 5, p. 735–738, 2015.

BELLARE, M.; CANETTI, R.; KRAWCZYK, H. Keying hash functions for message authentication. In: SPRINGER. *Annual International Cryptology Conference*. [S.l.], 1996. p. 1–15.

BELLO, L. L. The case for Ethernet in automotive communications. *ACM SIGBED Review*, ACM, v. 8, n. 4, p. 7–15, 2011.

BUCKL, C.; CAMEK, A.; KAINZ, G.; SIMON, C.; MERCEP, L.; STÄHLE, H.; KNOLL, A. The software car: Building ICT architectures for future electric vehicles. In: IEEE. *Electric Vehicle Conference (IEVC), 2012 IEEE International*. [S.l.], 2012. p. 1–8.

CAMEK, A. G.; BUCKL, C.; KNOLL, A. Future cars: necessity for an adaptive and distributed multiple independent levels of security architecture. In: ACM. *Proceedings of the 2nd ACM international conference on High confidence networked systems*. [S.l.], 2013. p. 17–24.

CHAKRABORTY, S.; FARUQUE, M. A. A.; CHANG, W.; GOSWAMI, D.; WOLF, M.; ZHU, Q. Automotive cyber–physical systems: A tutorial introduction. *IEEE Design & Test*, IEEE, v. 33, n. 4, p. 92–108, 2016.

CHAKRABORTY, S.; LUKASIEWYCZ, M.; BUCKL, C.; FAHMY, S.; CHANG, N.; PARK, S.; KIM, Y.; LETEINTURIER, P.; ADLKOFER, H. Embedded systems and software challenges in electric vehicles. In: EDA CONSORTIUM. *Proceedings of the Conference on Design, Automation and Test in Europe*. [S.l.], 2012. p. 424–429.

CHECKOWAY, S.; MCCOY, D.; KANTOR, B.; ANDERSON, D.; SHACHAM, H.; SAVAGE, S.; KOSCHER, K.; CZESKIS, A.; ROESNER, F.; KOHNO, T. et al. Comprehensive experimental analyses of automotive attack surfaces. In: SAN FRANCISCO. *USENIX Security Symposium*. [S.l.], 2011.

DAOUD, R. M.; AMER, H. H.; ELSAYED, H. M.; SALLEZ, Y. Ethernet-based car control network. In: IEEE. *2006 Canadian Conference on Electrical and Computer Engineering*. [S.l.], 2006. p. 1031–1034.

FREEMAN, W.; MILLER, E. An experimental analysis of cryptographic overhead in performance-critical systems. In: IEEE. *Modeling, Analysis and Simulation of Computer and Telecommunication Systems, 1999. Proceedings. 7th International Symposium on*. [S.l.], 1999. p. 348–357.

HAMALAINEN, P.; ALHO, T.; HANNIKAINEN, M.; HAMALAINEN, T. D. Design and implementation of low-area and low-power AES encryption hardware core. In: IEEE. *9th EUROMICRO Conference on Digital System Design (DSD'06)*. [S.l.], 2006. p. 577–583.

HANK, P.; MÜLLER, S.; VERMESAN, O.; KEYBUS, J. V. D. Automotive Ethernet: in-vehicle networking and smart mobility. In: EDA CONSORTIUM. *Proceedings of the Conference on Design, Automation and Test in Europe*. [S.l.], 2013. p. 1735–1739.

HOLLE, J.; LOTHSPEICH, T. Security concepts for Ethernet based e/e-architectures. In: IEEE. *IEEE-SA Ethernet I& IP Automotive Technology Day 2016, Paris, France*. [S.l.], 2016.

HUMAYED, A.; LIN, J.; LI, F.; LUO, B. Cyber-physical systems security–a survey. *arXiv preprint arXiv:1701.04525*, 2017.

IEEE. *802.1Qbu - Frame Preemption*. 2015. [Online: accessed 7-February-2017]. Available at: <URL:http://www.ieee802.org/1/pages/802.1bu.html>.

KAINZ, G.; BUCKL, C.; KNOLL, A. *Experimental Platform for Innovative ICT Car Architecture*. Citeseer, 2010. [Online: accessed 5-February-2017]. Available at: <URL:https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=f9bf94501f30f151ead3e35eb024a3c3e8e3da40>.

KLEBERGER, P.; OLOVSSON, T.; JONSSON, E. Security aspects of the in-vehicle network in the connected car. In: IEEE. *Intelligent Vehicles Symposium (IV), 2011 IEEE*. [S.l.], 2011. p. 528–533.

KOFTINOFF, J. *AVB/TSN Developer FAQ*. 2016. [Online; accessed 5-February-2017]. Available at: <URL:https://avb.statusbar.com/page/developer-faq/>.

KOSCHER, K.; CZESKIS, A.; ROESNER, F.; PATEL, S.; KOHNO, T.; CHECKOWAY, S.; MCCOY, D.; KANTOR, B.; ANDERSON, D.; SHACHAM, H. et al. Experimental security analysis of a modern automobile. In: IEEE. *2010 IEEE Symposium on Security and Privacy*. [S.l.], 2010. p. 447–462.

KRAWCZYK, H. The order of encryption and authentication for protecting communications (or how secure is SSL?). In: SPRINGER. *Annual International Cryptology Conference*. [S.l.], 2001. p. 310–331.

KRAWCZYK, H.; BELLARE, M.; CANETTI, R. HMAC: Keyed-hashing for message authentication rfc2104. *IETF, Feb*, 1997.

LAB, T. K. *Gateway Internals of Tesla Motors*. 2016. [Online: accessed 10-February-2017]. Available at: <URL:https://2016.zeronights.ru/wp-content/uploads/2016/12/Gateway{_}Internals{_}of{_}Tesla{_}Motors{_}v6.pdf>.

LEE, Y.; PARK, K. Meeting the real-time constraints with standard Ethernet in an in-vehicle network. In: IEEE. *Intelligent Vehicles Symposium (IV), 2013 IEEE*. [S.l.], 2013. p. 1313–1318.

LIM, H.-T.; HERRSCHER, D.; CHAARI, F. Performance comparison of ieee 802.1 q and ieee 802.1 avb in an ethernet-based in-vehicle network. In: IEEE. *Computing Technology and Information Management (ICCM), 2012 8th International Conference on*. [S.l.], 2012. v. 1, p. 1–6.

LIM, H.-T.; WECKEMANN, K.; HERRSCHER, D. Performance study of an in-car switched ethernet network without prioritization. In: SPRINGER. *International Workshop on Communication Technologies for Vehicles*. [S.l.], 2011. p. 165–175.

LTD., R. T. E. *About FreeRTOS*. 2016. [Online: accessed 5-February-2017]. Available at: <URL:http://www.freertos.org/RTOS.html>.

MACHER, G.; ARMENGAUD, E.; KREINER, C. Automated generation of autosar description file for safety-critical software architectures. In: *GI-Jahrestagung*. [S.l.: s.n.], 2014. p. 2145–2156.

MILLER, C.; VALASEK, C. Remote exploitation of an unaltered passenger vehicle. *Black Hat USA*, 2015.

MONTGOMERY, D. C.; RUNGER, G. C. *Applied statistics and probability for engineers*. [S.l.]: John Wiley & Sons, 2010.

NAVET, N.; SIMONOT-LION, F. *In-vehicle communication networks - a historical perspective and review*. [S.l.], 2013.

PETERSON, W. W.; BROWN, D. T. Cyclic codes for error detection. *Proceedings of the IRE*, IEEE, v. 49, n. 1, p. 228–235, 1961.

ROMANOW, A. 802.1 ae—media access control (mac) security. *URL: http://www.ieee802.org/1/pages/802.1ae.html*, 2006.

SORDO, I. C. M. D. *Infrastructure of Tomorrow: the Compound Data Center Architecture*. 2016. [Online: accessed 5-February-2017]. Available at: <URL:https://www.myecole.it/biblio/wp-content/uploads/2020/11/3_DK_2DS_Infrastructures_of_tomorrow_IBM.pdf>.

SPURGEON, C. *Ethernet: the definitive guide*. [S.l.]: " O'Reilly Media, Inc.", 2000.

STAEHLE, H.; MERCEP, L.; KNOLL, A.; SPIEGELBERG, G. Towards the deployment of a centralized ICT architecture in the automotive domain. In: IEEE. *2013 2nd Mediterranean Conference on Embedded Computing (MECO)*. [S.l.], 2013. p. 66–69.

STÄHLE, H.; HUANG, K.; KNOLL, A. Drive-by-wireless with the ecar demonstrator. In: ACM. *Proceedings of the 4th ACM SIGBED International Workshop on Design, Modeling, and Evaluation of Cyber-Physical Systems*. [S.l.], 2014. p. 19–22.

STALLINGS, W. *Cryptography and network security: principles and practices*. [S.l.]: Pearson Education India, 2006.

STANDARD, D. E. Federal information processing standards publication 46. *National Bureau of Standards, US Department of Commerce*, 1977.

STEFFEN, R.; BOGENBERGER, R.; HILLEBRAND, J.; HINTERMAIER, W.; WINCKLER, A.; RAHMANI, M. Design and realization of an IP-based in-car network architecture. *The First Annual International Symposium on Vehicular Computing Systems, Dublin, Ireland*, Citeseer, p. 1–7, 2008.

STEINBACH, T.; KORF, F.; SCHMIDT, T. C. Real-time Ethernet for automotive applications: A solution for future in-car networks. In: IEEE. *2011 IEEE International Conference on Consumer Electronics-Berlin (ICCE-Berlin)*. [S.l.], 2011. p. 216–220.

STUDNIA, I.; NICOMETTE, V.; ALATA, E.; DESWARTE, Y.; KAÂNICHE, M.; LAAROUCHI, Y. Survey on security threats and protection mechanisms in embedded automotive networks. In: IEEE. *Dependable Systems and Networks Workshop (DSN-W), 2013 43rd Annual IEEE/IFIP Conference on*. [S.l.], 2013. p. 1–12.

TEXAS INSTRUMENTS. *Tiva TM4C129ENCPDT Microcontroller DATA SHEET*. [S.l.], 2014. Rev. B.

TEXAS INSTRUMENTS. *TI Network Developer's Kit (NDK) v2.25 API Reference Guide*. [S.l.], 2016. Rev. J.

TEXAS INSTRUMENTS. *TI-RTOS 2.20 User's Guide*. [S.l.], 2016. Rev. M.

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, 2015.

TUOHY, S.; GLAVIN, M.; JONES, E.; TRIVEDI, M.; KILMARTIN, L. Next generation wired intra-vehicle networks, a review. In: IEEE. *Intelligent Vehicles Symposium (IV), 2013 IEEE*. [S.l.], 2013. p. 777–782.

WANG, M.-Y.; SU, C.-P.; HUANG, C.-T.; WU, C.-W. An HMAC processor with integrated SHA-1 and MD5 algorithms. In: IEEE PRESS. *Proceedings of the 2004 Asia and South Pacific Design Automation Conference*. [S.l.], 2004. p. 456–458.

APPENDIX A – EXPERIMENT'S RESULTS

	T recv (us)				
	Mean	Min	Max	Std Dev	Median
64/128/shamd5	96.3	95.5	161.5	3.4	95.5
64/128/sha1	96.5	95.4	160.9	3.4	95.4
64/128/sha224	101.0	100.2	165.3	3.2	100.2
64/128/sha256	103.4	102.2	167.4	3.7	102.2
64/192/shamd5	94.2	93.7	101.3	1.9	93.7
64/192/sha1	97.5	96.5	159.3	3.3	96.5
64/192/sha224	100.4	99.6	208.3	3.6	99.6
64/192/sha256	102.8	101.6	164.8	3.5	101.7
64/256/shamd5	97.6	96.4	162.2	3.5	96.4
64/256/sha1	97.0	96.5	104.1	1.9	96.5
64/256/sha224	103.8	102.5	165.8	3.4	102.5
64/256/sha256	105.3	104.4	167.7	3.5	104.5
128/128/shamd5	140.7	139.8	248.0	3.4	139.9
128/128/sha1	144.0	142.6	205.6	3.7	142.6
128/128/sha224	146.3	145.5	153.2	2.3	145.5
128/128/sha256	149.1	147.5	215.0	4.4	147.5
128/192/shamd5	144.2	141.3	257.5	6.1	141.3
128/192/sha1	144.3	143.4	251.4	3.0	143.4
128/192/sha224	149.2	147.2	263.3	5.2	147.2
128/192/sha256	151.8	149.3	270.9	5.9	149.3
128/256/shamd5	147.1	145.9	159.0	2.6	145.9
128/256/sha1	151.6	148.6	217.0	4.6	148.7
128/256/sha224	151.4	149.8	217.1	4.0	149.8
128/256/sha256	152.6	151.9	159.4	2.2	151.9
736/128/shamd5	594.8	585.6	658.2	6.3	593.2
736/128/sha1	615.1	608.0	724.2	8.2	615.6
736/128/sha224	598.4	591.4	662.4	5.9	599.0
736/128/sha256	602.7	593.4	664.5	6.8	601.0
736/192/shamd5	617.0	610.2	683.2	6.2	617.8
736/192/sha1	621.3	612.7	734.3	8.6	620.3
736/192/sha224	624.7	616.2	737.6	6.5	623.6
736/192/sha256	625.3	618.2	686.7	6.1	625.8
736/256/shamd5	624.4	615.0	794.9	9.2	622.6
736/256/sha1	623.4	617.4	686.3	6.1	625.0
736/256/sha224	626.3	620.9	689.4	5.8	628.4
736/256/sha256	629.1	622.9	739.0	7.2	630.4
1440/128/shamd5	1114.8	1109.2	1218.0	7.7	1115.5
1440/128/sha1	1155.5	1149.2	1271.9	8.0	1155.8
1440/128/sha224	1120.5	1115.1	1183.7	7.2	1121.3
1440/128/sha256	1122.9	1116.8	1228.5	7.4	1123.3
1440/192/shamd5	1162.0	1156.0	1334.3	7.9	1162.4
1440/192/sha1	1165.7	1158.3	1328.8	8.7	1165.0
1440/192/sha224	1168.0	1162.0	1277.1	8.2	1168.4
1440/192/sha256	1169.9	1164.0	1280.9	8.2	1170.4
1440/256/shamd5	1171.0	1165.3	1287.7	7.5	1171.6
1440/256/sha1	1173.6	1167.4	1290.1	7.4	1174.1
1440/256/sha224	1177.5	1171.8	1288.1	8.3	1177.5
1440/256/sha256	1179.3	1173.4	1295.4	7.9	1179.5
64/crc32	66.5	65.6	130.0	2.7	65.6
128/crc32	100.5	98.7	161.6	3.7	98.7
736/crc32	424.4	421.3	486.6	4.7	421.3
1440/crc32	802.4	793.8	864.5	6.3	801.3
64/no crypto	17.5	17.5	17.5	0.0	17.5
128/no crypto	21.8	21.3	29.1	1.7	21.3
736/no crypto	57.7	57.1	119.4	2.3	57.1
1440/no crypto	99.3	98.1	160.8	3.1	98.1

	T iteration (us)				
	Mean	Min	Max	Std Dev	Median
64/128/shamd5	526.1	490.4	594.9	7.5	523.9
64/128/sha1	530.5	492.9	596.9	7.3	530.5
64/128/sha224	539.8	497.5	633.4	7.9	537.6
64/128/sha256	544.4	520.0	618.8	9.0	541.0
64/192/shamd5	526.4	481.0	591.1	6.4	526.5
64/192/sha1	530.6	492.8	597.0	7.0	530.5
64/192/sha224	541.5	498.5	651.6	8.1	540.4
64/192/sha256	545.8	500.8	651.2	9.9	541.8
64/256/shamd5	528.0	491.4	593.8	6.6	528.3
64/256/sha1	530.5	494.3	593.2	6.5	528.8
64/256/sha224	543.7	501.7	608.3	7.6	543.3
64/256/sha256	549.4	521.3	654.7	7.7	547.3
128/128/shamd5	657.0	627.8	774.1	7.6	657.3
128/128/sha1	666.0	625.0	788.1	8.3	665.7
128/128/sha224	674.8	649.6	738.3	6.7	673.7
128/128/sha256	678.2	651.4	750.3	8.4	675.7
128/192/shamd5	666.1	624.5	783.1	8.5	665.4
128/192/sha1	666.8	628.7	774.9	7.7	667.2
128/192/sha224	677.6	652.1	795.9	8.7	676.3
128/192/sha256	685.9	653.0	804.1	9.3	684.8
128/256/shamd5	660.6	625.2	727.6	7.8	660.1
128/256/sha1	670.6	628.4	736.7	8.0	669.8
128/256/sha224	674.7	638.9	747.4	9.6	672.9
128/256/sha256	680.7	649.4	745.2	6.5	681.1
736/128/shamd5	1937.5	1918.2	2000.7	7.1	1937.4
736/128/sha1	1963.3	1922.6	2081.7	11.6	1961.3
736/128/sha224	1975.8	1947.6	2042.9	8.0	1977.5
736/128/sha256	1984.1	1950.6	2060.2	10.0	1983.9
736/192/shamd5	1963.4	1920.9	2034.4	8.1	1962.7
736/192/sha1	1978.9	1935.1	2094.3	12.6	1978.0
736/192/sha224	1990.1	1952.4	2105.6	10.7	1989.5
736/192/sha256	1989.5	1970.4	2056.0	8.6	1989.3
736/256/shamd5	1979.4	1943.4	2163.6	13.5	1978.0
736/256/sha1	1987.7	1963.6	2055.5	8.8	1989.0
736/256/sha224	2011.3	1965.8	2080.8	9.8	2012.1
736/256/sha256	2020.8	1984.3	2144.8	11.6	2018.4
1440/128/shamd5	3423.3	3395.3	3546.3	13.2	3422.9
1440/128/sha1	3470.8	3437.3	3590.0	11.7	3469.5
1440/128/sha224	3479.9	3455.7	3595.6	11.7	3477.6
1440/128/sha256	3488.2	3456.6	3607.3	12.5	3486.4
1440/192/shamd5	3484.0	3453.2	3642.8	12.1	3481.6
1440/192/sha1	3488.8	3448.9	3654.9	13.3	3487.0
1440/192/sha224	3496.0	3462.1	3609.9	12.4	3493.6
1440/192/sha256	3501.8	3464.1	3624.3	12.8	3499.0
1440/256/shamd5	3496.6	3475.8	3613.5	12.0	3493.5
1440/256/sha1	3505.8	3474.7	3618.0	11.7	3504.4
1440/256/sha224	3558.1	3524.2	3670.8	12.8	3556.3
1440/256/sha256	3561.1	3533.1	3677.8	12.2	3560.2
64/crc32	441.0	407.0	504.5	6.0	440.7
128/crc32	533.9	498.0	705.1	8.4	533.5
736/crc32	1421.0	1392.2	1484.9	6.0	1420.8
1440/crc32	2448.6	2403.9	2526.7	11.7	2448.1
64/no crypto	342.9	305.7	406.7	5.0	342.0
128/no crypto	371.8	334.9	433.7	5.8	370.4
736/no crypto	661.4	641.2	727.0	6.4	660.2
1440/no crypto	993.7	966.7	1062.4	5.8	992.2

T transm+recv (us)

	Mean	Min	Max	Std Dev	Median
64/128/shamd5	401.7	367.7	472.2	5.7	400.5
64/128/sha1	403.3	367.4	466.1	5.7	402.2
64/128/sha224	409.6	368.8	504.7	6.3	408.4
64/128/sha256	411.8	389.3	478.4	6.5	409.6
64/192/shamd5	401.2	357.4	467.5	4.9	400.4
64/192/sha1	403.4	366.5	463.7	5.5	402.8
64/192/sha224	410.1	369.0	522.1	6.8	408.7
64/192/sha256	412.4	369.2	474.8	7.0	410.1
64/256/shamd5	402.6	363.2	467.1	5.3	402.1
64/256/sha1	402.2	367.7	466.6	5.1	400.9
64/256/sha224	412.2	372.0	472.3	6.0	411.8
64/256/sha256	415.8	389.5	522.9	6.6	414.2
128/128/shamd5	465.2	438.3	574.1	6.3	463.4
128/128/sha1	471.5	431.2	567.4	5.5	471.3
128/128/sha224	477.1	454.3	542.9	4.8	476.0
128/128/sha256	478.7	454.0	552.8	7.1	476.8
128/192/shamd5	473.0	433.6	592.3	7.3	472.9
128/192/sha1	471.0	435.0	580.4	5.7	470.0
128/192/sha224	478.7	455.4	590.1	6.9	477.3
128/192/sha256	485.1	454.2	605.3	8.7	483.0
128/256/shamd5	471.4	438.1	532.5	6.9	469.0
128/256/sha1	478.5	438.6	543.8	6.6	477.7
128/256/sha224	480.0	446.1	554.6	8.5	476.5
128/256/sha256	483.7	454.6	544.2	4.6	483.0
736/128/shamd5	1127.0	1114.2	1196.7	7.4	1126.3
736/128/sha1	1149.2	1108.9	1259.1	8.9	1147.4
736/128/sha224	1135.6	1105.5	1202.3	6.3	1134.2
736/128/sha256	1142.2	1110.2	1206.6	7.9	1143.2
736/192/shamd5	1147.4	1104.8	1226.0	7.0	1145.9
736/192/sha1	1157.2	1116.5	1270.1	9.2	1158.5
736/192/sha224	1166.2	1138.1	1277.0	7.6	1167.4
736/192/sha256	1162.7	1153.2	1224.4	6.5	1160.5
736/256/shamd5	1156.3	1130.3	1324.4	9.8	1156.8
736/256/sha1	1162.2	1144.1	1223.8	6.4	1161.0
736/256/sha224	1162.4	1118.0	1224.9	7.9	1164.3
736/256/sha256	1169.8	1142.0	1278.7	7.8	1168.7
1440/128/shamd5	1885.8	1866.6	1993.5	9.2	1883.7
1440/128/sha1	1930.9	1901.0	2046.3	8.8	1930.3
1440/128/sha224	1896.3	1876.6	1964.8	8.0	1895.2
1440/128/sha256	1902.3	1874.1	2007.1	8.9	1899.9
1440/192/shamd5	1936.8	1910.3	2107.5	8.9	1936.2
1440/192/sha1	1939.6	1901.6	2107.3	10.2	1939.7
1440/192/sha224	1943.6	1913.3	2053.6	9.0	1942.4
1440/192/sha256	1947.4	1911.3	2066.0	9.0	1946.1
1440/256/shamd5	1941.0	1916.3	2061.5	9.0	1939.9
1440/256/sha1	1947.5	1920.3	2059.5	8.1	1947.7
1440/256/sha224	1956.1	1921.4	2065.9	8.8	1956.3
1440/256/sha256	1957.3	1926.1	2070.8	9.0	1955.8
64/crc32	365.2	332.3	426.7	5.4	364.0
128/crc32	423.1	388.7	533.1	6.4	421.8
736/crc32	952.2	928.4	1021.0	4.9	952.1
1440/crc32	1571.0	1528.1	1637.2	7.1	1569.2
64/no crypto	316.4	279.3	380.3	4.9	315.6
128/no crypto	339.9	303.4	400.6	5.5	338.5
736/no crypto	583.6	564.1	644.5	5.8	582.3
1440/no crypto	864.6	838.0	927.4	5.5	863.3

	T send (us)					
	Mean	Min	Max	Std Dev	Median	
64/128/shamd5	124.4	122.7	188.5	4.8	122.7	
64/128/sha1	127.2	125.5	191.1	4.9	125.5	
64/128/sha224	130.2	128.7	194.3	4.6	128.7	
64/128/sha256	132.6	130.7	196.3	5.0	130.7	
64/192/shamd5	125.3	123.6	189.5	5.0	123.6	
64/192/sha1	127.2	125.5	190.8	4.8	125.5	
64/192/sha224	131.4	129.5	195.3	5.1	129.5	
64/192/sha256	133.4	131.5	240.3	5.6	131.5	
64/256/shamd5	125.5	123.8	189.4	4.8	123.8	
64/256/sha1	128.3	126.6	192.1	4.8	126.6	
64/256/sha224	131.5	129.8	195.5	4.9	129.8	
64/256/sha256	133.6	131.8	197.4	5.0	131.8	
128/128/shamd5	191.8	189.5	305.8	5.7	189.5	
128/128/sha1	194.6	192.3	316.0	6.1	192.3	
128/128/sha224	197.7	195.4	262.3	5.5	195.4	
128/128/sha256	199.5	197.4	263.5	5.3	197.4	
128/192/shamd5	193.1	190.9	258.2	5.5	190.9	
128/192/sha1	195.8	193.7	302.4	6.1	193.7	
128/192/sha224	198.8	196.7	264.2	5.3	196.7	
128/192/sha256	200.7	198.8	266.3	5.3	198.8	
128/256/shamd5	189.2	187.0	254.0	5.4	187.0	
128/256/sha1	192.1	189.8	256.7	5.5	189.8	
128/256/sha224	194.8	192.8	259.8	5.2	192.8	
128/256/sha256	197.0	194.8	261.6	5.4	194.8	
736/128/shamd5	810.5	803.7	874.8	4.7	811.2	
736/128/sha1	814.1	806.1	933.1	8.3	813.7	
736/128/sha224	840.2	830.8	907.0	6.7	838.4	
736/128/sha256	841.9	832.9	913.3	7.4	840.4	
736/192/shamd5	816.0	808.4	882.5	5.8	816.0	
736/192/sha1	821.6	811.0	932.4	8.5	824.0	
736/192/sha224	823.9	814.3	936.0	9.0	821.8	
736/192/sha256	826.8	816.2	894.9	6.7	823.8	
736/256/shamd5	823.1	813.1	947.3	8.9	820.7	
736/256/sha1	825.5	815.6	894.2	7.3	823.2	
736/256/sha224	848.9	840.3	916.1	7.1	847.8	
736/256/sha256	851.0	842.3	971.3	8.7	849.9	
1440/128/shamd5	1537.5	1527.3	1654.9	9.3	1535.8	
1440/128/sha1	1539.9	1530.1	1604.3	8.3	1538.3	
1440/128/sha224	1583.7	1573.7	1695.4	9.4	1581.3	
1440/128/sha256	1586.0	1575.7	1697.5	9.6	1583.3	
1440/192/shamd5	1547.2	1535.3	1658.8	9.5	1544.9	
1440/192/sha1	1549.2	1539.8	1661.6	8.5	1547.6	
1440/192/sha224	1552.4	1543.1	1664.4	9.1	1550.8	
1440/192/sha256	1554.4	1545.0	1666.8	8.9	1552.7	
1440/256/shamd5	1555.6	1546.2	1668.1	8.7	1554.0	
1440/256/sha1	1558.2	1547.0	1670.6	8.8	1556.5	
1440/256/sha224	1602.0	1589.8	1713.7	9.7	1599.5	
1440/256/sha256	1603.8	1591.9	1715.7	8.9	1601.6	
64/crc32	75.8	74.7	140.1	3.8	74.7	
128/crc32	110.8	109.2	232.0	6.1	109.2	
736/crc32	468.8	463.8	476.9	3.6	471.3	
1440/crc32	877.7	868.2	952.4	9.9	879.2	
64/no crypto	26.5	26.4	88.9	1.4	26.4	
128/no crypto	32.0	31.6	94.5	2.4	31.6	
736/no crypto	77.8	77.1	144.5	3.3	77.1	
1440/no crypto	129.1	128.8	136.4	1.4	128.8	

	T transm (us)					
	Mean		Min		Max		Std Dev		Median	
64/128/shamd5	305.3		272.2		373.2		5.0		304.3	
64/128/sha1	306.8		271.9		369.3		5.4		305.5	
64/128/sha224	308.6		268.6		404.5		5.5		307.7	
64/128/sha256	308.3		287.1		376.2		5.2		307.3	
64/192/shamd5	307.0		263.7		373.8		4.9		306.2	
64/192/sha1	305.8		270.1		367.0		5.1		304.9	
64/192/sha224	309.6		269.4		422.5		5.6		308.9	
64/192/sha256	309.7		267.6		373.1		5.8		308.2	
64/256/shamd5	305.0		266.8		370.7		4.9		304.0	
64/256/sha1	305.2		271.2		370.1		4.7		304.2	
64/256/sha224	308.4		269.5		368.9		5.5		307.2	
64/256/sha256	310.5		285.0		418.4		5.1		309.6	
128/128/shamd5	324.5		290.8		389.7		5.2		323.4	
128/128/sha1	327.4		288.6		424.8		5.2		326.4	
128/128/sha224	330.8		308.7		391.9		3.9		330.2	
128/128/sha256	329.6		306.5		393.6		4.9		328.6	
128/192/shamd5	328.8		292.3		392.0		5.5		327.5	
128/192/sha1	326.7		291.6		431.3		5.2		325.7	
128/192/sha224	329.6		308.1		397.4		5.1		328.4	
128/192/sha256	333.3		305.0		400.7		6.2		332.3	
128/256/shamd5	324.3		292.2		386.6		5.8		322.7	
128/256/sha1	326.9		290.0		389.5		6.1		325.4	
128/256/sha224	328.5		296.2		392.9		6.8		326.5	
128/256/sha256	331.1		302.7		392.4		4.7		330.0	
736/128/shamd5	532.2		523.9		593.3		4.0		531.4	
736/128/sha1	534.1		500.9		631.1		4.3		532.9	
736/128/sha224	537.2		504.8		585.8		3.8		535.7	
736/128/sha256	539.5		509.2		600.1		4.7		537.5	
736/192/shamd5	530.4		494.6		590.8		4.3		529.2	
736/192/sha1	535.9		496.2		633.6		5.0		534.2	
736/192/sha224	541.5		514.3		607.8		5.5		540.0	
736/192/sha256	537.4		532.5		581.5		3.2		537.8	
736/256/shamd5	531.9		507.7		592.2		4.4		530.1	
736/256/sha1	538.8		526.7		601.1		4.7		537.7	
736/256/sha224	536.1		497.1		595.9		5.1		536.0	
736/256/sha256	540.7		519.2		609.1		4.4		539.4	
1440/128/shamd5	771.0		756.5		870.3		4.8		770.5	
1440/128/sha1	775.4		750.7		885.0		4.7		775.1	
1440/128/sha224	775.7		747.7		834.1		4.2		774.4	
1440/128/sha256	779.4		756.2		840.3		5.5		778.7	
1440/192/shamd5	774.8		747.9		842.2		5.0		773.8	
1440/192/sha1	773.9		738.3		833.7		5.1		773.4	
1440/192/sha224	775.6		750.3		884.1		4.1		774.5	
1440/192/sha256	777.5		746.6		839.2		4.5		776.7	
1440/256/shamd5	770.1		750.3		875.8		4.5		768.8	
1440/256/sha1	773.9		751.6		837.9		4.5		773.6	
1440/256/sha224	778.5		749.5		842.4		4.7		778.6	
1440/256/sha256	778.0		752.0		839.1		4.8		776.2	
64/crc32	298.7		266.7		358.7		5.0		297.6	
128/crc32	322.7		290.1		434.4		5.4		321.5	
736/crc32	527.8		507.2		599.8		3.6		527.3	
1440/crc32	768.5		734.3		833.5		4.2		767.6	
64/no crypto	298.9		261.9		362.8		4.9		298.1	
128/no crypto	318.1		282.1		379.3		5.2		317.0	
736/no crypto	525.9		507.1		585.1		5.1		524.8	
1440/no crypto	765.3		739.8		827.9		4.6		764.2	

	t owd (us)					
	Mean		Min	Max	Std Dev	Median
64/128/shamd5	277.1		258.8	340.3	5.1	275.5
64/128/sha1	280.6		261.5	344.9	5.6	278.8
64/128/sha224	284.5		263.0	348.4	5.3	282.9
64/128/sha256	286.8		274.3	355.5	5.9	284.7
64/192/shamd5	278.8		255.5	340.9	5.2	277.5
64/192/sha1	280.1		260.9	345.7	5.4	279.2
64/192/sha224	286.2		264.2	349.4	5.7	284.7
64/192/sha256	288.2		265.3	394.9	6.8	285.8
64/256/shamd5	277.9		259.4	341.9	5.2	276.9
64/256/sha1	280.9		262.2	344.1	5.1	279.3
64/256/sha224	285.7		264.5	349.0	5.3	284.5
64/256/sha256	288.9		274.3	351.9	5.4	287.2
128/128/shamd5	354.1		334.9	467.0	5.8	353.3
128/128/sha1	358.3		337.3	480.7	6.6	356.9
128/128/sha224	363.1		349.8	427.5	5.6	361.4
128/128/sha256	364.3		350.7	428.8	5.5	362.4
128/192/shamd5	357.5		337.0	423.0	5.7	356.5
128/192/sha1	359.2		339.5	466.9	6.4	357.7
128/192/sha224	363.6		350.8	427.2	5.5	361.8
128/192/sha256	367.4		351.3	432.4	5.8	365.9
128/256/shamd5	351.4		333.1	417.9	5.7	350.3
128/256/sha1	355.5		334.8	420.2	5.7	353.5
128/256/sha224	359.0		340.9	423.3	5.9	357.3
128/256/sha256	362.5		346.2	427.5	5.6	361.5
736/128/shamd5	1076.6		1066.1	1140.0	4.5	1076.5
736/128/sha1	1081.2		1064.1	1199.6	8.3	1080.2
736/128/sha224	1108.8		1090.7	1172.3	6.5	1107.7
736/128/sha256	1111.6		1095.0	1183.6	7.2	1111.4
736/192/shamd5	1081.2		1056.0	1146.5	5.5	1080.7
736/192/sha1	1089.6		1066.7	1199.9	8.5	1090.5
736/192/sha224	1094.7		1071.4	1209.4	8.9	1094.3
736/192/sha256	1095.5		1082.5	1165.8	6.6	1094.3
736/256/shamd5	1089.0		1067.0	1213.8	8.9	1088.3
736/256/sha1	1094.9		1080.3	1162.4	7.6	1094.0
736/256/sha224	1117.0		1096.4	1183.5	7.1	1115.9
736/256/sha256	1121.4		1101.9	1240.2	8.8	1119.8
1440/128/shamd5	1923.0		1906.7	2039.1	9.5	1921.8
1440/128/sha1	1927.6		1911.7	1991.0	8.5	1926.8
1440/128/sha224	1971.5		1953.0	2082.1	9.5	1970.7
1440/128/sha256	1975.7		1957.2	2090.1	9.9	1974.9
1440/192/shamd5	1934.6		1916.8	2045.4	9.6	1933.7
1440/192/sha1	1936.2		1914.8	2048.2	8.8	1934.3
1440/192/sha224	1940.2		1924.0	2052.7	9.2	1938.4
1440/192/sha256	1943.1		1926.0	2057.6	9.0	1942.0
1440/256/shamd5	1940.6		1929.3	2052.9	8.9	1939.2
1440/256/sha1	1945.2		1925.1	2057.3	9.1	1944.0
1440/256/sha224	1991.3		1972.2	2103.1	9.9	1990.6
1440/256/sha256	1992.8		1979.3	2109.8	8.9	1992.9
64/crc32	225.1		208.1	289.5	4.3	224.0
128/crc32	272.1		254.3	391.1	6.6	270.8
736/crc32	732.7		717.4	763.7	4.1	734.2
1440/crc32	1261.9		1242.9	1337.1	10.0	1263.0
64/no crypto	176.0		157.3	238.4	2.8	175.5
128/no crypto	191.0		172.6	253.5	3.4	190.2
736/no crypto	340.7		330.6	406.6	4.0	339.8
1440/no crypto	511.7		498.7	550.3	2.9	510.9

t iteration

	(%) Mean/CRC32	(%) Max/CRC32	(%) Mean/NO-Cryptc	(%) Max/NO-Crypto
64/128/shamd5	19%	18%	53%	46%
64/128/sha1	20%	18%	55%	47%
64/128/sha224	22%	26%	57%	56%
64/128/sha256	23%	23%	59%	52%
64/192/shamd5	19%	17%	54%	45%
64/192/sha1	20%	18%	55%	47%
64/192/sha224	23%	29%	58%	60%
64/192/sha256	24%	29%	59%	60%
64/256/shamd5	20%	18%	54%	46%
64/256/sha1	20%	18%	55%	46%
64/256/sha224	23%	21%	59%	50%
64/256/sha256	25%	30%	60%	61%
128/128/shamd5	23%	10%	77%	78%
128/128/sha1	25%	12%	79%	82%
128/128/sha224	26%	5%	81%	70%
128/128/sha256	27%	6%	82%	73%
128/192/shamd5	25%	11%	79%	81%
128/192/sha1	25%	10%	79%	79%
128/192/sha224	27%	13%	82%	84%
128/192/sha256	28%	14%	84%	85%
128/256/shamd5	24%	3%	78%	68%
128/256/sha1	26%	4%	80%	70%
128/256/sha224	26%	6%	81%	72%
128/256/sha256	27%	6%	83%	72%
736/128/shamd5	36%	35%	193%	175%
736/128/sha1	38%	40%	197%	186%
736/128/sha224	39%	38%	199%	181%
736/128/sha256	40%	39%	200%	183%
736/192/shamd5	38%	37%	197%	180%
736/192/sha1	39%	41%	199%	188%
736/192/sha224	40%	42%	201%	190%
736/192/sha256	40%	38%	201%	183%
736/256/shamd5	39%	46%	199%	198%
736/256/sha1	40%	38%	201%	183%
736/256/sha224	42%	40%	204%	186%
736/256/sha256	42%	44%	206%	195%
1440/128/shamd5	40%	40%	245%	234%
1440/128/sha1	42%	42%	249%	238%
1440/128/sha224	42%	42%	250%	238%
1440/128/sha256	42%	43%	251%	240%
1440/192/shamd5	42%	44%	251%	243%
1440/192/sha1	42%	45%	251%	244%
1440/192/sha224	43%	43%	252%	240%
1440/192/sha256	43%	43%	252%	241%
1440/256/shamd5	43%	43%	252%	240%
1440/256/sha1	43%	43%	253%	241%
1440/256/sha224	45%	45%	258%	246%
1440/256/sha256	45%	46%	258%	246%
64/crc32			29%	24%
128/crc32			44%	63%
736/crc32			115%	104%
1440/crc32			146%	138%
64/no crypto				
128/no crypto				
736/no crypto				
1440/no crypto				

t send

	(%) Mean/CRC32	(%) Max/CRC32	(%) Mean/NO-Cryptc	(%) Max/NO-Crypto
64/128/shamd5	64%	35%	369%	112%
64/128/sha1	68%	36%	379%	115%
64/128/sha224	72%	39%	391%	119%
64/128/sha256	75%	40%	400%	121%
64/192/shamd5	65%	35%	372%	113%
64/192/sha1	68%	36%	379%	115%
64/192/sha224	73%	39%	395%	120%
64/192/sha256	76%	71%	403%	170%
64/256/shamd5	66%	35%	373%	113%
64/256/sha1	69%	37%	384%	116%
64/256/sha224	74%	39%	395%	120%
64/256/sha256	76%	41%	404%	122%
128/128/shamd5	73%	32%	500%	224%
128/128/sha1	76%	36%	509%	234%
128/128/sha224	78%	13%	518%	178%
128/128/sha256	80%	14%	524%	179%
128/192/shamd5	74%	11%	504%	173%
128/192/sha1	77%	30%	513%	220%
128/192/sha224	79%	14%	522%	180%
128/192/sha256	81%	15%	528%	182%
128/256/shamd5	71%	9%	492%	169%
128/256/sha1	73%	11%	501%	172%
128/256/sha224	76%	12%	509%	175%
128/256/sha256	78%	13%	516%	177%
736/128/shamd5	73%	83%	942%	505%
736/128/sha1	74%	96%	947%	546%
736/128/sha224	79%	90%	980%	528%
736/128/sha256	80%	92%	982%	532%
736/192/shamd5	74%	85%	949%	511%
736/192/sha1	75%	96%	956%	545%
736/192/sha224	76%	96%	959%	548%
736/192/sha256	76%	88%	963%	519%
736/256/shamd5	76%	99%	958%	555%
736/256/sha1	76%	88%	961%	519%
736/256/sha224	81%	92%	992%	534%
736/256/sha256	82%	104%	994%	572%
1440/128/shamd5	75%	74%	1091%	1114%
1440/128/sha1	75%	68%	1093%	1077%
1440/128/sha224	80%	78%	1127%	1143%
1440/128/sha256	81%	78%	1129%	1145%
1440/192/shamd5	76%	74%	1099%	1117%
1440/192/sha1	77%	74%	1100%	1119%
1440/192/sha224	77%	75%	1103%	1121%
1440/192/sha256	77%	75%	1104%	1122%
1440/256/shamd5	77%	75%	1105%	1123%
1440/256/sha1	78%	75%	1107%	1125%
1440/256/sha224	83%	80%	1141%	1157%
1440/256/sha256	83%	80%	1143%	1158%
64/crc32			186%	58%
128/crc32			247%	146%
736/crc32			503%	230%
1440/crc32			580%	598%
64/no crypto				
128/no crypto				
736/no crypto				
1440/no crypto				

t transm

	(%) Mean/CRC32	(%) Max/CRC32	(%) Mean/NO-Cryptc	(%) Max/NO-Crypto
64/128/shamd5	2%	4%	2%	3%
64/128/sha1	3%	3%	3%	2%
64/128/sha224	3%	13%	3%	12%
64/128/sha256	3%	5%	3%	4%
64/192/shamd5	3%	4%	3%	3%
64/192/sha1	2%	2%	2%	1%
64/192/sha224	4%	18%	4%	16%
64/192/sha256	4%	4%	4%	3%
64/256/shamd5	2%	3%	2%	2%
64/256/sha1	2%	3%	2%	2%
64/256/sha224	3%	3%	3%	2%
64/256/sha256	4%	17%	4%	15%
128/128/shamd5	1%	-10%	2%	3%
128/128/sha1	1%	-2%	3%	12%
128/128/sha224	3%	-10%	4%	3%
128/128/sha256	2%	-9%	4%	4%
128/192/shamd5	2%	-10%	3%	3%
128/192/sha1	1%	-1%	3%	14%
128/192/sha224	2%	-9%	4%	5%
128/192/sha256	3%	-8%	5%	6%
128/256/shamd5	1%	-11%	2%	2%
128/256/sha1	1%	-10%	3%	3%
128/256/sha224	2%	-10%	3%	4%
128/256/sha256	3%	-10%	4%	3%
736/128/shamd5	1%	-1%	1%	1%
736/128/sha1	1%	5%	2%	8%
736/128/sha224	2%	-2%	2%	0%
736/128/sha256	2%	0%	3%	3%
736/192/shamd5	0%	-1%	1%	1%
736/192/sha1	2%	6%	2%	8%
736/192/sha224	3%	1%	3%	4%
736/192/sha256	2%	-3%	2%	-1%
736/256/shamd5	1%	-1%	1%	1%
736/256/sha1	2%	0%	2%	3%
736/256/sha224	2%	-1%	2%	2%
736/256/sha256	2%	2%	3%	4%
1440/128/shamd5	0%	4%	1%	5%
1440/128/sha1	1%	6%	1%	7%
1440/128/sha224	1%	0%	1%	1%
1440/128/sha256	1%	1%	2%	1%
1440/192/shamd5	1%	1%	1%	2%
1440/192/sha1	1%	0%	1%	1%
1440/192/sha224	1%	6%	1%	7%
1440/192/sha256	1%	1%	2%	1%
1440/256/shamd5	0%	5%	1%	6%
1440/256/sha1	1%	1%	1%	1%
1440/256/sha224	1%	1%	2%	2%
1440/256/sha256	1%	1%	2%	1%
64/crc32			0%	-1%
128/crc32			1%	15%
736/crc32			0%	3%
1440/crc32			0%	1%
64/no crypto				
128/no crypto				
736/no crypto				
1440/no crypto				

t owd

	(%) Mean/CRC32	(%) Max/CRC32	(%) Mean/NO-Cryptc	(%) Max/NO-Crypto
64/128/shamd5	23%	18%	57%	43%
64/128/sha1	25%	19%	59%	45%
64/128/sha224	26%	20%	62%	46%
64/128/sha256	27%	23%	63%	49%
64/192/shamd5	24%	18%	58%	43%
64/192/sha1	24%	19%	59%	45%
64/192/sha224	27%	21%	63%	47%
64/192/sha256	28%	36%	64%	66%
64/256/shamd5	23%	18%	58%	43%
64/256/sha1	25%	19%	60%	44%
64/256/sha224	27%	21%	62%	46%
64/256/sha256	28%	22%	64%	48%
128/128/shamd5	30%	19%	85%	84%
128/128/sha1	32%	23%	88%	90%
128/128/sha224	33%	9%	90%	69%
128/128/sha256	34%	10%	91%	69%
128/192/shamd5	31%	8%	87%	67%
128/192/sha1	32%	19%	88%	84%
128/192/sha224	34%	9%	90%	69%
128/192/sha256	35%	11%	92%	71%
128/256/shamd5	29%	7%	84%	65%
128/256/sha1	31%	7%	86%	66%
128/256/sha224	32%	8%	88%	67%
128/256/sha256	33%	9%	90%	69%
736/128/shamd5	47%	49%	216%	180%
736/128/sha1	48%	57%	217%	195%
736/128/sha224	51%	53%	225%	188%
736/128/sha256	52%	55%	226%	191%
736/192/shamd5	48%	50%	217%	182%
736/192/sha1	49%	57%	220%	195%
736/192/sha224	49%	58%	221%	197%
736/192/sha256	50%	53%	222%	187%
736/256/shamd5	49%	59%	220%	199%
736/256/sha1	49%	52%	221%	186%
736/256/sha224	52%	55%	228%	191%
736/256/sha256	53%	62%	229%	205%
1440/128/shamd5	52%	53%	276%	271%
1440/128/sha1	53%	49%	277%	262%
1440/128/sha224	56%	56%	285%	278%
1440/128/sha256	57%	56%	286%	280%
1440/192/shamd5	53%	53%	278%	272%
1440/192/sha1	53%	53%	278%	272%
1440/192/sha224	54%	54%	279%	273%
1440/192/sha256	54%	54%	280%	274%
1440/256/shamd5	54%	54%	279%	273%
1440/256/sha1	54%	54%	280%	274%
1440/256/sha224	58%	57%	289%	282%
1440/256/sha256	58%	58%	289%	283%
64/crc32			28%	21%
128/crc32			42%	54%
736/crc32			115%	88%
1440/crc32			147%	143%
64/no crypto				
128/no crypto				
736/no crypto				
1440/no crypto				

t recv

	(%) Mean/CRC32	(%) Max/CRC32	(%) Mean/NO-Cryptc	(%) Max/NO-Crypto
64/128/shamd5	45%	24%	451%	824%
64/128/sha1	45%	24%	452%	821%
64/128/sha224	52%	27%	478%	846%
64/128/sha256	56%	29%	492%	858%
64/192/shamd5	42%	-22%	439%	480%
64/192/sha1	47%	22%	458%	811%
64/192/sha224	51%	60%	475%	1092%
64/192/sha256	55%	27%	488%	843%
64/256/shamd5	47%	25%	459%	828%
64/256/sha1	46%	-20%	455%	496%
64/256/sha224	56%	28%	494%	849%
64/256/sha256	58%	29%	503%	860%
128/128/shamd5	40%	53%	547%	753%
128/128/sha1	43%	27%	562%	607%
128/128/sha224	46%	-5%	573%	427%
128/128/sha256	48%	33%	586%	639%
128/192/shamd5	43%	59%	563%	785%
128/192/sha1	44%	56%	563%	764%
128/192/sha224	48%	63%	586%	805%
128/192/sha256	51%	68%	598%	831%
128/256/shamd5	46%	-2%	576%	447%
128/256/sha1	51%	34%	597%	646%
128/256/sha224	51%	34%	596%	646%
128/256/sha256	52%	-1%	601%	448%
736/128/shamd5	40%	35%	931%	451%
736/128/sha1	45%	49%	966%	506%
736/128/sha224	41%	36%	937%	455%
736/128/sha256	42%	37%	944%	456%
736/192/shamd5	45%	40%	969%	472%
736/192/sha1	46%	51%	977%	515%
736/192/sha224	47%	52%	982%	518%
736/192/sha256	47%	41%	984%	475%
736/256/shamd5	47%	63%	982%	566%
736/256/sha1	47%	41%	980%	475%
736/256/sha224	48%	42%	985%	477%
736/256/sha256	48%	52%	990%	519%
1440/128/shamd5	39%	41%	1023%	657%
1440/128/sha1	44%	47%	1064%	691%
1440/128/sha224	40%	37%	1028%	636%
1440/128/sha256	40%	42%	1031%	664%
1440/192/shamd5	45%	54%	1070%	730%
1440/192/sha1	45%	54%	1074%	726%
1440/192/sha224	46%	48%	1076%	694%
1440/192/sha256	46%	48%	1078%	696%
1440/256/shamd5	46%	49%	1079%	701%
1440/256/sha1	46%	49%	1082%	702%
1440/256/sha224	47%	49%	1086%	701%
1440/256/sha256	47%	50%	1088%	705%
64/crc32			281%	644%
128/crc32			362%	456%
736/crc32			635%	307%
1440/crc32			708%	437%
64/no crypto				
128/no crypto				
736/no crypto				
1440/no crypto				