eman ta zabal zazu

Universidad del País Vasco Euskal Herriko Unibertsitatea

Escuela Técnica Superior de Ingeniería de Bilbao

Departamento de Tecnología Electrónica

DOCTORAL THESIS.

System-on-Chip Architecture for Secure Sub-microsecond Synchronization Systems

Author: Naiara Moreira Ciruelos

Supervisor: Dr. Armando Astarloa Cuéllar

Bilbao, January 2017

Resumen

Las redes eléctricas de nueva generación, denominadas Smart Grids, deben proporcionar una serie de servicios inteligentes con estrictos requisitos de operación. Para cumplir tales requisitos es necesario modernizar las subestaciones instalando dispositivos electrónicos inteligentes que proporcionen las funciones de control y protección del equipamiento primario. Estos dispositivos pueden accederse de forma local o remota para asegurar una respuesta rápida y automática a posibles problemas en el suministro energético. Por tanto, estos deben interconectarse mediante buses digitales constituyendo una red de comunicación dentro de la subestación, y deben también proporcionar interfaces de comunicación hacia el exterior, por ejemplo, hacia un centro de control remoto.

Como consecuencia de esta digitalización de las subestaciones, surgen nuevas amenazas digitales que en las redes eléctricas tradicionales no existían. Dado que el equipamiento de protección está conectado a grandes redes de comunicación externas e incluso a la red pública de Internet, éstas quedan expuestas a ataques de individuos situados a cientos o miles de kilómetros que pueden desembocar en problemas gravísimos como la interrupción del suministro eléctrico en grandes áreas urbanas o polígonos industriales. En ese sentido, los dispositivos de protección normalmente están basados en sistemas de computación embebidos que requieren de un enfoque específico para afrontar los retos de ciberseguridad.

En la presente tesis, se pretende dar solución a los problemas que presenta la protección de uno de los protocolos de comunicación más sensibles de entre los considerados por los organismos de estandarización para su aplicación en subestaciones. Es el caso del Precision Time Protocol (PTP), el cual tiene como misión distribuir una referencia de tiempo desde un dispositivo maestro al resto de dispositivos esclavos dentro de la red de forma muy precisa. Este protocolo es altamente vulnerable, ya que introduciendo tan sólo un error de tiempo de un microsegundo, pueden causarse graves problemas en las funciones de protección e incluso detener el funcionamiento del equipamiento primario.

En primer lugar, se presentará el estado actual de la tecnología de las comunicaciones y la ciberseguridad en subestaciones. En particular, se describirá la problemática que supone la protección del protocolo de sincronización mencionado. A continuación, se realizará un estudio detallado de los componentes hardware y software existentes para el diseño de sistemas electrónicos basados en dispositivos reconfigurables, con el fin de integrar el protocolo PTP con el estándar de seguridad para redes Ethernet conocido como MACsec. En segundo lugar, se propondrá una nueva architectura System-on-Chip (SoC) basada en dispositivos reconfigurables que incluya soporte PTP y MACsec. Además, se introducirá un nuevo sistema de gestión de claves de grupo para entornos industriales protegidos mediante MACsec.

Para finalizar, el dispositivo Zynq de Xilinx será utilizado como plataforma de validación de las arquitecturas propuestas. La tercera y última parte de la tesis, por una parte, describirá todo el proceso de diseño llevado a cabo para la validación de la arquitectura PTP SoC segura. Este proceso incluirá el desarrollo de nuevas interfaces hardware y software necesarios para la integración de los módulos identificados en el estado del arte. Por otro lado, se estudiará el efecto de los mecanismos de seguridad MACsec propuestos sobre el rendimiento del protocolo de sincronización. Para ello, se analizarán los resultados de diversos experimentos realizados en una variedad de configuraciones, desde redes básicas experimentales implementadas en el laboratorio hasta redes más complejas en entornos industriales reales.

Abstract

New generation power grids, named Smart Grids, must offer some intelligent services with strong operation requirements. In order to fulfil these requirements, substations need to be modernised by installing new intelligent electronic devices to provide primary equipment control and protection functions. These intelligent devices might be accessed locally or remotely so as to assure a quick and automatic response to power supply problems. Therefore, they must be interconnected through digital buses that configure a communication network within the substation, and they must also provide external communication interfaces to a remote control center, for example.

As a consequence of the digitalization process in substations, new digital threats that did not happen in traditional power grids arise. Due to the interconnection from protection equipment to external communication networks or even to the Internet, substations are exposed to cyber attacks from people located at hundreds or thousands of kilometres. These attacks could result in severe failures like the interruption of the power supply over large urban or industrial areas. In this sense, protection devices are normally based on embedded computing systems that require an specific approach to face cyber-security challenges.

This thesis aims to solve the problem of securing one of the most sensitive communication protocols among those recommended to be used in substations. This is the case of the Precision Time Protocol (PTP), which distributes very precisely a time reference from a master to numerous networked slaves. PTP is highly vulnerable since a timer error of only one microsecond could lead protection functions to serious problems or even interrupt the operation of the primary equipment.

Firstly, the state of the art about communications and cyber-security in substations will be presented. Particularly, identified difficulties when securing the mentioned synchronization protocol will be described. Then, it will be performed a deep study on off-the-shelf hardware and software components to be considered in the design of electronic systems based on reconfigurable devices. The objective is to integrate the PTP protocol with the Ethernet security standard, known as MACsec, in the same silicon chip. Secondly, a new System-on-Chip (SoC) architecture based on reconfigurable devices including both PTP and MACsec support will be proposed. In addition, a new group key management and distribution scheme to be employed in industrial networks protected by MACsec will be introduced.

Finally, Xilinx Zynq device will be used as the validation platform to analyse the viability of proposed architectures. This last part of the thesis, on the one hand, will describe the whole design process performed to validate the secure PTP SoC architecture. It will include the development of new interfaces between those hardware and software modules identified in the literature. On the other hand, the effect of MACsec security mechanisms on PTP protocol performance will be tested. In this way, results obtained from diverse experiments executed over a variety of setup configurations will be analysed. The used setups range from basic experimental networks in the laboratory to more complex real networks in industrial plants.

Contents

List of Figures

List of Tables

List of Acronyms

AAA	*Authentication, Authorization and Accounting*
ACSI	*Abstract Communication Service Interface*
AES	*Advanced Encryption Standard*
API	*Application Programming Interface*
ASIC	*Application-Specific Integrated Circuit*
AXI	*Advanced eXtensible Interface*
BC	*Boundary Clock*
BMC	*Best Master Clock algorithm*
BSD	*Berkeley Software Distribution license*
CAK	*Connectivity Association Key*
CAM	*Content Addressable Memory*
CMAC	*Cipher-based Message Authentication Code*
CP	*Controlled Port*
CPPS	*Cyber-Physical Production Systems*
CRC	*Cyclic Redundancy Check*
CTs/VTs	*Current and Voltage Transformers*
DAN	*Double Attached Node*
DDR	*Double Data Rate memory*
DMA	*Direct Memory Access*
DNP3	*Distributed Network Protocol version 3*
DoS	*Denial of Service*
E2E TC	*End-to-End Transparent Clock*
EAP	*Extensible Authentication Protocol*
EAPOL	*Extensible Authentication Protocol over LAN*
ECDSA	*Elliptic Curve Digital Signature Algorithms*
EDK	*Embedded Development Kit*

FCS	*Frame Check Sequence*
FIFO	*First In First Out*
FMC	*FPGA Mezzanine Card*
FPGA	*Field Programmable Gate Arrays*
FSBL	*First Stage Boot Loader*
FSM	*Finite State Machine*
GCM	*Galois Counter Mode*
GDOI	*Group Domain of Interpretation*
GIC	*Generic Interrupt Controller*
GMAC	*Gigabit Media Access Controller*
GMII	*Gigabit Medium Independent Interface*
GOOSE	*Generic Object Oriented Substation Events*
GPIO	*General Purpose Input Output*
GNU GPL	*GNU General Public License*
GNU LGPL	*GNU Lesser General Public License*
HC	*Hybrid Clock*
HMAC	*keyed-Hash Message Authentication Code*
HMI	*Human Machine Interface*
HSR	*High-availability Redundancy*
ICK	*MACsec ICV Key*
ICV	*Integrity Check Value*
IEC	*International Electrotechnical Commission*
IED	*Intelligent Electronic Device*
IEEE	*Institute of Electrical and Electronics Engineering*
IETF	*Internet Engineering Task Force community*
IFG	*Inter Frame Gap*
IKE	*Internet Key Exchange protocol*
IO	*Input/Output*
IP	*Internet Protocol*
	Intellectual Property core
IPsec	*Internet Protocol security*
IRIG-B	*Inter-Range Instrumentation Group time code B*
IV	*Initialization Vector*
KaY	*MACsec Key Agreement Entity*
KDF	*Key Derivation Function*
KEK	*MACsec Key Encryption Key*
LAN	*Local Area Network*

LLC	*Logical Link Control protocol*
MAC	*Media Access Control*
	Media Access Controller
	Message Authentication Code
MACsec	*Media Access Control Security*
MII	*Media Independent Interface*
MITM	*Man-In-The-Middle attack*
MKA	*MACsec Key Agreement protocol*
MM2S	*Memory-Mapped-to-Stream channel*
MMS	*Manufacturing Message Specification*
MPDU	*MAC Protocol Data Unit*
MSK	*Master Session Key*
MTU	*Master Terminal Unit*
MU	*Merging Unit*
NAPI	*Linux New Application Programming Interface*
NAT	*Network Address Translation*
NIC	*Network Interface Card*
NIST	*National Institute of Standards and Technology*
NTP	*Network Time Protocol*
NTS	*Network Time Security*
OC	*Ordinary Clock*
OSI	*Open Systems Interconnection*
OS	*Operating System*
P1588 Security SC	*IEEE P1588 Security Subcommittee*
P2P TC	*Peer-to-Peer Transparent Clock*
PAE	*Port Access Entity*
PACP	*Port Access Control Protocol*
PHC	*Linux PTP Hardware Clock*
PHY	*Physical Interface Transceiver*
PL	*Zynq Programmable Logic section*
PN	*Packet Number*
PPS	*Pulse Per Second signal*
PRP	*Parallel Redundancy Protocol*
PS	*Zynq Processing System section*
PTP	*Precision Time Protocol*
RADIUS	*Remote Authentication Dial-In User Service*
RFC	*Request For Comments*

RSA	*Rivest, Shamir and Adleman cryptographic algorithm*
RSTP	*Rapid Spanning Tree Protocol*
RTC	*Real Time Clock*
RTU	*Remote Terminal Unit*
S2MM	*Stream-to-Memory-Mapped channel*
SA	*Security Association*
SAK	*Security Association Key*
SAS	*Substation Automation System*
SCADA	*Supervisory Control And Data Acquisition*
SCI	*MACsec Secure Channel Identifier*
SCL	*Substation Configuration Language*
SCSM	*Specific Communication Service Mapping*
SDK	*Software Development Kit*
SecY	*MACsec entity*
SecTAG	*MACsec Security Tag*
SHA	*Secure Hash Algorithm*
SoC	*System-on-Chip*
SV	*Sampled Values*
TC	*Transparent Clock*
TCP/IP	*Transmission Control Protocol/Internet Protocol*
TESLA	*Time Efficient Stream Loss-tolerant Authentication protocol*
TICTOC	*Timing over IP Connection and Transfer of Clock working group*
TLS	*Transport Layer Security*
TLV	*Type-Length-Value*
TSN	*Time Sensitive Networking*
TSU	*Timestamp Unit*
UART	*Universal Asynchronous Receiver-Transmitter*
UCA	*Utility Communication Architecture*
UES	*Unmanaged Ethernet Switch IP core from SoC-e*
UDP	*User Datagram Protocol*
UP	*Uncontrolled Port*
VHDL	*VHSIC Hardware Description Language*
VLAN	*Virtual Local Area Network*
WAN	*Wide Area Network*
WB	*Wishbone Bus*
WPA	*WiFi Protected Access protocol*
XML	*EXtensible Mark-Up Language*

Chapter 1

Introduction

1.1 Problem statement

The evolution of information and communication technologies have motivated the transformation of traditional power grids to modern Smart Grids. The Smart Grid defined as the next-generation electrical power system, has to fulfil strong requirements regarding reliability, flexibility, efficiency and environmentally friendly operation. The communication systems in conjunction with the cyber-security are critical parts in the power system that will allow a more reliable and efficient generation and distribution of electricity. The 'security-by-obscurity' of old proprietary solutions in Substation Automation Systems (SAS) gives way to the use of very well-known cryptographic software and public standards [1]. The robustness of standard solutions is based on the complexity of the algorithms rather than the inaccessibility to source codes. Therefore, while new possibilities are emerging, also new digital threats must be properly handled by communication systems.

Several significant events have exhibited the vulnerabilities of the communication framework in power systems since 2003, when a nuclear plant crashed because its control system network was infected by the Slammer worm that bypassed firewalls causing the safety monitoring system to be disable for nearly five hours [2]. Similarly, the computer worm Stuxnet infected the software of many industrial sites in Iran, including a nuclear plant which uses Siemens industrial control programs based on Microsoft Windows [3]. Stuxnet was the first worm known to attack Supervisory Control And Data Acquisition (SCADA) systems. It spies on the operations of the system and, then, it uses this information to take control of machines turning them into failure. The worm is believed to have been created with the support of a nation or state in an attempt of starting a cyber-war caused by geopolitical conflicts. After that, other variants of the Stuxnet worm were discovered such as Duqu, Flame and Gauss with the aim of spying on industries, people and banks. These worms can be quickly spread over USB sticks and networks without being detected by automated detection systems.

In last ten years, several organisms, such as those that formed the International Council on Large Electric Systems (in French, *Conseil International des Grands Réseaux Électriques or CIGRÉ*), the International Electrotechnical Commission (IEC) and the Institute of Electrical and Electronics Engineering (IEEE) have put a great effort concerning cyber-security on power systems [1]. IEC 62351 Security Standards address security issues for the different power system operations and communication standards defined by the IEC Technical Committee 57 (IEC TC57) [4]. On the one hand, this standard mandates that an asymmetric cryptosystem must be used to provide source authenticity of Generic Object Oriented Substation Event (GOOSE) and Sampled Values (SV) messages. However, despite expensive processors with crypto accelerators were utilized, execution times would exceed the maximum transfer times stated in the standard for most time critical applications [5]. On the other hand, the recommended synchronization solution is the Precision Time Protocol (PTP), as defined in IEEE 1588-2008, which introduced an optional security extension based on old keyed hash algorithms that has also been demonstrated to be suboptimal due to latency times and required resources.

This PTP security extension was defined in Annex K of the second version of the standard, but presented several vulnerabilities. The third edition of the standard, on which 1588 working group is now intensively working, will include a set of security mechanisms to be used individually or in combination to address security requirements of each particular application. In general terms, these security mechanisms will include an end-to-end security mechanism integrated in PTP based on a new Security Type-Length-Value (TLV) field to provide source authentication and, a hop-by-hop security solution based on some external security protocols to provide hop-by-hop authentication and integrity protection.

In this sense, data that is not modify on each hop should not rise a problem if a suitable key management scheme is used and symmetric cryptographic algorithms are efficiently implemented in end nodes. The real challenge is the integration of a hop-by-hop security solution without compromising the SAS performance. In fact, one of the main problems of deploying secured and redundant substation networks is the increased propagation delays of packets due to cryptographic units in intermediate nodes: since they modify the content of PTP messages, they must recalculate the associated security checksums before transmitting the messages. At the same time, the need for the participation of intermediate nodes makes difficult the management and distribution of cryptographic keys.

In this thesis, the problem of integrating hop-by-hop security mechanisms in PTP layer 2 networks is addressed. Media Access Control security (MACsec) as defined in IEEE 802.1AE-2006 standard was identified as an appealing alternative to provide cyber-security on each hop within PTP over Ethernet systems. Since modern Field Programmable Gate Array (FPGA) devices provide flexibility of reconfigurable logic and coexistence of hardware and software processing, they acquire special relevance in the development of future substation Intelligent Electronic Devices (IEDs) that will integrate authentication and integrity protection capabilities. As a consequence, the objective was focused on taking advantage of the experience of the research group in the design of System-on-Chip (SoC) electronic systems in conjunction with the knowledge on industrial communications and cryptographic algorithms implementation, with the aim of defining a secure PTP SoC architecture and validating the protocol performance using Xilinx FPGAs.

1.2 Objectives

As commented above, the aim of this work is to provide cyber-security mechanisms to PTP over Ethernet networks by defining a new SoC architecture with security capabilities. Bearing in mind this research line, this thesis deals with several objectives:

- An analysis of the state-of-the-art in communication technologies and cyber-security standards in substations, as well as the evolution of security mechanisms to protect the recommended precise clock synchronization protocol in SAS.
- A deep study on different techniques to design secure SoC architectures and deploy IED nodes with PTP and security support. This study includes a comparison of current available solutions and an early detection of required improvements and possible enhancements.
- A new secure PTP SoC architecture proposal, taking into consideration the issues discovered regarding the utilization of current commercial off-the-shelf solutions to deploy secure IEDs. To accomplish this objective, also new techniques to minimize the impact of cyber-security mechanisms

on substation communication performance are presented.

- The design and development of several FPGA-based implementations and different test setups in order to validate proposed secure SoC architectures and demonstrate their operation without compromising PTP protocol performance.

1.3 Organization

This manuscript is divided into three parts, where some parts span over multiple Chapters:

- **State of the art.** Standards related to substation communications and cyber-security measures in substations are presented and described in Chapter 2. With regard of securing the precision time protocol, the work done by the scientific community and the security committee within the standardization group is also deeply analysed in Chapter 3. Chapter 4 gives an overview on current commercial off-the-shelf and SoC architectures found in the literature related to the development of IEDs with precise synchronization and cyber-security support.
- **Contribution.** After summarizing all the information found in the literature regarding cyber-security in substations and synchronization systems as a list of design and security requirements, a MACsec-based proposal is presented in Chapter 5. Concretely, as a result of this thesis two main contributions are described: a new key scheme for the management and distribution of MACsec group keys for the particular case of PTP over Ethernet Local Arean Networks (LANs) and a new secure SoC architecture for the development of substation IEDs including PTP and MACsec support.
- **Validation.** The viability of implementing the proposed SoC architectures is demonstrated in Chapters 6 and 7. The former describes initial experiments without MACsec support that were performed over different test scenarios with the aim of acquiring a better knowledge of the design guidelines and testing techniques when working with PTP SoC architectures. The latter means a step forward in the validation process, since MACsec hardware units are integrated in the design.

Finally, Chapter 8 summarizes the complete manuscript and gives an outlook to future research lines based on the scientific findings in this work.

Chapter 2

Cyber-security in substation automation systems

2.1 Introduction

As it has been mentioned in Chapter 1, the cyber-security of several industrial plants has been compromised for the last years by some worms and viruses, such as Stuxnet, which was able to take control of the SCADA system of a nuclear plant in Iran. Consequently, the research community and the international standardization committees raised their awareness about protecting information in SASs. IEC 61850-5 and IEC 62351-6 standards respectively describe communication models and the security mechanisms to be deployed in current substations, but they present some inconsistencies that are addressed in this Chapter.

Section 2.2 gives an overview of the evolution of substation communications from parallel copper wiring architectures to modern SASs. The problem of interoperability raised in traditional SCADA systems was solved with the publication of the IEC 61850 family of standards, which is presented in Section 2.3. Security aspects regarding the protection of different communication models defined in IEC 61850 was targeted by the IEC 62315 family of security standards described in Section 2.4. Other protocols in substations are introduced in Section 2.5 and security issues to be considered in the near future are summarized in Section 2.6.

2.2 Evolution of substation communications

Conventional power generation, transmission and distribution networks are naturally evolving to Smart Grids and, consequently, the interest of the research community in the development of new functions has been growing considerably. The Smart Grid technology is defined by the European Technology Platform as "an electricity network that can intelligently integrate the actions of all users connected to it - generators, consumers and those that do both - in order to efficiently deliver sustainable, economic and secure electricity supplies" [6]. In order to optimize the operation of the interconnected elements in power systems, it is required a two-way flow of power and information in electrical and communication networks respectively [7].

Substations are the nodes in the electrical power network that connect the lines and cables to transmit and distribute electricity. Local functions in substations include data acquisition from the power grid via the switchgear (sensors) and activation of changes by commands to switchgear devices (actuators). For instance, a protection function may issue a trip command to the allocated circuit breaker in case of fault detection.

In the eighties, the substation architecture gradually evolved from rigid parallel copper wiring to some proprietary solutions based on telephone, teletype or modem technologies [8]. Also, some standard protocols for industrial automation were employed in power systems, such as the Distributed Network Protocol version 3 (DNP3). Local functions in substations were performed through centralized SCADA systems which consisted of a Master Terminal Unit (MTU) placed at a control center, and several Remote Terminal Units (RTUs) geographically distributed. RTUs were devices that gathered data and sent control commands to switchgear components. The MTU continuously checked the state of the RTUs and if they intended to use the shared bus, by sending them messages periodically.

Modern SASs consist of a variety of microprocessor based IEDs and primary equipment (high-voltage circuit breakers, disconnectors, earthing switches, gas insulators, transformers, etc.) that provide local or remote access to the power system, manual or automatic functions and communication links and interfaces to the switchgear. In contrast to RTUs, SASs perform all local tasks in a more decentralized structure and the communication function of the RTU is often implemented in a gateway IED, which should convert communication protocols in both directions. Thus, the information collected and stored in IEDs is transferred to the SCADA master via this gateway [9]. SASs improve the control of the network, allowing a self-response to problems in few seconds. Hence, the highest benefits of SASs are the minimization of the number of outages and their duration, the reduction of operating costs, the increase of productivity and the improvement of power system performance [9]. In addition, since SASs improve the quality of the service and the power quality, the positive impact on end customer experience increases his satisfaction.

In traditional SCADA systems, the existence of many proprietary solutions made difficult the interoperability between devices, even between different versions of devices from the same supplier. Expensive protocol converters or re-engineering were needed to mitigate the problem of interoperability. In 1994, the IEC started to work on developing a common standard for substation communications, while the IEEE started a similar work on developing a common communication framework called Utility Communication Architecture (UCA). After that, in 1997, both organisms started to work together on the development of the standard IEC 61850, titled "Communication networks and systems for power utility automation" [7]. The different parts of the standard were first published between 2002 and 2005 becoming the only standard that provides an open architecture and assures interoperability between IEDs from different vendors. The main objectives of the standard were interoperability, free architecture and long term stability [10].

2.3 IEC 61850 standard overview

While communication technologies change very rapidly, substations have lifetimes of 30 years and longer. Consequently, the standardization was focused on the object data model rather than the communication technology. Objects are breakers, controllers or other protection elements that exchange data with each other using standardized services [9].

Although the IEC 61850 standard was initially oriented to communications within the substation, it has been expanded to the whole power system including communications between substations and other inter domain communications, as shown in Figure 2.1. IEC 61850 parts are listed in Table 2.1. Parts from 3 to 5 define general and functional requirements as well as project management. Part 6 defines the Substation Configuration Language (SCL) that is based on Extensible Mark-Up Language (XML) and it is used to exchange configuration information. The rest of parts define communication services, data models and the mapping of services to different network protocols for both intra and inter substation communications.

In order to meet the free configuration and long-term stability requirements, the IEC 61850 standard follows an Open Systems Interconnection (OSI) 7 layer

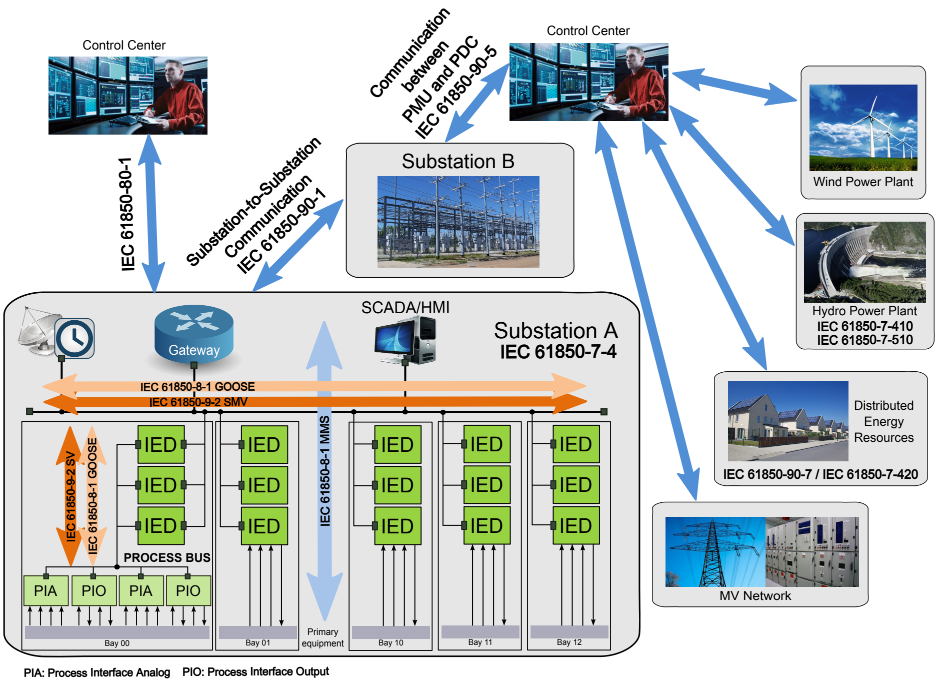


Figure 2.1: IEC 61850 standards family scope

Tabla 2.1: IEC 61850 standard parts

IEC61850	Title	Version	Date
Part 1	Introduction and Overview	ed2.0	2013/03
Part 2	Glossary	ed1.0	2003/08
Part 3	General requirements	ed2.0	2013/12
Part 4	System and project management	ed2.0	2011/04
Part 5	Communication requirements for functions and device models	ed2.0	2013/01
Part 6	Configuration description language for communication in electrical substations related to IEDs	ed2.0	2009/12
Part 7-1	Basic communication structure - Principles and models	ed2.0	2011/07
Part 7-2	Basic communication structure - Abstract Communication Service Interface (ACSI)	ed2.0	2010/08
Part 7-3	Basic communication structure - Common data classes	ed2.0	2010/12
Part 7-4	Basic communication structure - Compatible logical node classes and data classes	ed2.0	2010/03
Part 7-410	Basic communication structure - Hydroelectric power plants - Communication for monitoring and control	ed2.0	2012/10
Part 7-420	Basic communication structure - Distributed energy resources logical nodes	ed1.0	2009/03
Part 7-510	Basic communication structure - Hydroelectric power plants - Modelling concepts and guidelines	ed1.0	2012/03
Part 8-1	Specific Communication Service Mapping (SCSM) - Mappings to MMS (ISO 9506-1 and ISO 9506-2) and to ISO/IEC 8802-3	ed2.0	2011/06
Part 9-2	Specific Communication Service Mapping (SCSM) - Sampled values over ISO/IEC 8802-3	ed2.0	2011/09
Part 9-3	Precision time protocol profile for power utility automation	ed1.0	2016/05
Part 10	Conformance testing	ed2.0	2012/12
Part 80-1	Guideline to exchanging information from a CDC-based data model using IEC 60870-5-101 or IEC 60870-5-104	ed2.0	2016/07
Part 80-3	Mapping to web protocols - Requirements and technical choices	ed1.0	2015/11
Part 80-4	Translation from the COSEM object model (IEC 62056) to the IEC 61850 data model	ed1.0	2016/03
Part 90-1	Use of IEC 61850 for the communication between substations	ed1.0	2010/03
Part 90-2	Using IEC 61850 for communication between substations and control centres	ed1.0	2016/02
Part 90-3	Using IEC 61850 for condition monitoring diagnosis and analysis	ed1.0	2016/05
Part 90-4	Network engineering guidelines	ed1.0	2013/08
Part 90-5	Use of IEC 61850 to transmit synchrophasor information according to IEEE C37.118	ed1.0	2012/05
Part 90-7	Object models for power converters in Distributed Energy Resources (DER) systems	ed1.0	2013/02
Part 90-8	Object model for E-mobility	ed1.0	2016/04
Part 90-12	Wide area network engineering guidelines	ed1.0	2015/07

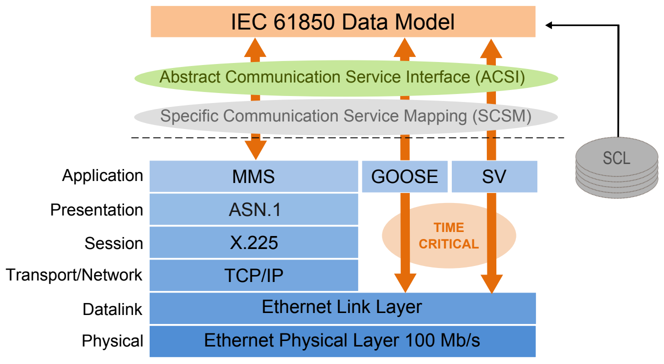


Figure 2.2: IEC 61850 communication stack

model, where substation data services and applications are built above the application layer. The communication stack is represented in Figure 2.2. The Abstract Communication Service Interface (ACSI) is a virtual interface to an IED that provides abstract services independently of the communication stack, while the Specific Communication Service Mapping (SCSM) defines the concrete mapping of ACSI services and objects into a particular protocol (IEC 61850-8-1 [11], IEC 61850-9-2 [12]).

The standard specifies three types of communication models for these ACSI services: client/server communication, publisher/subscriber model with GOOSE messages and multicast Ethernet based SV model. Most of IEC 61850 communications are based on client/server communication using the Manufacturing Message Specification (MMS) standard over Transmission Control Protocol/Internet Protocol (TCP/IP), as defined in [13], which provides a more reliable data transfer. An example of these services is the fault and event recording. For time critical applications, such as a protection function issuing a trip command, data is directly mapped to the Ethernet data link layer as GOOSE messages or SVs transmission using connectionless multicast addressing of frames. For instance, a protection function issuing a trip command requires transfer times below 3 milliseconds and, hence, information is directly mapped into a GOOSE message. While GOOSE messages generally transmit binary data such as indications, alarms and tripping signals, SVs are used to transfer current and voltage raw samples from Current and Voltage Transformers (CTs/VTs) to IEDs.

Figure 2.3 shows the substation automation model and its communication interfaces [14], although not all the interfaces enumerated in the figure must be present in any substation. There are three different levels: the station level, the bay level and the process level. The process bus is the communication network which connects IEDs at primary equipment level, whereas the station bus connects IEDs at bay level and IEDs at station level. Process level devices are typically switchyard apparatus, intelligent sensors and actuators, remote IOs and CTs/VTs. Bay level devices are needed for control, protection and monitoring functions. Therefore, the primary equipment and the protection and control IEDs are in the process and bay levels respectively. The station level consists of a computer with a database and a Human Machine Interface (HMI) to be controlled by an operator and the interfaces for remote communication. Originally, the process bus was specified as the carrier of the SV traffic and the station bus as the carrier of MMS and GOOSE traffic, but both can carry all types of traffic. Interfaces in Figure 2.3 are listed below:

1. Protection-data exchange between bay level and station level.
2. Protection-data exchange between bay level and remote protection.
3. Data exchange within bay level.
4. Analogue data exchange between process level and bay level.
5. Control data exchange between process level and bay level.
6. Control data exchange between bay level and station level.
7. Data exchange between the substation and a remote operator.
8. Direct data exchange between the bays especially for fast functions.
9. Data exchange within station level.
10. Control data exchange between the substation and remote control centre.
11. Control data exchange between substations.

For a detailed description on each communication model within substations according to IEC 61850 standard, reader is encouraged to read parts 8-1 [11] and 9-2 [12].

2.4 IEC 62351 standard for security in IEC 61850 communications

The interest of the research community in the protection of critical infrastructures, specially the electricity generation and distribution grids, has increased over the past decade: when the electricity stops, everything stops [15]. As it has

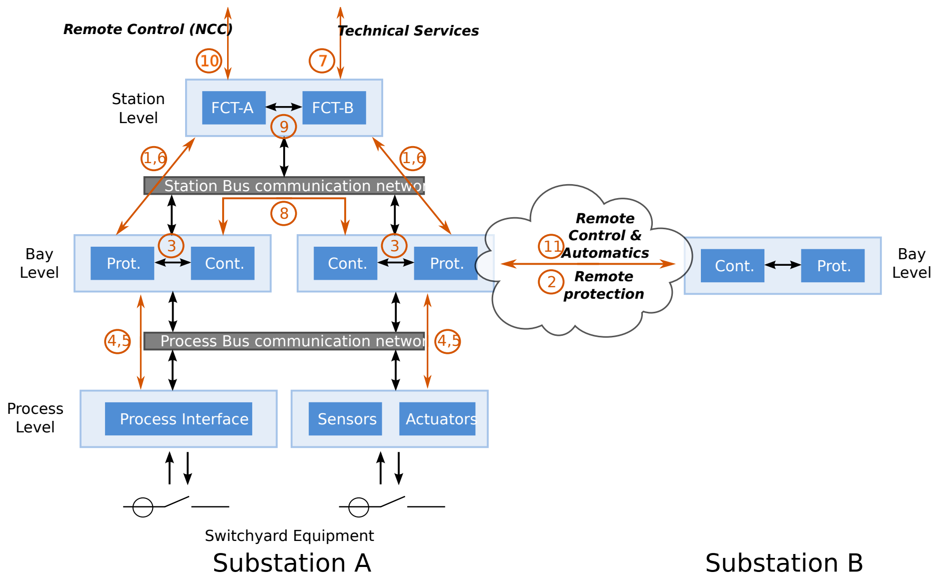


Figure 2.3: Interface model within a SAS

been introduced in Section 2.2, power systems are becoming computerised and control equipment interconnection is evolving from proprietary solutions to standard communication networks. This evolution reduces the costs of deployment but also opens up new digital vulnerabilities, since clear information packets can be easily sniffed, altered or recorded and played black. Examples of real-world cases of cyber-security intrusions have already been described in Chapter 1.

A lot of effort has been put on preventing that people from the Internet could gain unauthorized access to substation control systems. Therefore, the use of firewalls and other boundary control devices to control the information flow between a power plant and the external network, like the Waterfall Unidirectional Security Gateway [16], has been the highest priority. However, this perimeter defences are insufficient as it was demonstrated with the infection of SCADA control systems via USB flash drives, where no Internet access was needed. The ideal scenario would be the protection of each individual device within the control network. In addition, security firewalls and gateways are very difficult to maintain, since devices and networks in power systems are managed by multiple firms and constantly require changes in network configurations [15].

The IEC TC57 working group 15 works in the development of cyber-security

Tabla 2.2: IEC 62351 standard parts

IEC62351	Title	Version	Date
Part 1	Communication network and system security - Introduction to security issues	ed1.0	2007/05
Part 2	Glossary of terms	ed1.0	2008/08
Part 3	Communication network and system security - Profiles including TCP/IP	ed1.0	2014/10
Part 4	Profiles including MMS	ed1.0	2007/06
Part 5	Security for IEC 60870-5 and derivatives	ed2.0	2013/04
Part 6	Security for IEC 61850	ed1.0	2007/06
Part 7	Network and System Management (NSM) data object models	ed1.0	2010/07
Part 8	Role-based access control	ed1.0	2011/09
Part 9	Key management	ed1.0	Pending
Part 10	Security architecture guidelines	ed1.0	2012/10
Part 11	Security for XML documents	ed1.0	2016/09
Part 12	Resilience and security recommendations for power systems with Distributed Energy Resources (DER) cyber-physical systems	ed1.0	2016/04
Part 13	Guidelines on security topics to be covered in standards and specifications	ed1.0	2016/08

standards for power system communications, with the aim of covering both information infrastructure and communication security [4]. In particular, they focused in the communication protocols defined in IEC 60870-5/6, 61850, 61970 and 61968 series. In 2007, the IEC 62351 security standard for the power system information infrastructure was first published and the work is still in process. Table 2.2 lists the parts that currently compose the standard. The first part describes the background on security for power system operations and introduces the remaining parts [17]. Parts 3-6 specifies how to provide security services for the protocols mentioned above and listed in Table 2.1.

In particular, IEC 62351-6 [18] specifies the security mechanisms for protecting communications defined in the IEC 61850 family of standards. The implementation of cryptographic algorithms in power system devices with constrained memory and processing power is a challenge that is partially acknowledged in this part. Thereby, applications based on GOOSE and SVs, which require short transfer times, should only use authentication for ensuring data integrity and source authenticity, but not confidentiality. This standard proposes protecting GOOSE and SV messages with Message Authentication Codes (MACs) using the Secure Hash Algorithm (SHA), which are digitally signed using Rivest, Shamir and Adleman (RSA) public-key cryptosystem to provide source authenticity.

The main drawback of RSA digital signatures are the long execution times for both computation and verification of the signature. Thus, even though a high-end ARM processor with a crypto accelerator core were utilized in substation equipment, RSA signature with 1024-bit keys could not be computed and verified within 3 milliseconds, which is the maximum transfer time required by some GOOSE messages. In addition, deployment costs would increase considerably. Another alternative to that proposed in the standard, would be the implementation of Elliptic Curve Digital Signature Algorithms (ECDSA) in dedicated crypto cores, which could offer latency times required by IEC 61850 for fast messages [15].

IEC is now working on the first edition of the Part 9 regarding key management, which is expected to be published as an international standard in 2017. This standard will be based on the Group Domain of Interpretation (GDOI). Similarly, Part 6 is planned to be updated shortly as a second edition based on security requirements defined in IEC 61850-90-5 for synchrophasor communications over wide area networks [19]. In this case, communications are based on User Datagram Protocol (UDP) over IP, which also allows multicast transmission, and they require confidentiality protection apart from integrity and authenticity. In order to minimize the security impact on the performance of field devices, instead of digital signatures, symmetric cryptography is proposed as the protection mechanism. A shared group key should be distributed from a key center to the group participants after being authenticated using GDOI protocol [20].

The rest of applications, which are based on MMS, should also include data confidentiality in addition to authentication and they are secured at application and transport levels as described in IEC 62351-3 and IEC 62351-4. End-to-end authentication is provided using Transport Layer Security (TLS) Version 1.0, as defined by the Request for Comments (RFC) 2246 [21]. Some of the included cryptographic algorithms are RSA for key exchange, Advanced Encryption Standard (AES) for data encryption and SHA for message authentication.

2.5 Other communication protocols for efficient and reliable SASs

Apart from cyber-security mechanisms to prevent attacks against the legitimacy of the exchanged information, in order to deploy reliable and efficient communication networks in substations, additional protocols are required. In particular, in this Section, some protocols that are needed to provide precise time synchronization and redundancy are introduced.

2.5.1 Synchronization protocols

When the blackout from August 2003 in the Northeast United States and Ontario [22] occurred, the alignment of fault records from different locations in the post-fault analysis was so difficult that time synchronization became the focus in such systems. In the past, the identification of simultaneous incidents was done by skilled personnel who observed continuously at the waveforms. However, an automated evaluation of these incidents could quickly deliver hints that could immediately be used for remedial action, not just for post-fault analysis. This automated evaluation is only possible by assigning accurate timestamps during the recording of faults, and thus, a powerful time synchronization system is required [23].

All devices in SASs must have the same time reference so as to analyse globally the response of the system and, in case of fault, analyse precisely why, where and when this fault occurred [9]. For current and voltage samples, a time synchronization accuracy in the order of one microsecond is required. Even for fault detection and location in power transmission lines of the power grid, in order to measure the time that a travelling wave takes to traverse the line, a precise synchronization accuracy below one microsecond is desired. As an example, a time error of one microsecond results in a fault location error of 300 meters [23].

Traditionally, the time transmission standard defined by the Inter-Range Instrumentation Group (IRIG-B) [24, 25] was the common synchronization scheme in substations, which provides one microsecond accuracy using dedicated cabling infrastructure. This infrastructure is normally not redundant, increasing considerably implementation and maintenance costs. Usually, substations need long cables, in the range of 300-400 meters, from control building to instrument transformers resulting in varying propagation delays that must be compensated with complicated and bothersome calibration processes [26].

In addition, the tendency is to gather all communications over the same Ethernet based data network, particularly over a shared network process bus in SASs. The IEC Smart Grid Strategy Group and the National Institute of Standards and Technology of the United States (NIST) [27] recommend PTP protocol, as defined in IEEE 1588-2008 standard [28], for high precision time synchronization in substations. Concretely, with the introduction of field specific profiles, the IEEE firstly published a PTP Power Profile as the IEEE C37.238 standard [29] for Power Systems.

PTP automatically compensates for propagation delays and distributes absolute time across a substation directly over Ethernet providing accuracies in the range of nanoseconds. Furthermore, in contrast to IRIG-B systems, time can be

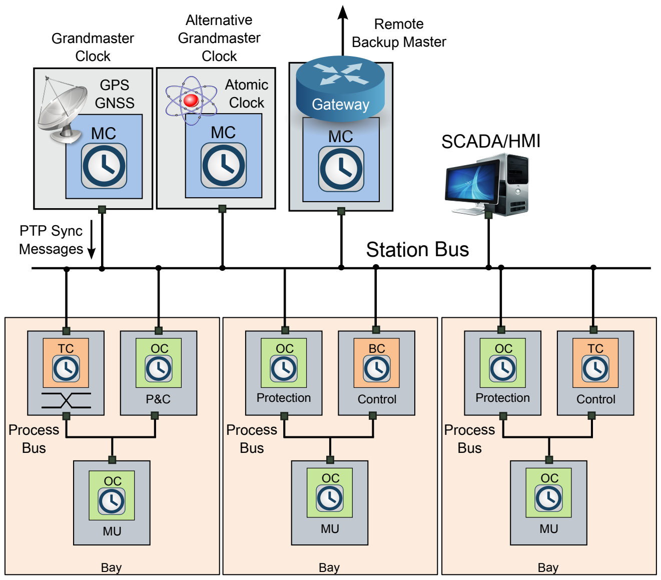


Figure 2.4: Example of a PTP network within the substation

transmitted over redundant Ethernet networks to increase the reliability of time distribution.

PTP systems follow a master-slave hierarchy, where the master imposes the time by sending regular *Sync* messages with accurate timestamps and slaves synchronize with it in both phase and frequency. In Figure 2.4, an example of IEEE 1588-aware substation network is shown [30]. End devices are called Ordinary Clocks (OCs) and may be Merging Units (MUs) that generate SV traffic or protection and control units. Intermediate nodes are Boundary Clocks (BCs) or Transparent Clocks (TCs) and may also hold protection and control functionalities. In contrast to BCs, TCs modify the content of PTP messages so as to consider latencies introduced by network nodes when computing the propagation delay. They basically measure the time the message takes to traverse the TC, named residence time, and accumulate it in a special PTP field called the *correctionField*. A detailed description of each type of node is addressed in Section 3.2.2.

However, PTP presents some security vulnerabilities and threats that might lead to different attacks on PTP systems ranging from Denial of Service (DoS) to selective packet delay attacks passing through clock manipulation by inserting, removing or modifying PTP packets [31–33]. As a consequence, a slave clock could be forced to be aligned to a false time or interruptions of PTP protocol could be occasioned. Although PTP firstly introduced security as an optional extension in Annex K of the second version of the standard, due to its limitations, it was never formalized into a properly-defined security protocol. For more information on Annex K security extension, see Section 3.2.4.

2.5.2 High availability networks

Apart from precise time synchronization, there are other critical issues in SASs such as the need for high availability networking. For instance, no traffic interruption is allowed for busbar protection functions in case of link failure. IEC 61850-90-4 considers the Rapid Spanning Tree Protocol (RSTP), the Parallel Redundancy Protocol (PRP) and the High-availability Seamless Redundancy (HSR) for IEC 61850-8-1 and IEC 61850-9-2 substation communications. The highest level of availability is achieved using HSR and PRP [34]. Both protocols provide zero recovery time and no frame loses in case of network failure. Otherwise, RSTP does not provide seamless recovery, but it recovers fast enough for most applications that use the station bus. In this Section, both PRP and HSR are considered, as defined in IEC 62439-3 standard [35].

On the one hand, PRP protocol uses a completely doubled network topology and Doubled Attached Nodes (DANs) as network interfaces. DANs send and receive all network traffic over both networks all at once. Detection and removal of duplicated frames is handled by protocol interfaces in a completely transparent way to the rest of devices [36]. On the other hand, HSR nodes only need an additional physical port to build a ring topology where DANs are daisy-chained. The sender sends two copies of the Ethernet frame in both directions of the ring and the destination passes the first arrived frame to the upper layers, while the second one is discarded in case of unicast communications. For multicast and broadcast ones, the HSR node must also forward it.

The topology of substation communication networks may differ depending on the physical location of IEDs as a consequence of electrical primary equipment configuration. Normally, a group of IEDs per bay is attached to a bridge as shown in Figure 2.1, although exceptions with IEDs serving several bays are also possible. Thus, the interconnection of IEDs in substations vary from a star topology to a daisy-chain or to a ring. With the aim of increasing the resiliency of the substation network, it can be segmented into multiple redundancy domains. An example topology with two redundancy domains for station and process bus is depicted in Figure 2.5, where the block diagram of PRP/HSR nodes and PTP clocks of a complex substation is represented [30]. In this Figure, a double LAN network is used on the station bus, which consist of two RSTP rings. The process bus is an HSR ring per each bay. In small substations, also HSR could fit in the station bus.

In order to couple non-redundant network nodes, such as the grandmaster clock or the substation gateway, and couple PRP and HSR networks, Redundancy Boxes (RedBoxes) are used. In the example network in Figure 2.5, there are two RedBoxes in each bay: RedBox A couples the orange RSTP ring in the station bus with the HSR ring in the process bus, while the RedBox B couples the green one with the same HSR ring. Although RedBoxes can also be TCs, in this case they are BCs and are treated as redundant clocks in the HSR ring. This means that only one of them sends Sync messages. Otherwise, in case they were TCs, they would inject four Sync messages with the same sequence number into the HSR ring. HSR end nodes in the process bus have an Hybrid Clock (HC) that is a combination of a TC and an OC.

2.5.3 Communication protocols incompatibility issues

PRP and HSR assume some principles about message propagation over the network that are quite incompatible to PTP [37, 38]. Annex A of IEC 62439-3 [35] defines how redundant PTP messages must be handled. This annex also specifies that PTP messages must be directly transported in multicast Ethernet frames and Peer-to-Peer Transparent Clock (P2P TC) functionality must be included in intermediate nodes. The P2P TC is a type of transparent clock that use the peer delay mechanism to measure the link delay and update the *correctionField* with both the residence time and the link delay associated with the ingress transmission path of Sync messages. Chapter 3 provides a detailed description on PTP operation and types of devices.

The IEC SC 65C subcommittee continued working on these incompatibility issues and on a new new edition of IEC 62439-3 standard, which was released in March 2016 [39]. From this third edition, it was concluded that a new PTP profile for Power Systems had emerged from IEC groups, named Utility Profile and specified as an Annex. Consequently, the existence of two different PTP profiles for Power Systems, the Power Profile defined in IEEE C37.238 standard [29] and this Utility Profile, arose a new incompatibility problem. Since the Utility Profile also considered the utilization of redundancy protocols, IEC and IEEE recently agreed on the utilization of the Utility Profile for SASs. The definition of this unique profile was moved from IEC SC 65C to IEC TC57 working group and it was published as the IEC 61850-9-3 standard in May 2016 [40].

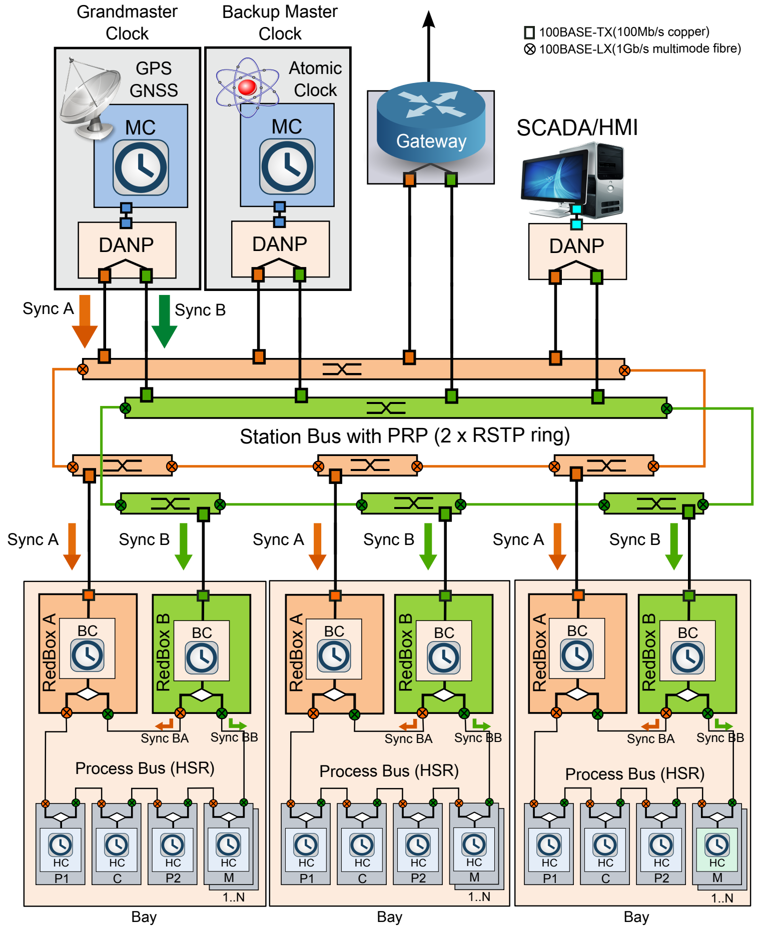


Figure 2.5: Station and process bus with redundancy and synchronization

In [41], authors firstly implemented a full hardware solution with IEC 62439-3 and IEEE 1588 support. They demonstrated to be cost effective and fast. In the worst case, the propagation delay through an HSR ring would be the product of each node bridging delay by the number of nodes. Since this delay must be acceptable for time critical messages in SASs, IEC 62439-3 estimates that each node in the HSR ring should forward the frames within 5 microseconds. Hence, the utilization of cut-through bridging is suggested, where the frame is forwarded before it is entirely received. Even though cut-through switching were implemented, only the average forwarding delay would be improved. In the worst-case scenario of all nodes injecting a maximum size frame at the same time, the overall propagation delay would not experience any reduction. In order to overcome this problem and fully exploit the cut-through properties, a pre-allocated time window would restrict the nodes to sending frames in particular moments and, accordingly, a common precision clock is needed.

However, the utilization of PTP in secured HSR networks implies that all nodes in the ring must recalculate the authentication code while the message passes through the nodes, in order to consider changes in *correctionField*. P2P TC functionality modifies the content of this field by adding the residence time and the link delay and, therefore, cryptographic units located in ingress and egress ports must respectively verify and recalculate security checksums that protect the integrity of PTP messages. In fact, the main drawback of implementing security mechanisms in substation redundant networks is the increased propagation delays of packets, which should be considered and minimized. Otherwise, GOOSE or SV messages will suffer long transfer times that will not meet timing requirements.

2.6 The future of cyber-security in SASs

Since the security mechanisms specified in both IEC 62351-6 and IEEE 1588-2008 standards have been demonstrated to be suboptimal, a dramatic change should be carried out in future versions. On the one hand, the use of digital signatures to protect GOOSE and SV messages is computationally expensive and presents long execution times that are not permissible for time critical applications. On the other hand, Annex K of IEEE 1588-2008 standard presents several vulnerabilities that will be further described in Section 3.3.1.

Rather than specifying an independent security mechanism for each communication service in substations, in order to save computational resources in restricted IEDs, the utilization of a common security framework to protect all substation communications should be considered in future versions of standards. Differ-

Tabla 2.3: Summary of security requirements for SAS communication services

Security Requirements	Communication Services in SASs					Security Protocols		
	MMS	GOOSE	SVs	End PTP	Peer PTP	TLS	IPsec	MACsec
Source authentication	MUST	MUST	MUST	MUST	-	X	X	-
Group authentication	-	-	-	MUST	MUST	-	-	X
Hop-by-hop integrity	-	MAY	MAY	MUST	MUST	-	-	X
End-to-end integrity	MUST	MUST	MUST	MUST	-	X	X	-
Confidentiality	MUST	-	-	-	-	X	X	X
Unicast key management	MUST	-	-	SHOULD	-	X	X	-
Multicast key management	-	MUST	MUST	SHOULD	SHOULD	-	-	X

ent types of traffic in SASs should meet different timing requirements, as well as different security requirements. Table 2.3 summarizes the general security requirements for different communication services found in substations. Apart from the three communication models described in Section 2.3, also PTP synchronization service is included. In fact, PTP messages have been classified into two categories. While Peer PTP messages are those involved in the peer delay mechanism and exchanged between adjacent nodes, End PTP messages are those transmitted from the master to slaves, and vice versa, through the network.

Before continuing with the specification and development of a new security solution for SAS communications, it is worth performing a preliminary study about the applicability of Information Technology (IT) security solutions in substation environments. Figure 2.6 shows available security protocols and standards in the OSI model that provide security services on each layer independently. Security services offered by these security suites are very similar and also their cryptographic mechanisms are often the same. However, they differ in the applicability scope: while TLS provides security at application and transport layers, network and data link layers are protected by Internet Protocol security (IPsec) and Media Access Control Security (MACsec) respectively.

TLS was already considered by IEC 62351-6 standard to provide end-to-end authentication of MMS-based client-server communications, as mentioned in Section 2.4. IPsec and MACsec, in contrast, have not been considered to protect substation communications yet because they show some drawbacks. While IPsec provides a great security solution in almost all IT applications, it is limited to data traffic that is transported over Internet Protocol (IP) at network layer. In the case of real time communications within the substation, all events need to be rapidly handled so as not to loss information and data is transported directly over Ethernet at link layer, as it has been seen in Section 2.3. Therefore, security at network layer is not feasible. On the other hand, MACsec provides

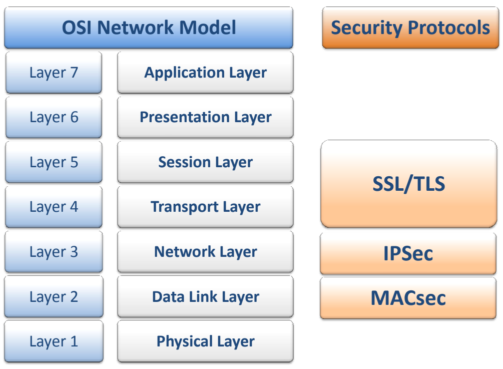


Figure 2.6: Security protocols in the OSI model

hop-by-hop integrity and authenticity, but not end-to-end source authenticity. Moreover, since each node in the path has to verify the integrity and authenticity of the message in the reception path and regenerate the authentication code in the transmission path, long latencies could be introduced in cascaded topologies, such as star or ring network configurations.

Apart from message protection using authentication codes, since many communications in SASs are multicast transmissions, a multicast key management protocol is also desired. In [19], several multicast key management protocols are presented. These multicast key management protocols are generally used to distribute symmetric group keys, therefore only group authentication is assured. In most cases, the protection offered by symmetric group keys only guarantees that a message was sent or last modified by a group membership, since all nodes in the group know the shared key and can both send and receive messages.

In order to achieve source or sender authentication, the most immediate method could be the asymmetric cryptography [42]. The main drawback of this approach is the computational cost, which is estimated to be about three times the cost of symmetric encryption. This is mainly due to the utilization of complex arithmetic functions with large numbers [43], since the length of asymmetric keys must be almost ten times the length of symmetric keys to achieve an equivalent security level. Nevertheless, the key management in asymmetric cryptography is simpler, so it is basically used to authenticate entities using digital signatures and establish symmetric keys that will encrypt session data.

As depicted in [19], the smarter solution to the problem of sender authentication in multicast communications is the utilization of key chains, such as the Timed

Efficient Stream Loss-tolerant Authentication (TESLA) protocol, which provides source authentication using symmetric cryptography. In such a key management scheme, keys are generated as the result of applying a cryptographic algorithm repeatedly. Thus, each key is valid for sending only during a limited time interval; when this interval expires, the sender release it in a secure way to receivers. The main drawback of using key chains to protect real time traffic like GOOSE and SV messages is the delay introduced when processing the received message while the node is waiting for the key. In [19], authors propose the utilization of early control command execution, which consists in performing the control action in the receiver immediately after the message reception but, if later on the MAC is invalid, the inverse operation is performed.

2.7 Conclusions

In this Chapter, the current outlook of cyber-security measures in substation communications has been presented. Process bus in substations has recently evolved to digital communication networks that connect microprocessor based smart devices. In order to assure interoperability between devices from different vendors, standard protocols defined in the IEC 61850 family of standards must be employed. Apart from communication models described in this standard, additional protocols such as IEEE 1588 and IEC 62439-3 for precise synchronization and redundancy are required.

With the introduction of standard protocols and software, new digital threats must be properly handled by communication systems and, hence, the IEC 62351 standard specifies the security mechanisms to protect communications defined in IEC 61850 standard. Similarly, the IEEE 1588 standard firstly introduced security as an optional extension in Annex K of the second version of the standard. However, both approaches have been demonstrated to be inefficient due to computational costs and processing latencies. Therefore, with the aim of saving computational resources, future versions of standards should specify a common security framework to protect all substation communications.

In order to address specific application requirements, the common security framework should consider an hybrid solution with hop-by-hop group authentication and integrity protection using symmetric cryptography and end-to-end source authentication using efficient cryptographic methods. Data that is not modified by IEEE 1588 transparent clock functionality in network nodes should not raise a problem if a suitable key management scheme is used and symmetric cryptographic algorithms are efficiently implemented in end nodes. The real challenge is the integration of a hop-by-hop security solution without compromising the SAS performance. In fact, as mentioned previously, one of the main problems of deploying secured and redundant substation networks is the increased propagation delays of packets due to cryptographic units in TCs: since they modify the content of PTP messages, they must recalculate the associated security checksums before retransmitting the messages. At the same time, the need for the participation of intermediate nodes makes difficult the management and distribution of cryptographic keys.

Chapter 3

Cyber-security in precise time synchronization networks

3.1 Introduction

As presented in Chapter 2, the use of time based systems in substations required all nodes to share a common sense of time. Traditionally, IRIG-B was the common synchronization scheme in substations. However, it requires a dedicated cabling infrastructure and complex calibration processes to compensate the varying propagation delays caused by disproportionate cable lengths in substations [24, 25]. Consequently, PTP protocol, as defined in IEEE 1588-2008 standard [28], is the synchronization protocol recommended by the IEC Smart Grid Strategy Group for SASs. In contrast to IRIG-B, PTP provides automatic compensation of propagation delays and can distribute absolute time across a substation directly over Ethernet, providing accuracies in the range of nanoseconds.

Due to its hop-by-hop synchronization approach, PTP introduces a security challenge: while an end-to-end security approach is generally more robust and secure, the need for the participation of intermediate nodes becomes a real challenge from the security perspective, since every cryptographic mechanism is as weak as the number of nodes that hold the cryptographic keys [33]. The Security Subcommittee within the IEEE P1588 Working Group (P1588 Security SC), is immerse in specifying the PTP security solution.

In this Chapter, a detailed review of PTP protocol and the evolution of the proposed cyber-security mechanisms to protect PTP traffic are drawn.

3.2 IEEE 1588 standard overview

The work in IEEE 1588 begun around 1990 in the Hewlett-Packard laboratories, and continued at Agilent Technologies headed by Dr John Eidson [26]. The initial goal of the protocol was to distribute timing in control and measurement systems. In November 2002, the first version of the standard was published as IEEE 1588-2002. Two years later, the IEC approved it as IEC 61588 First Edition.

After that, IEEE continued specifying the second version of the protocol, which was finally published in March 2008 as IEEE 1588-2008 [28] and adopted by the IEC as IEC 61588 Edition 2.0 in February 2009 [44]. This second version emerged because of the need to cover new areas such as telecommunications, audio/video bridging networks and power industry. In addition, some new requirements needed to be targeted: higher accuracy, variable update rates, rapid reconfiguration and fault tolerance, among others.

Some of the most important improvements in the new version were the support for hardware timestamping and the addition of a new device type, the transparent clock, a kind of IEEE 1588-aware bridge. Other relevant advancement was the introduction of application specific profiles, which are subsets of attributes and parameters that are specific for an application field. Also, an optional security extension was firstly introduced to provide integrity and authentication.

PTP systems follow a master-slave hierarchy, where the master imposes the time by sending regular *Sync* messages, and slaves synchronize to it in both phase and frequency. Accuracies in the nanoseconds range are achieved by simply exchanging regular PTP messages, which contain accurate timestamps as described in Section 3.2.1. These timestamps must be compensated by slaves for the propagation delay, which may be computed using two measurement mechanisms that need the participation of intermediate nodes, as explained in Sections 3.2.2 and 3.2.3. Finally, in Section 3.2.4, a description of the optional security extension included in the second version of the standard is reported.

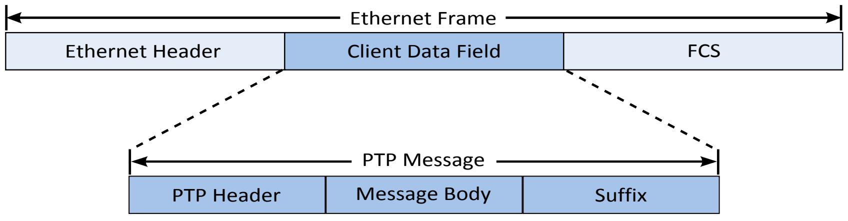


Figure 3.1: PTP over IEEE 802.3/Ethernet mapping

3.2.1 PTP messages

PTP messages can be transported over several network protocols such as UDP over IP, Ethernet, DeviceNET, ControlNET and Fieldbus. Since the tendency in substations is to gather all communications over the same Ethernet based data network, in this document only the mapping of PTP messages directly into Ethernet frames is considered. In this case, in order to identify PTP messages, the Ethertype field in the Ethernet header is set to 0x88F7 in hexadecimal.

Figure 3.1 shows the mapping of PTP messages over Ethernet frames. PTP messages consist of header, body and optional suffix. The header is common to all PTP messages and its format can be seen in Figure 3.2. The length of the message is indicated by the third and fourth octets within the PTP header and the *sourcePortIdentity* field contains a value that identifies the port that originated the message. In order to protect PTP protocol against replay attacks, a sequence number is included in the *sequenceId* field. The *correctionField* is the sum of all corrections made in transparent clocks as described in Section 3.2.2.3. More information about the rest of the fields may be found in the standard [28].

The type of PTP message is indicated by the *messageType* field in the header, so as to identify the information contained in the body. Table 3.1 lists the values of this field. The *Announce* message is used to indicate the capabilities of a clock to the rest of clocks in a domain, in order to establish the master-slave hierarchy. The *Sync* message is periodically sent by a master clock and it contains the time when the *Sync* message was sent, which is called the *originTimestamp*. If the master has not the capability to include the precise *originTimestamp* in the proper *Sync* message, it will be sent in the associated *Follow_Up* message. The *Delay_Req* message allows a slave to start the delay request-response mechanism to calculate the propagation delay, as explained in Section 3.2.3. After receiving a *Delay_Req*, the master responses with a *Delay_Resp* message to finish this process.

On the other hand, *Pdelay_Req* and *Pdelay_Resp* messages are used to compute

Bits								Octets
7	6	5	4	3	2	1	0	
transportSpecific				messageType				1
Reserved				versionPTP				1
messageLength								2
domainNumber								1
Reserved								1
Flags								2
correctionField								8
Reserved								4
sourcePortIdentity								10
sequenceID								2
controlField								1
logMessageInterval								1

Figure 3.2: PTP header format

the link delay with the peer delay mechanism. The former is sent by a delay requester clock to initialize the process and the latter is sent by a delay responder. If the responder clock cannot include the timing information required to compute the link delay in the *Pdelay_Resp*, also a *Pdelay_Resp_Follow_Up* is sent containing the timestamp. For a detailed explanation see Section 3.2.3.

Signalling messages are used to send additional information and to perform optional or implementation-specific functionalities that are defined in the standard or outside the scope of the standard. *Management* messages are used to transmit information from a clock to a management node and vice versa for system management operations.

3.2.2 PTP device types

Normally, time distributed systems consist of a combination of PTP and non-PTP devices. PTP devices are classified as ordinary clocks, boundary clocks, transparent clocks and management nodes. Additionally, although they are not defined in IEEE 1588-2008 standard, manufacturers widely deploy hybrid clocks. They are a combination of a transparent clock and an ordinary clock. On the other hand, non-PTP devices might include bridges, routers and other network devices such as computers, printers, or field specific devices that are not IEEE 1588-aware.

Tabla 3.1: PTP message types

Message Class	Message Type	Mechanism
Event Messages	*Sync*	Delay request-response mechanism
	Delay_Req	
	Pdelay_Req	Peer delay mechanism
	Pdelay_Resp	
General Messages	*Announce*	Synchronization hierarchy establishment
	Follow_Up	Delay request-response mechanism
	Delay_Resp	
	Pdelay_Resp_Follow_Up	Peer delay mechanism
	Management	Initialization and configuration of nodes
	Signalling	Other

In addition, each PTP device may work as either one-step or two-step mode for the transfer of timestamps. One-step clocks generate egress timestamps on-the-fly and transport them directly over *Sync* and *Pdelay_Resp* event messages. Otherwise, two-step clocks include this timestamping information in the corresponding *Follow_Up* and the *Pdelay_Resp_Follow_Up* general messages. Since the utilization of two-step clocks requires more network bandwidth and data processing, in order to maximize the efficiency of the protocol, the use of one-step operation is recommended.

3.2.2.1 Ordinary clock

An Ordinary Clocks (OC) is a system end device that contains a PTP clock with a single physical network connection. It can adopt the role of PTP master or slave. Figure 3.3 depicts the architecture of an OC device. One of the logical interfaces is used to send and receive PTP event messages and the other one sends and receives PTP general messages, as defined in Table 3.1. The former uses the Timestamp Unit (TSU) to timestamp event messages based on the value of the local clock and, then, passes the messages to the protocol engine. The latter directly passes general messages to the protocol engine.

An OC only has a single copy of the protocol, i.e. the protocol engine. When it is in the slave state, it deals with the computation of the reference time from master, based on the received PTP timing messages and local timestamps generated by

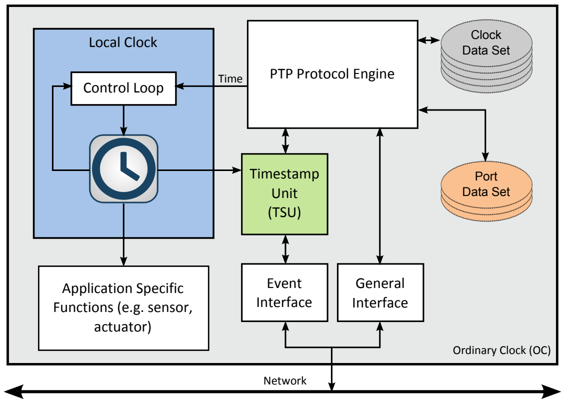


Figure 3.3: Model of an Ordinary Clock (OC)

the event interface. The local clock normally consist of a counter and a control loop, which adjusts the counter parameters until agreeing with the time of its master. When it is in the master state, the local clock acts as a free-running clock, although it might be possibly synchronized to an external reference clock, such as a GPS receiver.

In addition, the protocol engine of an OC maintains a clock data set and a port data set with the necessary attributes to run the protocol.

3.2.2.2 Boundary clock

A Boundary Clocks (BC) is a network device that contains specialized IEEE 1588 functionality and, unlike ordinary clocks, it has several physical ports. While OCs are usually used as end devices, BCs are essentially used as network elements without additional functionalities. They are used by the protocol to segment the network logically and to create the master-slave hierarchy in each region.

They also have event and general logical interfaces per physical port, as shown in Figure 3.4. There is a single local clock and a clock data set common to all of the ports. In addition, one protocol engine per port must resolve its state to be slave or master and, hence, it must determine which port provides the time signal used to synchronize the local clock.

In a BC, PTP received messages are not forwarded, except from *Management* messages that are not restricted to be propagated through different clock regions. That is, a BC synchronizes to the master and generates new PTP messages

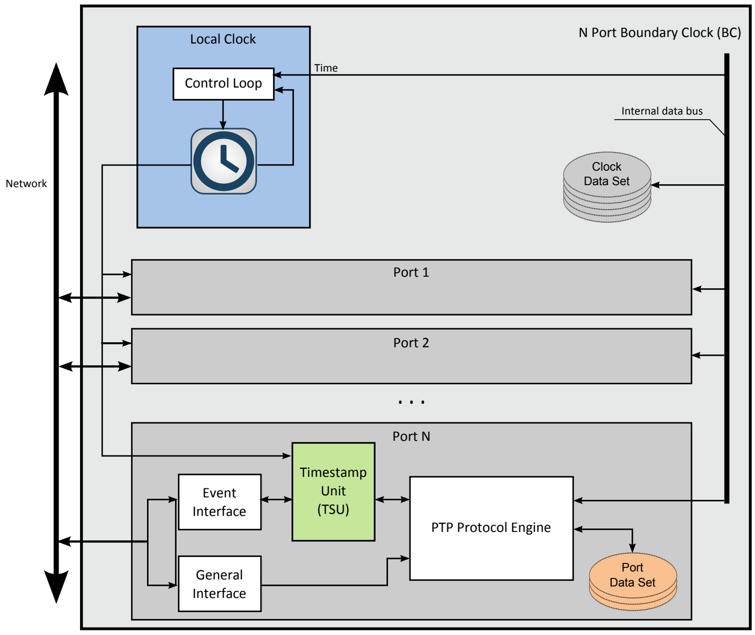


Figure 3.4: Model of a Boundary Clock (BC)

to share its own timing reference with the rest of slaves. As a consequence, time errors are accumulated in cascaded topologies where PTP messages passed through several BCs. All non-PTP messages are managed as if it was a normal network component, such as a bridge, a switch or a router.

3.2.2.3 Transparent clock

Similarly to BCs, Transparent Clocks (TCs) are embedded in network elements but, on the contrary, they forward PTP and non-PTP messages. The most significant difference is that TCs measure the time the message takes to traverse the switch or router, called the residence time, and accumulates it in a special PTP field of event messages, named *correctionField*.

The value of the residence time is computed as the difference between the ingress and egress timestamps. This computation is usually based on the difference between the timestamps generated when the message enters and leaves the node. A TC device might have two or more ports and, therefore, the paths of messages

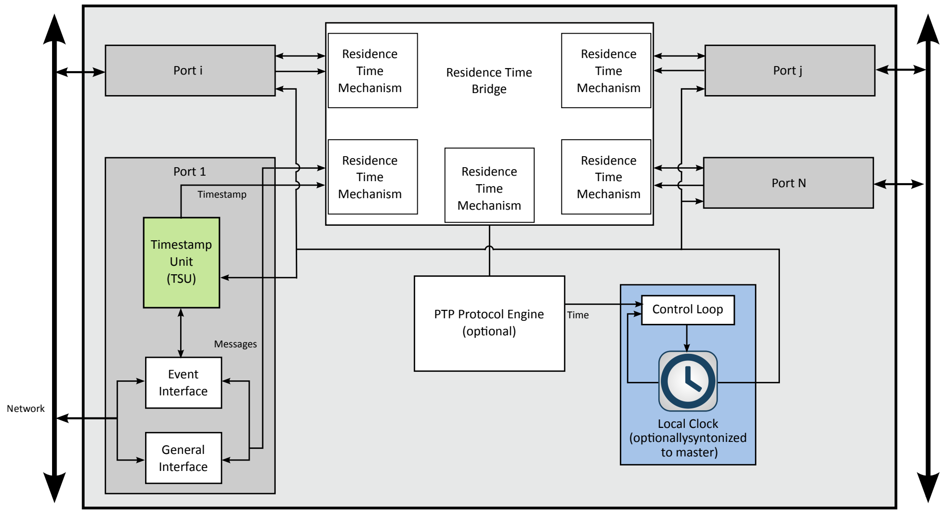


Figure 3.5: Model of Transparent Clock (TC)

crossing it might differ depending on the source and destination addresses. Consequently, associated residence times will vary due to queueing delays and traffic prioritization. Since the payload content will change as message crosses a TC, all network protocol affected checksums must be updated before leaving the device.

Figure 3.5 depicts a TC block diagram. The ingress and egress timestamps are generated from the local clock, which may be a free-running[1] or a synchronized clock. The implementation of the residence time bridge function in the TC is not specified in the standard, but it needs to add a kind of logic interface to each port in order to get the captured timestamps and compute the time difference.

There are two types of TCs: End-to-End Transparent Clocks (E2E TCs) and Peer-to-Peer Transparent Clocks (P2P TCs). E2E TCs measure the residence time and accumulate it in the *correctionField* of event messages, in order to allow the slave to compute the propagation delay by using the delay request-response mechanism, as explained in Section 3.2.3.3. Otherwise, P2P TCs use the peer delay mechanism to measure the link delay, as described in Section 3.2.3.4. They update the *correctionField* with both the residence time and the

[1]Due to the fact that the rates of the master and the TC are likely to be different, the introduced errors in the correction factor must be considered when using free-running clocks. For instance, with a typically difference of 0.02% in clock rates, the maximum error for a residence time of 1 milliseconds will be 200 nanoseconds, which may be or not acceptable for some applications.

link delay associated with the ingress transmission path of *Sync* messages. Since there is no point in measuring the propagation delay after changing the *Sync* path, P2P TCs allow a faster reconfiguration after network topology changes. How PTP messages traverse E2E and P2P TCs is shown in Figures 3.6(a) and 3.6(b) [45].

The peer delay mechanism is not compatible with the delay request-response mechanism and, as a consequence, P2P TCs only can work with ports supporting the peer delay mechanism as link peers. In order to connect a region using E2E TCs to another one with P2P TCs, a BC might be used as the interconnection element.

3.2.2.4 Management nodes

Management nodes might be combined with any of the mentioned clock types. They have one or more physical network connections and serve as an interface to PTP management messages.

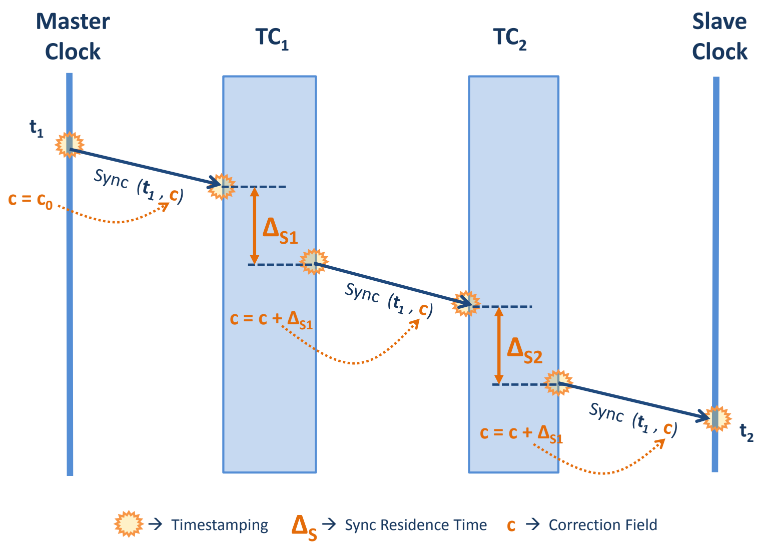
3.2.3 PTP operation

The protocol consists of four operational processes:

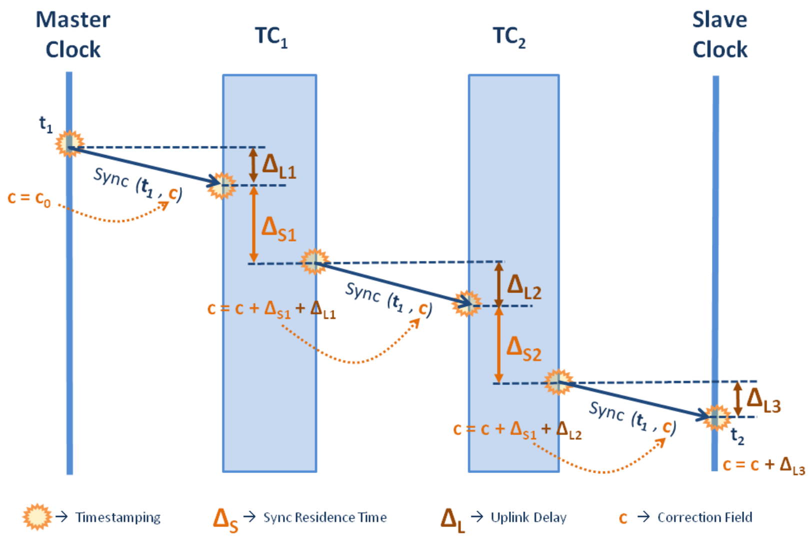
1. **Establishment of system boundaries**, through which domains with the same time reference are established. The standard defines a domain as an area in which the protocol is executed and, as a consequence, different independent synchronization systems can be maintained within a physical network. In order to limit the extent of a domain, in Ethernet networks for example, the propagation of multicast messages is typically managed by configuring switches and routers.

2. **Master-slave hierarchy establishment**, through which all the PTP nodes are logically configured into a tree structure where the root is the grandmaster clock, OCs are the leaves of the tree and BCs or TCs create the branch points. The Best Master Clock (BMC) algorithm allows the selection of the grandmaster based on the clock quality, accuracy and stability. For a detailed description of the BMC algorithm, the reader is encouraged to read the standard [28].

3. **Synchronization and syntonization**, which are two independent process executed by OCs and BCs to synchronize their local clocks in phase and frequency respectively. Since they are the most important procedures in PTP protocol operation they are thoroughly explained below.



(a) *correctionField* updating in E2E TCs



(b) *correctionField* updating in P2P TCs

Figure 3.6: *Sync* message traversing two TCs

4. **System and clock management**, which is done via *Management* messages, as it has been mentioned in Section 3.2.1. Although the default management mechanism is described in the standard, another alternative might be used if necessary.

Synchronization and syntonization processes involve:

- The computation of the clock offset from master in the slave, called *offsetFromMaster*.
- The computation of the fractional frequency offset from master in the slave, called *frequencyDrift*.
- The measurement of propagation delay, called *meanPathDelay*, using the delay request-response mechanism in the slave or the peer delay mechanism in every P2P port.
- The correction of the *meanPathDelay* by adding the residence time to the *correctionField* value of *Sync* and *Follow_Up* messages in any TC.
- The correction of the *meanPathDelay* for path asymmetries.

3.2.3.1 Clock offset computation

The computation of clock offset is done in OCs and BCs and its value will depend on the *twoStepFlag* bit of the *flagField* of the *Sync* message. If this bit is set to FALSE, one-step mode was selected and a *Follow_Up* will not be received. On the contrary, if it is set to TRUE, the peer clock is using two-step mode, therefore a *Follow_Up* message will be received.

The clock offset is computed based on Equation 3.1 following the rules shown in Table 3.2 for one-step and two-step operation mode.

$$offsetFromMaster = syncIngressTimestamp - correctedMasterTimestamp \tag{3.1}$$

Tabla 3.2: Correction of master timestamps

Mode	***correctedMasterTimestamp***
One-Step	*originTimestamp* + *meanPathDelay* + *correctionField* of *Sync* message
Two-Step	*preciseOriginTimestamp* + *meanPathDelay* + *correctionField* of *Sync* message + *correctionField* of *Follow_Up* message

Continuing with the clock offset computation process, the terms used in Equation 3.1 and Table 3.2 are explained below:

- The *syncIngressTimestamp* is the value of the timestamp generated by the slave upon the receipt of the *Sync* message.
- The *originTimestamp* is the value of the *originTimestamp* field of *Sync* message.
- The *preciseOriginTimestamp* is the value of the *preciseOriginTimestamp* field of the *Follow_Up* message.
- The *meanPathDelay* shall be computed accordingly to the propagation delay measurement mechanism.

3.2.3.2 Drift computation

The frequency drift, or fractional frequency offset, is computed using the Equation 3.2 and it is used by the slave to adjust the frequency of its local clock. It is estimated as the ratio of the elapsed time of the slave to the elapsed time of the master between two *Sync* messages, which are N *syncInterval* apart from each other ($N > 0$).

$$frequencyDrift = \frac{syncIngressTimestamp_{2N} - syncIngressTimestamp_{21}}{correctedMasterTimestamp_{1N} - correctedMasterTimestamp_{11}} \tag{3.2}$$

Figure 3.7 and Equation 3.3 represent an example of frequency drift computation. t'_{11} and t'_{1N} are master timestamps t_{11} and t_{1N} but corrected by the propagation delay, as seen in Table 3.2.

$$frequencyDrift_{FIG} = \frac{t_{2N} - t_{21}}{t'_{1N} - t'_{11}} \tag{3.3}$$

3.2.3.3 Delay request-response mechanism

Timestamps contained in PTP messages must be corrected in slaves by the propagation delay, which can be measured using one of the two mechanisms defined in the standard: the delay request-response or the peer delay mechanism. The first one uses the messages *Sync*, *Delay_Req*, *Delay_Resp* and possibly *Follow_Up* as represented in Figure 3.8.

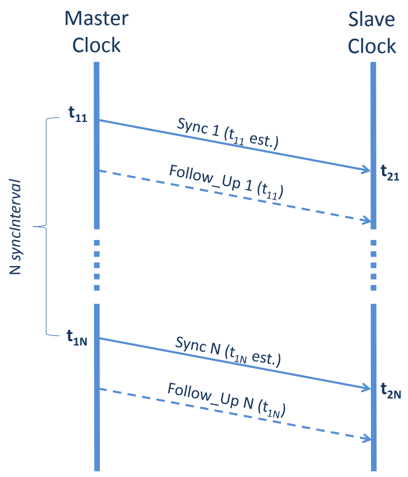


Figure 3.7: IEEE 1588 frequency drift correction

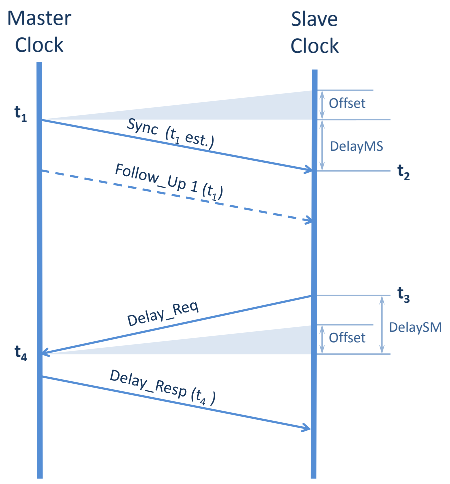


Figure 3.8: IEEE 1588 E2E propagation delay correction

The slave needs four measurement values t_1, t_2, t_3 and t_4 to compute the propagation delay, which are the ingress and egress timestamps of *Sync* and *Delay_Req* messages as shown in Figure 3.8. *Follow_Up* and *Delay_Resp* messages transport the time values captured by the master down to the slave. The propagation delay from master to slave and from slave to master, named $Delay_{MS}$ and $Delay_{SM}$ respectively, are:

$$Delay_{MS} + Offset = t_2 - t_1 \tag{3.4}$$

$$Delay_{SM} - Offset = t_4 - t_3 \tag{3.5}$$

From Equation 3.5:

$$Offset = Delay_{SM} - (t_4 - t_3) \tag{3.6}$$

and substituting from 3.4 above:

$$Delay_{MS} + Delay_{SM} = (t_2 - t_1) + (t_4 - t_3) \tag{3.7}$$

Assuming that the message transit delay is the same for both directions, the simplified equation for the propagation delay can be obtained:

$$Delay_{MS} = Delay_{SM} = \frac{(t_2 - t_1) + (t_4 - t_3)}{2} \tag{3.8}$$

Actually, in order to compensate latencies in network nodes, the value of t_1 and t_4 must be corrected with the values of the *correctionField* in the *Sync*, *Follow_Up* and *Delay_Resp* messages. Therefore, in the standard, the value of the *meanPathDelay* is computed as follows:

$$meanPathDelay = \frac{(t_2 - t_1') + (t_4' - t_3)}{2} = \frac{(t_2 - t_3) + (t_4' - t_1')}{2} \tag{3.9}$$

where t_1' and t_4' are the corrected timestamps, as defined in Table 3.3.

Tabla 3.3: Correction of master timestamps in delay request-response mechanism

Mode	***t₁'***
One-Step	*originTimestamp* of *Sync* message + *correctionField* of *Sync* message
Two-Step	*preciseOriginTimestamp* of *Follow_Up* message + *correctionField* of *Sync* message + *correctionField* of *Follow_Up* message

Mode	***t₄'***
One-Step	*receiveTimestamp* of *Delay_Resp* message + *correctionField* of *Delay_Resp* message
Two-Step	*receiveTimestamp* of *Delay_Resp* message + *correctionField* of *Delay_Resp* message

3.2.3.4 Peer delay mechanism

While the delay request-response mechanism measures the *meanPathDelay* between a pair of PTP ports using the *Sync*, *Delay_Req*, *Delay_Resp* and possible *Follow_Up* messages, the peer delay mechanism measures the link delay between two peer ports[2] that support this mechanism using the *Pdelay_Req*, *Pdelay_Resp* and possibly *Pdelay_Resp_Follow_Up* messages. This approach is represented in Figure 3.9.

Similarly to Equation 3.9, the *meanPathDelay* between Ports A and B is computed as follows:

$$meanPathDelay = \frac{(t_2 - t_1) + (t_4 - t_3)}{2} = \frac{(t_4 - t_1) - (t_3 - t_2)}{2} \tag{3.10}$$

The elapsed time $t_3 - t_2$ is directly transported from Port-B to Port-A in the *correctionField* of the *Pdelay_Resp* message when Port-B is a one-step clock. For two-step clocks two possibilities are available. On the one hand, the *correctionField* of the *Pdelay_Resp_Follow_Up* message can be used to transport the difference $t_3 - t_2$. On the other hand, both the *Pdelay_Resp* and the *Pdelay_Resp_Follow_Up* messages can be used to transport t_2 and t_3 respectively.

When measuring link delay by the peer delay mechanism, timestamps are not corrected by the propagation delay but asymmetries should be considered if known. For a detailed understanding, the reader is encouraged to read the standard.

[2]Peer ports are communicating ports of two adjacent nodes that are directly connected in the network, e.g. the peer port of a master port can be the port of the adjacent P2P TC.

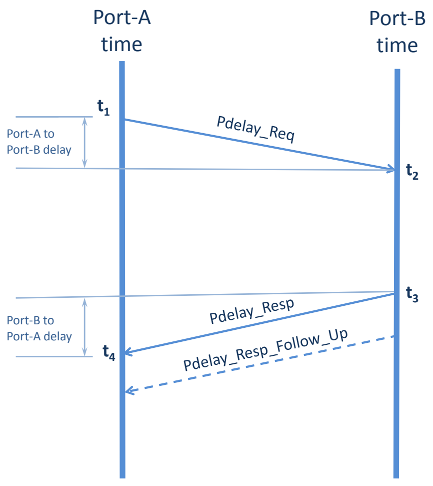


Figure 3.9: IEEE 1588 P2P propagation delay correction

3.2.3.5 Residence time computation

The residence time is the time that an event message takes to traverse a TC, which is calculated as shown in Equation 3.11:

$$residenceTime = egressTimestamp - ingressTimestamp, \tag{3.11}$$

ingressTimestamp and *egressTimestamp* terms in the above Equation are respectively the times of reception and transmission of an event message on the ingress and egress port of the TC.

The way the residence time is transferred from the TC to the slaves depends on the type of message and on the type of TC. As a summary, in Table 3.4, the main concepts involved in residence time computation have been reflected. E2E TCs has to measure the residence time for all event messages listed in this Table, whereas P2P TC only has to modify *Sync* messages. In addition, *Pdelay_Req* and *Pdelay_Resp* messages terminate at P2P TCs.

Tabla 3.4: A summary of residence time transferring options

Message Type	TC type	Mode	Residence time transfer method
Sync	E2E/P2P	One-Step	Added to the *correctionField* of the transmitted *Sync* message
		Two-Step	Added to the *correctionField* of the transmitted *Follow_Up* message
Delay_Req	E2E	One-Step	Added to the *correctionField* of the transmitted *Delay_Req* message
		Two-Step	Added to the *correctionField* of the transmitted *Delay_Resp* message
Pdelay_Req	E2E	One-Step	Added to the *correctionField* of the transmitted *Pdelay_Req* message
		Two-Step	Added to the *correctionField* of the transmitted *Pdelay_Resp_Follow_Up* message
Pdelay_Resp	E2E	One-Step	Added to the *correctionField* of the transmitted *Pdelay_Resp* message
		Two-Step	Added to the *correctionField* of the transmitted *Pdelay_Resp_Follow_Up* message

3.2.3.6 Asymmetry correction

Although the protocol cannot detect asymmetries in the propagation paths, they are sometimes known. For instance, the asymmetry introduced by physical layer due to different delays in Ethernet transceivers might be determined from basic experiments. If they are known, a PTP node must apply corrections for asymmetries. The way to manage *Sync*, *Delay_Req*, *Pdelay_Req* and *Pdelay_Resp* messages when correcting the path asymmetry for the path connected to the corresponding ingress or egress port of a device is specified in the standard. For instance, when managing the path asymmetry correction of a *Sync* message by an ordinary clock, the asymmetry correction must be done for incoming *Sync* messages through any ingress port, but not for the path connected to the egress port.

3.2.4 Cyber-security protection of PTP messages

Annex K in [28] defines an optional security extension that provides group source authentication, message integrity and replay attack protection for PTP messages using symmetric MAC functions. This extension is composed by two basic mechanisms:

- An integrity protection mechanism, which uses the MAC functions to verify that a received message was sent by an authenticated source, it was not modified in transit, and it is fresh (i.e. not a message replay).
- A challenge-response mechanism, which is a three way mutual authentication process used to affirm the authenticity of new sources and to maintain the freshness of the trust relations.

The Hash-based Message Authentication Code (HMAC) functions supported by

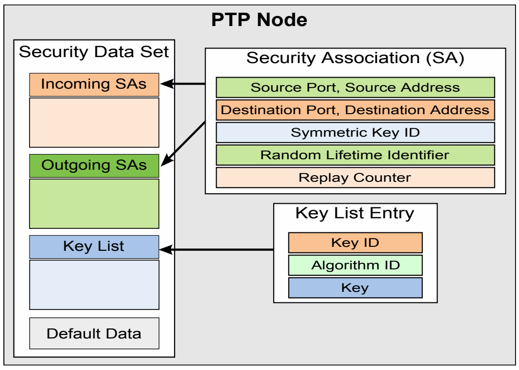


Figure 3.10: Annex K security data set

the protocol are HMAC-SHA1-96 and HMAC-SHA256-128, but the addition of other MACs in the future is allowed. The implementation of these algorithms must be in accordance with several RFC documents from the Internet Engineering Task Force (IETF) community and with NIST standards: RFC 2104 [46], RFC 2404 [47], NIST Secure Hash Standard [48] and NIST HMAC Standard [49].

Master and slaves share secret symmetric keys, which could be shared by the whole domain or by subsets of the domain. The generation of secret keys and the key distribution, which may be done either manually or by an automatic key management protocol, are out of scope of the standard.

3.2.4.1 Security data set

PTP nodes maintain Security Associations (SA) to perform the aforementioned mechanisms, which will be explained in next Sections. These associations are established using the challenge-response mechanism between two nodes to confirm their identity. In addition, they are unidirectional and, consequently, they can be used to protect traffic from the source to the destination, but no viceversa.

Each node maintains a table of incoming SAs and a table of outgoing SAs, which are respectively used for verification and protection of incoming and outgoing traffic. The entire list of incoming and outgoing SAs together with a list of keys and a list of default data set parameters, composes the security data set, as it has been represented in Figure 3.10.

SAs can be static and configured in advanced. They can also be created dynamically on-the-fly once the clock receives a PTP message from a port that does not

Tabla 3.5: Annex K ICV and pad length

Algorithm	ICV length (octets)	Pad length (octets)
NULL	0	16
HMAC-SHA1-96	12	4
HMAC-SHA256-128	16	0

match with any entry of the incoming SAs table. There are two possibilities for creating dynamic SAs by the sender:

1. A single SA for each source and all destinations. The sender creates an SA with its source address being one of its interface unicast addresses and the destination being "all".
2. An SA per source and destination. That is, the sender creates multiple SAs with the same source address and different multicast and unicast destinations.

3.2.4.2 Secure transmit and receive processing

To use the security extension, the security flag bit in the *flagField* of the PTP message header must be set to TRUE in all secured transmitted PTP messages, so as to indicate that the message carries the security AUTHENTICATION TLV field. This TLV includes security information of the associated outgoing SA and the Integrity Check Value (ICV), as shown in Figure 3.11(a). The ICV is the result of computing the MAC algorithm over all PTP message fields beginning with the first octet of the common header and ending with the last octet of the security AUTHENTICATION TLV.

In order to facilitate hardware implementations of the security protocol, the AUTHENTICATION TLV is the final TLV field appended to the PTP message. The last 16 octets are distributed between the padding (all zeros) and the ICV field depending on the selected algorithm as stated in Table 3.5.

When a PTP message is received, the ICV in the message is checked. This checksum must match with the result of applying the MAC function specified by the algorithm identifier, which is retrieved from the key list data set indexed by the key identifier. If the key identifier in the message is neither valid nor known, or the algorithm identifier in the message is not equal to the one associated in the incoming SA, the ICV check fails. If the computed ICV does not match the ICV carried in the message the ICV test also fails. Therefore, if any of the previously

Bits								Octets
7	6	5	4	3	2	1	0	
tlvType = AUTHENTICATION								2
lengthField								2
lifetimeId								2
replayCounter								4
keyId								2
algorithmId								1
reserved								1
pad								M
ICV								N

(a) AUTHENTICATION TLV

Bits								Octets
7	6	5	4	3	2	1	0	
tlvType = AUTHENTICATION_CHALLENGE								2
lengthField								2
challengeType								1
reserved								1
requestNonce								4
responseNonce								4

(b) AUTHENTICATION_CHALLENGE TLV

Bits								Octets
7	6	5	4	3	2	1	0	
tlvType = SECURITY_ASSOCIATION_UPDATE								2
lengthField								2
addressType								1
reserved								1
nextKeyId								2
nextLifetimeId								2

(c) SECURITY_ASSOCIATION_UPDATE TLV

Figure 3.11: Annex K security TLVs

mentioned checks fails the packet should be silently discarded, i.e. the message is discarded without further processing and without internal or external resource allocation, so as to limit the ability of a DoS attack.

After checking the ICV, the received PTP message is matched against the corresponding incoming SA in the SA lookup process, to determine whether they are received from a trusted source. The source port identifier of the PTP header and the source protocol address must match the source port and the source address of the incoming SA. Then, the key identifier value in the AUTHENTICATION TLV must be checked. If it is equal to the one stored in the incoming SA the test passes; if it does not match, the message is silently discarded. After that, the replay counter is checked following the replay protection mechanism. The incoming lifetime identifier is compared with that of the incoming SA. If they are identical, then the replay counter is compared. On the contrary, if it is smaller or equal to the stored one, the message must be silently discarded[3].

3.2.4.3 The challenge-response exchange

The challenge-response exchange is a three-way mutual authentication protocol, in which two nodes are able to affirm their authenticity and freshness. It is initialized when a clock receives a message from an untrusted source.

During the challenge-response exchange the AUTHENTICATION-CHALLENGE TLV is used. In Figure 3.11(b) the format of this TLV is represented. Whereas AUTHENTICATION TLV described above is appended at the end of all PTP secure messages, the AUTHENTICATION-CHALLENGE TLV must be appended as the first TLV of the secured signalling message. The *challengeType* field defines the challenge type as *challengeRequest*, *challengeResponseRequest* or *challengeResponse*. Thus, the sender of the *challengeRequest* and the *challengeResponseRequest* sets the request nonce to a random number. After that, the receiver of the *challengeResponse* and the *challengeResponseRequest* matches the request nonce it sent before in the request to the reply. If it does not match the challenge-response check fails and the challenge message is silently discarded.

[3]The replay protection mechanism relies on the fact that the probability for the *sourcePortIdentity*, *lifetimeId* and *replayCounter* triplet appearing twice is extremely low and depends on the shared keys lifetime. There is a trade-off between this probability and the key exchange frequency, i.e. the smaller the shared keys lifetime, the higher the key exchange frequency.

3.2.4.4 The security association update exchange

A challenge-response exchange is also initiated to update SA parameters, such as the lifetime and the key identifiers, of the outgoing SA using the SECURITY-ASSOCIATION-UPDATE shown in Figure 3.11(c), which is sent in challengeResponse and challengeResponseRequest signalling messages. By specifying the address type the TLV can be used to deliver the SA update relevant to all addresses or, on the contrary, deliver SA update information for a particular address type if different outgoing SAs are maintained by the challenge-response.

3.3 The future of PTP security

PTP presents some security vulnerabilities and threats that might lead PTP systems to different attacks ranging from DoS to selective packet delay attacks passing through clock manipulation by inserting, removing or modifying PTP packets [31–33]. PTP firstly introduced security as an optional extension in Annex K of the second version of the standard, but it presented some limitations that have been studied during last years [50–52]. In this Section, firstly, the most significant security flaws of Annex K are collected. After that, a summary of the contributions regarding PTP security of related standardization committees is presented.

3.3.1 PTP security extension limitations

The security extension for PTP, as defined in Annex K of the second version of the standard, was never implemented in real networks due to several security vulnerabilities identified in the literature. An attacker could directly manipulate nodes and maliciously modify protocol stacks or install some malware in order to take advantage of these weaknesses. Some of the most significant threats are summarized below:

Unsecure and congested startup phase

In the startup phase, a manipulated PTP node can take advantage of the lack of verification of announced accuracies and, hence, it might become a byzantine master[4] and gain system control by announcing a wrong time. In addition, since the first *Sync* messages from all non-authenticated nodes will not have associated an incoming SA, every node will trigger a challenge

[4]A byzantine master is an attacker that becomes PTP master, for example by announcing false clock accuracies in the startup phase, in order to establish a wrong reference time.

request-response procedure for every received message causing a storm of messages jamming the network, which might lead the system to a DoS attack [32].

Source address modification

Network protocol addresses used for the SA lookup process are not protected bacause the ICV value is computed over all PTP message fields excluding network or transport protocol addresses. Therefore, the source address can be changed by an attacker, who might cause DoS attack by overloading the SA table with new entries or produce replay packets as a man-in-the-middle attacker. The attacker could take advantage of this security flaw by creating a twin SA or resetting an existing one as authors affirmed in [51].

Sub-optimal ICV calculation

In the transmission path, the ICV must be computed after drawing the timestamp in order to allow one-step mode of operation. Hence, on-the-fly calculation of the ICV should be performed close to physical layer, which requires MAC computation of one block to be finished before fetching the next block. Both HMAC-SHA1 and HMAC-SHA256 operate on blocks of 512 bits, which need 64 cycles at 1 Gbit/s or 512 nanoseconds to process an entire block. For SHA1 algorithm collision attacks were demonstrated, and therefore the utilization of SHA256 is recommended. However, hardware implementations of SHA256 need loop unrolling techniques to compute the hash in less than 64 cycles with the consequent area cost. In addition to this, HMAC architecture introduces an additional delay because the final hash value of the message has to be digitally signed [53]. Modern MAC hardware implementations based on the AES algorithm offer the same security level than HMAC-SHA256 with reduced latency and area [52].

Additional overhead

The additional overhead introduced by the security extension can be divided into network load and computational load. Network load is caused by the need for additional messages due to security management and additional AUTHENTICATION TLVs appended to each message. Thus, the overhead of the payload is around 60% of the acutal message volume [50]. The computational load relates to the computational resources required to implement security extension. In [54], several performance tests were executed over a practical implementation of a software security layer to analyse CPU load under both normal operation and when a DoS attack was simulated. These tests showed that the CPU load considerably increases with the number of PTP nodes, specially in the startup phase.

Security for switches

In case of using TCs, the highest difficulty is to manage and distribute the high number of keys for different SAs running on a switch [50]. Supposing that multicast addressing is used to distribute timing information, keys for different groups must be additionally distributed to all TCs to recalculate the ICV after modifying the *correctionField*. Authors in [50] also identified the need for implementing a single security unit for each sending path in a TC, so as to minimize the jitter under load conditions. In addition, long execution times of SHA algorithm must be taken into account for the calculation of the residence time.

Unnecessary challenge-response exchange

The three-way authentication scheme generates additional traffic, consumes resources and interrupts the useful traffic for a while [52]. As synchronization frames are broadcast, the challenge-response mechanism with all destinations increases the message traffic. Consequently, the grandmaster suffers from high CPU load. This mechanism could be replaced by a one-way authentication with the support of a key distribution scheme.

3.3.2 TICTOC working group contributions

The Timing over IP Connection and Transfer of Clock (TICTOC) working group belongs to the IETF community and is concerned with highly accurate time and frequency distribution over packet switched networks. After some years of analysis for protecting synchronization messages over Internet, the TICTOC group moved forward with the definition of security requirements for Network Time Protocol (NTP) and PTP networks in a succession of Internet Drafts that had an informational purpose. There are already twelve versions of the draft, with the last one published in September 2014. Finally, in October 2014, the work of the group resulted in the publication of the RFC 7384 [55]. There, the possible threats and the required security services for packet based synchronization networks are defined and security requirements for time synchronization protocols are specified.

Tables 3.6 and 3.7 list types of attacks and security requirements respectively. As it can be seen in Table 3.6, attacks are classified according to two criteria: the accessibility of the attacker to the security information and the localization of the attacker within the network. On the one hand, an attacker can be internal or external. An internal attacker has access to a trusted segment of the network or to an intermediate node that possess the security keys. Consequently, it can tamper information as well as generate traffic maliciously. Otherwise, external

Tabla 3.6: Summary of PTP cyber-attacks

Types of attacks	Impact			Attacker Type			
				Internal		External	
	False Time	Accuracy Degrad.	DoS	MITM	Injector	MITM	Injector
Manipulation	X			X			
Spoofing	X			X	X		
Replay attack	X			X	X		
Rogue master (rogue TC/rogue BC)	X			X	X		
Interception and removal		X	X	X		X	
Packet delay manipulation	X			X		X	
Layer 2/layer 3 DoS attacks			X	X	X	X	X
Cryptographic performance attacks			X	X	X	X	X
Time protocol DoS attacks			X	X	X		
Master time source attack (e.g. GPS spoofing	X			X	X	X	X

attackers do not have access to security keys.

On the other hand, an attacker can be Man-In-The-Middle (MITM) or packet injector. The former is located somewhere in the network that allows intercepting and modifying packets, like an attacker that has gained control of one of the nodes. The latter generates new traffic, even from an external network that is connected to the network under risk. An injector could also perform replay attacks.

Attackers may exploit different threats to align slaves to a false time or frequency value, to induce accuracy degradation[5] or to cause DoS. For example, by packet interception and manipulation, an internal MITM attacker can modify the *originTimestamp* and *correctionField* of PTP *Sync* packets causing the receiving nodes to be aligned to a false time. Other important threats are spoofing and replay attacks. An internal MITM or Injector attacker can spoof a clock and send messages that appear to receivers to be generated by the legitimate master, or even record PTP messages and replay them later.

A rogue master could be the result of manipulating the best master clock selection, which is based on the quality of the clocks that is advertised in the startup phase through *Announce* messages. If an attacker generates malicious *Announce* messages with false clock quality or Media Access Control (MAC) address[6], it

[5]All attacks with a false time alignment effect implicitly causes accuracy degradation, but in the table they represent two levels of severity

[6]When multiple clocks have the same clock quality the selection is done based on the MAC addresses.

could become an illegitimate master clock.

The rest of attacks can be performed by both internal and external attackers in common PTP networks. For a detailed description of each attack the reader is encouraged to look up the document [55].

Tabla 3.7: Summary of PTP cyber-security requirements

Security Requirement	**Level**	**Corresponding Threat**
Authentication & authorization of sender	MUST	Spoofing attack and rogue master
Authentication & authorization of master	MUST	
Recursive authentication & authorization	MUST	
Authentication & authorization of slaves	MAY	DoS against the master
Authentication & authorization of P2P TCs by master	MAY	
Authentication & authorization of *Announce* messages	MUST	Spoofing attack and rogue master
Authentication & Authorization of *Management* messages	MUST	
Authentication & authorization of *Signalling* messages	MAY	
Integrity protection	MUST	Packet manipulation
Master/slave/P2P TC spoofing protection	MUST	Spoofing attacks
Availability	SHOULD	DoS attacks against the time protocol
Replay protection	MUST	Replay attacks
Key freshness	MUST	
Security association	SHOULD	-
Unicast and multicast associations	SHOULD	-
Performance: no degradation in quality of time transfer	MUST	-
Performance: computation load, storage and bandwidth	SHOULD	-
Confidentiality protection	MAY	MITM attacks
Protection against delay and interception attacks	MUST	Packet delay attack
Secure mode	MUST	-
Hybrid mode	SHOULD	-

In order to represent the impact level and the difficulty of performing the corresponding attacks in case the requirement in question were not implemented, different requirement levels are defined in Table 3.7. MUST means the definition of an absolute requirement, whereas SHOULD and MAY terms relate to recommendations and optional features that should be considered. When securing clock synchronization networks in general, confidentiality and non-repudiation are not usually considered as crucial security services in today's industrial environments. In particular, time information is public in the network. In this way, confidentiality is not required and encryption can be avoided. Therefore, the work on cyber-security issues in PTP systems has been focused on integrity, authentication and availability.

Regarding clock identity authentication and authorization, which are implemented to prevent spoofing and rogue master attacks, the requirement level varies depending on the device type. Authentication refers to verify that a node is who it claims to be, whereas authorization refers to verify that a node is permitted to play the role it plays. For example, only nodes that are authorized to be masters could announce their clock capabilities to participate in the best master clock selection. While slaves must authenticate and authorize masters, it is not absolutely necessary to authenticate slaves by masters. Similarly, masters are not forced to authenticate P2P TCs that are directly connected to it.

In addition, the recursive authentication and authorization of masters must be supported when intermediate clocks participate in the protocol. In those cases, the PTP slave authenticates the intermediate clock as well as the master, and the intermediate clock authenticates another attached TC through which time information is forwarded if necessary. It is important to note that *Announce* messages in the startup phase and *Management* messages must also be protected by authentication mechanisms.

The integrity protection prevents from packet manipulation by ensuring that the packet has reached the destination intact. In this sense, there is a discussion regarding hop-by-hop versus end-to-end integrity protection. The hop-by-hop integrity protection acquires importance in PTP networks because intermediate nodes modify the content of PTP packets and, as a consequence, the integrity protection tags must be verified and re-generated on each hop. The drawback of such a solution is that it allows rogue TCs to modify protocol packets. In contrast, end-to-end integrity protection is maintained from source to destination, limiting the ability of rogue TCs to maliciously modify the packet, but it needs a separate integrity protection mechanism to protect the *correctionField.*

The security solution must additionally include a replay protection mechanism to prevent replay attacks, which typically consists on an increasing packet sequence number that is commonly included in the authentication mechanism. Replay attacks are also avoided by guaranteeing the freshness of cryptographic keys by replacing current keys by new ones frequently[7]. Moreover, it is recommended the utilization of a security association protocol that might establish unicast and multicast authentication relationships between two or more clocks to agree on session keys.

Another challenging requirement is the protection against packet delay and inter-

[7]The value of the frequency at which keys must be refreshed will depend on the system requirements.

ception attacks to prevent that a MITM attacker could degrade the time accuracy. A possible solution to packet delay protection is the utilization of redundant masters or redundant paths between the master and the slave, and ignore time values that significantly differ from the others [56].

Finally, time transfer quality must not suffer degradation due to security measures. Also, the performance of the protocol should not be compromised in terms of computation load, storage or bandwidth overhead. In order to get detailed information on each security requirement, the reader is encouraged to read [55].

3.3.3 P1588 security subcommittee contributions

With the aim of facing all the security challenges described before, cyber-security solutions have to be implemented in PTP systems. When employing authentication and integrity protection mechanisms, the impact of external attackers is drastically reduced. If it is combined with sophisticated key management and security association establishment protocols, the robustness to internal attacks is strengthened.

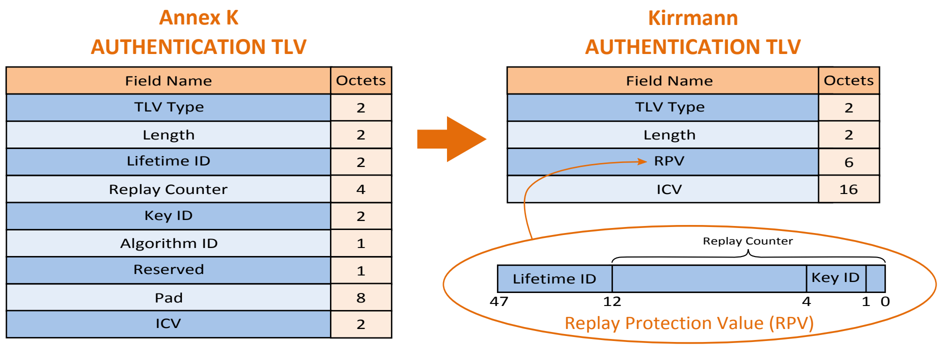
Now the P1588 Working Group is working on the third edition of the standard. Concretely, the P1588 Security SC is immerse in specifying the most suitable PTP security solution. There are five security proposals that were discussed by members of the subcommittee. In this Section, these approaches, identified by the names of their forefathers, are presented and discussed.

3.3.3.1 First proposal: Kirrmann - modified annex K

Hubert Kirrmann's proposal is the result of the work done in [52], where the PTP security extension was thoroughly analysed. The computational and bandwidth overhead caused by the challenge response mechanism, the lack of key management and distribution and the suboptimal MAC computation using HMAC were the main drawbacks of Annex K solution identified by authors.

The aim of this work was to demonstrate the necessity of improving current security extension and to propose the following modifications:

1. Replacement of the three way mutual authentication with one way authentication by using a modified AUTHENTICATION TLV with replay attack protection. The signalling messages involved in the challenge-response mechanism are removed and, consequently, the network overhead is reduced. Thus, when a node joins a network, it shall send periodical supervision frames, e.g. PRP supervision frames, which must include an ICV.



Field Name	Octets
TLV Type	2
Length	2
Lifetime ID	2
Replay Counter	4
Key ID	2
Algorithm ID	1
Reserved	1
Pad	8
ICV	2

Field Name	Octets
TLV Type	2
Length	2
RPV	6
ICV	16

Figure 3.12: Annex K vs Kirrmann's proposal AUTHENTICATION TLV

2. Removal of the security association updating mechanism. The freshness of authentication relationships are maintained by listening to these supervision frames and verifying a replay counter. If a message from a source is successfully verified for the first time, a new SA is created. If no more messages are received from that source in a given time, the SA is removed.

3. Utilization of modern MAC algorithms to compute the ICV. HMAC shall be replaced by the Cipher-based Message Authentication Code (CMAC) algorithm with AES block cipher, known as AES-CMAC, for on-the-fly ICV calculation and verification. Moreover, the new ICV shall protect protocol destination and source addresses in addition to PTP message payload.

4. Definition of a new AUTHENTICATION TLV format, as shown in Figure 3.12. The replay counter of the associated outgoing SA is copied into the TLV in every sent message and then increased by two for future message transmissions. Hence, the key ID is automatically changed for each transmitted message.

5. Distribution of a list of shared and symmetric secret keys to authenticated participants. IPsec shall be used to distribute this list, where participants authenticate themselves by certificates. They receive updated key lists often enough in order for the replay counter not to roll over. Instead of being associated to particular SAs, each key is used only once since the key ID in the TLV changes in every message.

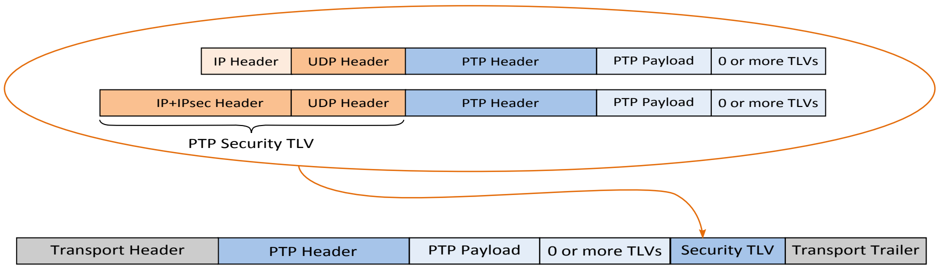


Figure 3.13: Mizrahi's proposal IPsec-based Security TLV

3.3.3.2 Second proposal: Mizrahi - IPsec-based security TLV

Tal Mizrahi began his work on PTP security in 2011 when he analysed the utilization of IPsec and MACsec as potential security solutions to protect PTP messages as well as the security threats and types of attacks [33]. After that he started to collaborate on the development of the security requirements draft document for time synchronization protocols within the TICTOC group. He was the main author of subsequent Internet Drafts prior to the publication of the RFC 7384 introduced in Section 3.3.2.

In April 2014, he proposed the definition of a new IPsec-based Security TLV [57] applicable to all transport protocols, so as to provide authentication and integrity protection in two-step mode of PTP operation. The Security TLV contains IP and UDP headers where IP addresses represent PTP clocks and can be unicast or multicast. These headers are generated as if the PTP packet were encapsulated using UDP over IP and protected with IPsec [58], as it can be seen in Figure 3.13 taken from [57]. IPsec header refers to fields containing the Encryption Security Payload (ESP) with null encryption based on the IP packet, which shall be used to provide authenticity and integrity protection.

In this case, the challenge is how to manage IP addresses contained in the Security TLV. On the one hand, using native addresses is simple but presents problems when traversing Network Address Translation (NAT) routers in private-public network schemes[8]. On the other hand, if specific IP addresses were allocated for PTP, the management would be too difficult. A viable solution could be generating clock identity-based IP version 6 addresses.

This proposal only covers the encapsulation issue but it does not specify neither

[8]With NAT technology, a private IP address space is remapped into another public IP address space, and vice versa, when packets traverse a NAT routing device in face of IPv4 address exhaustion.

a key management scheme nor cryptographic algorithms to be used. Mizrahi suggests the utilization of Internet Key Exchange (IKE) [59] encapsulation into the PTP TLV for key distribution.

One of the main drawbacks of an IPsec-based TLV as proposed by Mizrahi is that it only works in two-step mode of PTP operation. In addition, a problem rises with regard to the insertion of a layer 3 address into the PTP packet body, because it brakes OSI layer model.

The main advantage is that the Mizrahi solution is based on a very well-known and widely deployed security protocol. Therefore, it should not require the definition of a new security protocol, but rather reuses tools defined for IPsec. IPsec is thoroughly maintained by IETF and there is open source code to implement it in software. However, the off-the-self IPsec software cannot be used because an specific mapping has to be created.

3.3.3.3 Third proposal: Fries - new authentication TLV and group key management

Steffen Fries from Siemens is an important cornerstone in the development of cyber-security solutions for energy automation in Smart Grids. His work on P1588 Security SC consists on extrapolating the conclusions derived from protecting GOOSE and SV messages in SASs to PTP scenarios [60].

Security for SASs communications was initially covered in IEC 62351-6 standard, in which digital signatures were considered to provide source authenticity. However, as explained in Section 2.4, using asymmetric cryptography and digital signatures do not fulfil substation communications performance requirements. As a consequence, the IEC working group has moved forward to the utilization of symmetric group keys and GDOI to distribute them.

Two main tasks are covered in Fries' proposal [61]. On the one hand, the key management is handled out of band, i.e. using a separate distribution channel outside PTP messages, instead of the triple handshake of Annex K. In this case, the key management might be unicast or multicast and manual or automatic. For multicast and automatic key management, a group key distribution scheme like GDOI is suggested, but also other distribution mechanisms could be used [19].

On the other hand, a new Security TLV is appended to the PTP message to provide authentication and integrity protection. The content of this TLV will depend on chosen key management protocol. Figure 3.14 shows two example TLV formats [61]. Since the security TLV B contains a key identifier corresponding to

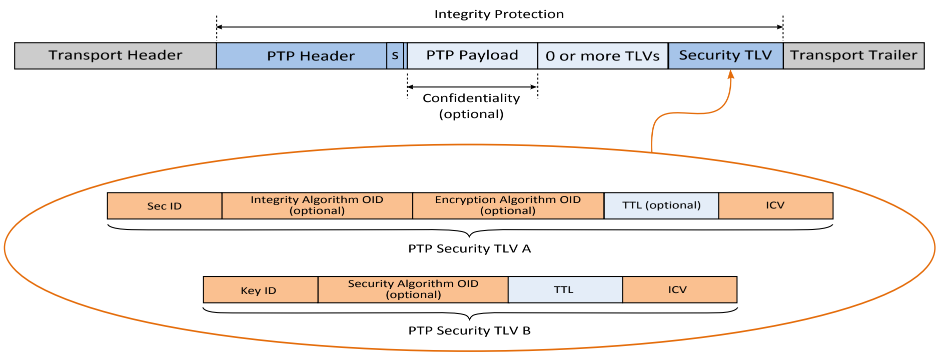


Figure 3.14: Fries' proposal new Security TLV

an available key on the receiver side, it cannot be used with TESLA. The security TLV A format may be usable with TESLA because the SAId field identifies the security association that is previously established by the key management protocol. Also the validity time of the key or time-to-live (TTL) and the cryptographic algorithm identifiers are negotiated by the key management protocol.

In this proposal, the ICV can be computed using cryptographic algorithms utilized by other applications that use PTP, such as GOOSE communications. PTP payload encryption could be optionally provided, but confidentiality is preferably achieved by external protocols like MACsec. For early recognition and ICV preprocessing, one security flag in the PTP header must indicate the utilization of security TLV. Once a message is received, the receiver looks for the corresponding SA based on the sourcePortId field of the PTP header and the sequenceId field for replay protection.

In addition, there is available open source code for GDOI implementation with small footprint [62]. The small processing overhead and the possibility of on-the-fly ICV computation makes this proposal very interesting for applications that demand cut-through operation to reduce frames switching latency. As Mizrahi's proposal, this approach is applicable to all transport protocols and only software or firmware upgrading would be required if two-step operation is selected.

3.3.3.4 Fourth proposal: Ellegaard - MACsec security

In the security group also MACsec, as defined in IEEE 802.1AE-2006 standard [63], has been contemplated as an appealing security approach to provide hop-by-hop user data confidentiality (optionally), frame data integrity, authenticity and

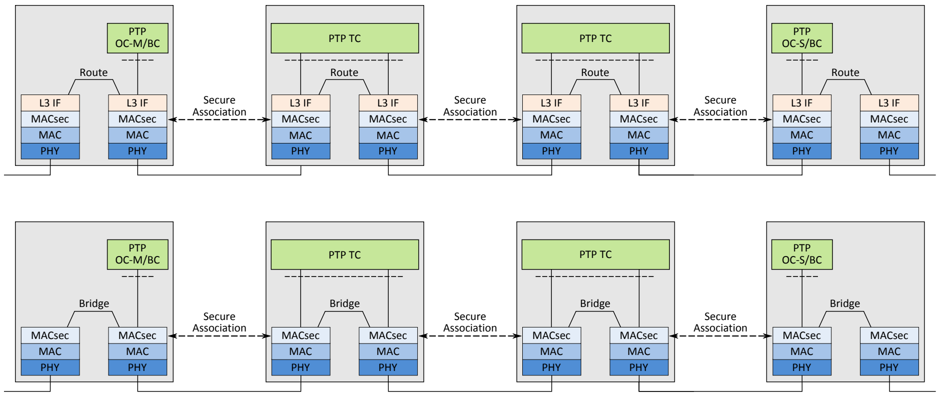


Figure 3.15: Ellegaard's proposal to secure PTP with MACsec

replay protection. The first member to suggest MACsec as an external security solution for protecting PTP traffic was Lars Ellegard from Vitesse Semiconductor Corporation, acquired by Microsemi Corporation in 2015. Ellegard proposed to include as an appendix in the standard a document on how MACsec can be used to protect PTP networks [64]. He affirmed that MACsec could be added to PTP packets, even in one-step operation if cryptographic algorithms were implemented in hardware, without compromising the clock accuracy. Nevertheless, considerable latencies introduced by cryptographic units on each hop must be taken into account.

One of the key points of MACsec is that it can be used in both point-to-point and shared LANs, and in both layer 2 bridged and layer 3 routed networks. However, Ellegaard's proposal was only focused on the point-to-point LAN case with pairwise connectivity associations, as Figure 3.15 shows for routed and bridged networks. Moreover, he does not contemplate neither key management nor authorization of clocks.

External security solutions like MACsec could also protect the startup phase, where the protection of announce messages could avoid attacks on BMC algorithm. However, this proposal provides link-based security due to the utilization of pairwise associations. Additional mechanisms to verify the clock identity are required to provide source authenticity and authorization. On the contrary, an attacker could 'hijack' a TC, for example, and act as a rogue master forcing the slaves to align to a false time.

3.3.3.5 Fifth proposal: Sibold/Dickerson - NTS and TESLA for source authentication

Dieter Sibold from the Physikalisch-Technische Bundesanstalt in Berlin, is currently engaged in the definition of a new security protocol for NTP, named Network Time Security (NTS) [65]. He begun his collaboration with P1588 Security SC to discuss the applicability of NTS for IEEE 1588 protocol [66].

NTS provides time servers authentication by clients and end-to-end integrity protection using X.509 certificates and HMAC algorithm respectively. There are two modes of operation: using pairwise keys for unicast mode, which are exchanged via asymmetric cryptography, or broadcast mode using TESLA [67] with a posteriori verification of buffered packets. In Figure 3.16 the time diagram of TESLA operation is represented, where the whole operation is divided in N time intervals. Firstly, the sender randomly generates the last key K_N of the chain. The rest of the symmetric keys are generated using a one-way hash function over the next key to be used in the next interval following the Equation 3.12:

$$K_0, K_1, K_2, ...K_N \quad / \quad K_{i-1} = H(K_i) \qquad 1 \leq i \leq N \tag{3.12}$$

As represented in Figure 3.16, during T_0, the sender releases K_0 protected with a digital signature. Then, it starts sending messages protected with MACs generated with K_1 during time interval T_1. During time interval T_2, MACs are generated with K_2, and K_1 is distributed in clear to group members and so on. Thus, once the receiver owns a single source authenticated key of the chain, i.e. K_0, subsequent keys of the chain are self-authenticating.

Since TESLA requires time synchronization, in the startup and convergence phases of PTP protocol, messages have to be protected by other means. NTS uses pairwise symmetric keys in this phase, until slave clock reaches sufficient accuracy. In the unicast mode of operation, a cookie is negotiated between the master and a slave and exchanged via asymmetric cryptography. This cookie is used as a key for computing the HMAC-based MAC in unicast time synchronization messages. Once the slave is well enough synchronized, NTS might be employed in a broadcast mode of operation. Then, in the bootstrapping phase, all TESLA parameters are transmitted form sender to destination nodes using asymmetric cryptography: time intervals, one-way function, disclosure delay and the initial key. Then, broadcast transmitted messages are protected by symmetric keys as TESLA mandates. Regarding subsequent key distribution for using TESLA schemes in PTP systems, two alternatives were identified in the group: using GDOI, out-of-band as proposed by Fries in Section 3.3.3.3, or carrying key information inside an specific security TLV, within PTP protocol.

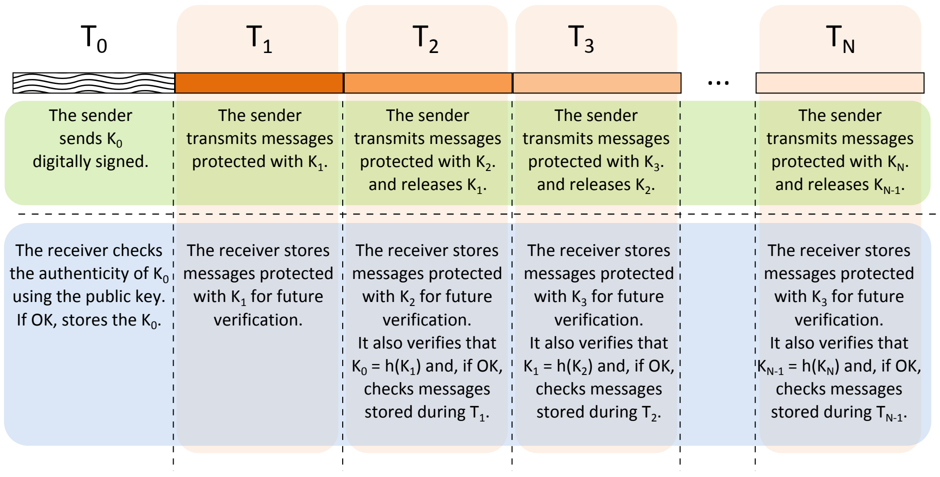

T_0
T_1
T_2
T_3
...
T_N
The sender sends K_0 digitally signed.
The sender transmits messages protected with K_1.
The sender transmits messages protected with K_2. and releases K_1.
The sender transmits messages protected with K_3. and releases K_2.
The sender transmits messages protected with K_N. and releases K_{N-1}.
The receiver checks the authenticity of K_0 using the public key. If OK, stores the K_0.
The receiver stores messages protected with K_1 for future verification.
The receiver stores messages protected with K_2 for future verification. It also verifies that $K_0 = h(K_1)$ and, if OK, checks messages stored during T_1.
The receiver stores messages protected with K_3 for future verification. It also verifies that $K_1 = h(K_2)$ and, if OK, checks messages stored during T_2.
The receiver stores messages protected with K_3 for future verification. It also verifies that $K_{N-1} = h(K_N)$ and, if OK, checks messages stored during T_{N-1}.

Figure 3.16: Time representation of TESLA protocol operation

Bill Dickerson, from Arbiter Systems, also promotes the utilization of TESLA for source authentication in PTP networks in security subcommittee discussions. In addition, in order to provide hop-by-hop traceability in PTP networks, Dickerson proposed the introduction of a new type of message, named *Hop_Delay* [68]. This message is sent from each TC and contains the residence time and peer delay values for the previously transmitted *Sync* message. The slave is responsible for authenticating *Sync* and all *Hop_Delay* messages, besides correcting the *preciseOriginTimestamp* by the delays received in *Hop_Delay* messages.

The highest benefit from using TESLA to secure PTP networks is that it provides good source authentication using computationally efficient symmetric cryptography. However, TESLA presents a compromise between the authentication delay and the propagation delay of messages. On the one hand, the authentication delay due to the key delivery time should be as short as possible. On the other hand, this time should be enough not to overtake any of the messages protected with that key.

3.3.3.6 Comparison

Table 3.8 shows the comparison between all solutions discussed in the P1588 Security SC. The third column in the Table shows the compliance of security requirements for PTP, whereas the second column collects those that were adopted from the RFC 7384 [55] for time synchronization protocols over packet switched networks in general. In the table, security requirements marked with an 'X' are completely fulfilled, whereas an '?' symbol indicates partially accomplishment and the need for additional definitions or specifications. A '-' indicates lack of mechanisms to fulfil the requirement. Many features should be enhanced on each solution to address PTP security requirements defined by TICTOC and P1588 Security SC.

In Kirrmann's solution, authentication, data integrity and replay counter protection are performed hop-by-hop using symmetric cryptography. At the same time, nodes are authenticated using certificates and they periodically transmit supervision frames in order to maintain freshness of security associations. Although authorization of PTP nodes is not implicitly provided by this solution, some authorized role information could be carried out in these supervision frames. Nevertheless, if multicast security associations are employed, this role based authorization would be based on protocol addresses or clock identifiers. As a consequence, a 'hijacked' authenticated node, for example, could perform spoofing attacks because all authenticated nodes posse the group key associated to that multicast association. In contrast, the utilization of unicast security associations with pairwise symmetric keys guarantees that a packet was sent by the legitimate node because only it knows the key. However, that results in complex key management and distribution in large networks with numerous TCs.

With Mizrahi's and Fries' solutions, the same problem of using unicast or multicast security associations for symmetric cryptography and the lack of authorization protection are presented.

On the other hand, the utilization of MACsec and pairwise keys according to Ellegaard's solution, apart from raising the problem of managing pairwise keys, also lacks a mechanism to provide node authorization. Therefore, additional and specific PTP signalling is required to verify that the sender of a *Sync* message is authorized to be a master, for instance. Consequently, authenticated nodes that belong to the secure domain could become rogue masters by performing spoofing attacks. Similarly, internal attackers could perform DoS attacks that might compromise the availability of the PTP service. Moreover, performance degradations should be further analysed due to the processing delay at each hop.

The last solution, promoted by Sibold and Dickerson, provides sender authentication using TESLA and recursive authentication by introducing the *Hop_delay* message, but the network overhead and the computation overhead should be deeply analysed. Particularly, key management for protecting unicast messages in the initial phase could be too complex in large networks with many cascaded TCs, which at the same time could flood the network with *Hop_delay* messages. Master authorization can be easily achieved by maintaining a list of authorized

Tabla 3.8: Security proposals discussed by P1588 Security SC

Security Requirement	RFC 7384	P1588	Kirrmann	Mizrahi	Fries	Ellegaard	Sibold/Dickerson
Authentication & authorization of sender	MUST	MUST	?	?	?	?	X
Authentication & authorization of master	MUST	MUST	?	?	?	?	X
Recursive authentication & authorization	MUST	MUST	?	?	?	?	X
Authentication & authorization of slaves	MAY	MAY	?	?	?	?	X
Authentication & authorization of P2P TCs by the master	MAY	MAY	?	?	?	?	X
Authentication & authorization of *Announce* messages	MUST	MUST	?	?	?	?	X
Authentication & Authorization of *Management* messages	MUST	MUST	?	?	?	?	X
Authentication & authorization of *Signalling* messages	MAY	MAY	?	?	?	?	X
Integrity protection	MUST	MUST	X	X	X	X	X
Master/slave/P2P TC spoofing protection	MUST		-	-	-	-	X
Availability (DoS attacks on PTP protocol)	SHOULD	SHOULD	?	?	?	-	?
Replay protection	MUST	MUST	X	X	X	X	X
Key freshness	MUST	MUST	X	-	X	?	X
Security association	SHOULD	SHOULD	X	-	X	?	X
Unicast and multicast associations	SHOULD	SHOULD	X	-	X	-	X
Performance: no degradation in quality of time transfer	MUST	MAY	X	X	?	?	?
Performance: computation load, storage and bandwidth	SHOULD	MAY	X	?	X	?	?
Confidentiality protection	MAY	MAY	-	-	X	X	-
Protection against delay and interception attacks	MUST	MUST	-	-	-	-	-
Secure mode	MUST	MUST	X	X	X	X	X
Hybrid mode	SHOULD	SHOULD	?	?	?	?	?

masters on each node, which can be delivered by the authentication server in conjunction with public keys.

The risk of DoS, message manipulation and packet delay attacks can be reduced but never eliminated. Without data encryption, lots of information are available in PTP messages such as vendor's name, address, PTP message types and lengths, master clock identity and class or accuracy and timestamps. All this information could be maliciously used to perform an attack that could compromise the availability of the synchronization protocol. In risky PTP networks, encryption could be supported optionally and externally by mappings to IPsec or MACsec security protocols. Nevertheless, even if encryption is performed, the transmission pattern defined by packet lengths and transmission frequency could ease the detection of PTP messages.

Packet delay attacks can be minimized using redundancy techniques as stated by Mizrahi in [56]. Other additional goals such as algorithm agility and mappings should be considered as guidelines in the development of next IEEE 1588 security solution. For instance, PTP security solution should support multiple cryptographic algorithms and provide clean and tested transition strategies between algorithms.

3.3.4 The way to the third edition of IEEE 1588 standard

As a result of all this work done by the security subcommittee, three categories were identified to classify PTP security mechanisms: end-to-end security, hop-by-hop security and network redundancy. This classification was called the 'three pronged approach' and consists of three aspects. First, strong source authentication of PTP messages sent by grandmaster to provide end-to-end security. Then, shared group key authentication to provide hop-by-hop security for messages that are modified in TCs. Finally, architectural enhancements such as the utilization of redundant communication paths and redundant grandmasters to facilitate the detection of tampering or performance degradation attacks.

In October 2014, they agreed on the need for editing a document, named Security Standing Document, with the aim of reflecting the current status of the Security Subcommittee discussions. Immediately, another prong regarding monitoring and management was added to the multi-pronged approach.

Finally, in January 2015, they agreed on a 'four pronged approach' [69]:

- Prong A: a security mechanism integrated in PTP, which will define a new Security TLV and the utilization of GDOI/TESLA/NTS to provide end-to-end strong source authentication.

- Prong B: a security mechanism external to PTP, like MACsec or IPsec, to provide hop-by-hop security. The test scenarios for using PTP over MACsec and IPsec need to be investigated further.
- Prong C: architectural guidance that will describe the use of redundant communication paths and/or grandmasters to facilitate the detection of tampering or performance degradation attacks.
- Prong D: monitoring and management guidance.

The Security Standing Document is based on the assumption that industry profiles will specify security approaches and mechanisms to protect specific applications from threats and deployment paradigms for each particular application. Therefore, the standard will include several optional security mechanisms to be used individually or in combination.

This document evolved to a draft of the informative annex that will target security features in the next edition of the IEEE 1588 standard. During last months, special attention was made by group members to Prong A specifications such as the Security TLV format. The rest of Prongs will only include guidelines to include external security mechanisms, descriptions of some redundancy architectures and a brief summary on performance monitorization. As a consequence, Prongs B, C and D need further clarification that will be left for future versions of the annex.

3.4 Conclusions

PTP protocol security vulnerabilities, like the manipulation of the best master clock selection by a rogue master, have been intensively studied for last ten years. Although an experimental security extension was firstly introduced in the second version of the IEEE 1588 standard, it was never formalized into a proper defined security solution due to its limitations.

In this Chapter, the evolution from this experimental security extension to currently discussed approaches is outlined. The great challenge of PTP security is the hop-by-hop scope because intermediate nodes need to participate in the security protocol. Hence, all nodes in a PTP domain must share the cryptographic keys in order to modify the *correctionField* in TCs. The higher the number of nodes sharing the keys, the weaker becomes the security solution.

Nowadays, the security subcommittee is still working on the development of the new annex that will be indexed in the third edition of IEEE 1588 standard. This annex will be a set of security mechanisms, rather than a security protocol, which is intended to satisfy different design and security requirements for each protocol application field. The mentioned set of mechanisms were collected on the four Prongs A, B, C and D described at the end of this Chapter.

However, the work done by the members of the subcommittee has been mainly focused on the definition of the new security TLV to address the problem of end-to-end security integrated in PTP messages as stated by Prong A. On the contrary, Prongs B, C and D need further contribution and will only be described as informative guidelines. It is expected that future work on these Prongs would be included in next revisions of the security annex. Therefore, contributions from the research community to any of the mechanisms targeted in Prong B, C and D are crucially important.

Chapter 4

Implementing secure PTP SoC architectures

4.1 Introduction

Conclusions section in Chapter 2 states the challenging process of integrating a hop-by-hop security solution to protect PTP frames in SASs without compromising substation communication performance. This hop-by-hop security solution is needed to protect synchronization because PTP messages are modified on each hop. TCs must verify incomming messages, modify the correctionField to include the residence time and regenerate the corresponding security checksums of outgoing messages. In order to provide hop-by-hop message authentication and integrity protection, in this Chapter, several tools and technologies to provide MACsec based layer 2 security in PTP networks are explored.

This Chapter consists of two clearly separate parts. On the one hand, in Section 4.2, a general hardware-software architecture to provide IEEE 1588 support is described. In addition, the most relevant PTP software and hardware modules used by the research community and industrial manufacturing companies are presented. These approaches have been identified in the bibliography as well as in the numerous attended conferences and plugfests.

On the other hand, Section 4.3 is about how to integrate layer 2 security mechanisms in PTP networks. In particular, different alternatives to include MACsec in hardware and the needed software support are briefly explored.

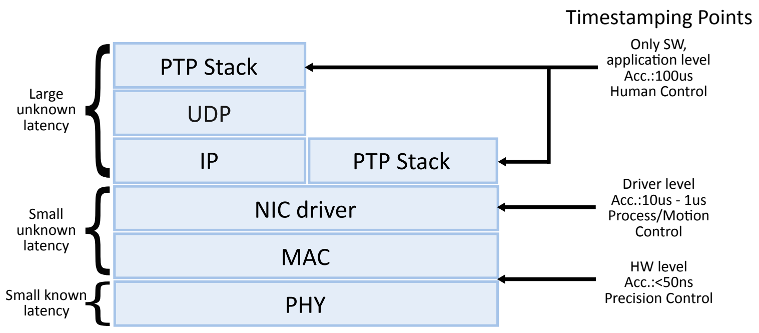

Timestamping Points
Only SW, application level Acc.:100us Human Control
Driver level Acc.:10us - 1us Process/Motion Control
HW level Acc.:<50ns Precision Control
Large unknown latency
Small unknown latency
Small known latency
PTP Stack
UDP
IP
PTP Stack
NIC driver
MAC
PHY

Figure 4.1: Possible timestamping points

4.2 Designing SoC architectures with IEEE 1588 hardware support

Hardware assistance to precisely timestamp IEEE 1588 messages is currently supported on Ethernet network interface controllers and on Ethernet physical layer devices, which can be found in multiple Network Interface Cards (NICs), such as the Intel 82576EB Gigabit Ethernet Controller [70] and the DP83640 Precision Physical Interface Transceiver (PHY) from Texas Instruments [71].

Although there are some pure software or hardware IEEE 1588 solutions, most of them are mixed solutions consisting of a CPU running the PTP software and some hardware modules capturing timestamps as close as possible to physical layer. In this Section some general mixed SoC architectures in conjunction with available IEEE 1588 software and hardware modules are presented.

4.2.1 Hardware vs. software timestamping techniques

In order to achieve precise synchronization using PTP, timestamps should reflect the send and receive times as precise as possible. PTP environment offers different possible timestamping points as shown in Figure 4.1 [72]. Two options are discussed in this Section:

1. The hardware assisted approach, in which time stamps are taken at the Medium Independence Interface (MII), between Media Access Controller (MAC) and PHY chips.

2. Pure software solutions take time stamps in the NIC driver or at the application layer.

Time stamping at the application layer has the advantage of platform independence, but it presents relative large variations on the message transit delay through the protocol stack, which is commonly known as jitter. Time stamping at the driver level is the optimal software solution but requires a modified network driver. Both options result on inaccurate measurements due to Operating System (OS) load and interrupt latency.

The closer to the physical layer are messages timestamped, the better is the achieved accuracy. In order to get better synchronization results, timestamps should be taken as near as possible to the physical layer and, therefore, the most accurate solution is that implemented in hardware. As a consequence, several vendor solutions have emerged in last years, which are enumerated in Section 4.2.5.

4.2.2 General SoC PTP ordinary clock architecture

Hardware assisted PTP solutions generally consist of a Central Processing Unit (CPU) containing the PTP stack and, at least, two specific hardware modules: the Real Time Clock (RTC) and the TSU. The RTC is normally a 80 bit counter that represents seconds and nanoseconds and can be tuned changing the values of some configuration registers of the core. Basically, these registers control the frequency at which the timer is updated and the amount of nanoseconds have to be added to it each time [72]. The TSU is the responsible for storing interesting information about the frame such as sequence ID, message type or clock identity, as well as time stamping information.

Figure 4.2 shows the SoC architecture for a PTP OC with one port, which could take the role of master or slave depending on the configuration of the PTP stack that is running in the CPU. This general architecture is only valid for two-step operation. In case one-step operation is desired, the IEEE 1588 Intellectual Property (IP) core should be able to modify PTP packets on-the-fly before throwing them to the PHY.

As it can be seen in Figure 4.2, other additional modules might be required. This is the case of the Frame Parser and the Register Interface. The Frame Parser sometimes is included inside the TSU module and it is connected to the PHY device interface sniffing all Ethernet frames and detecting which ones are IEEE 1588. Depending on the hardware solution, it can detect frames IEEE 1588 tagged in layer 2 or/and 3. Also, some implementations generate an interrupt signal

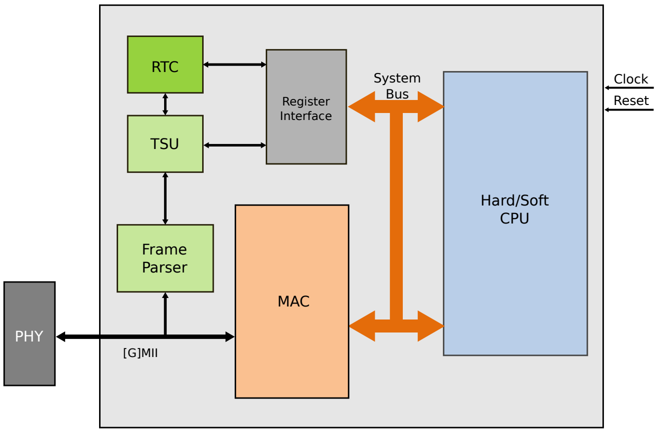


Figure 4.2: General SoC PTP OC architecture with hardware timestamping

when detect an event message (*Sync*, *Delay_Req*, *PDelay_Req*, *PDelay_Resp*) so as to become noticeable on the PTP stack, because these are time sensitive.

The Register Interface may be required by the CPU as an interface between the processor and the PTP hardware modules. Through these registers, the corresponding IEEE 1588 drivers might read the information about the frame and the timestamping, as well as control the RTC configuration. These drivers provide an abstraction layer and define the set of functions to be employed by user applications to manage the TSU and RTC cores.

Finally, other additional peripherals that are not represented in the Figure might also be needed to run the PTP stack, like timers, Double Data Rate (DDR) memory controller, Universal Asynchronous Receiver-Transmitter (UART) modules, etc. The system configuration depends on the specific application and the software to be run in the CPU.

4.2.3 General SoC PTP hybrid clock architecture

The architecture in Figure 4.2 can be extrapolated to a multiple port PTP HC by adding Ethernet switching and TC capabilities as shown in Figure 4.3. In this case, apart from the PTP stack and the IEEE 1588 drivers, also additional software modules to manage Ethernet switch could be needed.

TC functionality in Ethernet switches can be implemented following different techniques found in the literature [73]. They are basically divided into two cat-

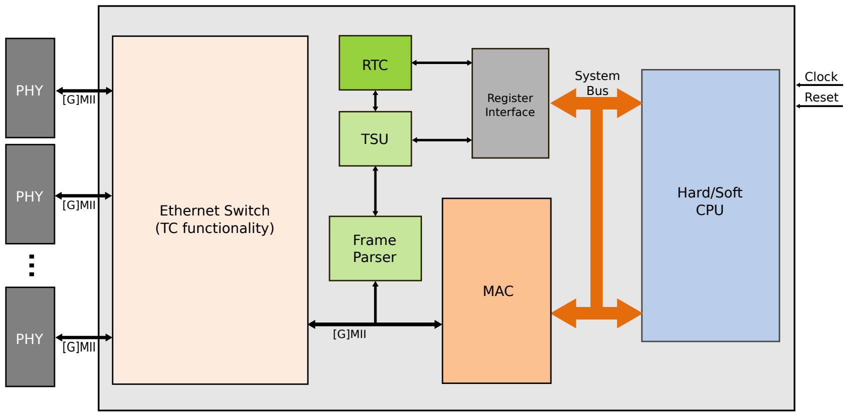


Figure 4.3: General SoC PTP multiport HC architecture with hardware timestamping

egories depending on how they address the problem of transferring information from input ports to output ports. On the one hand, some techniques forward the ingress timestamp with the frame itself and the transmission unit captures it to compute the residence time. On the other hand, timestamps can be stored in a buffer or memory which is writeable from the reception unit and readable from transmission unit, which also computes the residence time.

The later was demonstrated to be viable by implementing a Content Addressable Memory (CAM) based design and using Wishbone Bus (WB) as represented in Figure 4.4 for two ports TCs [73]. In the experiment, better residence times were achieved if compared with Spider TC solution [74], which forwards ingress timestamp in a special TLV field. However, the required resources would proportionally increase with the number of ports because several modules should be replicated. Also, proportional delay should be introduced in the transmission unit in order to avoid failures due to congested WB.

The optimal technique would be forwarding the ingress timestamp in the PTP header, for example in the reserved field, from the ingress port to the egress port. In this case, the residence time experienced by Spider TC would be considerably reduced, since the transmission unit wouldn't have to wait to the end of the frame to start transmitting it to the output port. In this sense, there is already a patent from Siemens that uses the reserved field of PTP over UDP packets to transport the reception or transmission timestamp from the physical layer to application layer [75].

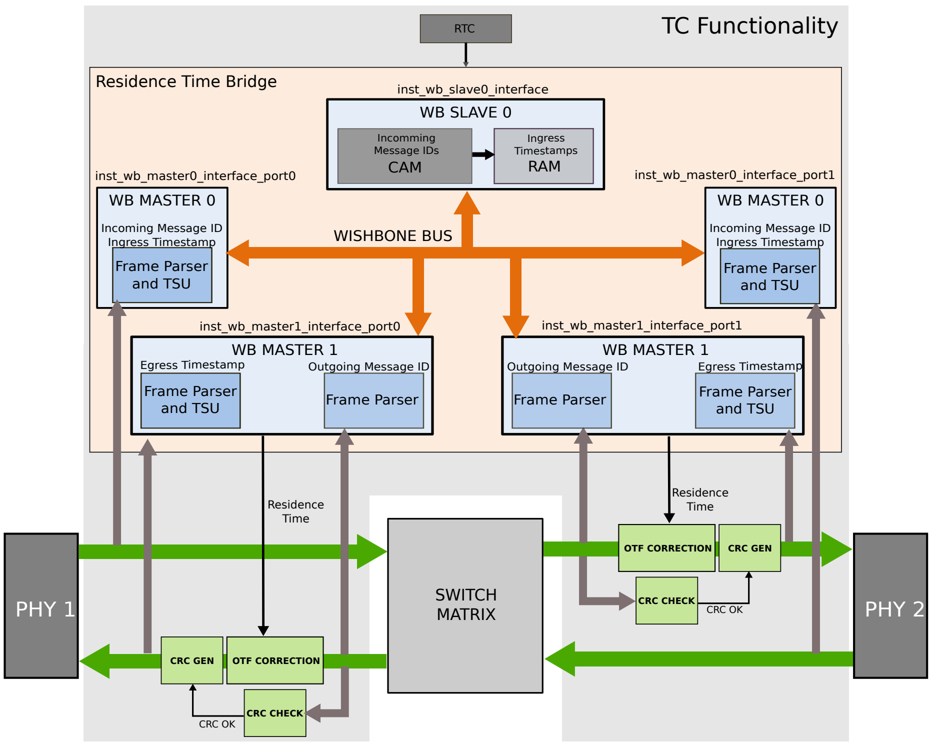


Figure 4.4: Residence time bridge WB architecture based on shared memory

4.2.4 Open source available PTP stacks

In order to run the PTP stack on the CPU of an OC SoC device, several solutions may be considered as shown in Table 4.1. They can be classified into open source and licensed software. Licensed software solutions are usually offered in conjunction with complete maintaining and training services packages, while open source are normally poorly documented and support is usually provided through mailing lists. Most of them support hardware timestamping to achieve synchronization accuracies in the nanoseconds range.

Apart from the lower cost, using open source software has the advantage of agile evolution because there are so many developers testing it that any flaws will be quickly detected and fixed. In addition, with Berkely Software Distribution (BSD) license and GNU General Public License (GNU GPL), users are free to customize the code in order to add a specific functionality, like hardware timestamping support if needed.

4.2.4.1 PTP daemon and its derivatives

The PTP daemon, named 'PTPd', is an open source software available under BSD-style lincense, which was originally developed by two engineering students at Case Western Reserve University in Cleveland (Ohio, USA) as a software-only system supporting IEEE 1588-2002 standard [86]. It was written in C language, for Unix based OSs and its first release was hosted on SourceForge in 2005. This code was capable of synchronizing computer clock frequency and time to an IEEE 1588 grandmaster without the support of specialized hardware. Neither hardware timestamps nor hardware clocks were employed by this stack. 'PTPd' performs time stamping of incoming packets using the SO_TIMESTAMP socket option of the IP stack, and outgoing packets are also timestamped using this feature after being looped back via IP_MULTICAST_LOOP. The SO_TIMESTAMP option enables the receiving of a control message containing the reception time of the last packet passed to the application using the socket system call. These timestamps are captured from system clock.

The code of the first release evolved to improved versions of the stack and, in 2010, it led to the first release of 'PTPd' implementing IEEE 1588-2008 standard. This 'PTPd' stack version 2 continued evolving to a more robust implementation of PTP version 2. The current version of the stack is the 2.3.1, which was hosted in SourceForge in 2015 [78].

The major components of 'PTPd' code are:

Tabla 4.1: Available PTP stacks

Name	Vendor/Organism	Ref.	Device Type	HW supp.	Operating Systems	License
Timekeeper	FSMLabs, Inc	[76]	OC/BC	Yes	Linux/Windows	Commercial
Domain Time II	Greyware Automation Products	[77]	OC	No	Linux/Windows	Commercial
PTPd	Various	[78]	OC	No	Linux/FreeBSD	BSD
PPSi daemon: PTP Ported to Silicon	CERN (White Rabbit Project)	[79]	OC/BC	Yes	Linux/Windows/Baremetal	GNU LGPL 2.0
RTS IEEE 1588 Network Stack	Real Time System GmbH	[80]	OC	Yes	Windows	Commercial
IXXAT IEEE1588 protocol software	IXXAT	[81]	OC/BC	Yes	Linux/Windows/Baremetal	Commercial
XR7 PTP	Flexibilis	[82]	OC/BC	Yes	Linux (Portable)	Commercial
syn1588 PTP stack	Oregano Systems	[83]	OC/BC	Yes	Linux (Portable)	Commercial
LinuxPTP project	Various	[84]	OC/BC	Yes	Linux	GNU GPL 2.0
PTP/IEEE 1588 Protocol Software OC	Zurich University	[85]	OC	No	Portable	Commercial

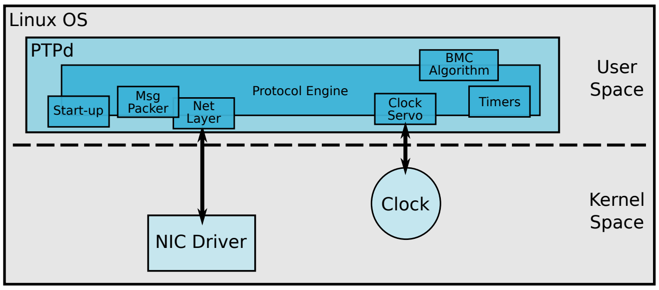


Figure 4.5: Software components of 'PTPd' stack

- Protocol Engine. It contains the main protocol state machine implemented as a forever loop and executed after Start-up.
- BMC Algorithm. This part of code implements the best master clock algorithm and returns the proper state, master or slave.
- Clock Servo. This component computes the offset from master and the corresponding clock tick rate to minimize it.
- Message Packer. It is responsible for gathering data into and extract data from PTP messages.
- Network Layer. It initializes connections, sends and receives data between PTP clocks, as well as retrieves timestamps from the control message.
- Timer. This component consists of several low resolution timers for controlling periodic *Sync* and *Delay_Req* messages, in addition to periodic runs of the BMC and timeouts.
- Start-up. It retrieves command line arguments and parameters in the configuration file as run-time options.

The first open source hardware assisted time stamping solutions were based on 'PTPd'. For example, in 2008, engineers from Intel released a fork of that daemon that takes advantage of the hardware timestamping capability of the Intel network card by using a specially modified driver and 'ioctl' calls to get hardware timestamps [87]. Two main modes of operation were added to the original 'PTPd' stack. On the one hand, the assisted system time mode synchronizes the system time but, each time the NIC timestamps a PTP packet, the NIC-system time offset is measured and added to it. On the other hand, the two-level mode synchronizes the NIC time to the grandmaster time over the network using PTP, and system time is also synchronized against NIC via local PTP.

In [88], synchronization accuracies were demonstrated to be better when using hardware timestamping of prototype Intel NIC cards, if compared to those accuracies in the order of one sub-microsecond achieved with software-only solutions.

After that, P. Ohly developed a patch to incorporate hardware timestamping into Linux kernel using the Intel Gigabit Ethernet driver, which added a new socket option called SO_TIMESTAMPING and a new Application Programming Interface (API) as defined in 'net_tstamp.h' header file [89]. He also developed a new modified 'PTPd' making use of this new API [90]. The patch was finally integrated into version 2.6.30 of the kernel, where the Intel driver was the only Linux MAC driver which supported the new feature. In version 2.6.35 of the kernel new MAC drivers were included to support it, such as the Freescale's PowerPC and Analog Devices's Blackfin microprocessors. Nowadays, the list of Linux MAC drivers that support SO_TIMESTAMPING option has been extended.

4.2.4.2 The Linux PTP project

The linuxptp stack is another open source software developed and maintained by R. Cochran that implements the PTP stack under GNU GPL. The code is available within the Linux PTP project webpage [84]. It supports both hardware and software timestamping via the SO_TIMESTAMPING socket option and uses the Linux PTP Hardware Clock (PHC) infrastructure.

This PHC infrastructure provides an interface to access and program hardware clocks in the Linux kernel, in order to use them as time sources in PTP devices. It was originated as a result of the work presented by Cochran in [89]. He started working on PTP software using the 'PTPd', but more and more problems arose when trying to keep the code up-to-date while supporting new hardware. With the aim of reducing the need for hardware-specific code in user space, Cochran's contribution provide a standardized PTP clock API for developing clock drivers and synchronizing Linux with external clocks, as well as control them from user applications. There is a PHC class driver that is used by clock drivers to register themselves and to pass events such as alarms and timestamps. At the same time, this class driver creates a character device for each registered clock, so user space programs are able to control the clock through 'ioctl', 'read()' and 'poll()' methods.

The PHC class finally merged into kernel version 3.0 in July 2011. After that, Cochran continued his work by implementing a mechanism to synchronize the Linux kernel to the very precise PTP clock. Instead of running two instances of PTP stack in the node, as the Ohly's two-level method proposes, the utilization

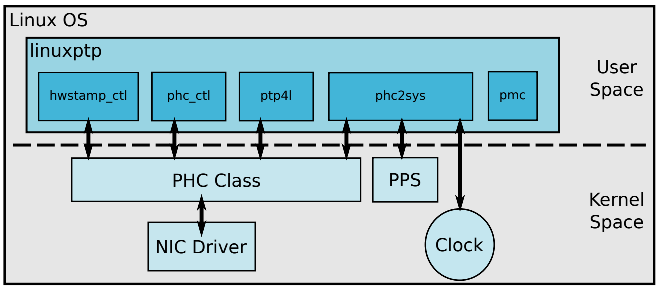


Figure 4.6: Software components of linuxptp stack

of a Pulse Per Second (PPS) signal to synchronize the system clock to the PHC was firstly presented by Cochran in [91].

As a consequence of all his efforts, in December 2012, the first release of the linuxptp stack was available and it evolved for three years until September 2015, which is the publication date of the last release. Two main programs are available within the code: the 'ptp4l' and the 'phc2sys'. However there are other three programs available:

- Ptp4l. It is the protocol engine and implements the PTP Boundary Clock (BC) and Ordinary Clock (OC) according to IEEE 1588-2008 standard.
- Phc2sys. This program synchronizes two or more clocks in the system; for example, it is typically used to synchronized the system clock to a PHC clock that is synchronized to a grandmaster using the 'ptp4l' program.
- Hwstamp_ctl. It can be used to get and set the hardware timestamping policy at the network driver level with the SIOCSHWTSTAMP 'ioctl' as a a debugging tool ('ptp4l' program automatically sets the policy in an appropriated way).
- Phc_ctl. The user can use this program to directly control a PHC clock for debugging purposes.
- Pmc. This is the PTP management client which reads actions specified by name and management identifier, sends them over the selected transport protocol and prints received replies.

By using the 'phc2sys' application, two modes of synchronization are supported. A PPS signal can be provided by the source clock and the Linux PPS API can be used to synchronize the system clock to the PHC. Otherwise, the source clock is directly read and set using general system calls like 'clock_gettime' and 'clock_adjtime'.

Depending on the source clock capabilities, one mode or another should be selected in function of desired accuracy, because it will depend on latencies introduced by hardware and driver implementation as discussed in [91]. Normally, the PPS mode is usually preferred, because reading the PHC is slow and introduces variable errors in the readings. However, not all PTP clocks are able to provide the PPS signal.

4.2.5 Commercial off-the-shelf PTP hardware assisted solutions

Besides implementing the protocol engine in software, dedicated hardware modules are required in PTP-aware SoC architectures if sub-microsecond accuracy is a requirement. Several solutions to assist PTP in hardware have been identified in the literature, which can be classified into five groups:

- IEEE 1588-aware PHYs
- IEEE 1588-aware Ethernet MACs
- Microcontrollers with on-chip IEEE 1588 hardware logic
- IEEE 1588 IP cores
- MAC IP cores with IEEE 1588 support

Ethernet PHYs and MACs are hardware modules implemented in Silicon to provide network interface to processors. Microsemi and Texas Instruments are leader suppliers of external IEEE 1588-aware Ethernet PHYs and controllers. Also, some high end microcontrollers integrate IEEE 1588 hardware logic in the same chip, like the F28M36x family of Concerto microcontrollers from Texas Instruments [92]. Also, the IXP46X product line of network processors from Intel [93] and the SAM4E series of microcontrollers from Atmel [94] provide IEEE 1588 hardware support.

Other vendors within the IP core business have been developing dedicated IP cores to provide IEEE 1588 support in reconfigurable devices. Some of the most distinguished have been collected in Table 4.2. There different types of cores available in the market have been identified using the following notation: (1) dedicated IEEE 1588 IP cores only consisting in TSU and RTC components, (2) MAC IP cores with IEEE 1588 support, (3) IEEE 1588-aware Ethernet switch IP cores and (4) full IEEE 1588 engines that do not require software stack.

Tabla 4.2: Commercial off-the-shelf IEEE 1588 IP cores

Vendor	Type of IP core				Funtionalities	Targeted FPGA
	(1)	(2)	(3)	(4)		
Flexibilis	–	X	X	–	OC/BC/TC	Altera
SoC-e	X	–	X	X	OC/BC/TC	Xilinx
MorethanIP	–	X	X	–	OC/BC/TC	Altera
Oregano	X	–	–	X	OC/BC	Lattice
Verylogic	–	–	X	–	TC	Altera
Arasan	–	X	–	–	OC/BC	Altera/Xilinx
Cadence	–	X	–	–	OC/BC	Xilinx

In modern substations, the integration of latest communication models and protocols in the process bus is really challenging due to new specific hardware and significant embedded software integration. The need for flexibility of these emerging Ethernet based protocols makes FPGAs, and reconfigurable devices in general, the best candidates to implement substation network devices [95, 96]. FPGAs offer hardware processing capabilities to achieve low switching latency times and flexibility enough to adapt the design to specific customer requirements, protocol updates and complex protocol combinations (e.g. IEC 62439-3 [35] and IEEE 1588).

Furthermore, IP core business is mature for FPGAs allowing time-to-market being dramatically reduced and scaled to different types of nodes required in networks topologies. Using IEEE 1588 IP cores and FPGAs seems to be the most precise and accurate method for implementing flexible embedded solutions capable of timestamping PTP messages in hardware. There are two main competitors that supply IEEE 1588 IP cores to companies who demand FPGA-based solutions for industrial communications. They are System-on-Chip engineering (SoC-e) and Flexibilis, which clearly target Xilinx and Altera families of FPGAs respectively.

4.2.5.1 System-on-Chip engineering IP cores

System-on-Chip engineering, commonly named SoC-e, offers communication solutions based on FPGA technology for critical systems. Their solutions consist of optimized hardware and software architectures that face the challenges of implementing standards for synchronization and reliability in substation communications like IEEE 1588 and IEC 62439-3. Their product line includes networking and synchronization IP cores, as well as boards and modules, to allow the implementation of custom routers/switches and end-equipment with powerful networking capabilities.

SoC-e provides different IP cores to integrate IEEE 1588 functionalities in network nodes:

- PreciseTimeBasic: master/slave OC and BC
- 1588Tiny: CPU-less slave-only OC
- Managed and Unmanaged Ethernet Switch: TC functionality
- HSR/PRP Switch: TC with redundancy support

The PreciseTimeBasic IP core [97] provides a hardware TSU capable of accurately timestamp IEEE 1588-2008 compliant event messages and an adjustable timer with sub-microsecond precision. The TSU was designed to be connected to the [Gigabit] Medium Independient Interface ([G]MII), between the MAC and the PHY, supporting 10/100/1000 Mbit/s interfaces. The reference design targets different Xilinx FPGA families and it also includes software modules and PTP stack are provided with the core for implementing OC/BC clocks.

In contrast to PreciseTimeBasic, the 1588Tiny IP [98] is a hard core that provides basic slave functionalities using minimum resources. This IP requires neither an embedded processor to run the PTP stack nor a generic Ethernet MAC. All these functionalities, including an optimized Ethernet MAC to process PTP frames, are implemented as hardware modules.

SoC-e's both Managed Ethernet Switch and Unmanaged Ethernet Switch IP cores [99, 100] implement a store-and-forward switching approach using non-blocking crossbar matrix and forwarding optimization techniques to achieve latencies in the nanoseconds range. Both IPs support TC functionalities, even in one-step mode of operation, implemented in an independent hardware module for each port. While the Managed Ethernet Switch can be managed through several interfaces, Unmanaged Ethernet Switch does not require external configuration and it is focused on providing Ethernet switching capabilities on plug-and-play devices.

Finally, the HSR/PRP Switch IP core [101] implements redundancy protocols as defined in IEC 62439-3 for Reliable Ethernet communications and supports 10/100/1000TX- 1000FX interfaces. This switch is an all-hardware module and, consequently, it does not need software running on a microprocessor. It provides

P2P TC functionality as specified in PTP power profile.

4.2.5.2 Flexibilis IP cores

Flexibilis Oy also develops solutions for critical communication systems based on FPGA technology. They are mainly focused on wired Ethernet and offer several IP cores with IEEE 1588 support, as well as redundancy protocols. Flexibilis work together with Altera Corporation in the development of reference designs and solutions to integrate these cores in Altera FPGAs.

Some of the IP cores that Flexibilis offers are:

- Advanced Flexibilis Ethernet Controller: Ethernet Controller with IEEE 1588 support
- Flexibilis Ethernet Switch: TC functionality
- Flexibilis Redundant Switch: TC functionality with HSR/PRP support
- Flexibilis Deterministic Switch: TC functionality with Time Sensitive Networking (TSN) support

The Advanced Flexibilis Ethernet Controller IP core [102] resulted from the natural evolution of their first Ethernet controller, and provides both copper and fiber 10/1000/1000 Mbit/s interface support. As PTP protocol was becoming more and more deployed, they also decided to add IEEE 1588 support to it. In conjunction with their XR7 PTP stack running on an embedded Linux host, master/slave OC and BC can be implemented in latest Altera families of FPGAs. An evaluation version can be obtained through their webpage.

The rest of the IP cores offered by Flexibilis all implement an Ethernet layer 2 switch with TC functionality and 10/100/1000 Mbit/s interface support. Firstly, the Flexibilis Ethernet Switch IP core [103] is a store-and-forward switch with error-checking that provides E2E TC functionality. An evaluation setup is provided by Flexibilis to evaluate this core using the Altera Cyclone IV FPGA.

Secondly, the Flexibilis Redundant Switch IP core [104] also provides seamless redundancy compatible with IEC 62439-3 standard for HSR/PRP networking and E2E TC functionality (P2P TC functionality is also supported with control software running on a CPU). This switch provides both cut-through and store-and-forward operation, although broken frames are also forwarded in cut-through mode. Several reference designs for Altera Cyclone IV and V FPGAS are available on the webpage.

Thirdly, the Flexibilis Deterministic Switch switch [105] makes possible to combine deterministic Ethernet with seamless redundancy protocols implementing TSN as defined in IEEE 802.1CB protocol. In addition, it also provides E2E/P2P TC functionality and cut-through or store-and-forward operation. At the moment, neither a reference design nor an evaluation version are available on the webpage.

4.3 MACsec support for securing PTP networks

Despite MACsec had already been considered by the P1588 Security SC as described in Section 3.3.3, they only contemplated the utilization of pairwise keys to protect point-to-point links and no key management was taken into account.

This Section addresses both the key management scheme and the SoC architecture for implementing MACsec aware devices. Firstly, from a networking engineer point of view, the utilization of MACsec and 802.1X standard in PTP networks is outlined. Secondly, different solutions for integrating MACsec encryption units in general SoC architectures are explored. Finally, available software and hardware implementations are identified.

4.3.1 Overview of MACsec and 802.1X standard

As it has been presented before, MACsec is a security framework [63] which provides hop-by-hop user data confidentiality, data integrity, authenticity and replay protection. A secured Ethernet frame is identified by the MACsec Ethertype (0xh‘88E5’) and it includes a security tag (SecTAG) and an ICV as shown in Figure 4.7.

The SecTAG includes parameters that identify the protocol and the key, and also provides the replay protection through the Packet Number (PN) field. If confidentiality is required, the Secure Data is the User Data but encrypted. The ICV ensures the integrity of the MAC Destination Address, the MAC Source Address, SecTAG and Secure Data. The mandatory cipher suite to compute the ICV uses the AES block cipher with 128-bit keys and the Galois Counter Mode (GCM) symmetric cryptography algorithm, which is known as AES-GCM-128. The AES-GCM-256 algorithm is optionally supported. In addition, other cipher suites may be implemented in the future whenever security requirements specificed in [63] were met.

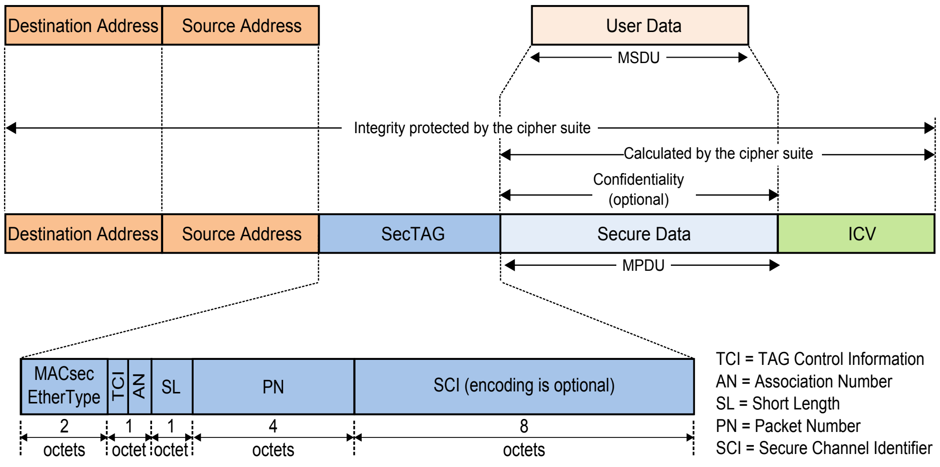


Figure 4.7: MACsec frame format

On the other hand, MACsec relies on the IEEE 802.1X authentication standard [106], which specifies a port-based network access control and defines the state machines to implement the MACsec Key Agreement (MKA) protocol among others. Consequently, different entities are involved in the normal operation of MACsec, as they have been represented in Figure 4.8. Each MACsec-aware port within a station has one MACsec entity (SecY), which provides the secure MAC service to its clients, such as the Logical Link Control (LLC) entity. Apart from the SecY, there is a MAC Security Key Agreement Entity (KaY), which discovers other authenticated KaYs in other stations attached to the same LAN by confirming mutual possession of a Connectivity Association Key (CAK) as a result of a successful authentication procedure. The KaY also uses the MKA protocol to agree the Security Association Keys (SAKs) to be used by the SecY [63].

As defined in IEEE 802.1X-2010 standard, the Port Access Entity (PAE) consists of several state machines and components represented in Figure 4.9: the Port Access Control Protocol (PACP) state machines, the Controlled Port (CP) state machine and the KaY component among others. The KaY is responsible for executing the MKA protocol, which allows PAEs to discover other PAEs in the LAN, to confirm mutual possession of a Connectivity Association Key (CAK) as a result of a successful authentication procedure and to agree the Security Association Keys (SAKs) to be used by the SecY. This CAK is derived from the Master Session Key (MSK) after an authentication procedure using the Key Derivation Function (KDF) for automatic CAK management. However, a pre-

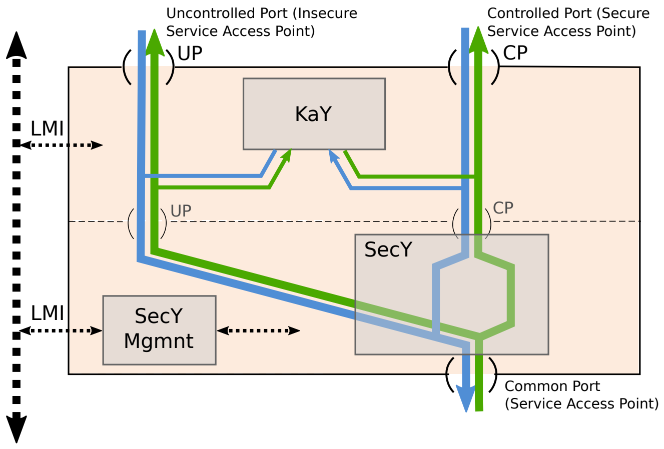

Uncontrolled Port (Insecure Service Access Point)
UP
Controlled Port (Secure Service Access Point)
CP
KaY
LMI
UP
CP
SecY
LMI
SecY Mgmnt
Common Port (Service Access Point)

Figure 4.8: Entities involved in MACsec normal operation

shared CAK might also be used. Moreover, a group CAK can be distributed by other MKA instance in a PAE entity that acts as a key server to create a group connectivity association. In any case, the CAK is never used directly but two further keys are derived from it using the KDF function, as represented in Figure 4.10.

On the one hand, the ICV Key (ICK) is used to verify the authenticity and integrity of the MKA protocol data unit. On the other hand, the Key Encryption Key (KEK) is used to transport a succession of secret keys from the key server to other members of the connectivity association that will be used by the SecY entities to protect user data. The KDF function is compatible with the counter mode KDF described in the NIST Special Publication 800-108 [107] and it uses the AES-CMAC-128 or the AES-CMAC-256 algorithm as the pseudorandom function, as specified in IEEE 802.1X-2010 standard [106].

Therefore, in order to allow the PAE to participate on the MKA protocol, it needs to be authenticated against an authentication server using a remote user authentication and accounting protocol, such as the Remote Authentication Dial-In User Service (RADIUS), as it can be seen in Figure 4.11. The result of the authentication process would be the CAK, the key that is used by the MKA protocol to establish the connectivity association and negotiate the key material.

Although IEEE 802.1X standard does not define any authentication procedure, it specifies the use of Extensible Authentication Protocol (EAP) as defined in RFC 3748 [108] and, concretely, EAP over LAN (EAPOL) to transmit EAP messages

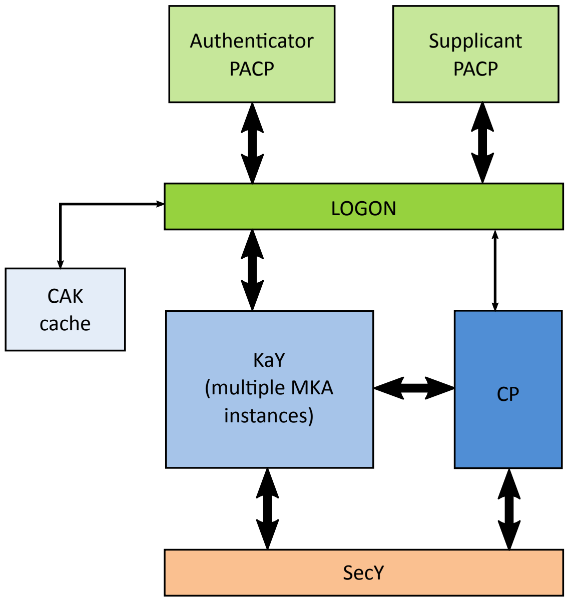
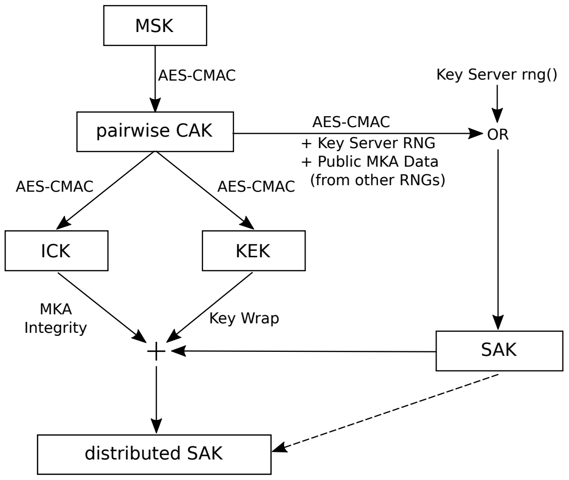


Figure 4.9: PAE state machines and components

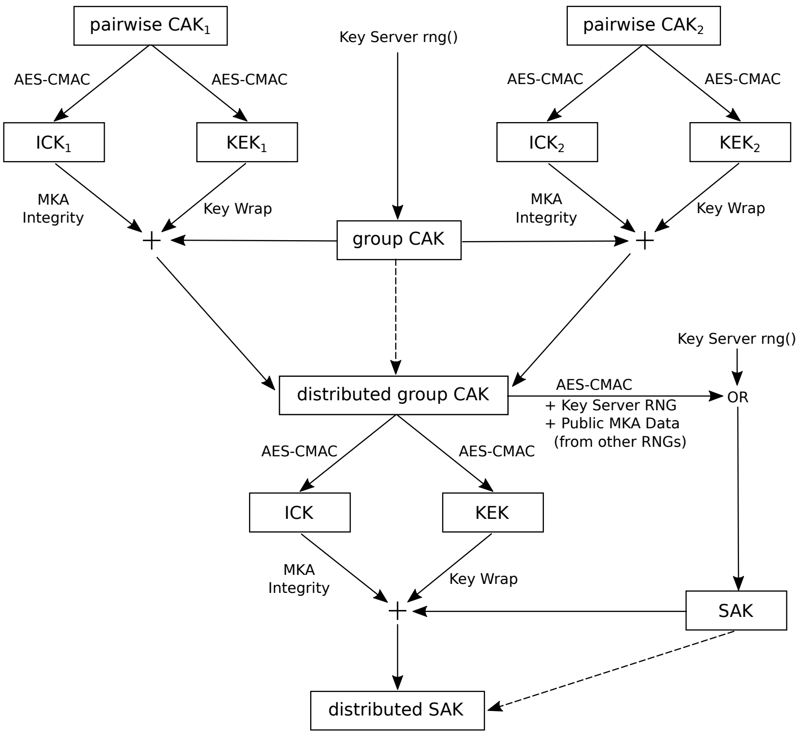
between PAEs. Besides components represented in Figure 4.9, the PAE also implements a higher layer supported by PACP which provides EAP functionality in the supplicant, and EAP functionality and Authentication, Authorization and Accounting (AAA) functionality in the authenticator. Thus, each PAE entity adopts the role of authenticator or supplicant in order to perform the EAP authentication exchange, and there is an authentication server that can be located in the same device as the authenticator or in a remote system.

To sum up, regarding key management in MACsec-based secured networks, two main solutions can be distinguished: using pre-configured keys or distribute them dynamically. Taking into account these two key management alternatives, three approaches have been identified to provide MACsec key material in PTP networks:

- **Using pre-shared SAKs.** This is the simplest solution to introduce MACsec in PTP networks because it does not need key distribution schemes. Several pairwise keys are installed in all PTP nodes before launching the PTP stack and they are periodically used after some time interval defined in the security policy. The utilization of static keys that are used repeatedly is recognized as a strong weakness due to brute force attacks



(a) Use of pairwise CAKs to distribute pairwise SAKs



(b) Use of pairwise CAKs to distribute group SAKs

Figure 4.10: MACsec and IEEE 802.1X key hierarchy

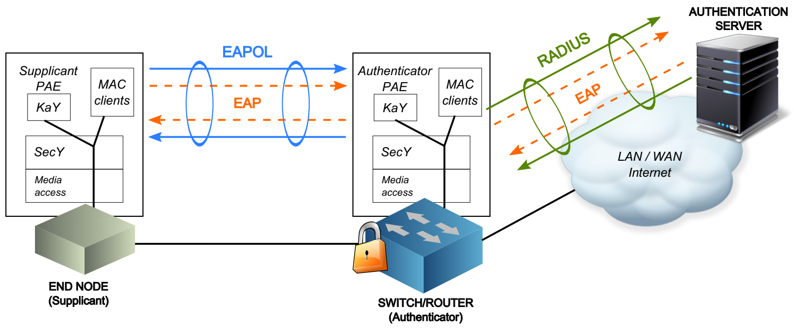


Figure 4.11: Authentication and key agreement exchange between PAEs

from external attackers. Therefore, using pre-shared SAKs is only recommended for evaluation purposes in experimental networks to test the viability of MACsec operation in PTP systems. The only additional entity to be implemented in all PTP nodes is the SecY.

- **Using pre-shared CAKs.** Instead of installing SAKs directly, pre-shared CAKs might be installed, which can be used periodically to negotiate the SAKs or each time a new device is connected to the same LAN. In this case, the volume of messages protected with the same static key is much lower than in the previous solution, and so the risk of brute force attacks is considerably reduced. Besides implementing the SecY, also one KaY per port should be implemented in this solution, in order to run the MKA protocol. Using pre-shared CAKs could be a good solution as a transition from a laboratory experimental network to a full authenticated and secured network.

- **Using EAP methods to derive pairwise CAKs.** A full authenticated network needs additional key management and distribution schemes, as the one proposed in IEEE 802.1X standard, which uses EAP methods to authenticate end nodes against an authentication server, as it can be seen in Figure 4.11. After a successful authentication, the server delivers the pairwise CAK to both supplicant and authenticator. This CAK would be used to negotiate group CAKs and fresh SAKs using the MKA protocol. Although this solution only requires the implementation of supplicant state machines in end nodes, both supplicant and authenticator state machines must be implemented in intermediate nodes, as shown in Figure 4.12 taken from [106]. Additionally, PAEs are completed with a SecY unit and a KaY instance for every PTP port.

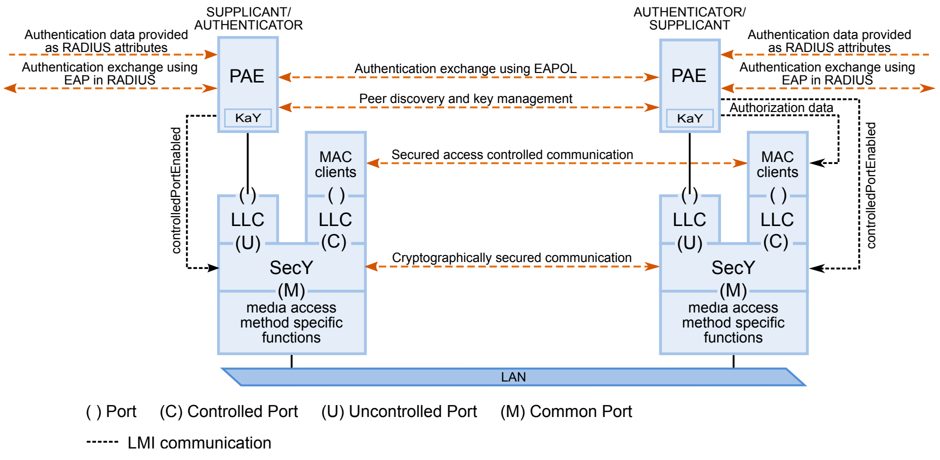

SUPPLICANT/ AUTHENTICATOR
AUTHENTICATOR/ SUPPLICANT
Authentication data provided as RADIUS attributes
Authentication exchange using EAP in RADIUS
Authentication exchange using EAPOL
Peer discovery and key management
Authorization data
Secured access controlled communication
Cryptographically secured communication
PAE
KaY
MAC clients
LLC
(U)
(C)
SecY
(M)
media access method specific functions
controlledPortEnabled
LAN
( ) Port
(C) Controlled Port
(U) Uncontrolled Port
(M) Common Port
LMI communication

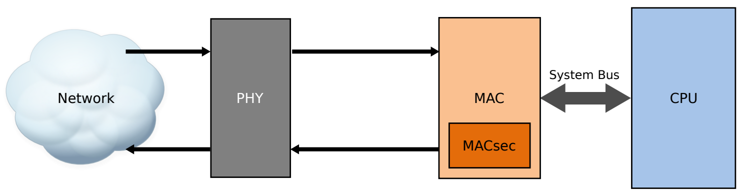
Figure 4.12: Entities involved in EAP authentication and key agreement

4.3.2 Adding MACsec capability to a general SoC architecture

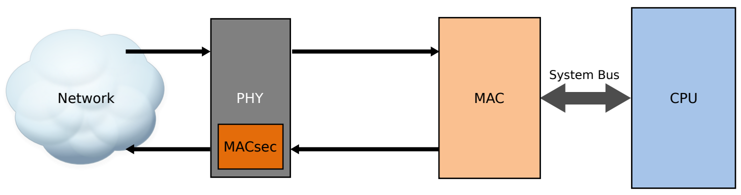
In order to provide hop-by-hop security using MACsec, two development planes must be considered. On the one hand, the MACsec control plane that is responsible for key management should consist in a set of software modules running on the processor. This software might implement the PAE state machines and components, such as the KaY entity, which have been introduced in previous Section.

On the other hand, regarding MACsec data plane, encryption units should be implemented in hardware to avoid throughput bottlenecks in the CPU due to heavy processing demands. Cryptography implemented in software to be executed on general purpose microprocessors is not as fast as hardware implementations that are designed in hardware description languages, such as the VHSIC Hardware Description Language (VHDL). Due to parallel processing and pipelining capabilities as well as instruction efficiency, hardware implementations offer really good performance in terms of processing speed. Hence, the total number of clock cycles required by an FPGA implementation is notably reduced if compared with software implementations running on a CPU [109].

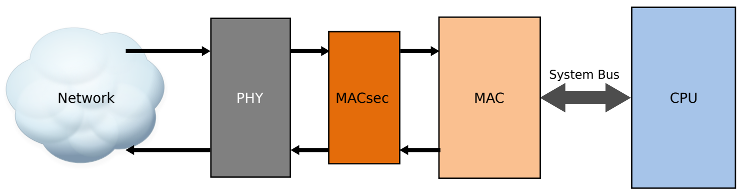
Particularly, the AES block cipher used in MACsec was designed to be efficient in both software and hardware. In spite of being well suited for 8-bit processors, software implementations of AES are not particularly efficient on 32-bit or 64-bit



(a) In the MAC



(b) In the PHY



(c) Between the MAC and the PHY

Figure 4.13: MACsec data plane entity placement options

machines [110]. Due to the fact that all time-critical functions in an AES round operate on individual bytes, processing one byte per instruction is inefficient on 32-bit machines like the ARM Cortex-A9 processor. Even with fast software implementations based on optimized round functions that operate on 32-bit words, the fastest known software implementation achieves a theoretical throughput of 1.6 Gbit/s approximately, whereas commercial hardware implementations can easily exceed throughputs of 10 Gbit/s.

Three different options have been identified to integrate MACsec data plane hardware modules in conventional Ethernet hardware subsystems. These approaches have been summarized in Figure 4.13. In Figure 4.13(a) the MACsec unit is placed in the MAC, while in Figure 4.13(b) it is place in the PHY. An alternative solution represented in Figure 4.13(c) is to place the MACsec unit between the MAC and the PHY [111].

If the MACsec entity is implemented in the MAC, the available MAC cores need modifications to support MACsec and only two-step operation is feasible, since timestamps are captured just before the PHY. Otherwise, if the MACsec entity is placed between the MAC and the PHY, neither the MAC nor the PHY need modifications because they do not need to be MACsec-aware. In addition, one-step operation should be possible if the IEEE 1588 IP core is capable of adding egress timestamps on-the-fly before passing PTP packets through the MACsec entity. The only requirement for one-step operation would be to introduce constant delays in MACsec entities. Specific MACsec-aware PHY devices would be required if MACsec entity is placed in the PHY and variable latencies are also not allowed in this case.

4.3.3 Open source available 802.1X software

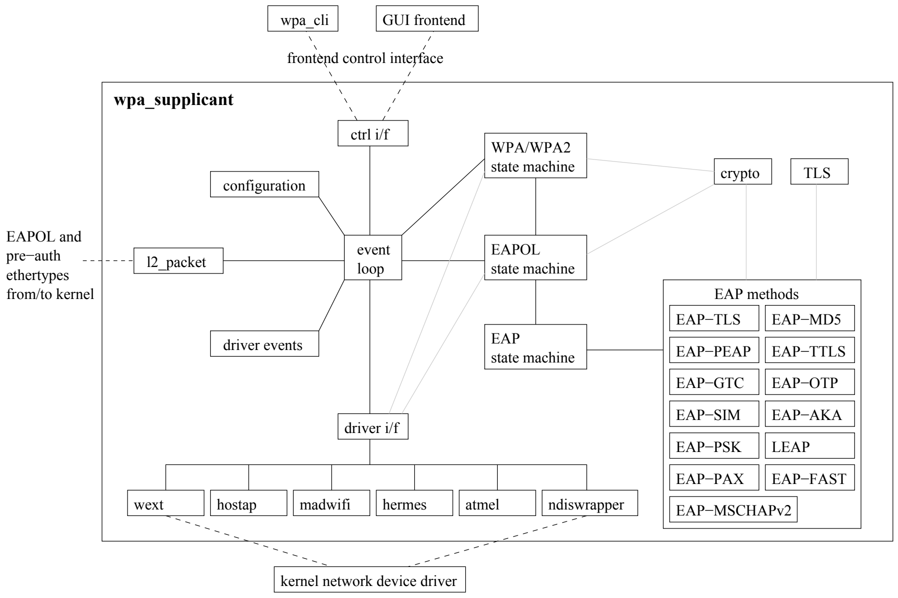
As it has been commented in previous Section, the addition of specific software modules is required to implement MACsec control plane responsible for key management as specified in IEEE 802.1X standard. Table 4.3 summarizes all IEEE 802.1X related open source software found in the literature. There are mainly three types of 802.1X software: supplicants, authenticators and servers. Some authenticator solutions may also implement local servers and some stacks allow operating as both supplicant and authenticator.

Tabla 4.3: IEEE 802.1X software solutions

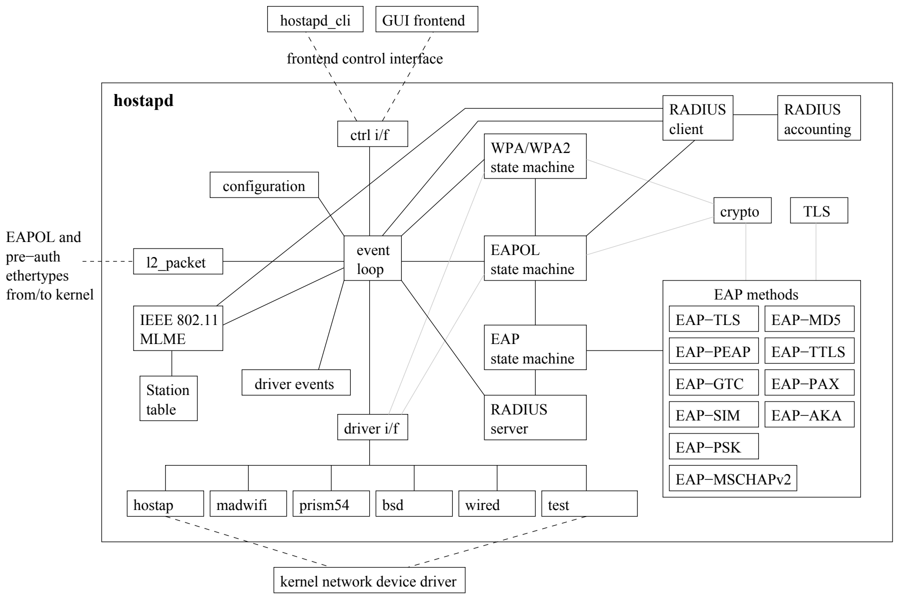
Name	**Ref.**	**Functionality**			**Platform**	**Licence**
		Supplicant	**Authenticator**	**Server**		
xsupplicant (Open1X)	[112]	X	–	–	Linux	BSD
hostapd	[113]	–	X	X	Linux/FreeBSD	BSD
wpa_supplicant	[114]	X	–	–	Linux/BSD/Windows/Mac	BSD
FreeRADIUS	[115]	–	–	X	Linux/OpenBSD/Solaris	BSD

One of the most deployed software implementations of 802.1X standard is the one maintained by J. Malinen [116], which contains two main applications: *wpa_supplicant* and *hostapd*. The former was designed to act as a WiFi Protected Access (WPA) supplicant in client stations for key negotiation with the peer WPA authenticator and authentication of the wireless station following the IEEE 802.11i standard [117]. This software uses some IEEE 802.1X security mechanisms and architecture concepts [43]. *hostapd* application implements a WiFi access point including IEEE 802.1X/WPA/WPA2 authenticator functionalities, but it is also able to act as an EAP server with RADIUS functionality.

These open source applications are available as portable C code, which includes



(a) *wpa_supplicant* modules



(b) *hostapd* modules

Figure 4.14: Block diagrams of the *wpa_supplicant* and *hostapd* applications

separate files with hardware specific functionality implementing a driver API, a control interface and EAP methods as represented in Figure 4.14 [116]. In addition, another module allows the implementation of the EAPOL state machines conforming to the specifications in the IEEE 802.1X-2004 [118] and RFC 4137 [119]. Otherwise, in the IEEE 802.1X-2010 standard, new PACP state machines are defined in order to enforce a stricter separation between them and the proper EAP state machines. Additionally, a new component implementing KaY for managing MKA protocol instances is also defined in the last version of the standard. Therefore, those EAPOL state machines used by *wpa_supplicant* and *hostapd* applications are obsolete and they do not include MKA protocol support for establishing and refreshing MACsec key material.

In J. Malinen applications, a software library that implements the PAE state machines as defined in 802.1X-2010 is also included, but the utilization of these functions in this library is limited to QCA MACsec Driver from Qualcomm. Modifications on wpa_supplicant and hostapd applications to support MKA protocol should be deeply analyzed.

4.3.4 Commercial off-the-shelf MACsec hardware solutions

Besides software modules running the control plane previously described, MACsec encryption units must also be implemented in hardware. In this Section, nowadays available out-of-the-box MACsec hardware based modules are presented, which are available as IP cores for being integrated into Application-Speci

c Integrated Circuit (ASIC) or FPGA devices. It can be distinguished between general AES-GCM crypto cores that can help building MACsec IP cores and proper full MACsec hardware cores which include frame classification and key material lookup to facilitate the integration on MAC interfaces.

Examples of AES-GCM crypto cores are those offered by IP Cores [120], Helion [121], Barco Silex [122] and Algotronix [123]. However, apart from Algotronix, these companies only provide the code for hardware processing of AES-GCM data blocks but they do not offer proper SecY functionalities to comply with IEEE 802.1AE standard.

On the other hand, there are several companies that offer hardware modules with full MACsec functionality support:

- Microsemi Corporation. The MACsec product offered by this company is based on MACsec-aware PHY devices from Vitesse combined with Intellisec technology [124, 125]. Vitesse Semiconductor was acquired by Microsemi in 2015.

- Inside Secure. This company provides a complete solution to provide secure Ethernet networking using MACsec. It is based on a MACsec software toolkit implementing the control plane and the family of SafeXcel hardware IP cores implementing the data plane [126].
- Algotronix Ltd. This company offers a complete crypto-core portfolio for Xilinx FPGAs. They provide different versions of MACsec IPs to support wire-speed processing from 1 Gbit/s to 40 Gbit/s. Algotronix claims low-latency and area-efficient in their IPs for equipment manufacturers [127].
- Synopsis, Inc. MACsec offered solutions by this company are based on integrating MACsec unit in the MAC, in the PHY or between the MAC and the PHY [111, 128]. These technologies were developed by Elliptic Technologies that was acquired by Synopsis in 2015.

Special attention deserve Microsemi and Algotronix among the companies that are offering hardware solutions for MACsec integration. Microsemi currently offers multiple physical transceiver solutions with MACsec support such as the VSC8584 or the VSC8490, which also includes VeriTime technology to enable timestamping of IEEE 1588 packets [124]. Since there is no Packet Delay Variation (PDV), there is no negative effect on PTP performance. On the other hand, the MACsec block operates in cut-through mode and, hence, frames with wrong ICV cannot be discarded. Otherwise, they must be dropped by the line or the system MAC when the frame abort signal is asserted by the MACsec block.

Microsemi also has a patent pending of approval [129] which proposes the utilization of their MACsec-aware PHYs in Wide Area Network (WAN) environments by leaving Virtual Local Area Network (VLAN) tags unencrypted, in order to forward frames and provide security in an end-to-end basis, instead the natural hop-by-hop operation of MACsec [130]. In that way, MACsec only needs to be implemented at the end nodes, reducing the cost of deployment.

Regarding MACsec IP cores, the one from Algotronix has gained popularity due to its partnership with Xilinx to be listed on their webpage [131, 132]. The core is supplied as VHDL/Verilog source code and it can be configured via compilation options to fulfil different performance and area trade-offs. Its pipelined implementation is able to provide throughputs from 1 to 40 Gbit/s. The host system should provide the core with the keys for each secure channel through a specific processor interface managed by a driver software, on top of which the KaY entity should be implemented. The MACsec core includes a SecY management module responsible for storing key material in a CAM.

4.4 Conclusions

In this Chapter, different hardware and software solutions to build MACsec-aware PTP nodes are explored. The definition of the SoC architecture that integrates MACsec hardware entities and PTP cores becomes a challenging process because of time sensitiveness of PTP event messages. The mixed software-hardware architecture should not negatively affect on PTP protocol performance and on time critical substation communications in general.

Hardware modules would add IEEE 1588 timestamping capability as well as MACsec authentication and integrity services, whereas software modules would be responsible for running PTP and IEEE 802.1X authentication stacks. Additionally, multiple port nodes would need the implementation of switching and TC functionalities between ports and also MACsec hardware units would be replicated on each port.

IEEE 1588 IP cores offered SoC-e and Flexibilis are off-the-self components to provide hardware timestamping support in FPGA-based designs. In the same way, MACsec core from Algotronix or MACsec-aware physical transceivers from Microsemi could be used to complete the secure PTP SoC architecture.

Regarding software support, the ‘PTPd’ and the linuxptp daemons have been identified as the most used open source solutions for executing PTP protocol engine in Linux systems. On the other hand, the *wpa_supplicant* and *hostapd* applications seem to be suitable candidates for implementing IEEE 802.1X stacks in embedded Linux hosts, but further modifications of the EAPOL state machines should be developed so as to conform to the last version of the standard that includes the MKA protocol needed by MACsec to establish session keys.

Chapter 5

SoC architecture for secure PTP

5.1 Introduction

In the literature, several embedded and SoC architectures for implementing IEEE 1588-2008 and IEC 61850 standards were found, but no one includes cyber-security mechanisms. The main objective of this thesis is to propose a new PTP SoC architecture that integrates MACsec hardware units, taking into consideration all the information gathered in Chapter 4.

A general overview of the proposal and a whole set of design and security requirements that the new SoC architecture should fulfil is outlined in Section 5.2. Sections 5.3 and 5.4 will truthfully be the key contributions: a modification of the general procedure described in IEEE 802.1X-2010 standard for MACsec key management and distribution and the proposed SoC architecture. Finally, conclusions are summarized in Section 5.5.

5.2 MACsec-based proposal for securing PTP

Chapters 2 and 3 explain issues derived from securing SAS communications and PTP messages in particular. In Table 5.1 each issue is associated with a security or design requirement. Bearing in mind the information found in the literature regarding security mechanisms for substation communications and the work done by the P1588 Security SC, an hybrid security solution is proposed: the utilization of MACsec protocol to provide hop-by-hop group authentication, in combination with a new PTP Security TLV to address the problem of end-to-end source authentication. How these two main security mechanisms might face major PTP security problems, is explained in Annex A and summarized in Table 5.1.

Tabla 5.1: MAcsec-based proposal

No.	**Requirement**	**Issue description**	**Proposal**
R1	Source authentication and authorization	Some PTP messages need source authentication. Group keys used to protect multicast PTP messages do not guarantee source authentication.	A new Security TLV includes an ICV computed with a symmetric group key that belongs to a TESLA key chain. This ICV is only verified in destination nodes.
R2	Group authentication and authorization	Some PTP messages are modified by TCs. Intermediate nodes need to demonstrate they belong to the group of nodes authorized to modify PTP packets.	The MKA protocol is used between peer nodes to verify the possession of the group CAK and establish pairwise SAKs.
R3	Hop-by-hop integrity	The security checksums of some PTP messages must be verified and regenerated on each hop.	PTP messages are encapsulated in MACsec frames, and the ICVs are verified/regenerated on each hop using pairwise SAKs.
R4	End-to-end integrity	Most PTP messages need to protect end-to-end integrity of PTP fields that are not modified in the forwarding path.	The ICV included in the new Security TLV guarantees the end-to-end integrity.
R5	Replay protection	PTP accuracy might be degraded if an attacker captures messages and replays them later.	Both the Security TLV and MACsec SecTAG include a kind of sequence number to provide end-to-end and hop-by-hop replay protection mechanism.
R6	Unicast key management	Unicast addressing of PTP messages is not employed in PTP power profile. A unicast key management scheme could be used in early stages of authentication protocols.	Protocols like TLS or IKE might be used to derive unicast symmetric keys. Also, IEEE 802.1X standard distributes unicast keys at the beginning of the authentication mechanism.

Continued on next page

Tabla 5.1 – *Continued from previous page*

No.	Requirement	Issue description	Proposal
R7	Multicast key management	An efficient key management and distribution scheme to provide group keys is required.	TESLA keys may be distributed within the Security TLV from master to slaves. IEEE 802.1X standard is proposed to distribute group CAKs from a server to all authorized PTP nodes.
R8	Sub-microsecond accuracy	In PTP power profile, the synchronization accuracy is specified to be better than one microsecond, so hardware timestamping and TC functionalities should be integrated in the design.	IEEE 1588 IP cores and IEEE 1588-aware switch IP cores are integrated in the SoC architecture.
R9	Avoid negative impact on time-critical SAS communications	Within SAS communications, traffic with the most restrictive latency requirements are GOOSE and SV, which require response times of 4 milliseconds. Processing and forwarding latency in SAS network nodes should be minimized.	Traffic separation is performed in MACsec crypto units depending on the Ethertype. Time-critical messages like GOOSE or SV are transmitted without SecTAG-ICV fields through the Uncontrolled Port (UP) of the SecY.
R10	Avoid throughput bottlenecks	Security checksums of frequent PTP messages, like Sync messages, which are in addition modified on each hop, should be verified and regenerated in hardware, in order to avoid a negative impact on PTP stacks or on other time critical software.	MACsec crypto units are implemented in hardware. The utilization of crypto accelerator cores is also recommended to check and compute the ICV in the Security TLV, or to protect EAPOL frames.
R11	Reduce the probability of external attackers	Since PTP messages are modified on each hop, an attacker could gain access to network, intercept packets and modify packets with the aim of making slaves to align to a false time value.	With MACsec external attackers have not access to cryptographic keys and, as a consequence, they can only perform packet delay attacks on PTP protocol, apart from interception and removal of packets.
R12	Reduce the probability of internal attackers	A TC could be highjacked and used to perform a MITM attack, for instance, modifying maliciously the *correctionField.*	If TC and switching functionalities are implemented in hardware, the risk of internal attacks is reduced to end nodes.

The definition of a new Security TLV has been the main focus of the P1588 Security SC for last years and different approaches were considered within Prong A of the Security Standing Document, which was introduced in Section 3.3.3.6. As a result, there is currently a draft document of the informative Annex that will address PTP security approach in next edition of IEEE 1588 standard. The proposed Security TLV in the draft can operate in two modes: using group-based key management or using TESLA-based key management. Since our proposal involves the utilization of MACsec to provide hop-by-hop group authentication, there is no sense on using a group-based key management mechanism with that additional TLV. The aim of specifying a new Security TLV is to provide an end-to-end source authentication and, consequently, only the TESLA-based key management is supported by this proposal.

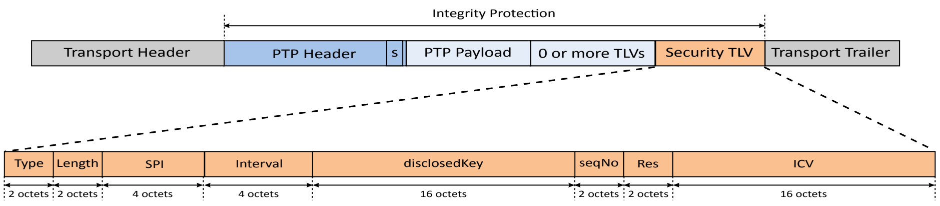


Figure 5.1: New Security TLV to be used with TESLA as defined by P1588 Security SC

The Security TLV format when using TESLA-based key management is represented in Figure 5.1. The TESLA key scheme can be employed to share keys between masters and slaves in a delayed fashion, while the Security TLV protects all PTP fields that are not modified by intermediate nodes, such as the *correctionField*. Instead of the key management being handled above layer 3 communications as stated in the mentioned draft, old TESLA keys to be disclosed might be included in a dedicated field, named *disclosedKey* of the Security TLV, as it can be seen in Figure 5.1. Since the operation of TESLA-based key schemes need participants to be synchronized, in the first stage of PTP protocol, additional key management schemes using asymmetric cryptography should be employed to distribute TESLA parameters and even the full TESLA key chain (or alternatively the first key, from which the whole chain can be generated).

Regarding the proposed hop-by-hop group authentication mechanism, Ethernet frames carrying PTP messages are encapsulated with MACsec security tag and the ICV as represented in Figure 4.7, where the MAC Protocol Data Unit (MPDU) is the PTP message to be transported over layer 2 networks. Peer nodes use the MKA protocol described in Section 4.3.1 to distribute MACsec session keys, which are pairwise SAKs, after they had been authenticated and authorized to a key server using EAP methods. Nevertheless, EAP methods as described in IEEE 802.1X standard can be employed to provide pairwise CAKs, but not to stablish and distribute group CAKs. Therefore, a new approach scheme to derive group CAKs using EAP methods is proposed in Section 5.3.

Taking into account this proposed key management scheme for PTP over layer 2 networks protected by MACsec, a new SoC architecture is also presented in Section 5.4. This architecture should provide sub-microsecond accuracy with hardware timestamping units and implement security units in hardware to avoid throughput bottlenecks. In this sense, the whole system throughput and latency need to be analysed in order to study the impact of MACsec on protocol performance.

5.3 TimeWardenKey: a new key management approach for PTP networks

5.3.1 Group keys distribution problem

Since the nature of MACsec is hop-by-hop operation, SAKs distributed by the MKA protocol are only valid at one physical link, no matter they are pairwise keys or group keys. The utilization of group SAKs acquires sense when protecting broadcast and multicast traffic with MACsec over wireless LANs, because several stations share the physical media. In contrast, in wired LANs, two stations are directly connected through point-to-point links and pairwise SAKs can be used to protect all types of traffic between them.

It can be thought that the only advantage of using group SAKs in wired networks is the utilization of a shared memory for storing group SAKs in intermediate nodes, which is accessible from all SecY units with the consequently saving of storing resources. However, a dedicated SA memory for storing SAKs is typically included in MACsec hardware implementations, such as the Algotronix IP core introduced in Section 4.3.4, and one instantiation of that core is required per port. Hence, if the same SAK is aimed to be used by all ports, it must be replicated on every SA memory implemented on each port, avoiding users to be able to save storing resources even using group SAKs.

In order to negotiate and transport pairwise SAKs between peer ports using the MKA protocol, both must own the same CAK as a result of an authentication procedure that was ended successfully. Instead of using pre-shared CAKs, IEEE 802.1X-2010 standard specifies the derivation of pairwise CAKs from EAP methods to achieve a full authenticated scheme. But, if this full authentication scheme is employed in wired Ethernet networks, intermediate nodes like switches must implement both supplicant and authenticator state machines, in addition to AAA functionality, as explained in Section 4.3.1.

Otherwise, if a more centralized authentication approach is employed, where there is only a multi-host authenticator and all PTP nodes in the LAN act as supplicants, intermediate nodes only need to implement the supplicant state machines. As a consequence, this approach could suit better on substation nodes with limited resources. In the next Section, a new key management scheme to address the problem of limited resources in substation nodes is proposed, which uses EAPOL messages to perform EAP methods and to derive the group CAK.

This new key scheme must overcome two main identified problems:

- EAPOL messages are filtered by IEEE 802.1D and IEEE 802.1Q conformance switches and bridges. The destination group addresses of EAPOL-EAP frames as specified in Table 11-1 of 802.1X-2010 standard are not forwarded by neither switches nor bridges. Hence, a new addressing scheme must be defined in order to allow these frames to be forwarded through multi-hop transmission paths.
- All PTP nodes in the LAN must share a group CAK. As a result of the EAP authentication each station obtains a pairwise key, named the Master Session Key (MSK), which is only known by the supplicant and the authenticator and it is not used directly to protect data. In order to allow the secure communication between peer ports of PTP nodes an additional group CAK must be distributed from the authenticator to all the supplicants, before using the MKA protocol to establish SAKs.

In the next Section, a new full authenticated scheme to derive group keys using EAP methods is proposed for PTP layer 2 networks. The goal of the proposed solution is to implement a simplified 802.1X authentication scheme targeted for applications that require layer 2 PTP and have constrained resources, like Substation Automation Systems.

5.3.2 TimeWardenKey description

As introduced above, all PTP nodes are expected to act as supplicants in the proposed key management scheme, and there is a multi-host authenticator node that also implements the AAA functionality if required to transport the authentication messages to the authentication server. However, IEEE 802.1X standard currently does not support such a centralized scheme in wired networks due to two main problems that have already been identified: the EAPOL frames addressing and the derivation and distribution of the group CAK.

Figure 5.2 represents a PTP network where the gateway acts as the multi-host authenticator and all PTP clocks assume the role of supplicants. The multi-host

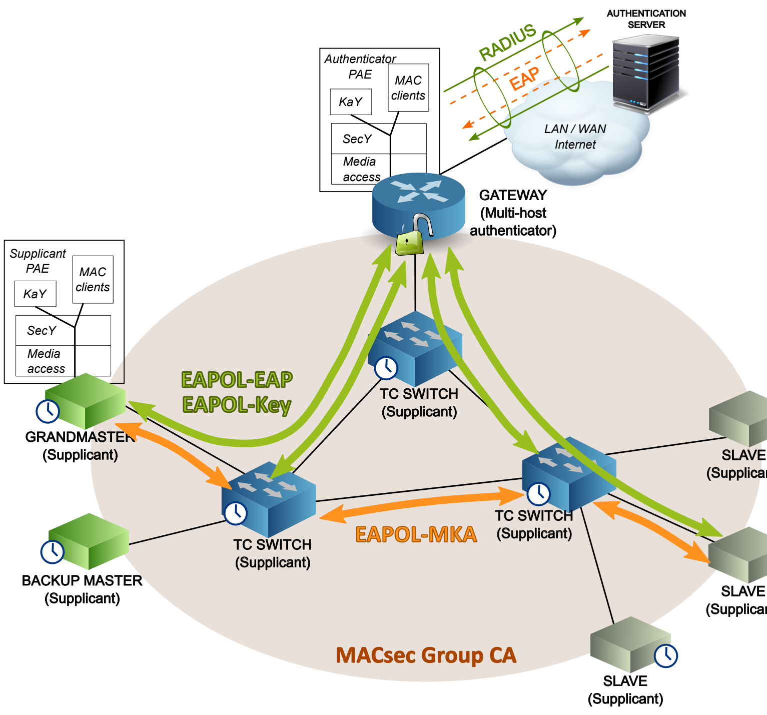


Figure 5.2: Example of the utilization of TimeWardenKey scheme in PTP layer 2 networks

authenticator delivers the group CAK to all PTP nodes as a result of successful individual authentication exchanges. By this way, TC switches only have to implement one supplicant state machine, which installs the received group CAK on each MKA instance associated to each PTP port.

Nevertheless, IEEE 802.1X-2010 standard only specifies the derivation of pairwise CAKs from EAP methods. Concretely, input parameters of the key derivation function are the MSK and the MAC addresses of both supplicant and authenticator, among others. Therefore, additional message exchange to negotiate this group CAK is needed. The proposal described in this Section is inspired in the IEEE 802.11i standard for wireless networks [117] and includes a group key handshake, which is presented below.

Moreover, with the aim of deriving and distributing the group CAK, a new key hierarchy is defined as depicted in Figure 5.3. From the MSK, a pairwise CAK is generated with the KDF function as specified in IEEE 802.1X. Similarly, ICK and

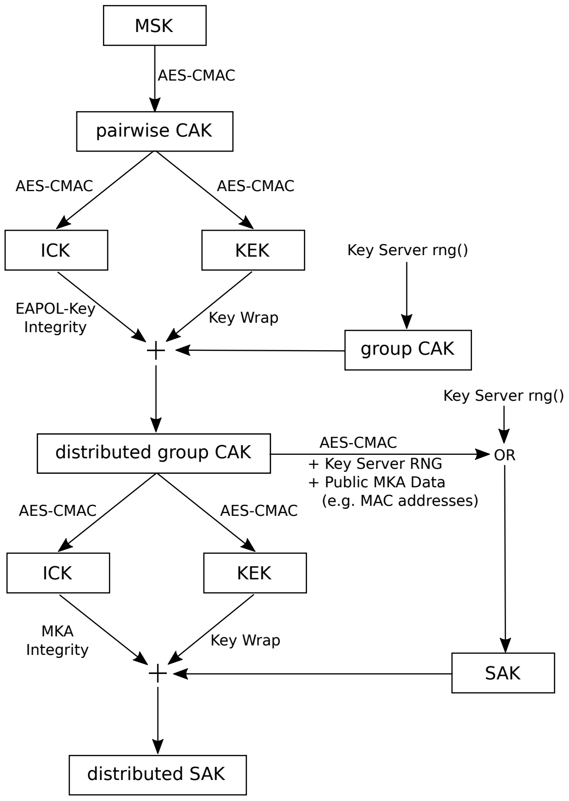


Figure 5.3: New key hierarchy proposed by TimeWardenKey

KEK keys are generated to distribute a group CAK using EAPOL-Key frames. A new key descriptor could be defined in order to transport the group CAK using EAPOL-Key, which might be based on the key descriptor defined in IEEE 802.11i standard, but using the KEK and AES Key Wrap functions to encrypt the frame and the ICK and AES-CMAC algorithm to compute the ICV.

After that, PTP nodes start the MKA protocol to demonstrate the possession of the group CAK. On each physical link, one peer node assumes the role of key server, generates the SAK and distributes it as specified in IEEE 802.1X-2010 standard. Now, all stations belonging to the group of authorized PTP nodes share the SAKs with their peers and are able to secure PTP frames using MACsec encapsulation.

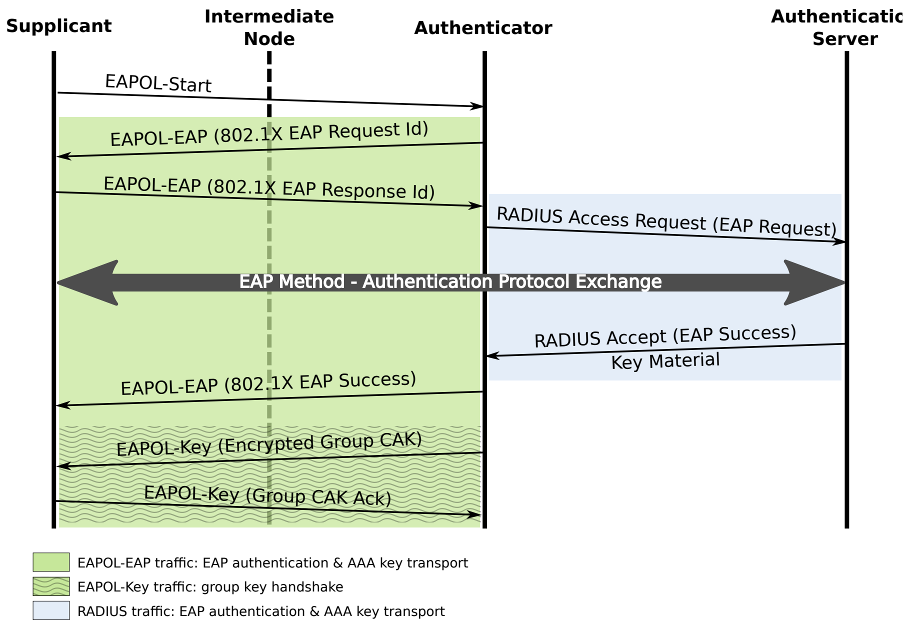

Supplicant
Intermediate Node
Authenticator
Authenticatic Server
EAPOL-Start
EAPOL-EAP (802.1X EAP Request Id)
EAPOL-EAP (802.1X EAP Response Id)
RADIUS Access Request (EAP Request)
EAP Method - Authentication Protocol Exchange
RADIUS Accept (EAP Success)
Key Material
EAPOL-EAP (802.1X EAP Success)
EAPOL-Key (Encrypted Group CAK)
EAPOL-Key (Group CAK Ack)
EAPOL-EAP traffic: EAP authentication & AAA key transport
EAPOL-Key traffic: group key handshake
RADIUS traffic: EAP authentication & AAA key transport

Figure 5.4: New EAPOL message exchange defined by TimeWardenKey

The resulting EAPOL message exchange for the new group CAKs distribution scheme has been represented in Figure 5.4. In order to exchange EAP authentication messages between supplicants and authenticator, as well as to deliver the group CAK from the multi-host authenticator to all supplicant PTP nodes, EAPOL-EAP messages must traverse intermediate nodes. The new addressing scheme for EAPOL messages is proposed to be as following:

- Supplicants send an EAPOL-Start frame with a multicast address as the destination MAC address, in case the MAC address of the authenticator is unknown. For example, the PTP multicast address 01:1B:19:00:00:00 might be used. An EAPOL-Start frame is only processed by the authenticator.
- Subsequent EAPOL-EAP and EAPOL-Key frames to derive the MSK and the group key respectively must contain a unicast destination MAC address.
- EAPOL-MKA frames used to negotiated and establish the pairwise SAKs between peer ports must use a group MAC address as specified in Table 11-1 of IEE 802.1X-2010 standard, in order to avoid accidentally or maliciously

creation of multi-hop MACsec tunnels.

- EAPOL-Logoff frames from supplicants employ the individual address of the authenticator to terminate the EAP authentication, but MKA protocol and the connectivity between peer ports continue until MKA fails because of an error or the deletion of all state associated with the authentication by the supplicant, including MKA and EAP derived state.

5.4 TimeWardenSoC: a new SoC architecture for secure PTP

Bearing in mind the security and design requirements collected in Section 5.2, as well as the new key management scheme proposed in Section 5.3.2, a new SoC architecture to protect PTP traffic over layer 2 networks like the process bus in substations will be presented. After being described the block diagram of the proposed architecture, it is modelled in terms of throughput and latency. Finally, other design issues such as an estimation of area cost are also outlined.

5.4.1 TimeWardenSoC architecture description

With the aim of defining a new SoC architecture for IEDs that integrates both MACsec and IEEE 1588 protocols, MACsec hardware units have been added to the architectures explained in Section 4.2. These MACsec cores have been integrated between the MAC IP core and the PHY external device. The resulting SoC architecture has been represented in Figure 5.5. This placement requires a MAC 'sandwich' around MACsec core to adapt its specific data transmit and receive interfaces to standardized [G]MII interfaces.

The MACsec 'sandwich' should be responsible for adapting data rates by changing data bus width, as well as generating flow control signals based on [G]MII signals comming from the PHY and the MAC. In addition, a frame filter should be implemented to classify frames. The ones that require MACsec protection, like PTP event messages, have to be delivered through the controlled port. The rest of frames must be forwarded through the uncontrolled port, for key management purposes for example.

Furthermore, the MACsec 'sandwich' must verify and regenerate the Frame Check Sequence (FCS) checksum of ingress and egress Ethernet frames that traverse the controlled port using the Cyclic Redundancy Check (CRC) algorithm. This operation is requested because the SecY unit changes the content of the

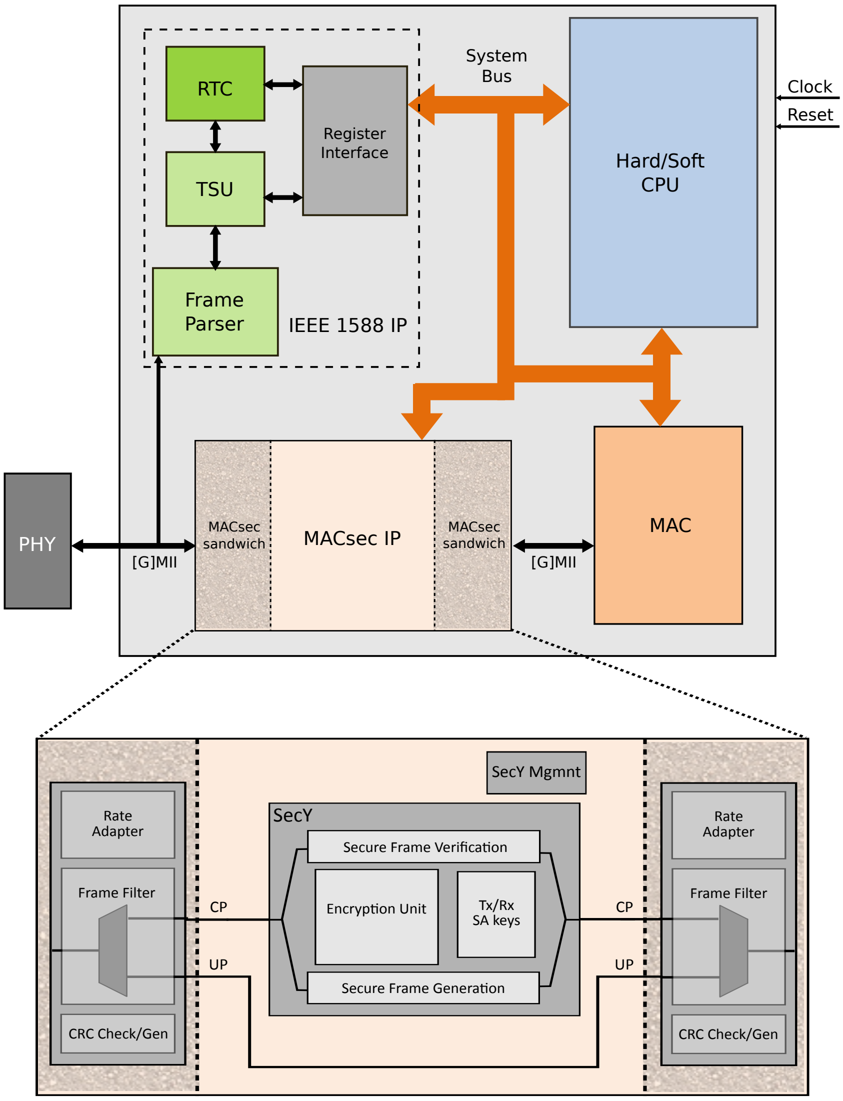


Figure 5.5: TimeWardenSoC architecture for an OC with a single port

frames in both directions: from network to processor, the SecTAG and the ICV are removed and, from processor to network, the SecTAG and the ICV are added. In the transmit path, the CRC Checker in the MAC side of the 'sandwich' removes the FCS of the Ethernet frame, while the CRC Gen in the PHY side appends the new generated FCS at the end. In the receiver path, the same happens in the opposite direction. Network errors detected by a wrong FCS code at one side might be notified to the other side with an error signal, which should take the corresponding action. For instance, when the calculated FCS in the CRC Checker does not match with the value of the received FCS, the CRC Gen is ordered to append a well known false FCS value of all ones, for example.

The module called SecY within the MACsec IP core provides secure verification of receiving frames and secure generation of transmitting frames using an encryption unit. This encryption unit implements two instances of the AES-GCM algorithm, because receive and transmit paths cannot share the same block cipher [1]. Session keys, named SAKs in IEEE 802.1AE standard, are written by the SecY Management module under the order of control plane software, which at the same time controls the security parameters of MACsec channels and associations between peers.

The proposed architecture represented in Figure 5.5 for a single port PTP node can be extrapolated to multiple ports by adding an Ethernet switch IP core with TC functionality, and replicating the MACsec IP core on each port. The resulting architecture is depicted in Figure 5.6.

In substation communications, traffic separation should also be implemented by the frame filter in IEDs, throwing time-critical messages to the uncontrolled port. For instance, in the case of an HSR ring topology for the process bus, GOOSE and SV messages would be transmitted with Ethertype=0xh'892F' within the HSR tag immediately after the Ethernet source address, and without SecTAG-ICV fields. In case the IED is not connected to an HSR ring, but to a PRP switch, the frame filter within the MACsec 'sandwich' should also detect GOOSE and SV frames with Ethertypes 0xh'88b8' and 0xh'88ba' respectively, and deliver them through the uncontrolled port. The particular case of an IED SoC architecture with MACsec and High Availability Ethernet support is shown in Figure 5.7.

A CPU is needed in all proposed architectures in order to run the PTP stack and the MACsec control plane as explained in Sections 4.2 and 4.3.2. The block diagram of the resulting software architecture is depicted in Figure 5.8. At the left side of the block diagram, software modules related to IEEE 1588 standard are included. It can be seen that a generic PTP stack retrieves timestamping

[1]The encryption unit must be able to handle receiving and transmitting frames simultaneously.

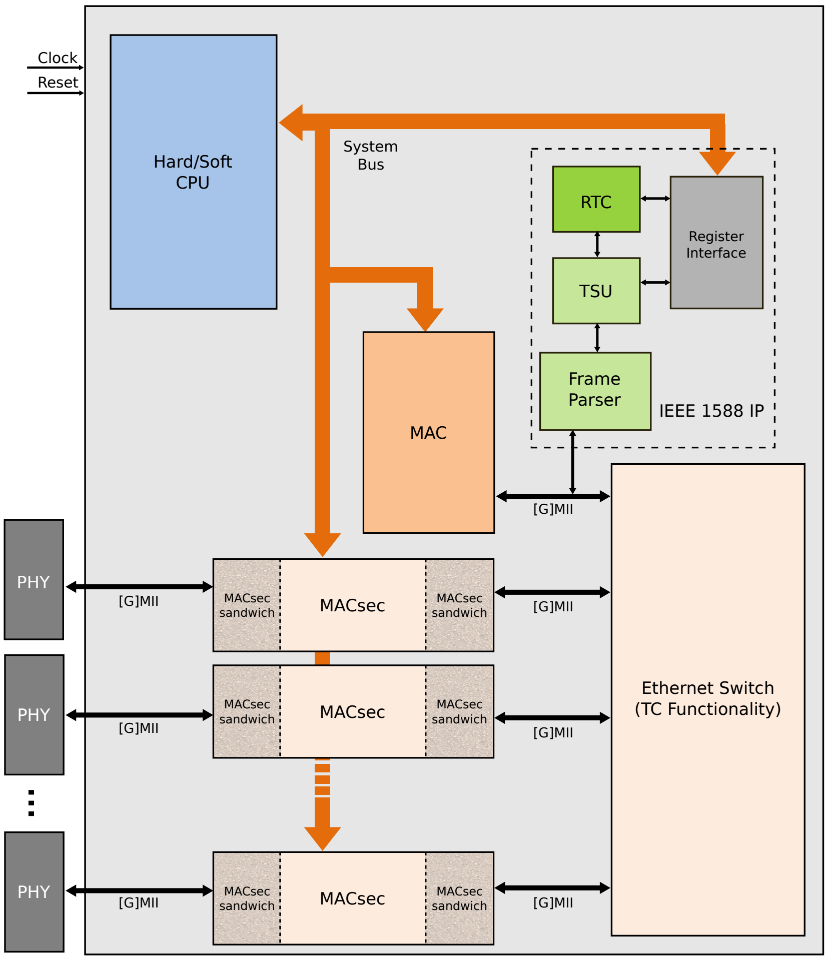


Figure 5.6: TimeWardenSoC architecture for a HC with multiple ports

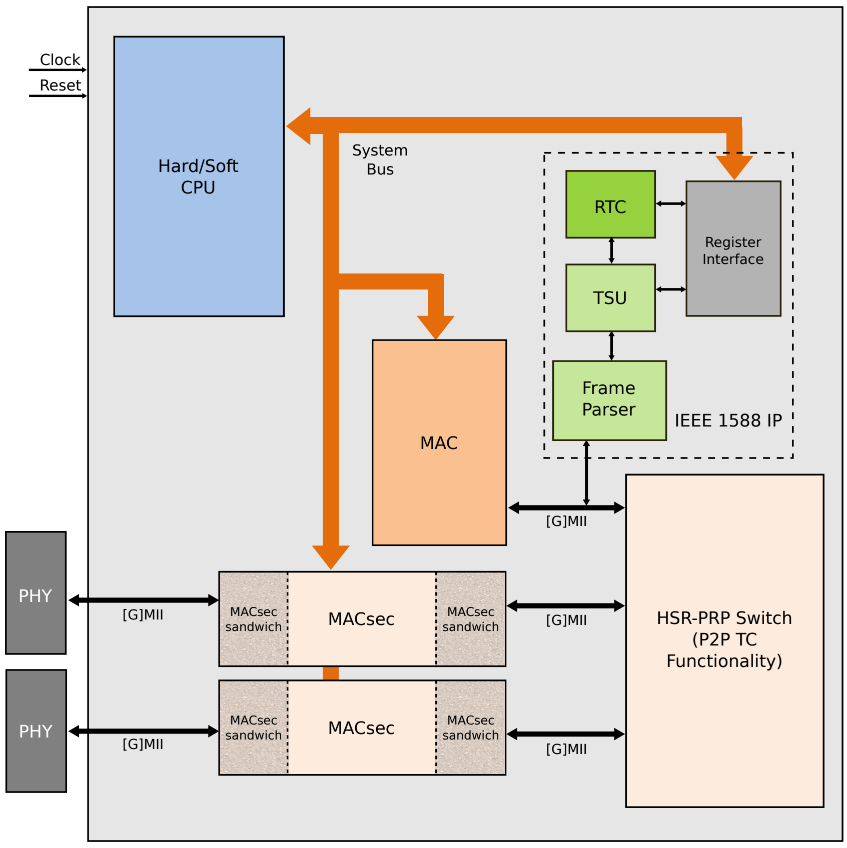


Figure 5.7: TimeWardenSoC architecture for an HSR node

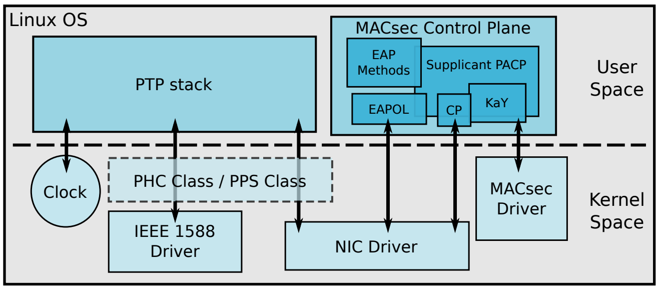


Figure 5.8: Software components for the TimeWardenSoC architecture

information from the driver of the IEEE 1588 IP core. The stack can also optionally make use of the PTP Hardware Clock infrastructure and the PPS Class as explained in Section 4.2.4.2, as represented with a dashed line in the Figure.

In Figure 5.8, the software related to IEEE 802.1X standard is typified as MACsec control plane. According to the new MACsec key management approach presented in Section 5.3.2, there is an only Supplicant PACP module that implements modified EAP state machines to include the distribution of a group CAK using EAPOL-Key frames. This module orders the KaY entity to manage as many MKA instances associated to that group CAK as the number of ports. Each MKA instance allows the operation of the MKA protocol on each link and controls the parameters and the keys to be used by the SecY. All ports might also have in common a single Controlled Port module to configure MACsec security parameters, such as the cipher suite or the utilization of confidentiality services in addition to authentication if desired.

5.4.2 TimeWardenSoC architecture characterization

In this Section, the viability of implementing the proposed SoC architecture to protect IEEE 1588 frames without compromising PTP protocol performance is analysed. There are two paths within the SoC architecture shown in Figure 5.5 that meet at software user level on the execution of the PTP stack. These paths must be analysed separately.

On the one hand, timestamping information goes through the timestamping path defined by those components that constitute the IEEE 1588 IP core. Throughput and latency are determined in this path by the [G]MII interface bit-rate, because the information of the received PTP event message and the corresponding timestamp must be provided to the Register Interface before the arrival of the next PTP event message. Since interfaces between components within the IEEE 1588 IP core are vendor specific, the IP vendor is responsible for designing them appropriately in order to meet throughput requirements. Data throughput and processing latency analysis of IEEE 1588 IP core is outlined in Section 5.4.2.1.

On the other hand, there is the data path that forwards Ethernet frames from physical interface to CPU in order to be processed by the PTP stack, the MACsec control plane or other user applications. This path would include MACsec and Ethernet switch IP cores among other hardware modules.

Several issues should be considered to assure that PTP protocol is able to operate correctly, even introducing the MACsec IP core within the data path. The MACsec IP core should accomplish with network bandwidth or line frequency, which is defined for Ethernet networks as the channel capacity and is typically expressed in bits per second. Section 5.4.2.2 analyses data throughput and processing latency of MACsec IP core.

Both paths meet at Linux user level on the execution of the PTP stack; therefore, several issues must be considered to assure that PTP protocol is able to operate correctly, even introducing the MACsec IP core within the data path. Designers must implement the proper MAC and Ethernet Switch IP cores in order to support the required Ethernet bandwidth. As stated in [133], since SV messages in substation process bus consume approximately 5 Mbit/s of the Ethernet bandwidth with a single MU, tens of MUs connected to the same LAN would need Gigabit capacity. Hence, bit-rate of network links between switches in substations need to support bit-rates of 1 Gbit/s or more, while end nodes containing an OC might still operate with Fast Ethernet technology at 100 Mbit/s. Some guidelines to allow Ethernet subsystem to fulfil throughput and latency requirements are exposed in Section 5.4.2.3, in which time calculations are computed for Gigabit Ethernet and must be multiplied by 10 for Fast Ethernet.

5.4.2.1 Data throughput and processing latency analysis of IEEE 1588 IP core

When a PTP event message is detected by the Frame Parser, relevant PTP information in conjunction with the timestamp captured by the TSU must be stored in the Register Interface before the next PTP frame is acquired. Otherwise, the information about the first PTP message would be lost. The processing latency of the IEEE 1588 IP core should be short enough to assure information of two consecutive messages is registered correctly. This latency is the time since the timestamp is captured until all relevant PTP fields of the frame are gathered and registered in the Register Interface.

Apart from timestamps, the PTP fields that must be stored in order to identify the message in the stack are: *messageType*, *sequenceID* and *sourcePortIdentity*. Therefore, at Gigabit Medium Independent Interface (GMII), TSU takes 62 cycles from the end of the *preamble* until the end of the *sequenceID* field[2] to acquire all the information. The minimum latency of the core would be 496 nanoseconds at Gigabit Ethernet.

However, higher latencies could be experienced in the timestamping path if such high throughputs are not achieved by interfaces inside the IEEE 1588 IP core. The maximum latency times allowed not to overwrite PTP information are computed bellow for the cases of two received and two transmit non-stop PTP frames.

At the receiving side, a *Pdelay_req* message could be received immediately after a Sync message, for example. The timestamping information of both messages must be available as soon as possible at processor interface, but particularly the information of *Sync* message must be stored in the Register Interface before capturing the *Pdelay_req*. Supposing that the transmission of the latter is started from a peer port once the transmission of the *Sync* is finished, the maximum latency allowed is the transmission time of the *Sync* plus the Inter Frame Gap (IFG). The worst case would be the transmission of a Synq message directly over Ethernet and encapsulated with MACsec, which is 108 bytes long including the SecTAG and the ICV. At Gigabit Ethernet, the transmission time would be 816 nanoseconds plus 96 nanoseconds for the IFG. At transmission side, the same happens when transmitting a *Pdelay_req* message right after the *Synq*.

Consequently, the maximum allowed latency for the IEEE 1588 IP core is 912 nanoseconds in total. The throughput of the IEEE 1588 IP core is defined as the number of bits to be registered in this latency time, as specified in Equation 5.1. In the Equation, the $timestamp_{length}$ is 64 bits and the $PTPdata_{length}$ is 100 bits, which is the sum of the lengths of *messageType*, *sequenceID* and *sourcePortIdentity* fields.

$$Throughput_{1588} = \frac{timestampg_{length} + PTPdata_{length}}{Latency_{1588}} = \frac{164}{Latency_{1588}} \tag{5.1}$$

As a consequence, the minimum allowed throughput inside IEEE 1588 IP core is in the order of 180 Mbit/s:

$$Throughput_{1588_{min}} = \frac{164}{912 \cdot 10^{-9}} = 0.180 Gbit/s \tag{5.2}$$

[2]The sum of 12 bytes for Destination and Source MAC addresses, 16 bytes of the SecTAG, 2 bytes of the Ethertype and 32 bytes of the PTP header including the *sequenceID* is 62 bytes, which take 62 cycles for transmitting through GMII interface.

Since interfaces between components within the IEEE 1588 IP core are vendor specific, the IP vendor is responsible for designing these interfaces appropriately in order to meet throughput and latency requirements. As an example, with a Register Interface input frequency of 50 Mhz and an input word width of 4 bits, PTP timestamp and data registration would take 820 nanoseconds and the throughput would be 200 Mbit/s.

5.4.2.2 Data throughput and processing latency analysis of MACsec IP Core

In this Section, the MACsec IP core is referred to as the set of hardware modules that implement the SecY entity and its management. The throughput and latency of this MACsec core are extremely important in order to guarantee the selected Ethernet technology bandwidth, because the only requirement for on-the-fly computation of the ICV is to finish the current block computation before the next one is ready.

There are some factors that negatively influence MACsec IP core latency:

- First block processing
- Ethernet IFG assessment
- Detection of short frames
- MAC computation

The processing time of the first block, named L_{first_block}, is the time it takes to receive the Ethernet preamble and a whole block of input data. Since the input data width of an AES block cipher is 128 bits, a latency of 8 and 16 cycles for the preamble and the first block, 192 nanoseconds in total, is introduced when the MACsec core is placed between the MAC and the PHY at GMII. The latency due to the IFG assessment, L_{IFG}, is introduced when two consecutive Ethernet frames are going to be transmitted. Frames forwarded from the common port of the MACsec core to the PHY need more time to be retransmitted because the SecTAG and the ICV might have been added. This additional latency will vary depending on the nature of the previous retransmitted frame. If it is forwarded through the uncontrolled port no latency is introduced. In contrast, when it is forwarded through the controlled port, L_{IFG} can increase up to 192 nanoseconds at GMII in the case that the SecTAG does not include the Secure Channel Identifier (SCI), or 256 nanoseconds if the SCI is implicitily included in it.

Apart from assuring the IFG, the transmit path of the MACsec core also needs to wait in order to check if the Ethernet frame is short. An Ethernet frame

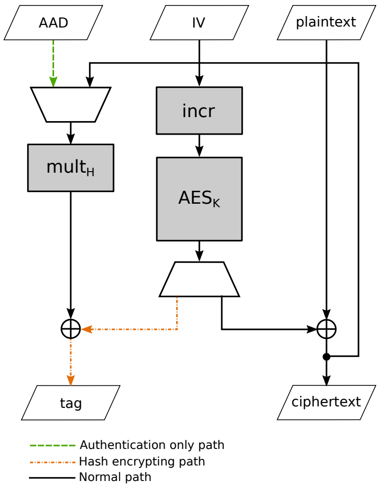
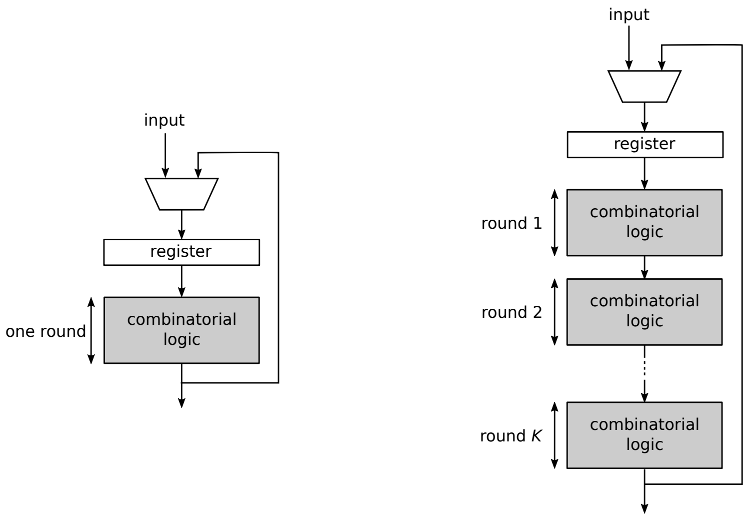


Figure 5.9: General hardware architecture of an AES-GCM crypto core

is considered short when its data length is less than 48 bytes, and so must be indicated in a specific field of the SecTAG. Therefore, before coding the SecTAG, the MACsec core must wait until it receives 60 bytes to start the computation of the first block. Since the latency corresponding to process the first block was already considered, the core should wait for receiving 44 bytes more. The time it takes to receive them is named L_{SL} and it will depend on MACsec core input word width. Considering the worst case of an input word width of 128 bits, up to three additional blocks need to be buffered before starting the MAC computation. In this case, L_{SL} would be 384 nanoseconds.

Finally, the time the MACsec core takes to compute the MAC is referred to as $L_{AES-GCM}$ and depends on the AES-GCM cryptographic algorithm implementation. In order to analyse the latency of the AES-GCM algorithm, a typical hardware architecture is depicted in Figure 5.9 extracted from [134]. AES_K denotes the AES block cipher using the key K and $mult_H$ stands for Galois field multiplication by the hash key H, which is computed as $H = AES(K, 0^{128})$. Hardware implementations usually utilize three different paths: the authentication only path, the hash encrypting path and the normal path.

The authentication-only data, represented as AAD in Figure 5.9, is fed into the multiplication function following the authentication only path. Then, the multiplication function input is switched to the ciphertext coming from the normal path. Also, the Initialization Vector (IV) is fed into the increment function and



(a) Basic iterative (b) Partial loop unrolling

Figure 5.10: General hardware architectures of block ciphers

successive counter values are obtained as outputs, which at the same time fed into the AES block cipher. The first output of the block cipher is exored with the output of the multiplication function, after all the data input has been processed, producing the authentication tag. This process follows the hash encrypting path.

On the one hand, if only authentication service must be provided, besides the computation of the binary Galois field multiplication over each block of authentication data, a single iteration of the AES block cipher is required. While the Galois field multiplication is especially suitable for hardware implementations and latency times of one clock cycle can be achieved with parallel techniques, the AES block cipher needs 10 or 14 cipher rounds, depending on the key size [134].

On the other hand, when both authentication and confidentiality are desired, one iteration of AES block cipher is performed each time a block of ciphertext is going to be generated. As a consequence, the latency of the AES-GCM algorithm in hardware will be determined by the architecture of the AES block cipher. Apart from pipelined architectures, designers can choose between basic iterative architecture or a more complex one using loop unrolling techniques. Figure 5.10(a) represents the basic architecture, where one round of the cipher is implemented as combinational logic and it is used iteratively for the number of rounds. The latency of this architecture, named $L_{iterative}$, is defined in Equation 5.3, where N_r is the number of rounds and T_{CLK} is the clock period.

$$L_{iterative} = N_r \cdot T_{CLK} \tag{5.3}$$

In contrast, architectures with partial loop unrolling have K rounds implemented as combinatorial logic, as it can be seen in Figure 5.10(b). Thus, the number of clock cycles necessary to encrypt/decrypt a block decreases by a factor of K. Nevertheless, the minimum clock period also increases by a factor slightly smaller than K and, consequently, latency times are hardly improved. In this case, the latency is L_{loop} as defined in Equation 5.4.

$$L_{loop} = \frac{N_r \cdot K' \cdot T_{CLK}}{K} = \frac{K'}{K} \cdot L_{iterative} \tag{5.4}$$

As an example, values of number of cycles per encryption, maximum frequencies and resulting latency times are listed in Table 5.2 for different values of K. Only with $K = 2$ and $K = 3$, the overall latency was improved in the range of few nanoseconds, as demonstrated in [53]. In this experiment, the effect of changing the K factor for a fast and open source AES block cipher implemented on a Xilinx Virtex 5 XC5VFX70T FPGA was analysed.

Tabla 5.2: R. Usselmann AES-128 hardware implementation results

	$K = 1$	$K = 2$	$K = 3$	$K = 4$	$K = 5$
Cycles per encryption	12	7	6	5	4
Max. frequency (MHz)	171.615	105.530	85.970	57.977	41.892
Latency (ns)	69.92	65.80	69.79	86.24	95.48
Max. throughput (Gbit/s)	1.830	1.945	1.834	1.484	1.341

The basic iterative implementation of the AES block cipher will be considered to estimate the latency of the MACsec IP core. Concretely, the latency of the AES-GCM, named $L_{AES-GCM}$, is computed as in Equation 5.5, where T_{MACsec} is the MACsec clock period.

$$L_{AES-GCM} = L_{iterative} = N_r \cdot T_{MACsec} \tag{5.5}$$

Cryptographic algorithms that are implemented in hardware are also characterized using throughput, which is defined as the number of bits encrypted/decrypted in a unit of time and is related to latency by Equation 5.6 [135].

$$Throughput_{cipher} = \frac{block\_size \cdot number\_of\_blocks\_processed\_simultaneously}{latency_{cipher}} \tag{5.6}$$

In the proposed SoC architecture, MACsec uses the AES-GCM block cipher to compute the ciphertext and the authentication tag of Ethernet frames as they are traversing the [G]MII interface. In contrast to file system encryption where large amounts of data must be encrypted or decrypted, data is captured in 8 bit nibbles at GMII interface for Gigabit Ethernet. Therefore, there is no sense in computing several blocks in parallel using pipelining techniques. Consequently, only one block is processed each time with this architecture and the throughput is calculated by Equation 5.7.

$$Throughput_{AES-GCM} = \frac{block\_size}{L_{AES-GCM}} = \frac{128}{N_r \cdot T_{MACsec}} \tag{5.7}$$

The minimum frequency of the MACsec IP core is the one that assures the throughput of Gigabit Ethernet, as stated in Equations 5.8 and 5.9, which results in 109.375 MHz when AES-256 ($N_r = 14$) and 78.125 MHz when AES-128 ($N_r = 10$).

$$Throughput_{AES-GCM} = \frac{128 \cdot f_{MACsec_{min}}}{N_r} = 1Gbit/s \tag{5.8}$$

$$f_{MACsec_{min}} = \frac{10^9 \cdot N_r}{128} \tag{5.9}$$

Similarly, the maximum latency of the AES-GCM algorithm can be calculated from the minimum frequency as stated in Equation 5.10, which results in 128 nanoseconds independently of the key size.

$$L_{AES-GCM_{max}} = \frac{N_r}{f_{MACsec_{min}}} = \frac{N_r \cdot 128}{N_r \cdot 10^9} = 128ns \tag{5.10}$$

Finally, the total latency of the MACsec IP Core is defined in Equation 5.11, with a maximum value of 960 nanoseconds.

$$Latency_{MACsec} = Lfirst\_block + L_{IFG} + L_{SL} + L_{AES-GCM} \tag{5.11}$$

5.4.2.3 Data throughput and processing latency analysis of the Ethernet subsystem

The data throughput and the overall processing latency of the system will be highly influenced by each hardware component in the Ethernet subsystem within the SoC architecture and by the software framework employed in the reception and transmission process. The impact of this Ethernet subsystem on system performance is analysed below for the single port and the multiple ports architectures, as well as for the particular case of an HSR node.

TimeWardenSoC architecture for an OC with a single port

In the simplest case, with only one Ethernet port, the Ethernet subsystem was simplified in Figure 5.5, where it was represented as a compact MAC IP core. Nevertheless, it is generally composed by a proper MAC unit, a transmission and reception buffer and the Direct Memory Access (DMA) engine.

The purpose of the DMA controller is to access to system memory or another device memory directly from a hardware component without requiring CPU intervention. Thus, the CPU is able to continue with its work and achieve good performance. A DMA controller is commonly used by Ethernet Controllers in order to read from system memory egress frames to be transmitted to network and write to system memory ingress frames received through the [G]MII interface.

Apart from hardware processing of Ethernet frames, software framework for frame reception and transmission also affects latency and throughput in Ethernet Subsystems. Most Linux based systems currently employ the New API (NAPI) as an alternative to pure interrupt driven technique, which presented the problem of receive-livelock under high traffic load [136]. In this state, CPU resources were spent on interrupt handling causing received frames not being processed and even dropped due to buffer overflows. NAPI was included in Linux kernel version 2.5.7 in order to reduce interrupt rate in case of high traffic load by processing multiple frames upon reception of an interrupt. Pure interrupt driven frameworks offer the lowest latencies between the reception of a frame and its processing at low loads, but do not work well under high load conditions. In contrast, with full CPU utilization, NAPI decreases the interruption rate and increases the maximum throughput at the cost of latency, but no frames are dropped. At low traffic loads NAPI offers the same latency than a pure interrupt driven mechanism. Readers are encourage to learn more about NAPI drivers and Ethernet frames processing in Linux based hosts in [136].

Bearing in mind the mode of operation of NAPI, in order to analyse the latency of the Ethernet subsystem, reception and transmission paths have been distinguished. On the one hand, a received frame is firstly buffered in the MAC unit and transferred via DMA to the system memory. Once the DMA transfer is finished, the DMA unit issues a receive interrupt so as to notify the CPU, which invokes the associated interrupt service routine defined in the network driver. Supposing that NAPI is supported by the driver, further processing by upper layers will wait until next time slot allocated to that device, when input frame is delivered to each protocol handler associated to it. Therefore, the latency of the reception path is the time elapsed between the reception of the frame and its processing by the corresponding protocol handler. This value is the sum represented in Equation 5.12, where L_{MAC} is the latency of frame buffering in the MAC unit, L_{DMA} is the latency of the DMA transfer, $L_{interrupt}$ is the receive interrupt latency and L_{input_queue} is the kernel latency due to NAPI scheduling.

$$Latency_{Rx} = L_{MAC} + L_{DMA} + L_{interrupt} + L_{input\_queue} \tag{5.12}$$

On the other hand, the latency experienced by a transmitted frame is in Equation 5.13. In this case, there are no interrupt handlers, but latency due to kernel processing for a determined egress frame will depend on the queueing discipline of the output queue.

$$Latency_{Tx} = L_{output\_queue} + L_{DMA} + L_{MAC} \tag{5.13}$$

The MAC latency, named L_{MAC}, depend on packet size ($frame\_length$), data width ($data\_width$) and frequency (f) on [G]MII interface, as specified in Equation 5.14. The DMA latency, named L_{DMA}, is in addition dependant on DMA unit implementation as defined in Equations 5.15 and 5.16. As an example, the latency introduced by the Advanced eXtensible Interface (AXI) DMA IP core from Xilinx due to the time elapsed between the input and output interfaces is estimated to be around 49 clock cycles in the reception path and 44 clock cycles in the transmission path [137]. Hence, L_{DMA_unit} could be in the order of 490 and 440 nanoseconds at 100 MHz. For a typical *Sync* message, $frame\_lentgh$ would be 560 bits and L_{MAC} would result in 560 nanoseconds. Supposing a DMA interface of 64 bits and 100 MHz, L_{DMA} would be around 500-600 nanoseconds.

$$L_{MAC} = \frac{frame\_length}{data\_width_{GMII} \cdot f_{GMII}} = \frac{frame\_length}{8 \cdot 125 \cdot 10^6} \tag{5.14}$$

$$L_{DMA} = L_{DMA\_unit} + L_{DMA\_transfer} \tag{5.15}$$

$$L_{DMA} = \frac{clock\_cycles}{f_{DMA}} + \frac{frame\_length}{data\_width_{DMA} \cdot f_{DMA}} \tag{5.16}$$

The interrupt latency is the time between the interrupt generation and the execution of the first instruction of the interrupt handler or service routine, whose value depends on different factors such as the interrupt controller implementation, priority of the interrupt and the operating system, among others. Linux embedded systems usually experience interrupt latency times in the order of few microseconds. For instance, for Zynq-7000 based systems, interrupt latencies around 5 microseconds under normal execution and 10 microseconds while invoking applications were achieved in [138].

The latency between the kernel and user space processing of ingress frames, as well as latency induced by kernel scheduling of egress grames, could be measured using the SO_TIMESTAMP option of sockets. Depending on where timestamps are taken by the network driver, this measured latency could also include the latency introduced by hardware components if hardware timestamping is supported by the driver. Also, a hardware timer could be used to measure software delays, as explained in [139], where experiments with different packet size were performed to evaluate frame release latencies when utilizing a Zynq-7000 SoC system. Authors admitted software latencies below 2 microseconds when transmitting packets even under traffic load. Therefore, values of around 1-2 microseconds for L_{output_queue} and L_{input_queue} are estimated for embedded systems.

The total latency experienced by transmitted *Sync* messages from PTP stack running on user space to an Ethernet port in such a SoC architecture, is estimated to be in de order of 3 microseconds. In the case of a received *Sync* message, also the interrupt latency has to be taken into account; hence, latency times could easily increase above 10 microseconds with CPU load.

TimeWardenSoC architecture for a HC with multiple ports

If a multiple port end point is designed, as represented in Figure 5.6, the latency introduced by the Ethernet Switch IP core must also be considered. This latency will vary depending on the forwarding strategy. While store-and-forward switches forward the frame once the whole packet has been received, a cut-through switch starts transmitting the frame immediately after knowing the destination address and the protocol, which are fields placed in the frame header. As a consequence, the forwarding latency in cut-through architectures is independent of the frame length and theoretical values around 200 nanoseconds can be achieved. Otherwise, in store-and-forward architectures, the introduced latency is one order of magnitude greater and depends on the frame length, ranging from 6 to 14 microseconds in typical Ethernet switches.

TimeWardenSoC architecture for an HSR node

In the particular case of HSR switch IP core of Figure 5.7, the latency time to be considered is the forwarding time between a redundant port and the interlink, which is also dependant on the frame length because store-and-forward mode of operation is mandatory in the interlink. In [140], latency times in the order of 2 and 10 microseconds are experienced by frames with payload size of 256 and 1280 bytes respectively.

5.4.3 Memory buffers size and silicon resources estimation

5.4.3.1 IEEE 1588 IP core register dimensioning

Typically, in IEEE 1588 IP cores used to provide PTP hardware support, there are two memory spaces where timestamps are stored: the Register Interface in hardware and a buffer implemented in software driver using kernel space memory. Normally, the IEEE 1588 driver operate in an interrupt driven mode, where the IEEE 1588 IP core generates a hardware interrupt to notify the CPU that a timestamp has been captured. An associated interrupt service routine, defined in the IEEE 1588 driver, is responsible for copying the contents of the registers in the IEEE 1588 core into the timestamping queue implemented in software, as a linked list for example.

The minimum number of timestamps that the Register Interface must be able to keep depends on the periodicity of the PTP messages and the time that the driver takes to retrieve them from hardware. In fact, if only the information of a single PTP frame and its timestamp were stored in the Register Interface, there would be a real chance of being overwritten upon arrival of the next PTP frame before being retrieved from software.

If a *Pdelay_req* message were received right after a *Sync* message, only 656 ns after the *Sync* message had been captured[3], a new timestamp should be stored in the Register Interface. Considering an average interrupt latency of 5 microseconds, as it has been commented above for experiments made in [138], if only one register were implemented in the Register Interface, *Sync* timestamp would be highly likely overwritten before having retrieved it by the interrupt service routine.

[3]This is the transmission time of a Sync message at Gigabit Ethernet plus the IFG.

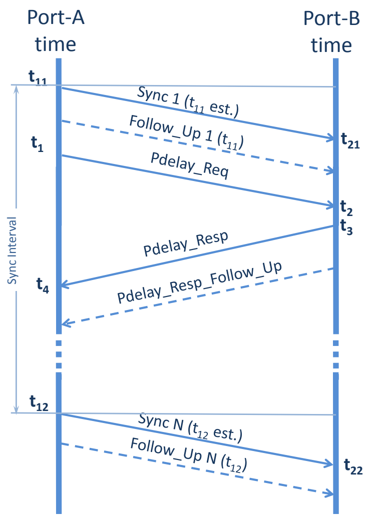


Figure 5.11: Maximum number of PTP messages exchanged in P2P mode

Therefore, register dimensioning of the memory space contained in the IEEE 1588 IP core, directly depends on the time spent by the CPU to handle the interrupt and the number of timestamps to be captured in this interval of time. The maximum number of messages that the core should be able to timestamp until the CPU handles the interrupt, will depend on the propagation delay mechanism configured in the PTP stack. The worst case is the P2P mode of operation, where up to 5 messages could be exchanged within each Sync interval as shown in Figure 5.11, or even within each peer delay interval if it were shorter. In IEEE C37.238-2011 standard for PTP power profile, *logSyncInterval* and *logMinPdelayReqInterval* attributes are specified to be 0 and, accordingly, the minimum time interval between two consecutive *Sync* or *Peer_Delay* messages would last one second. Thus, up to three timestamps must be captured in every second: t_{11}, t_1 and t_4 in Port A; t_{21}, t_2 and t_3 in Port B, as represented in Figure 5.11.

As a consequence, the memory space within the Register Interface must be capable of storing a minimum of three timestamps, together with PTP message information. As stated in Section 5.4.2.1, this information would at least consist of *messageType*, *sequenceID* and *sourcePortIdentity*. In total, for three PTP event messages, 492 bits of information should be buffered.

Although the dimensioning of the Register Interface and the size of the software input queue do not necessarily be the same, the software buffer should be capable of storing as many timestamps as the Register Interface or more. With too large latencies in the data path, there is a minimal risk of capturing a new *Sync* message and deliver the timestamping information to the driver before previous *Sync* message reaches the PTP stack. In this case, having space in system memory for more than three timestamps could be interesting. Nevertheless, this only could happen if the data path latency, including the received frame interrupt latency and frame processing by the Linux networking stack, were higher than the Sync or Peer Delay interval plus the latency of the timestamping path.

In this sense, modern Linux kernels implement several techniques to avoid large interrupt latencies and queues congestion, as it has been previously described the case of NAPI interface, which maintains low processing latencies under normal traffic conditions. Even when CPU is overloaded with more than 2000 frames per second, latencies of 2,300 microseconds were observed in experiments made in [141], where a traffic generator was connected to a Linux host that forwarded packets back to the generator in order to measure latencies at different loads. This large latency under overloaded conditions is still far from the one second time interval of *Sync* and *Pdelay_req* messages. Accordingly, an input buffer of three nodes managed by the driver would be large enough to store timing information of PTP event messages. However, since memory space can be allocated and deallocated dynamically, supporting greater buffer sizes should not be an inconvenient when designing IEEE 1588 drivers.

5.4.3.2 Area estimation

In this Section, the resource utilization of the proposed SoC architecture is estimated taking into account the implementation results included in the datasheets and brochures available from IP vendors when implementing them independently in Xilinx Zynq FPGA [142]. Consequently, Tables 5.3, 5.4 and 5.5 respectively summarize the area cost of an end node, an intermediate node and an HSR node with MACsec support. The intermediate node includes an Ethernet Switch IP core with four ports, providing three external ports and an internal port to the system.

From all IP cores presented in Chapter 4, the following ones were considered for estimating the area cost:

- IEEE 1588 IP core: PreciseTimeBasic (PTB) from SoC-e [97]
- Eth. Switch IP core: Unmanaged Ethernet Switch (UES) from SoC-e [100]
- HSR Switch IP core: HSR/PRP Switch (HPS) from SoC-e [101]

- MACsec IP core: MACSEC for 10 Gbit Ethernet from Algotronix [127]

The Zynq XC7Z020 FPGA available resources are: 106,400 Slice Registers, 53,200 Slice LUTs, 140 RAMB36, 280 RAMB18 and 220 DSP48E1 blocks. Bearing in mind this values, the percentage of utilization for each configuration is included in the last column of the Tables. Because all the listed percentages are below 100%, it can be expected that implementation of any of the three configurations would be viable in modern FPGAs. However, the high utilization of Slice LUTs with percentages around 90% in multiport configurations could drastically worsen system performance due to timing constraints. The less free space is in the FPGA, the smaller are the values of maximum frequency and, accordingly, the required latency and throughput of hardware components could be compromised.

Furthermore, the resource utilization of the MAC IP was not considered in the Tables because Zynq FPGA already has some hard Ethernet Controllers. As a consequence, designers could save reconfigurable logic resources by instantiating one of them when implementing these architectures. Otherwise, the area used by common soft MAC IP cores should be also taken into account. As an example, the minimum utilized resources by the AXI Ethernet Subsystem IP core from Xilinx [143] with GMII interface and without VLAN support are: 5,316 Slice Registers, 3,808 Slice LUTs and 5 RAMB36, which even more increases the chance to negatively affect system performance.

With the aim of overcoming the problem of limited area, a different FPGA device with more resources might be chosen. For example, instead of the Zynq XC7Z020 device, another superior FPGA within Zynq-7000 family could be employed, which contains the programmable logic equivalent to that of the Kintex-7 family [144].

Also, IP vendors usually offer compact versions of the cores with less memory space or even without some optional features that can be omitted depending on the application. In the case of the Algotronix MACsec IP core, for instance, the resource utilization can be minimized by reducing the number of implemented secure channels and virtual SecYs [4].

In Chapter 7, the resource utilization is further analysed for the implemented architectures that were used in the validation experiments.

[4]Implementation results of the Algotronix MACsec IP core shown in Tables 5.4 and 5.5 were obtained for 32 secure channels and 16 SecYs.

Tabla 5.3: Area cost of an IEEE 1588 OC with MACsec support in Zynq XC7Z020

Resources	IEEE1588 IP	MACsec IP	Total	Utilization
Slice Registers	1,333	14,618	15,951	14.99%
Slice LUTs	979	13,980	14959	28.12%
RAMB36	0	0	0	0%
RAMB18	0	32.5	32.5	11.61%
DSP48E1 blocks	0	0	0	0%

Tabla 5.4: Area cost of an IEEE 1588 HC with MACsec support in Zynq XC7Z020

Resources	IEEE1588 IP	Eth. Switch IP	MACsec IP	Total	Utilization
Slice Registers	1,333	8,321	14,618 (x3)	53,508	50.29%
Slice LUTs	979	6,897	13,980 (x3)	49,816	93.64%
RAMB36	0	31	0 (x3)	31	22.14%
RAMB18	0	8	32.5 (x3)	105.5	37.68%
DSP48E1 blocks	0	4	0 (x3)	4	1,82%

Tabla 5.5: Area cost of an IEEE 1588 HSR node with MACsec support in Zynq XC7Z020

Resources	IEEE1588 IP	HSR Switch IP	MACsec IP	Total	Utilization
Slice Registers	1,333	17,115	14,618 (x2)	47,684	44.82%
Slice LUTs	979	18,356	13,980 (x2)	47,295	88.90%
RAMB36	0	32	0 (x2)	32	22.86%
RAMB18	0	30	32.5 (x2)	95	33.93%
DSP48E1 blocks	0	16	0 (x2)	16	7.27%

5.5 Conclusions

An hybrid security solution consisting of a new PTP security field and an external security mechanism like MACsec, in combination with redundancy and monitoring techniques, seems to be the future in protecting distributed synchronized systems. Concretely, the contribution of this thesis is focused on the utilization of MACsec for providing hop-by-hop group authentication of PTP messages.

In this sense, the main contributions of the thesis are presented in this Chapter. On the one hand, a new key management scheme for distributing MACsec keys in substation networks is proposed. On the other hand, a secure PTP SoC architecture integrating MACsec hardware units is presented for the first time. The mentioned key scheme, named TimeWardenKey, relies on EAPOL-Key frames to distribute group CAKs and, consequently, modifications on current available IEEE 802.1X stacks are required. On the contrary, the integration of MACsec IP cores in PTP SoC architectures is expected to be an immediate step towards a full protected PTP system.

After analysing throughput and latency analysis of the TimeWardenSoC architecture proposed in this Chapter, the latency introduced by the MACsec IP core in the order of one microsecond is expected not to have a great influence on overall latency if compared with latencies introduced by the Ethernet subsystem. Next Chapters describe the experiments performed to deeply analyse implementations of the proposed SoC architectures and to validate PTP protocol performance when using them.

Chapter 6

PTP SoC architecture validation

6.1 Introduction

As stated in Chapter 4, with the increasing combination of industrial protocols in SASs, FPGAs and reconfigurable devices in general are acquiring more and more importance because of their flexibility to integrate new specific hardware and software into sophisticated embedded designs. In this sense, the technology offered by reconfigurable logic is moving forward to the next level: cost-affordable SoC devices, where all processing units and peripherals together with reconfigurable logic are integrated into a single chip.

Zynq device by Xilinx is a good example, as it allows developing powerful application specific embedded systems. Therefore, it was selected as the platform to validate PTP protocol performance using different SoC architectures. Firstly, some basic experiments were performed in the laboratory as depicted in Section 6.2. After that, the results of testing the most efficient designs on an interoperability plugfest are detailed in Section 6.3. Finally, Section 6.4 explains some tests made over real industrial networks in a manufacturing plant. The conclusions of this Chapter are discussed in Section 6.5.

6.2 Laboratory experimental setup

This Section outlines the first experiments performed in the laboratory based on the Zedboard developement board containing the Xilinx Zynq SoC device. Three different topologies over two setup configurations were tested employing laboratory equipment. In Section 6.2.1, the used material and the test setup configuration, as well as the three compared topologies, are described. The experimental results for each topology-setup are detailed in Section 6.2.2.

6.2.1 Setup description

Xilinx Zynq-7000 family of All Programmable SoC (AP SoC) devices combines a Processing System (PS) section containing an ARM dual core Cortex-A9 processor in conjunction with on-chip memory and a wide range of powerful peripherals (Gigabit Ethernet controllers, memory controllers, CAN bus, etc.) and a Programmable Logic (PL) section with last-generation 28nm reconfigurable logic [142]. The general block diagram of a Zynq-7000 AP SoC device is represented in Figure 6.1. Every device belonging to the Zynq-7000 family contains the same PS section, but the PL and the IO resources vary between devices to adapt to different application requirements.

Bearing in mind the general PTP SoC architecture introduced in Section 4.2 and using some of the presented software and hardware modules to provide PTP support, several SoC topologies were designed for the Zynq SoC device. The aim was to explore the benefits of using such a new reconfigurable logic technology and measure the achieved performance in terms of synchronization accuracy. All the studied topologies have some parts in common. They all include hardware PTP support to achieve accuracies below the microsecond. Also, an open source PTP stack is compiled to be executed by a Linux based OS running on the ARM. Additionally, some kernel drivers to exchange information from hardware registers to user space applications and viceversa are required.

In the experiments, the Zedboard development board from Avnet and Digilent was used [145]. This low-cost board contains a Zynq XC7Z020-CL484 FPGA and all the necessary memory units and peripherals to create a complete Linux system. It also provides a number of connectors to extend the implementation possibilities. The block diagram of the Zedboard development board is copied from [145] in Figure 6.2. In order to extend the number of Ethernet ports provided by the ZedBoard in multiport topologies, a TB-FMCL-GLAN board from Inrevium [146] is attached to it through the FPGA Mezzanine Card (FMC) connector.

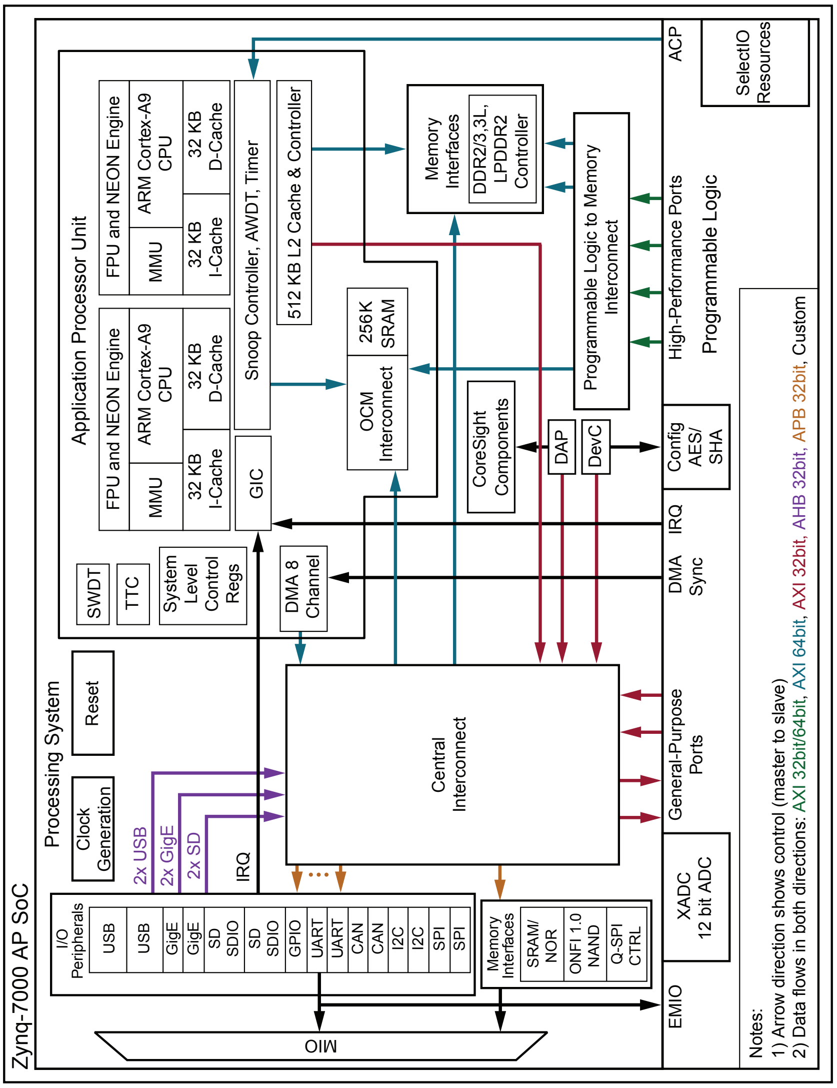


Figure 6.1: Zynq platform block diagram

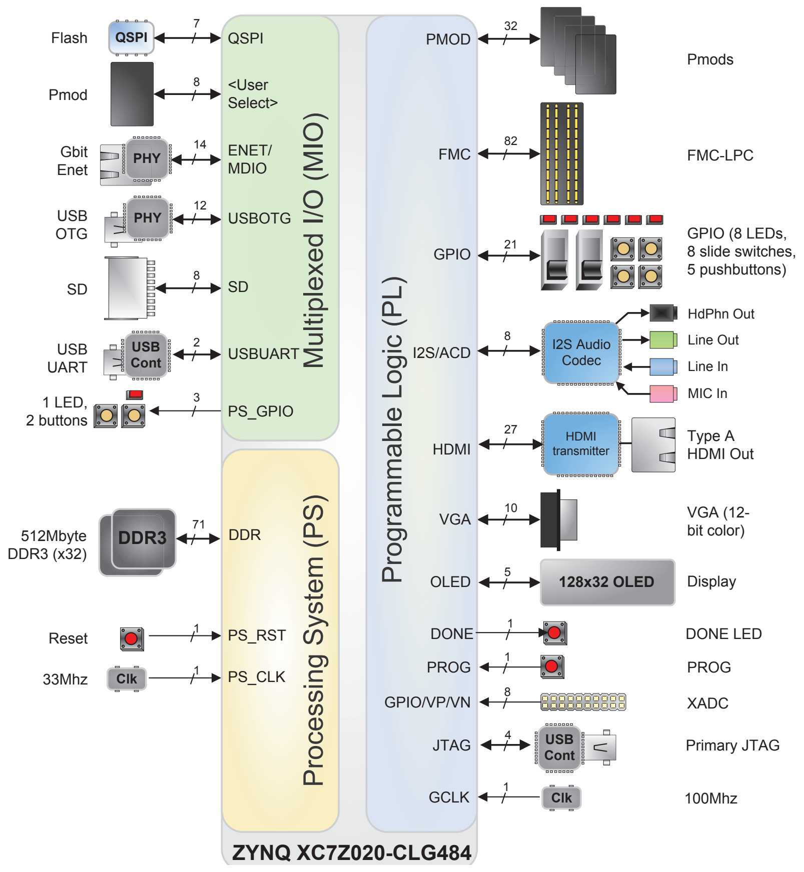


Figure 6.2: Zedboard development board block diagram

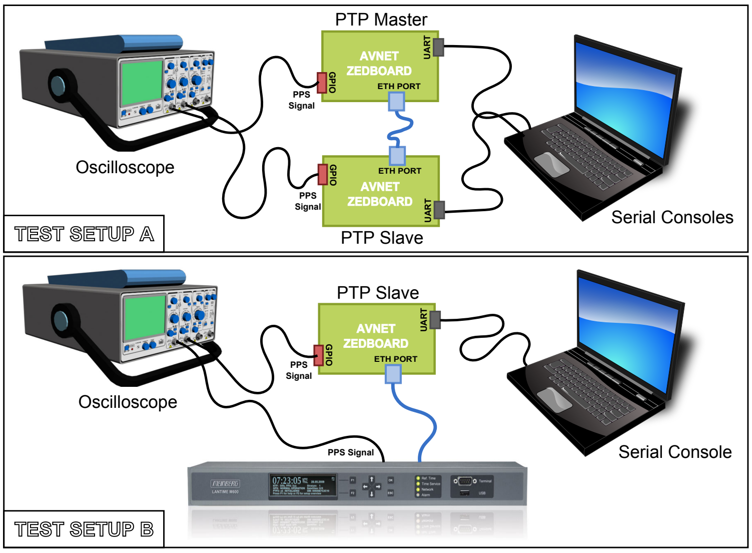


Figure 6.3: Diagrams of the laboratory experimental setups

The schemas of the two used setups are shown in Figure 6.3. The first setup, named Test Setup A, consists of two ZedBoard Zynq-7000 All Programmable SoC connected through an Ethernet link: one of the boards acts as a PTP master and the other one as a slave. In contrast, in the second setup, Test Setup B, a professional PTP master was used: the Meinberg LANTIME M600/MRS/PTP reference time source. In the setups, also a MSO7104A oscilloscope from Agilent Technologies and a laptop were used to see PPS signals generated by devices and messages on the serial console respectively. The pictures of both built setups can be seen in Figure 6.4.

The Linux OS booting files are stored in a SD card, which is connected through a 9-pin standard SD connector placed in the Zedboard. As it can be seen in Figure 6.5, the SD card was partiotioned into two partitions. The first partition contained the booting files, while the second one had stored the root filesystem. The embedded kernel compilation was executed on a Linux virtual machine created with VirtualBox, where Xilinx Vivado tools where also installed. The Xilinx Software Development Kit (SDK) contains the cross-compilation toolchains needed to build Linux-based software that runs in Xilinx processors. Particularly, the Xilinx tools provide the arm-xilinx-linux-gnueabi for the Zynq device, and so must be set in the CROSS_COMPILE environment variable in order to be invoked when building software for being run on it, such as the kernel drivers.

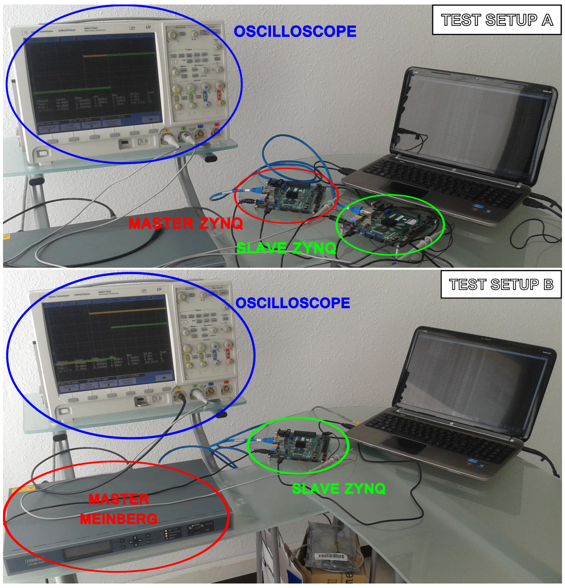


Figure 6.4: Pictures of the laboratory experimental setups

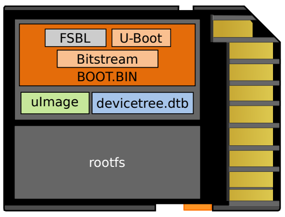


Figure 6.5: Block diagram of the SD card as the Linux boot medium

The implemented architectures for each topology are explained below.

Case 1: Zynq PS GMAC IEEE 1588-aware only

The block diagram of the first topology tested in the experiments is represented in Figure 6.6. This topology is the simplest one, without no specific logic in the PL section to support IEEE 1588 protocol. The integrated Gigabit Media Access Controller (GMAC) of the Zynq device already includes IEEE 1588 timestamping and timing capabilities, as shown in Figure 6.7 taken from [142]. There is a TSU

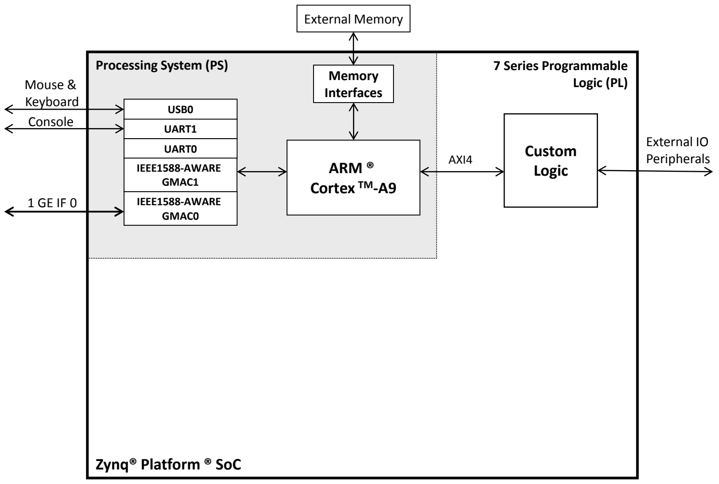


Figure 6.6: Case 1 topology block diagram

block within the controller that captures the time at which PTP event messages reach and leave the controller. Moreover, a timer to make timestamps and several registers to store PTP message information are contained in the TSU.

However, the IEEE 1588 support of the Zynq GMAC is limited by software addressing of TSU timer and registers, because they are only accessible from the PS portion. For example, since the controller cannot generate an interrupt upon the counter reaching a determined value, the timer in the TSU cannot be used to directly schedule user-events in hardware. As a consequence, if these hardware events are managed by software, the accuracy will be seriously deteriorated due to time variances in counter read access and software workload. Similarly, precise PPS signals commonly used for monitoring PTP synchronization accuracy cannot be generated.

The hardware-software architecture implemented for this topology is depicted in Figure 6.8. In order to demonstrate the limitations presented by this topology, a simple hardware design consisting of the processing unit and several Input/Output (IO) peripherals, such a s the GMAC and the UART, was implemented using Xilinx Vivado 2013.4 design tool. No bitstream[1] is downloaded to the Zynq device. Only if additional custom logic were instantiated, the PL section should also have to be programmed through a compiled bitstream.

[1]The bitstream is called to the file that is downloaded to the FPGA in order to configure connections between logic components in the PL section.

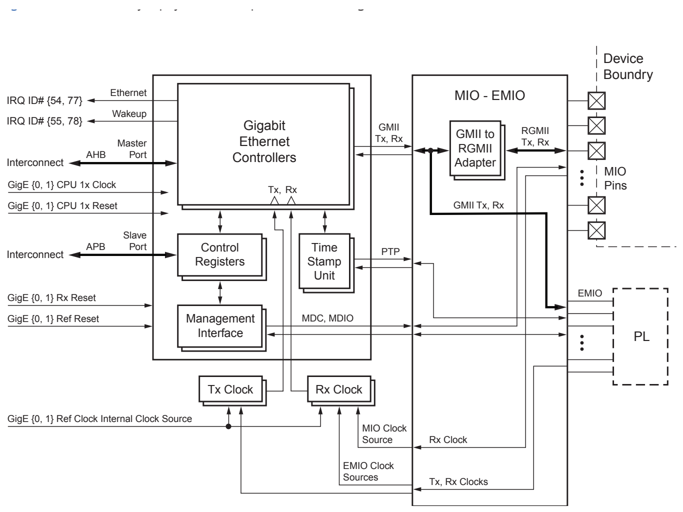


Figure 6.7: Zynq PS IEEE 1588-aware GMAC block diagram

Regarding the software, the ARM processor in the Zynq device holds the Linaro Ubuntu ARM Linux distribution based on kernel version 3.13.0, which runs the Linux PTP Project stack [147] described in Section 4.2.4.2. In order to boot the Linux OS in the ARM, the SD card was partitioned with two partitions [148], as it has been previously represented in Figure 6.5. The first partition contained the bootloader, the devicetree and the kernel image. The second partition had stored the root filesystem.

The bootloader is the BOOT.BIN file which was built with Xilinx SDK from the First Stage Boot Loader (FSBL) and the Universal Boot Loader (u-boot.elf) specific for the Zynq platform. In this case, as mentioned above, no bitstream is included in this step, since no logic had to be programmed. The devicetree was also built with Xilinx SDK and contains essential information about register addresses and interrupts of each peripheral connected to the system bus. The uImage file is the kernel image resulted from compiling the kernel sources in the Linux virtual machine. Some configuration options must be enabled in the kernel '.config' file before compiling the sources such as the hardware timestamping for the Xilinx PS GMAC and the PTP clock support [147, 149].

After compiling the Linux PTP project stack on each Zedboard, one is configured to act as the master and the other one as the slave executing the 'ptp4l' command with the corresponding parameters as command line arguments. In the slave, the Zynq PS GMAC PTP timer appears in the kernel tree as '/dev/ptp0', which is

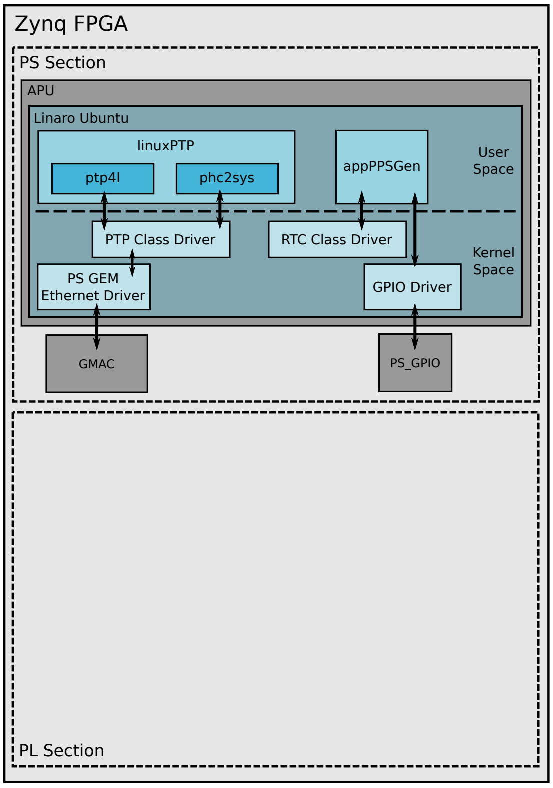


Figure 6.8: Hardware-software architecture for Case 1 topology

also synchronized to the master reference time using the 'ptp4l' application. At the same time, in both master and slave, the system clock is synchronized to '/dev/ptp0' using the 'phc2sys' application, which takes advantage of the PTP Class Driver within the Linux PHC infrastructure [150].

Since the Zedboard has no battery-backed RTC in hardware, by configuring the kernel to support the 'Test driver/device' option within the 'RTC Class' driver [151, 152], two software RTCs appear in the kernel tree as '/dev/rtc0' and '/dev/rtc1'. These software clocks get the time from the system clock, so a poor PPS signal can be generated via software by reading them using the 'ioctl' interface. Concretely, in this case it was generated from a user space application using the Linux 'RTC Class' drivers and the 'ioctl' interface to detect changes on the '/dev/rtc0' timer value using polling techniques. As a result, the state of some PS

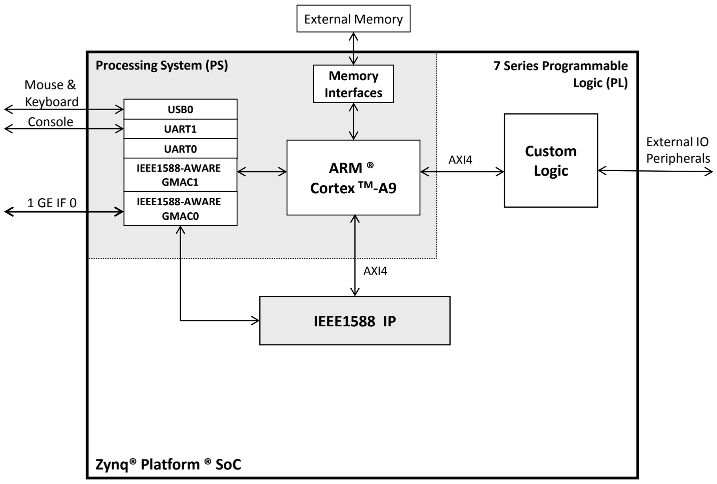


Figure 6.9: Case 2 topology block diagram

General Purpose Input Output (GPIO) ports, which were previously configured as outputs, was changed every second through the 'sysfs' Linux interface to be employed as PPS outputs [153, 154].

Case 2: Zynq PS GMAC IEEE 1588-aware combined with a IEEE 1588 IP on PL

With the aim of overcoming the limitations prompted by the utilization of the Zynq GMAC IEEE 1588 support, it is necessary to implement additional logic on the PL section. Figure 6.9 shows the block diagram of a design architecture that combines PS IEEE 1588-aware GMAC with an IEEE 1588 IP Core. Alike previous architecture, an IEEE 1588 IP core was instantiated in the reconfigurable portion of the Zynq device. Concretely, the Precise Time Basic Zynq Edition from SoC-e [155] was used in this case, which has been described in Section 4.2.5.

This IP takes benefit from the IEEE 1588 logic implemented on the PS GMAC and adds support for synchronized signal generation by hardware. Timestamps are still taken by the GMAC and passed through dedicated signals to the PL section. The IEEE 1588 IP core is responsible for interfacing with the PTP stack running on the ARM through the AXI bus, in order to send to it the timing information of PTP frames. In addition, it also maintains its PTP hardware clock synchronized and generates a precise PPS output that is routed to a PL

GPIO port. Hence, in order to implement this architecture, the PL section of the FPGA also had to be programmed.

Again, the SD card was partitioned with two partitions following the block diagram of Figure 6.5, but the bitstream was also included in the boot image creating process. The Xilinx ISE Design Suite 14.4 was the set of implementation tools used to build the Zedboard booting files. The hardware platform and the bitstream file was created using the Xilinx Embedded Development Kit (EDK) tool.

The general hardware-software architecture of this topology is represented in Figure 6.10. The same Linux kernel that in the previous Subsection was booted in the Zynq. However, in this case, instead of the Linux PTP Project stack, a modified 'PTPd' open source protocol stack was utilized. Moreover, in addition to the kernel and the PTP Stack, new software modules, named '1588 Drivers', have been included. These drivers consist of kernel code responsible for accessing the hardware within the IEEE 1588 IP core. Hence, the RTC and the TSU components of the IEEE 1588 IP core, called 1588Timer and 1588Capture in Figure 6.10, need drivers to allow user space applications to access to hardware registers that contain the configuration parameters and other revealing information. The sources of both drivers and 'PTPd' stack were provided by SoC-e in conjunction with the IP core.

The '1588 Drivers' for the IEEE 1588 IP core were cross-compiled in a Linux host using the cross-compilation tool provided by Xilinx SDK 201.4 and the embedded kernel sources. The resulting modules were loaded through the 'insmod' command before executing the PTP stack. 'PTPd' stack was directly compiled in the target and executed from command line passing through the corresponding configuration file name as an argument. This configuration file sets different PTP configuration options like master or slave, PTP transport type and propagation delay measurement mechanism among others.

Case 3: Zynq PS GMAC IEEE 1588-aware combined with a IEEE 1588 IP and Ethernet switch IP on PL

Typically, IEDs in SASs need two or more Ethernet ports and, consequently, hardware switching among all or many of its ports is sometimes desirable. This switching functionality can be implemented using Ethernet switch IPs that can be placed on the PL section of the platform. Figure 6.11 represents a Zynq-based architecture using an Ethernet Switch IP with three ports. Two of these ports are external to the reconfigurable device and the remaining one is internally connected to PS GMAC. Only the traffic addressed to the device MAC address

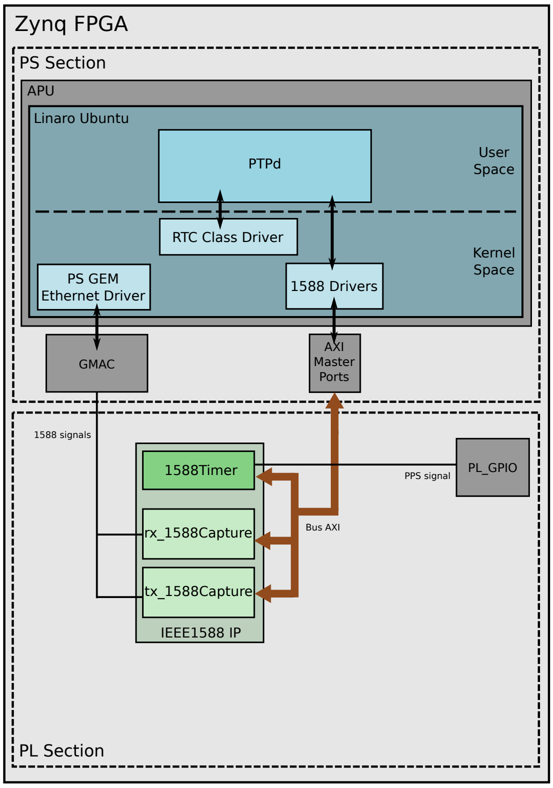


Figure 6.10: Hardware-software architecture for Case 2 topology

is derived to the switch port attached to the PS GMAC.

The Ethernet Switch IP used in the experiments was the Unmanaged Ethernet Switch (UES) IP core from SoC-e described in Section 4.2.5, which supports IEEE 1588 TC functionalities. This IEEE 1588 support allows to perform the peer delay mechanism, as explained in Section 3.2.3.4, and update the *correctionField* taking into account asymmetric latencies introduced by hardware switching due to frame processing or waiting queues.

The hardware-software architecture for this topology, shown in Figure 6.12, is nearly the same as in Case 2 topology, with the exception that also the UES IP core is added in the reconfigurable portion. Therefore, the UES IP was appended in the PL section in the EDK project and the GPIO pins routed to the FMC

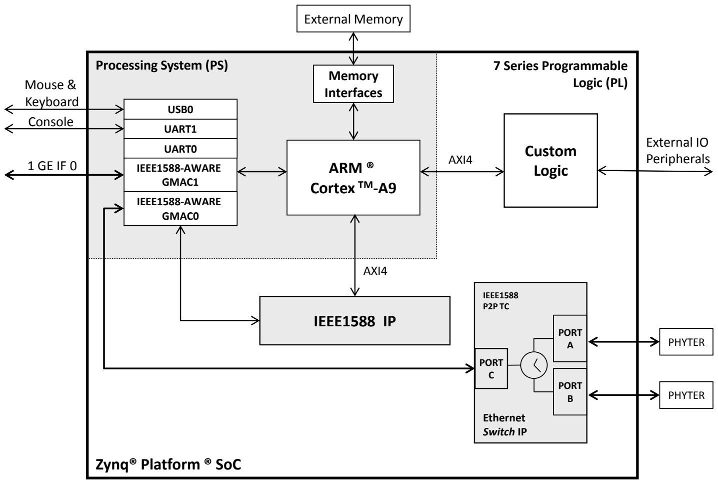


Figure 6.11: Case 3 topology block diagram

connector in the Zedboard are used as MII signals to be used by the TB-FMCL-GLAN board from Inrevium. With the aim of demonstrating the negative effect of asymmetric latencies in hardware switching, two different bitstreams were generated: firstly, the UES IP was configured not to implement the TC functionality; secondly, IEEE 1588 support was included. Consequently, since two new bitstreams were created, two new boot images had to be built with the Xilinx SDK tool to test the multiport proposed architecture.

If Figures 6.12 and 6.10 are compared, it can be seen that no software design changes are needed with respect to single port architecture described above. This happens because, as its name says, the Unmanaged Ethernet Switch IP is not managed by any software module.

6.2.2 Experimental results

This Section depicts the experimental results obtained for each topology presented in Section 6.2.1. Figures 6.13 to 6.16 show the screen captures from the oscilloscope when testing each topology with Test Setup A and B. In the graphics, the horizontal axis represents the time and the vertical axis represents voltages. The vertical scales are indicated at the upper-left corner of the screenshots, while the horizontal scale is specified near the upper-right corner next to the trigger term. The statistics for the mean and standard deviation values of the phase

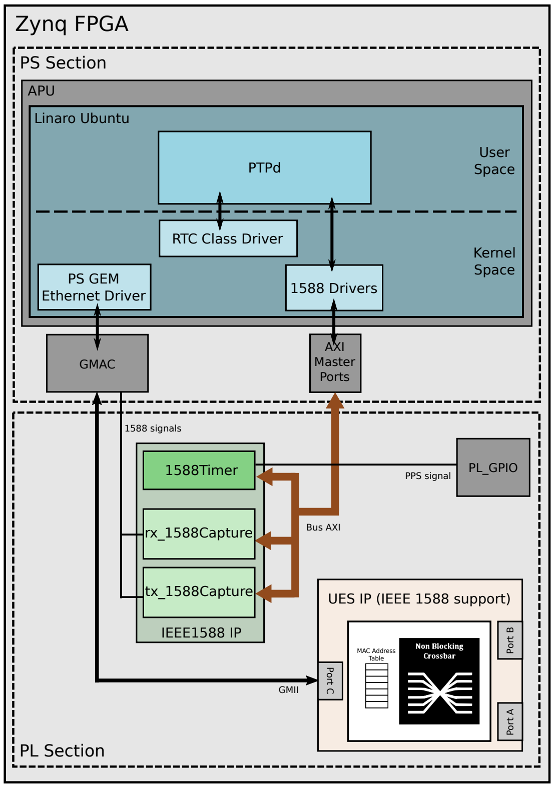
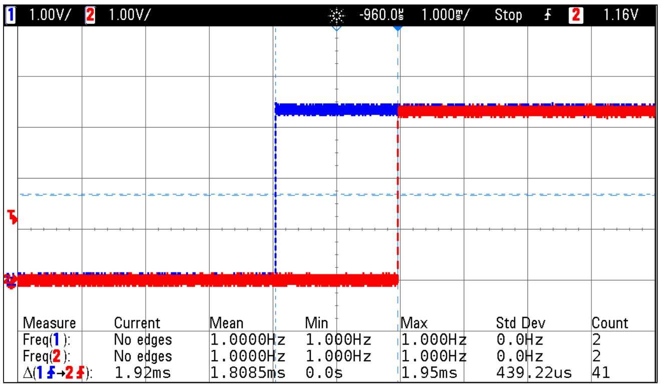


Figure 6.12: Hardware-software architecture for Case 3 topology

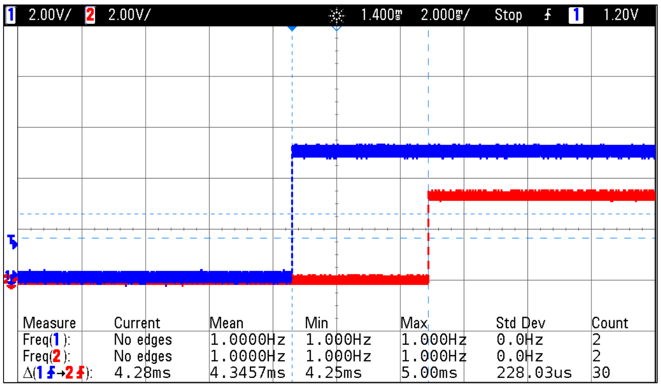
shift measurement between PPS signals from master and slave are included in the bottom part of the screen.

With the Case 1 topology, the results of synchronizing two ZedBoards (Test Setup A), and a ZedBoard to a Meinberg Time Reference Source (Test Setup B), are respectively shown in Figures 6.13(a) and 6.13(b). Despite using hardware timestamping, due to long and varying latencies introduced by software when synchronizing system clock and changing the PS GPIO state of the PPS output, both setups produced poor results regarding time offset of PPS signals. The slave offset from master is in the range of milliseconds with large standard deviations of hundreds of microseconds, although digital offsets showed by ‘ptp4l’ application on the serial console had values close to zero nanoseconds.



```
Measure      Current     Mean        Min        Max        Std Dev     Count
Freq(1):     No edges    1.0000Hz    1.000Hz    1.000Hz    0.0Hz       2
Freq(2):     No edges    1.0000Hz    1.000Hz    1.000Hz    0.0Hz       2
Δ(1→2):      1.92ms      1.8085ms    0.0s       1.95ms     439.22us    41
```

(a) Case 1, Test Setup A



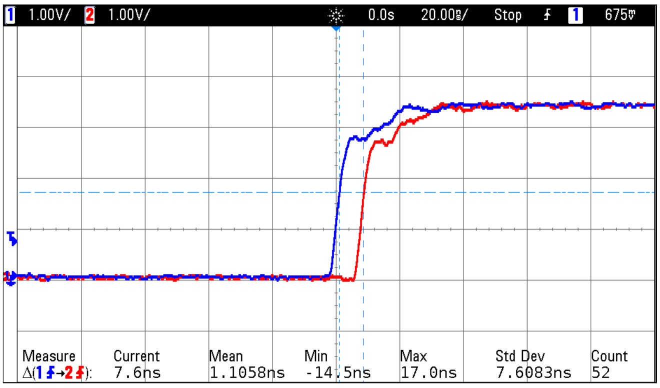
```
Measure      Current     Mean        Min        Max        Std Dev     Count
Freq(1):     No edges    1.0000Hz    1.000Hz    1.000Hz    0.0Hz       2
Freq(2):     No edges    1.0000Hz    1.000Hz    1.000Hz    0.0Hz       2
Δ(1→2):      4.28ms      4.3457ms    4.25ms     5.00ms     228.03us    30
```

(b) Case 1, Test Setup B

Figure 6.13: Time offset when using Case 1 topology

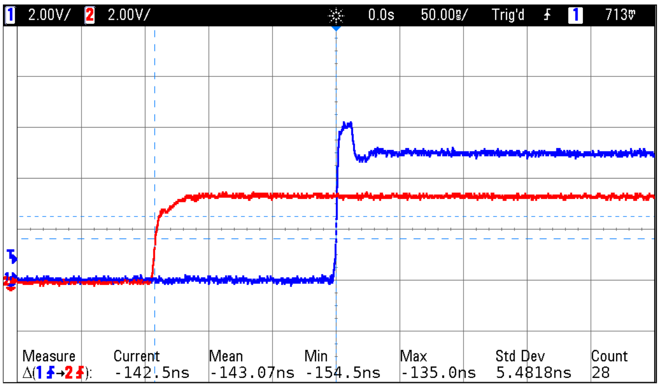
In contrast to results obtained in Case 1, Figures 6.14(a) and 6.14(b) demonstrate that a high precision PPS signal can be generated from a timer implemented in the PL section of the FPGA, when implementing Case 2 topology. On the one hand, with Test Setup A, accuracies in the range of 40 nanoseconds are achieved. In addition, the precision of the measurements is also highly improved because the standard deviation decreases to less than 10 nanoseconds.

On the other hand, in Test Setup B, the achieved accuracy is a bit worse, around 100-200 nanoseconds. However, the standard deviation maintains lower than 10 nanoseconds. This offset might happen due to the inaccuracy introduced by the



Measure	Current	Mean	Min	Max	Std Dev	Count
Δ(1→2):	7.6ns	1.1058ns	-14.5ns	17.0ns	7.6083ns	52

(a) Case 2, Test Setup A



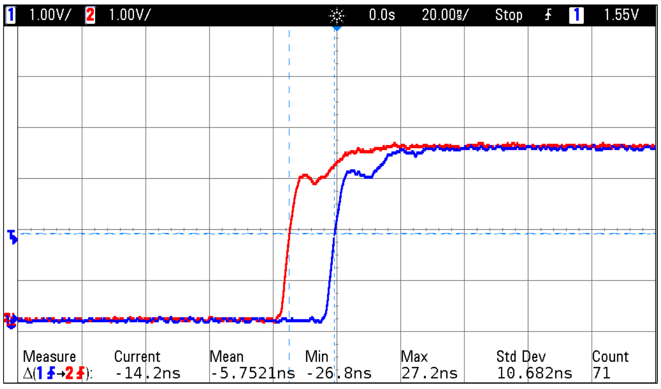
Measure	Current	Mean	Min	Max	Std Dev	Count
Δ(1→2):	-142.5ns	-143.07ns	-154.5ns	-135.0ns	5.4818ns	28

(b) Case 2, Test Setup B

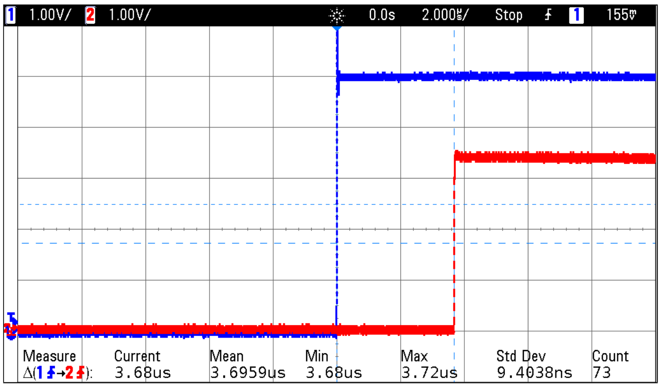
Figure 6.14: Time offset when using Case 2 topology

Meinberg equipment, because in its datasheet, a PPS output signal accuracy up to 250 nanoseconds is specified depending on the oscillator [156]. Nevertheless, this time offset could be reduced doing calibration processes in the slave. This calibration process consist on adjusting the inbound and outbound asymmetry parameters through software configuration until the time offset is closed to zero.

As it can be observed in Figure 6.15(a), if Case 3 topology without TC capabilities is implemented, the time offset keeps near to zero in the Test Setup A because no asymmetries are introduced. However, the standard deviation gets a bit worse due to the imprecision of hardware timestamping. This negative effect is the result of


1 1.00V/ 2 1.00V/
0.0s 20.00ns/ Stop 1 1.55V
Measure Δ(1→2): Current -14.2ns Mean -5.7521ns Min -26.8ns Max 27.2ns Std Dev 10.682ns Count 71

(a) Case 3 without TC, Test Setup A

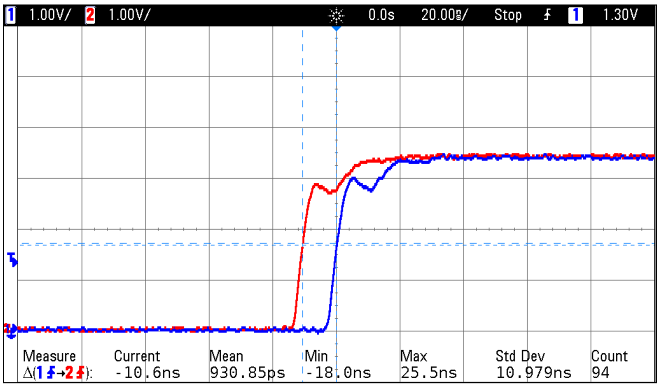

1 1.00V/ 2 1.00V/
0.0s 2.000us/ Stop 1 155mV
Measure Δ(1→2): Current 3.68us Mean 3.6959us Min 3.68us Max 3.72us Std Dev 9.4038ns Count 73

(b) Case 3 without TC, Test Setup B

Figure 6.15: Time offset when using Case 3 topology without IEEE 1588 support

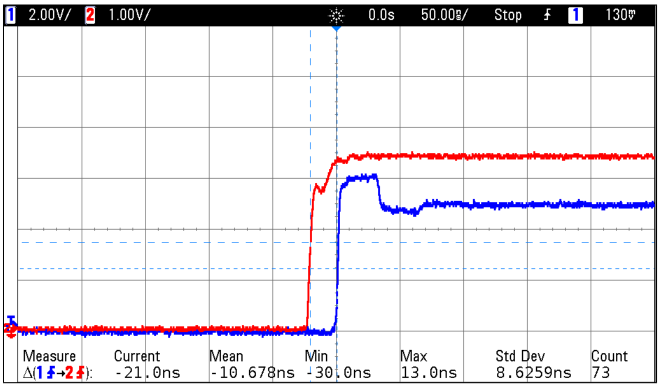
the timestamping quantization error imposed by hardware clock frequency when capturing ingress and egress PTP messages and also when the slave generates the PPS signal by hardware.

The effect of asymmetric latencies in hardware switching is demonstrated in Test Setup B. Since GMAC in the Zynq device works in Gigabit Ethernet and the Meinberg output is Fast Ethernet, the Ethernet Switch IP must do the speed conversion from Fast to Gigabit and viceversa. In this way, due to the Switch IP is store-and-forward, the whole frame has to be stored before starting the retransmission and this processing time differs in some microseconds on each



Measure	Current	Mean	Min	Max	Std Dev	Count
Δ(1→2):	-10.6ns	930.85ps	-18.0ns	25.5ns	10.979ns	94

(a) Case 3 with TC, Test Setup A



Measure	Current	Mean	Min	Max	Std Dev	Count
Δ(1→2):	-21.0ns	-10.678ns	-30.0ns	13.0ns	8.6259ns	73

(b) Case 3 with TC, Test Setup B

Figure 6.16: Time offset when using Case 3 topology with IEEE 1588 support

direction due to speed conversion. Therefore, significant asymmetries that cause time offsets of few microseconds in the slave are introduced by the switch, as observed with the oscilloscope in Figure 6.15(b).

Figure 6.16 show remarkable experimental results when introducing TC capabilities in Case 3 topology. In Test Setup A, the time accuracy of the slave is again in the range of few nanoseconds, with standard deviations around 10 nanoseconds. With the Meinberg equipment as the master in Test Setup B, the P2P TC in the UES IP eliminates time errors caused by asymmetries and the slave once more achieves accuracies in the range of 40 nanoseconds.

The measurements from the oscilloscope when comparing master and slave PPS outputs are summarized in Table 6.1 for all Cases.

Tabla 6.1: Synchronization results comparison of all topologies-setups

		Mean Offset	Min. Offset	Max. Offset	Std. Dev.
Case 1	Test Setup A	1.8085 ms	0 s	1.95 ms	439.22 us
	Test Setup B	4.3457 ms	4.25 ms	5 ms	228.03 us
Case 2	Test Setup A	1.1058 ns	-14.5 ns	17 ns	7.6083 ns
	Test Setup B	-143.07 ns	154.5 ns	-135 ns	5.4814 ns
Case 3	Test Setup A	-5.75 ns	-26.8 ns	27.2 ns	10.682 ns
	Test Setup B	3.696 us	3.68 us	3.72 us	9.404 ns
Case 3'	Test Setup A	930.85 ps	-18.0 ns	25.5 ns	10.979 ns
	Test Setup B	-10.678 ns	-30.0 ns	13.0 ns	8.626 ns

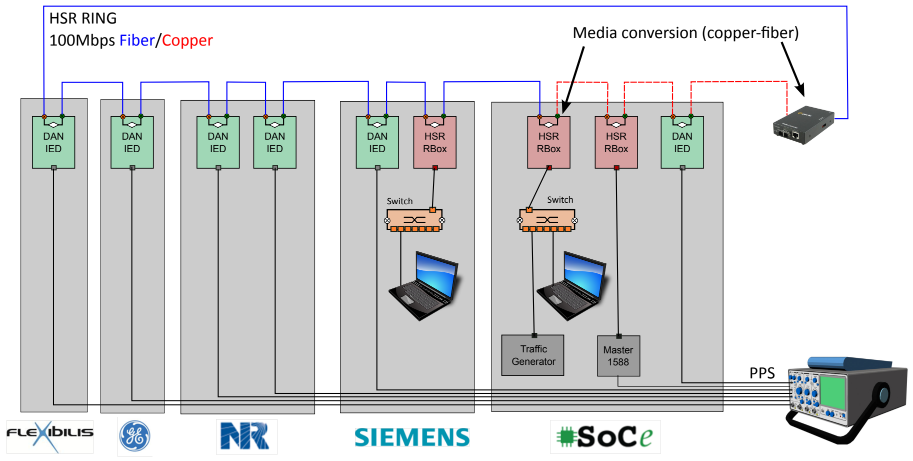
6.3 UCA interoperability testing

Some of the previously presented topologies were tested in the interoperability test that took place in Bilbao, in July 2014 [157]. This technical event was organized by UCA International Users Group, System-on-Chip engineering (SoC-e) and the University of the Basque Country (UPV/EHU), and consisted on performing several interoperability tests over an HSR ring. Participants were Siemens, RuggedComm, General Electric, Schneider Electric, ZIV, Ingeteam Power Technology, NR Electric, OMICRON, Flexibilis and SoC-e. This Section describes the test setups and the obtained results in the mentioned plugfest.

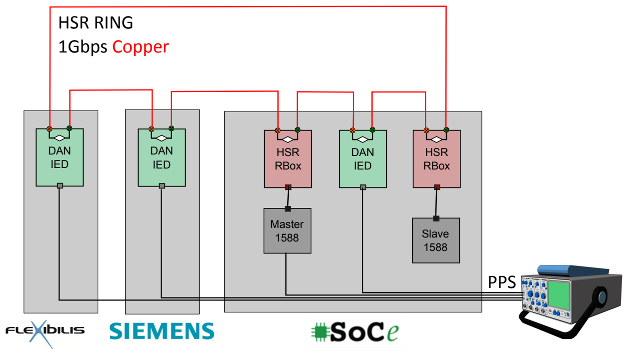
6.3.1 Setup description

SoC-e company decided to implement Case 2 and Case 3 topologies over Zedboard and ZC702 boards respectively, as explained in Section 6.2.1, to participate in two different tests and evaluate their performance in terms of synchronization accuracy when being connected to the HSR ring. Two different setups were configured: Test Setup A, as a Fast Ethernet HSR ring, and Test Setup B, as a Gigabit Ethernet HSR ring.

The first PTP test, named Test Setup A, consisted of 5 IEDs and 4 RedBoxes connected through the HSR ring, and the grandmaster connected to the ring through a TC, as represented in Figure 6.17(a). The ring employed both fiber and copper wiring and PPS signals from all slaves were checked with the oscilloscope. Concretely, SoC-e company employed the Case 3 topology to implement an IED



(a) Test setup A



(b) Test setup B

Figure 6.17: Diagrams of the HSR and IEEE 1588 network configurations tested in the UCA technical interoperability test

over a Xilinx ZC702 [158], but replacing the UES IP core shown in Figure 6.12 by an HSR switch with TC functionality.

In the Test Setup B, IEEE 1588-aware nodes were connected through a full copper HSR ring at Gigabit Ethernet, as represented in Figure 6.17(b). Apart from the SoC-e IED, an additional IEEE 1588 slave was connected through the RedBox. This slave was implemented over a Zedboard using Case 2 topology and the hardware-software architecture in Figure 6.10.

Figure 6.18 shows two pictures taken on the Interoperability Test.

6.3.2 Experimental results

After doing some HSR operation initial tests, the network was configured as Test Setup A in order to check that all slaves embedded in IEDs from different vendors were synchronized below one microsecond. Thus, time offsets between PPS signals from all slaves and the master were observed with the oscilloscope provided to each company. The PPS signal from the SoC-e IED can be seen in Figure 6.19(a)[2]. Through the persistence mode of operation in the oscilloscope, it can be conclude that the precision of the measurements was ±25 nanoseconds. Although the signal in the Figure is centred, it was moved horizontally to the left 596 nanoseconds, because a constant time offset of approx 450 nanoseconds was observed in all the nodes. This time offset was generated by asymmetries introduced by the non IEEE 1588-aware copper-fiber media converter. Additional traffic was also injected into the ring and all nodes showed good performance with up to 80% throughput with 64 and 1518 bytes length frames.

In the second case, where the ring configured as Test Setup B, all nodes were synchronized to the master below 500 nanoseconds. Both SoC-e slaves participating in the test with Case 2 and Case 3 topologies presented excellent synchronization results as shown in Figure 6.19(b). In this Figure, two PPS signals from the two slave clocks without horizontal shift can be seen. It can be conclude that the time offset from master has been reduced to 100 nanoseconds approximately, while the accuracy remained as expected.

[2]The PPS signal from master acted as the input trigger signal for the oscilloscope, so the horizontal position denotes the time offset from master.

6.4 Validation on a real manufacturing plant

With the evolution of traditional industrial automation systems to Cyber-Physical Production Systems (CPPS), distributed computed philosophy has also been adopted in advanced manufacturing plants. Therefore, sensors, actuators and controllers are integrated in the communication system and must be precisely synchronized, as well as in the substation automation scenario. Consequently, the layout of the an industrial plant in Ermua (Biscay, Spain) of a real manufacturing company called Microdeco was selected as the initial application scenario for the evaluation of the proposed Zynq-based SoC architecture.

6.4.1 Setup description

The network configuration for this setup consisted of nine lathe machines interconnected through fiber cables and Zynq-based devices from SoC-e, named CPPS Zynqbox [159], configured in a ring topology as represented in Figure 6.20. On each machine within the industrial plant, a CPPS Zynqbos was installed as shown by the picture illustrated in Figure 6.21.

The architecture inside CPPS Zynqbox nodes was the Case 3 topology presented in Section 6.2.1, with the addition of HSR support in the switch as described in Section 6.3.1. One node was configured as a PTP master, while the rest where configured as slaves.

6.4.2 Experimental results

In Figures 6.22 and 6.23, the time offset of 4 slaves is plotted based on the measurements collected over one hour of PTP synchronization (at 4 samples per second aprox.). As it can be seen in Figure 6.22(a), there was an unstable initial phase with huge offsets but, in less than 40 seconds, all slaves where synchronized below one microsecond, as plotted in Figure 6.22(b). In the stable phase, represented in Figure 6.23, time offsets oscillated between ± 50 nanoseconds.

The probability density function of the time offset is represented in Figure 6.24. It can be seen that most of the measured values were concentrated between ± 20 nanoseconds, and the probability density function of each slave could be modelled as a Gaussian distribution defined by parameters listed in Table 6.2, which were computed over 12,750 samples.

Tabla 6.2: Time offset statistical parameters

	Mean Offset (ns)	Std. Deviation (ns)
Slave 1	0.31	15.70
Slave 2	0.27	17.20
Slave 3	0.10	8.29
Slave 4	1.10	10.30

6.5 Conclusions

While the digital offset showed by user application was close to zero nanoseconds when using the IEEE 1588-aware GMAC alone in Case 1 topology, the offset between PPS signals was higher than 1 millisecond, with standard deviations in the range of hundred of microseconds. This time offset is not acceptable for many SAS applications such as the synchronization of MUs responsible for transmitting SV frames in the process bus network. With the support of an IEEE 1588 IP Core in the PL section, as demonstrated in Case 2 experiment, PPS signals can be generated very precisely in hardware. Thus, PPS signal precision is hugely improved achieving accuracies in the range of 40 nanoseconds, with standard deviations lower than 10 nanoseconds.

Case 3 topology was implemented in order to demonstrate the need of IEEE 1588 capability of internal hardware switching, when the device provides two or more Ethernet ports. If a non IEEE 1588-aware Switch IP was used, the processing times on each direction were different due to speed conversion causing undetectable asymmetries. Therefore, time synchronization errors were produced in the slave that was running on the ARM. In order to achieve highly accurate slave clocks, with accuracies in the range of nanoseconds, TC functionality must be implemented in the Switch IP that connects external Ethernet ports to the GMAC.

Since the most precise way of measuring the time offset from master is to compare the PPS signals, they must be as accurate as possible. In order to test that an IED with an integrated PTP slave fulfils strong synchronization requirements of SASs, a very precise PPS signal should be generated from a synchronized hardware clock as considered in Cases 2 and 3. The only method to achieve such a precise PPS output on new Xilinx Zynq devices is to add in the PL section an IEEE 1588 IP core with a synchronized timer capable of generating the PPS signal directly on hardware. Similarly, other user logic in the PL can use this hardware clock to perform protection and control functions or the PPS output may allow the slave to precisely synchronize MUs in the process bus with submicrosecond accuracy.

Case 2 and 3 topologies were also tested in the technical test that took place in Bilbao in July 2014, where synchronization accuracies were demonstrated to be in the nanoseconds range, even under traffic congestion. Additionally, Case 3 topology was also tested in a real manufacturing plant. The protocol performance was excellent when synchronizing several Zynq-based CPPS devices, as it has been demonstrated by the outstanding results obtained.

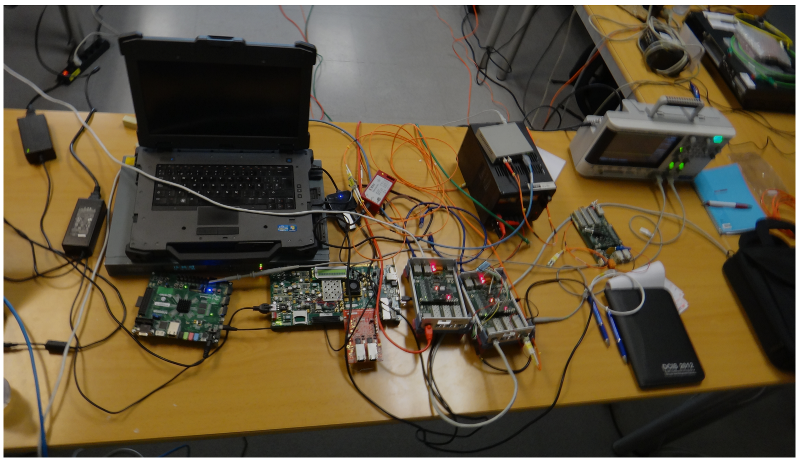
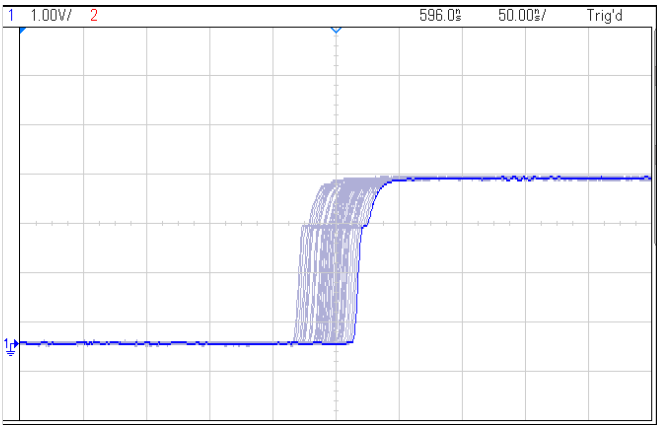
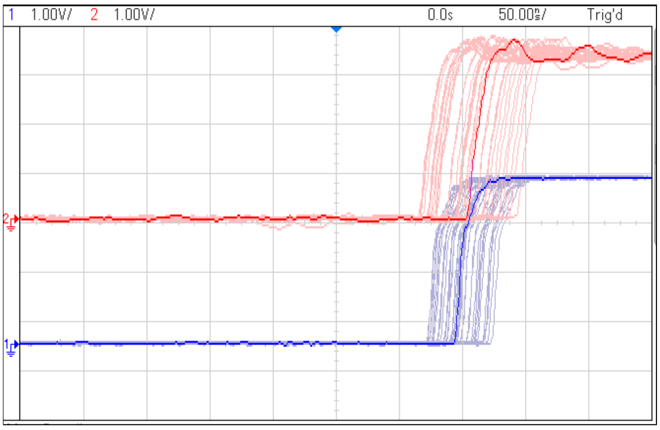


Figure 6.18: Pictures of the UCA technical interoperability test held in Bilbao



(a) Test setup A time offset



(b) Test setup B time offset

Figure 6.19: IEC 61850 and IEC 62439 technical interoperability test, Bilbao 2014. SoC-e IED time offset from grandmaster using Case 4 topology

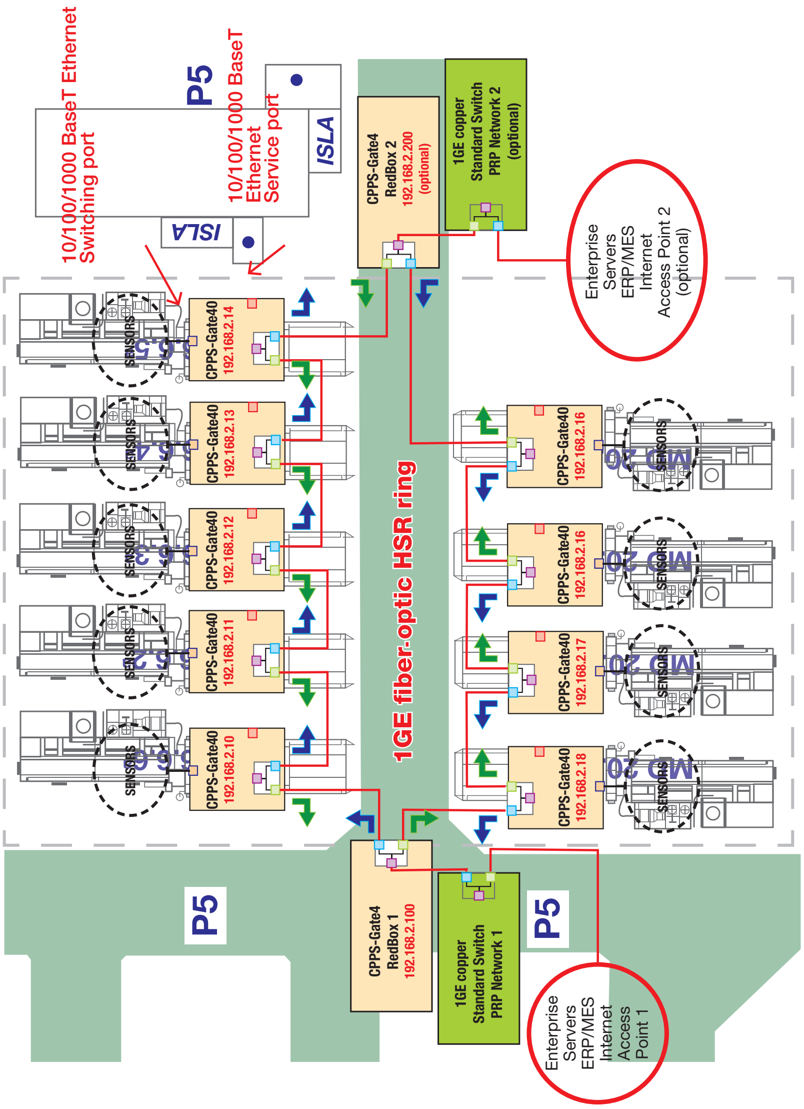

10/100/1000 BaseT Ethernet Switching port
10/100/1000 BaseT Ethernet Service port
P5
ISLA
ISLA
CPPS-Gate4 RedBox 2 192.168.2.200 (optional)
1GE copper Standard Switch PRP Network 2 (optional)
Enterprise Servers ERP/MES Internet Access Point 2 (optional)
CPPS-Gate40 192.168.2.14
CPPS-Gate40 192.168.2.13
CPPS-Gate40 192.168.2.12
CPPS-Gate40 192.168.2.11
CPPS-Gate40 192.168.2.10
SENSORS
1GE fiber-optic HSR ring
CPPS-Gate40 192.168.2.16
CPPS-Gate40 192.168.2.16
CPPS-Gate40 192.168.2.17
CPPS-Gate40 192.168.2.18
MD 20
CPPS-Gate4 RedBox 1 192.168.2.100
1GE copper Standard Switch PRP Network 1
P5
P5
Enterprise Servers ERP/MES Internet Access Point 1

Figure 6.20: Diagram of the Microdeco manufacturing plant layout

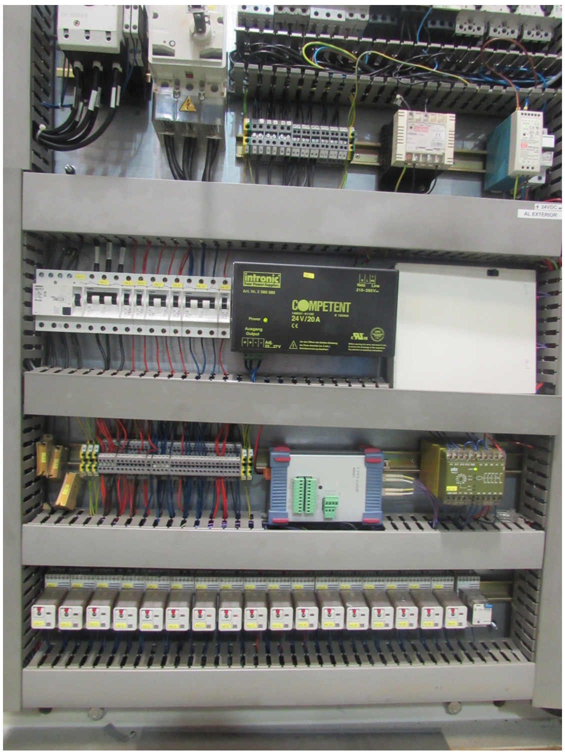
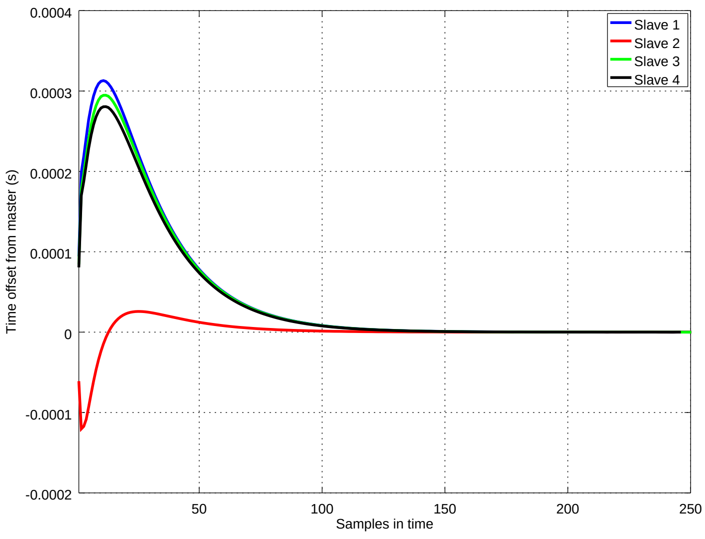
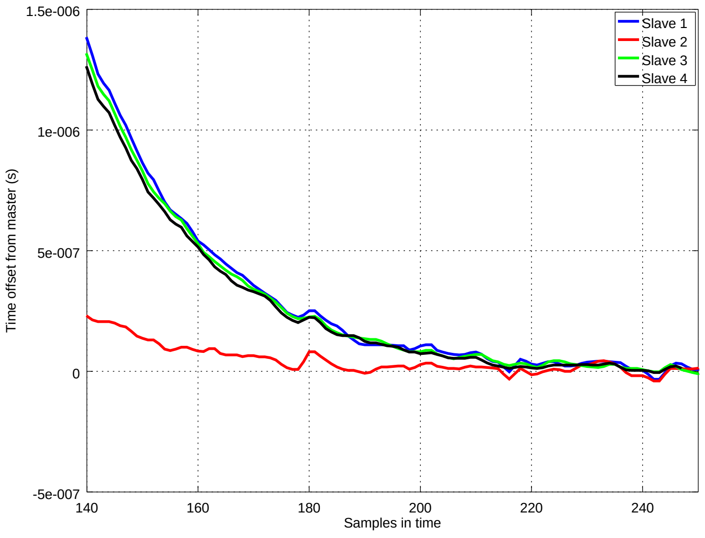


Figure 6.21: Picture of a CPPS Zynqbox installed in the industrial plant



(a) Initial phase synchronization



(b) Zoom of initial phase synchronization

Figure 6.22: CPPS time offset oscillation in the initial phase

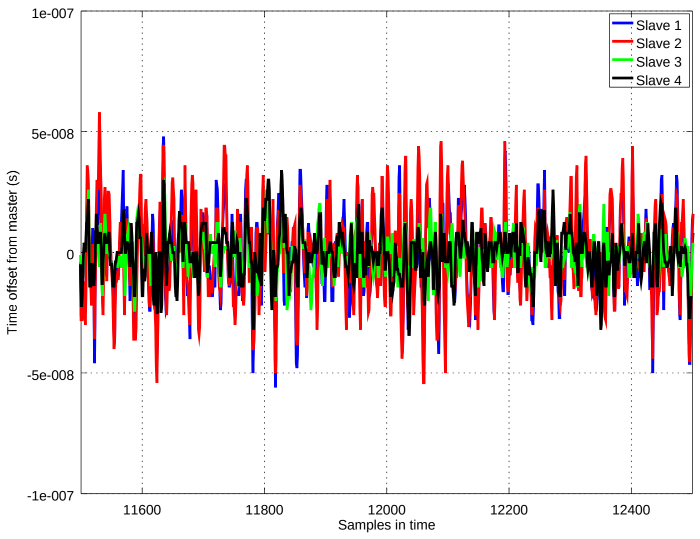


Figure 6.23: CPPS time offset oscillation in the stable phase

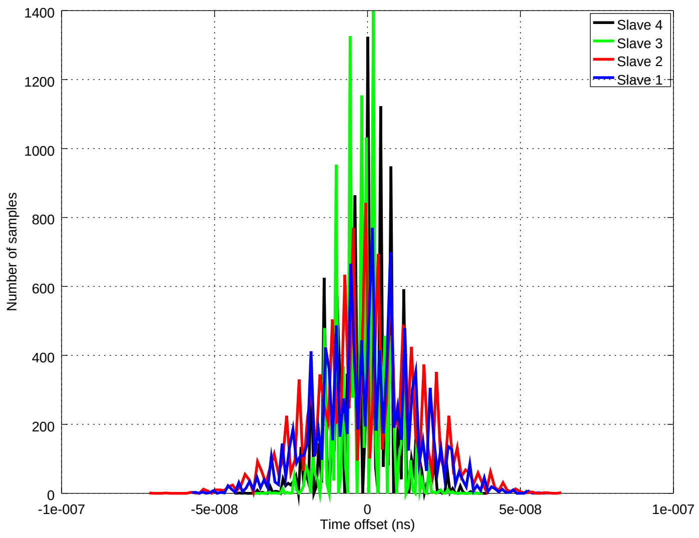


Figure 6.24: CPPS time offset probability density function

Chapter 7

Secure PTP SoC architecture validation

7.1 Introduction

In Chapter 6, beginning experiments without MACsec support were presented in order to obtain initial results that show the viability of PTP SoC architectures. This Chapter is a forward step in the validation of the proposed TimeWardenSoC architecture. For this purpose, a device running an IEEE 1588-2008 Ordinary Clock implementation will be secured.

Firstly, Section 7.2 describes the implemented topologies and the experimental setups. Secondly, implementation and synchronization results are compared for the different implemented topologies. Finally, Section 7.3 summarizes the conclusions.

7.2 Proof-of-concept design

Experiments described in this Section were conducted in the laboratory for two different topologies over the same setup configuration. These topologies and test setups are explained below.

7.2.1 Experimental setup description

With the aim of evaluating the effect of MACsec security mechanisms on PTP protocol performance, two similar SoC architectures represented in Figure 7.1 were designed for the Zynq device. Like in Section 6.2.1, the Avnet Zedboard has been used as the experimental platform. Both designs will include hardware PTP support to timestamp messages at physical layer and software PTP stack running on the ARM processor, as explained in previous Chapter. Additionally, one of the architectures will include a MACsec IP core to provide layer 2 security mechanisms. The selected MACsec IP core in this case was the one from Algotronix introduced in Section 4.3.4, which targets modern FPGA families from Xilinx and assures throughputs up to 40 Gbit/s. This company was interested in having a demonstration design based on the Zynq device and it agreed in delivering the VHDL code of the core in exchange for obtaining some experimental results using the mentioned platform.

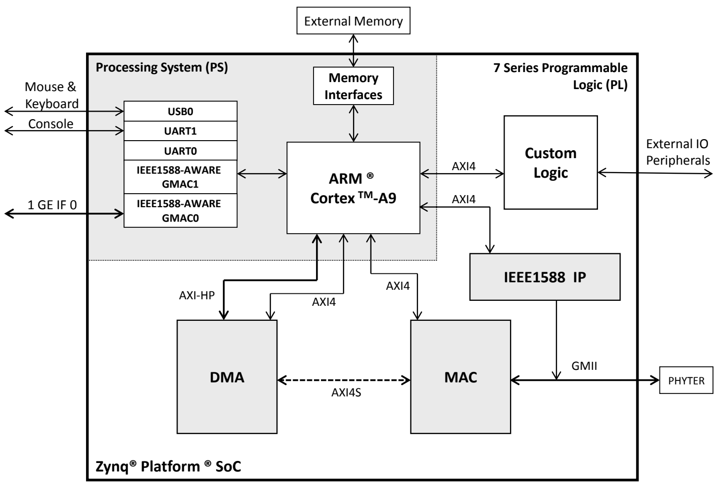
In contrast to the architecture represented in Figure 5.5, the MACsec IP core was not placed between the MAC and the PHY. It was placed after the MAC in the receiving direction, as depicted in Figure 7.1(b). This decision was taken to ease the integration of the Algotronix IP core, because its interfaces and data flow seemed to adapt better with the AXI system bus than with the GMII interface. As a consequence, instead of using the GMAC located in the PS part of the Zynq device, a soft MAC IP core and the corresponding DMA controller had to be instantiated in the PL section, so as to forward incoming and outgoing packets to and from processor through the AXI high-performance interface as detailed in [160]. The hardware-software architectures of the implemented designs are deeply described below.

The schema of the laboratory setup is displayed in Figure 7.2 and it consisted of two ZedBoards connected through a triple speed Ethernet link. In the setup, also the Tektronix DPO 7054C digital oscilloscope and a computer were used to display PPS signals generated by the ZedBoards and the serial output. The Linux OS booting files are stored in the SD card, which is partitioned once more into two partitions as represented in Figure 6.5.

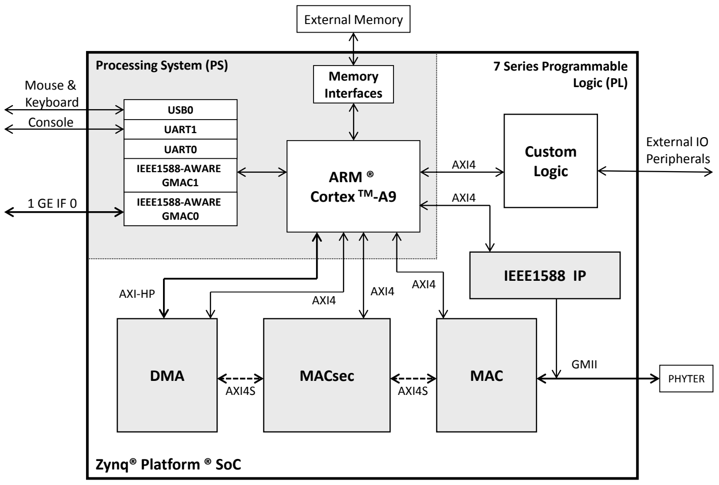
Two different test cases are distinguished in this Section: Case 1 corresponds to topology in Figure 7.1(a), while Case 2 implements Figure 7.1(b) topology.

Case 1: Zynq IEEE 1588-aware PL Ethernet design

Case 1 topology was built as the IEEE 1588-aware reference design from which the secure PTP SoC architecture would be validated. As mentioned above, the



(a) Case 1 topology



(b) Case 2 topology

Figure 7.1: TimeWardenSoC architectures with (below) and without (bottom) MACsec support

Algotronix MACsec IP core was designed to be integrated after the MAC; therefore, the GMAC Ethernet controller instantiated in the PS could not be used. Accordingly, an Ethernet controller was instantiated as soft logic in the PL region

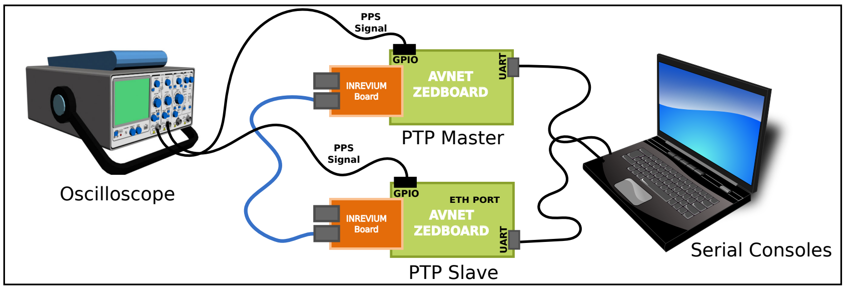


Figure 7.2: Diagram of the laboratory experimental setup for TimeWardenSoC validation

as specified in [160]. This hardware design was implemented using Vivado 2014.4 design tool, which mainly consisted of the AXI Ethernet, AXI DMA and AXI Interconnect. The IEEE 1588-2008 hardware support was included by instantiating the PTB IP core from SoC-e [97] described in Section 4.2.5.

Figure 7.3 shows the hardware-software architecture for Case 1 topology. For data movement purposes between the AXI Ethernet peripheral and the system memory, instead of using General Purpose AXI ports of the PS, the High-Performance AXI ports were employed. They provide high-bandwidth and low latency transfers to and from DDR3 memories [161]. In this sense, the AXI DMA IP core was needed to provide direct memory access between the AXI4 memory-mapped and AXI4-Stream interfaces through AXI4 Memory Mapped-to-Stream (MM2S) and Stream-to-Memory-Mapped (S2MM) channels. AXI Ethernet subsystem and AXI DMA were connected through an AXI4-Stream interface that provides point-to-point high-speed streaming data, as recommended for applications with high-bandwidth demand. In addition, the scatter/gather port of the AXI DMA was used to read descriptors from system memory, as required in processor-based systems [137] for offloading DMA management work from the CPU.

Details on hardware implementation using Vivado design tool are included in Appendix B. The resulted bitstream was used to create the new BOOT.BIN file with the Xilinx SDK, which was copied into the booting files partition of the SD card in conjunction with new generated devicetree.

Like in Chapter 6, the ARM software infrastructure to support IEEE 1588-2008 was composed by the same Linux distribution, the modified ‘PTPd’ software stack and the IEEE 1588 drivers for the PTB core.

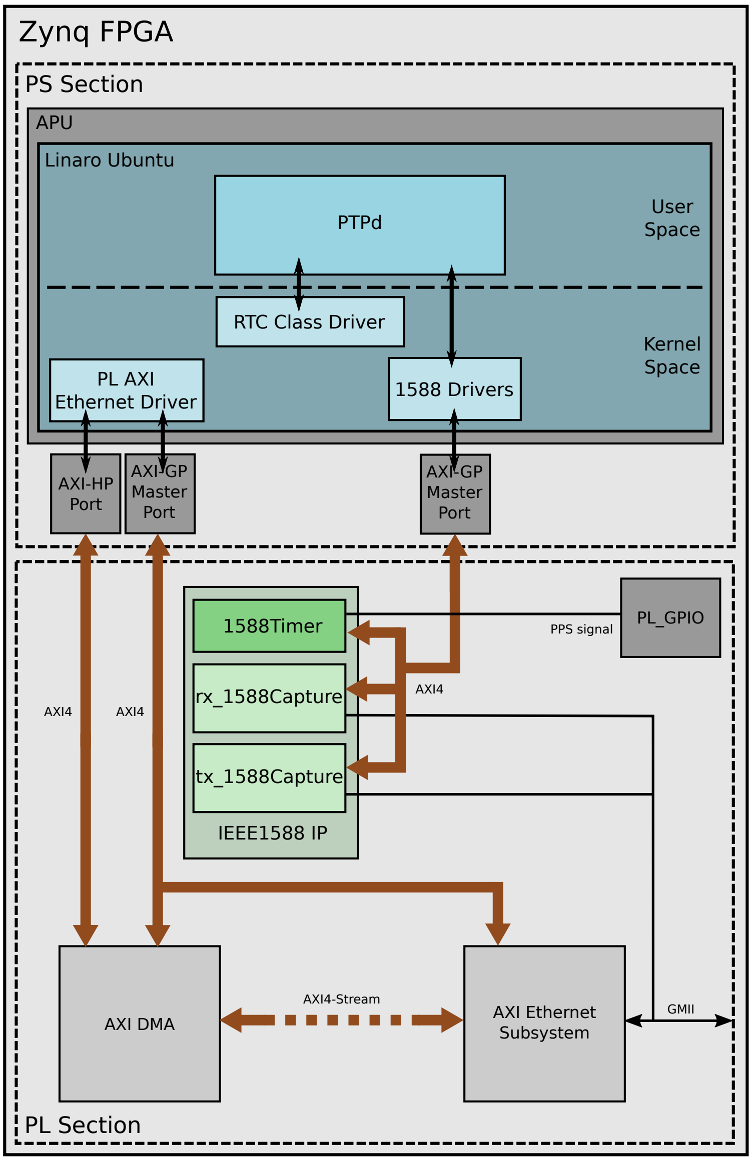


Figure 7.3: Hardware-software architecture for TimeWardenSoC Case 1 topology

Case 2: Zynq IEEE 1588-aware PL Ethernet design with MACsec support

MACsec hardware support was added to previous topology by integrating the Algotronix IP core between the AXI Ethernet and the AXI DMA. In order to adapt Algotronix interfaces to AXI4-Stream interfaces, the implementation of the so called MACsec 'sandwich' was needed, which consisted of some buffers and specifically designed state machines. The resulting core was named AXI Stream MACsec IP core and the corresponding hardware-software architecture of this topology is represented in Figure 7.4.

Concretely, the MACsec 'sandwich' consisted of a set of hardware modules designed in VHDL code as shown in Figure 7.5. With the aim of implementing the functionalities described in Section 5.4.1, two different state machines were designed on each submodule named TX Channel from System, TX Channel to MAC, RX Channel from MAC and RX Channel to System. These state machines control the writing and reading accesses to memory buffers and adapt data bus width.

On the other hand, the implemented memory buffers follow a First In First Out (FIFO) strategy to enqueue 64 bit data blocks, because the input data width of the Algotronix IP core is 64 bits. Thus, in the receive channel, Finite State Machine (FSM) RX number 1 in Figure 7.5 is responsible for managing AXI stream slave interface and writing incoming Ethernet frames to the FIFO buffer, while FSM RX number 2 reads the FIFO output and provides two 64-bit data words to the Algotronix IP core every 10 or 14 clock cycles[1]. FSM RX number 3 and 4 re-adapt proprietary Algotronix interface to standard AXI stream interface. At the transmitting side the processing path controlled by FSM TX number 1, 2, 3 and 4 is quite similar but in the opposite direction. Moreover, an AXI lite processor interface was also designed for configuration purposes, which includes a FSM to control the read and write accesses on Algotronix registers. Details on the designed FSMs are described in Annex B.

Moreover, an AXI lite processor interface was also designed for configuration purposes, as it is shown in Figure 7.5. This interface also includes a FSM to control the read and write accesses on the registers of the Algotronix IP. The Algotronix IP requires specific software support to manage these read and write operations. This software support was provided by pseudo drivers, which are parts of code compiled with 'gcc' as normal C files that provide simple hardware access from user space [162]. They basically map regions of memory assigned

[1]The FIFO output read time access depends on the cryptographic algorithm configured in the Algotronix IP core, which takes 10 or 14 clock cycles to process a 128-bit data block with AES-GCM-128 or AES-GCM-256 respectively.

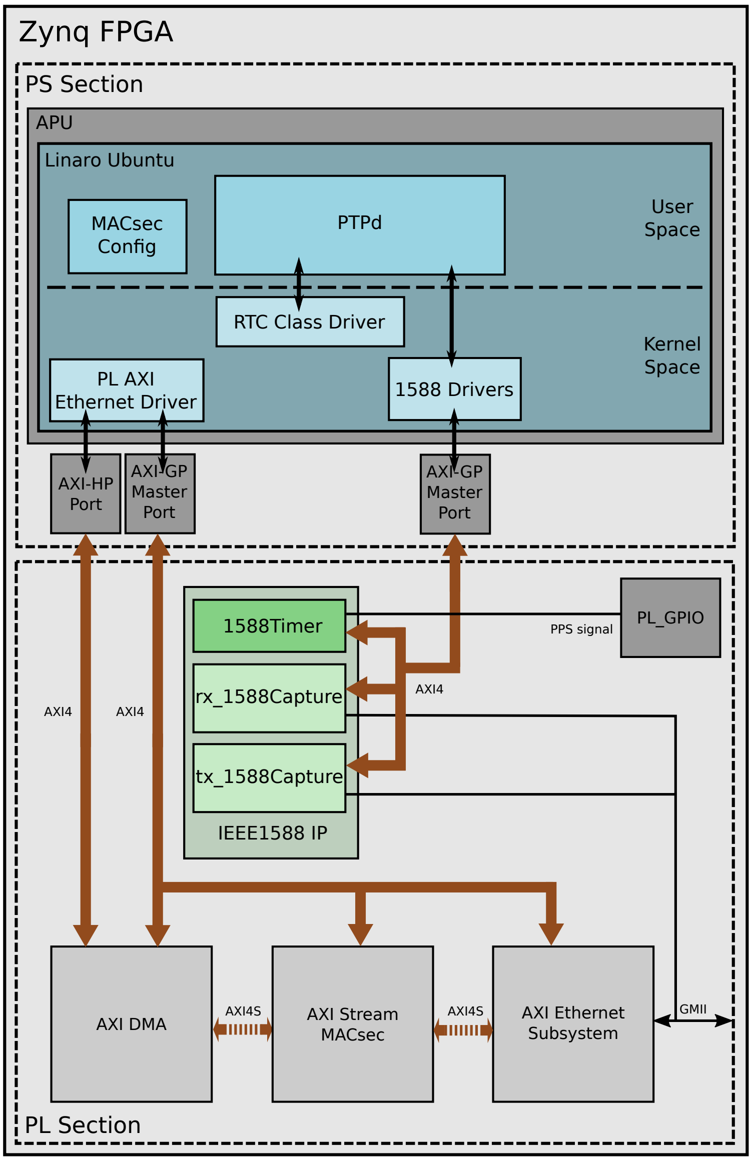


Figure 7.4: Hardware-software architecture for TimeWardenSoC Case 2 topology

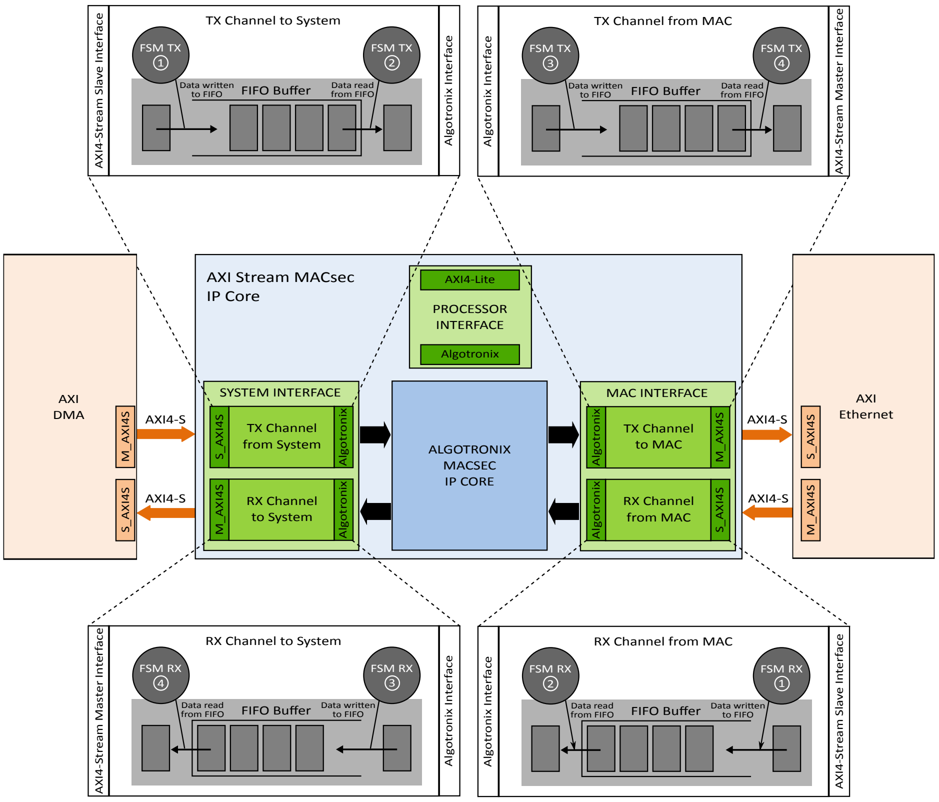


Figure 7.5: Block diagram of the AXI Stream MACsec IP Core

to the hardware device into user space memory through the '/dev/mem' Linux device. They do not replace a kernel mode driver, but provide a simpler method to allow designers accessing hardware registers infrequently.

Hence, the developed pseudo driver, named MACsec Config shown in Figure 7.4, initialized the Algotronix MACsec IP and assigned different Ethernet destination addresses to different SecYs. Firstly, it configured a SecY to be used as default to prevent frames with unassigned destination address from being dropped. Secondly, it assigned PTP multicast and peer delay addresses to a different SecY. The transmit and receive channels within this SecY were configured as represented in Figure 7.6, where the corresponding SCIs and other MACsec parameters are detailed. These configuration pseudo drivers also installed pre-shared SAKs. In this case, a unique SAK was used to protect the transmission and reception of

Tabla 7.1: Area cost of TimeWardenSoC Case 1 topology in Zynq XC7Z020

Resources	PTB IP	AXI Ethernet IP	AXI DMA IP	Others	Total	Utilization
Slice Registers	1,333	5,747	3,784	4,632	15,496	14.56%
Slice LUTs	1,412	4,592	4,465	4,680	15,149	28.48%
RAMB36	4	4	1	0	9	6.43%
RAMB18	2	0	1	0	3	1.07%
DSP48E1 blocks	0	0	0	0	0	0%

Tabla 7.2: Area cost of TimeWardenSoC Case 2 topology in Zynq XC7Z020

Resources	PTB IP	AXI Stream MACsec	AXI Ethernet IP	AXI DMA IP	Others	Total	Utilization
Slice Registers	1,538	17,320	4,984	3,406	3,830	31,078	29.21%
Slice LUTs	1,361	17,960	4,159	2,998	3,918	30,396	57.14%
RAMB36	4	20	4	1	0	29	20.71%
RAMB18	2	1	0	1	0	4	0.71%
DSP48E1 blocks	0	0	0	0	0	0	0%

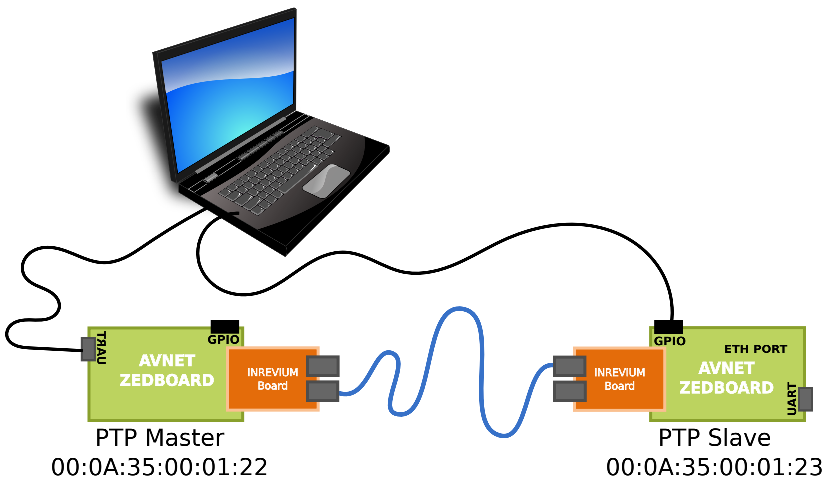
PTP frames.

Additionally, the pseudo driver also included some functionalities, in order to model a cyber-security attack that compromises the PTP protocol performance. A wrong SAK was maliciously installed in the master and, after a short period of time, the correct SAK was reinstalled. This experiment is thoroughly explained in Section 7.2.3.

7.2.2 Resource utilization

In this Section, the resource utilization of both implemented topologies, Case 1 and Case 2, is analysed so as to know the viability of the proposed solution. The utilization report was generated using the Vivado 2014.4 design tool for the Zynq XC7Z020 device. Implementation results are summarized in Tables 7.1 and 7.2 for Case 1 and Case 2 topologies respectively. Also, the device utilization was graphically captured from Vivado and shown in Figure 7.7.

As expected, the integration of the MACsec IP core increases considerably the area cost. Anyway, the obtained results ensures the viability of using the same low-cost device, the Xilinx Zynq XCZ7020 for the secure and non-secure designs. Consequently, a non-effective increment on the device cost would offer a very interesting solution for industry.



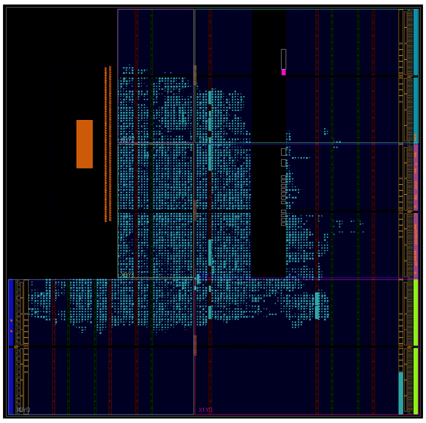
```
* TX Channel Config:
  - Destination Addresses:
       01:1B:19:00:00:00
       01:80:C2:00:00:0E
  - SecY Index: 1
  - SCI: 000A350001220001
  - Control parameters:
       Replay Protection: True
       Validate Frames: Check
       Confidentiality: False
       Protect Frames: True
       Use SCB: False
       Use ES: False
       CP Enabled: True
  - AN: 0
  - SAK: 97CFAA28 2FB33DAC
         AB82B1D0 4A4144FE
         270F36AA A64BE501
         4FC6B7BF 1C4E777A

* RX Channel Config:
  - SCI: 000A350001230001
  - SC Index: 0
  - AN: 0
  - SAK: 97CFAA28 2FB33DAC
         AB82B1D0 4A4144FE
         270F36AA A64BE501
         4FC6B7BF 1C4E777A
```

```
* TX Channel Config:
  - Destination Addresses:
       01:1B:19:00:00:00
       01:80:C2:00:00:0E
  - SecY Index: 1
  - SCI: 000A350001230001
  - Control parameters:
       Replay Protection: True
       Validate Frames: Check
       Confidentiality: False
       Protect Frames: True
       Use SCB: False
       Use ES: False
       CP Enabled: True
  - AN: 0
  - SAK: 97CFAA28 2FB33DAC
         AB82B1D0 4A4144FE
         270F36AA A64BE501
         4FC6B7BF 1C4E777A

* RX Channel Config:
  - SCI: 000A350001220001
  - SC Index: 0
  - AN: 0
  - SAK: 97CFAA28 2FB33DAC
         AB82B1D0 4A4144FE
         270F36AA A64BE501
         4FC6B7BF 1C4E777A
```

Figure 7.6: MACsec parameters configured from software


(a) Case 1 topology

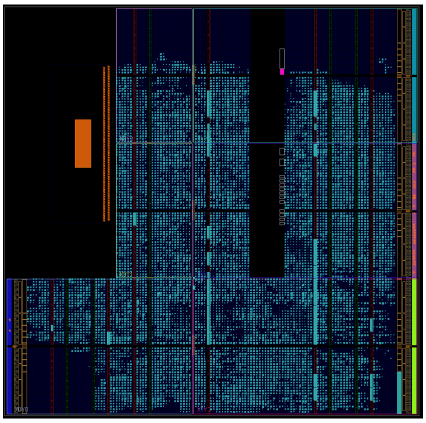

(b) Case 2 topology

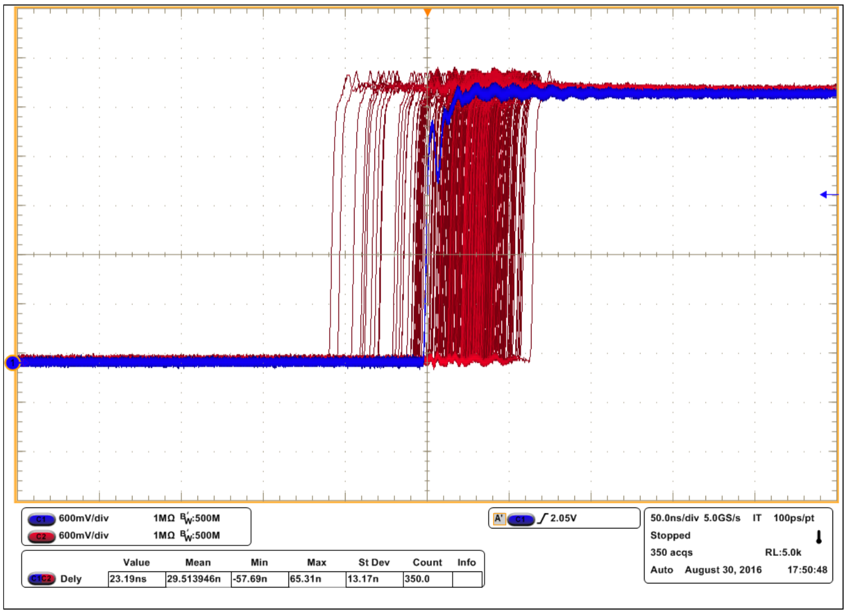
Figure 7.7: TimeWardenSoC FPGA utilization

7.2.3 Experimental results

This Section outlines the experimental results obtained for the non-secure and secure PTP topologies presented in Section 7.2.1. Screen captures from the oscilloscope when executing both topologies are shown in Figures 7.8 and 7.9. In the graphics, the horizontal axis represents the time and the vertical axis represents voltages. The statistics for the mean and standard deviation values of the phase shift measurement between PPS signals from master and slave ZedBoards are included in the bottom part of the screen.

Concretely Figure 7.8 presents the clock accuracies obtained for 'Case 1' and 'Case 2' topologies when P2P mode of operation is used. On the other hand, Figure 7.9 shows the equivalent results for the E2E mode of operation. Whereas the vertical scale was set to 60 millivolts/division, the value of the horizontal scale was 50 nanoseconds/division. The statistics shown at the bottom of the screenshots were computed over 350 samples captured within a short period of time of around five minutes. Time error mean and standard deviation values should not be representative of protocol performance for such a short time interval and, hence, synchronization accuracies obtained from slave's serial output over larger periods are presented below.

Three different experiments were performed in the laboratory over the test setup shown in Figure 7.2. Firstly, the slave ZedBoard was synchronized to the master



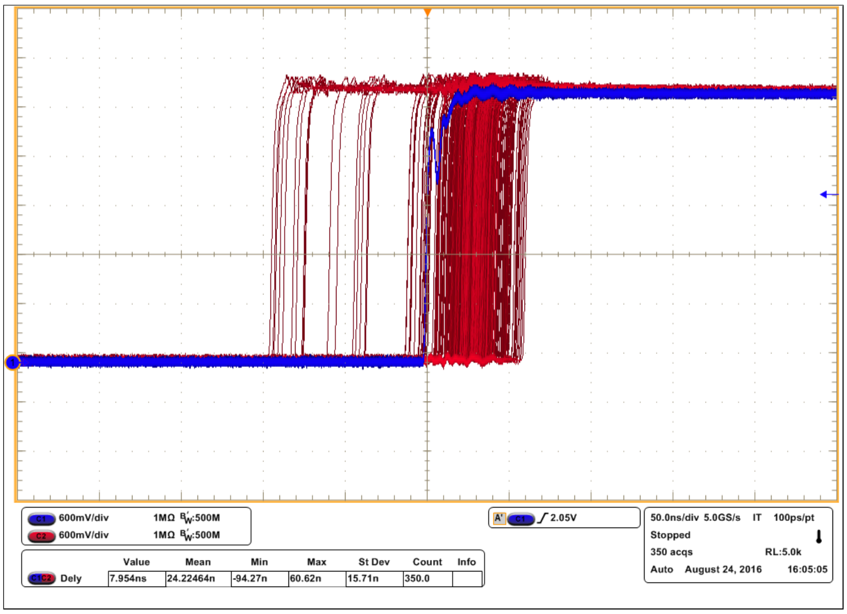
C1 600mV/div 1MΩ BW:500M
C2 600mV/div 1MΩ BW:500M

A' C1 ∫ 2.05V

50.0ns/div 5.0GS/s IT 100ps/pt
Stopped
350 acqs RL:5.0k
Auto August 30, 2016 17:50:48

	Value	Mean	Min	Max	St Dev	Count	Info
C1C2 Dely	23.19ns	29.513946n	-57.69n	65.31n	13.17n	350.0	

(a) Case 1, normal operation of PTP



C1 600mV/div 1MΩ BW:500M
C2 600mV/div 1MΩ BW:500M

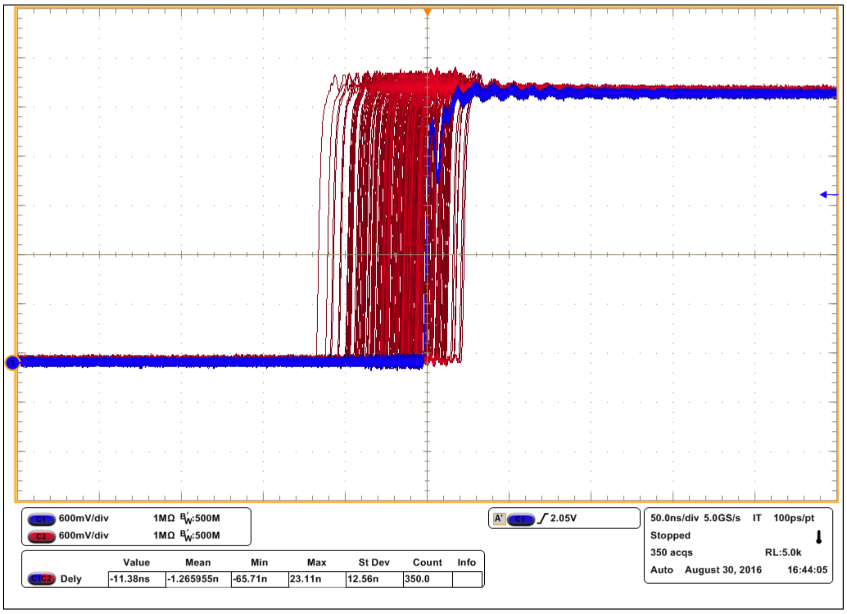
A' C1 ∫ 2.05V

50.0ns/div 5.0GS/s IT 100ps/pt
Stopped
350 acqs RL:5.0k
Auto August 24, 2016 16:05:05

	Value	Mean	Min	Max	St Dev	Count	Info
C1C2 Dely	7.954ns	24.22464n	-94.27n	60.62n	15.71n	350.0	

(b) Case 2, PTP protected with MACsec

Figure 7.8: Oscilloscope captures when using P2P mode of operation during 'Test 1' and 'Test 2' experiments



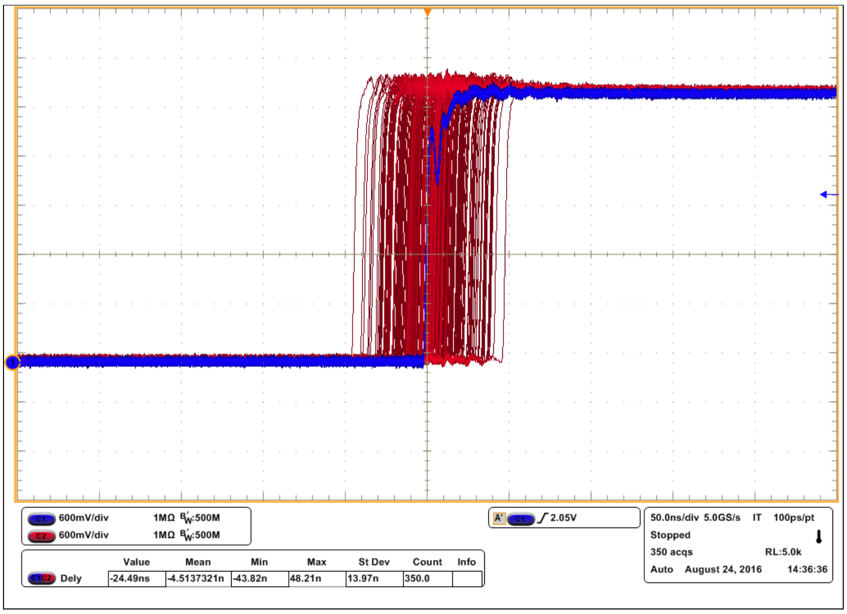
C1 600mV/div 1MΩ B_W:500M
C2 600mV/div 1MΩ B_W:500M

A' C1 ∫ 2.05V

50.0ns/div 5.0GS/s IT 100ps/pt
Stopped
350 acqs RL:5.0k
Auto August 30, 2016 16:44:05

	Value	Mean	Min	Max	St Dev	Count	Info
C1 C2 Dely	-11.38ns	-1.265955n	-65.71n	23.11n	12.56n	350.0	

(a) Case 1, normal operation of PTP



C1 600mV/div 1MΩ B_W:500M
C2 600mV/div 1MΩ B_W:500M

A' C1 ∫ 2.05V

50.0ns/div 5.0GS/s IT 100ps/pt
Stopped
350 acqs RL:5.0k
Auto August 24, 2016 14:36:36

	Value	Mean	Min	Max	St Dev	Count	Info
C1 C2 Dely	-24.49ns	-4.5137321n	-43.82n	48.21n	13.97n	350.0	

(b) Case 2, PTP protected with MACsec

Figure 7.9: Oscilloscope captures when using E2E mode of operation during 'Test 1' and 'Test 2' experiments

Tabla 7.3: Time offset statistical parameters during 'Test 1' and 'Test 2' experiments

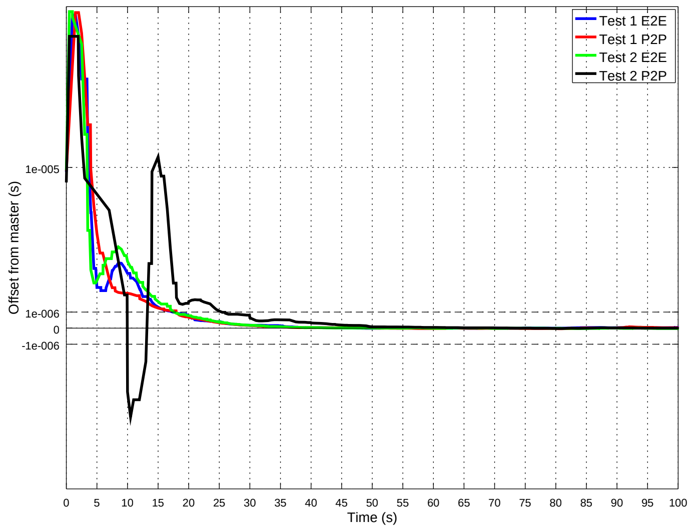
		Mean Offset (ns)	Std. Deviation (ns)
Case 1 - Conventional PTP	P2P	-0.82	18.54
	E2E	0.25	19.84
Case 2 - PTP over MACsec	P2P	-0.36	16.90
	E2E	-1.58	15.78

Zedboard during one hour approximately using the generated bitstream for the Case 1 topology represented in Figure 7.3. This experiment is called 'Test 1' in this Section. 'Test 2' was performed during one hour using the Case 2 topology represented in Figure 7.4 that additionally allows the protection of PTP messages with a static pre-configured MACsec SAK. The installation of such a key was done by means of the initialization and configuration pseudo drivers executed from Linux console, as explained in Section 7.2.1. Finally, during 'Test 3' experiment an incorrect SAK was installed in the master after some minutes and the correct one was reinstalled later. This experiment was performed using the Case 2 topology to test the protocol behaviour under an unsuccessful re-authentication process that might result in different SAKs installed in PTP nodes, for example.

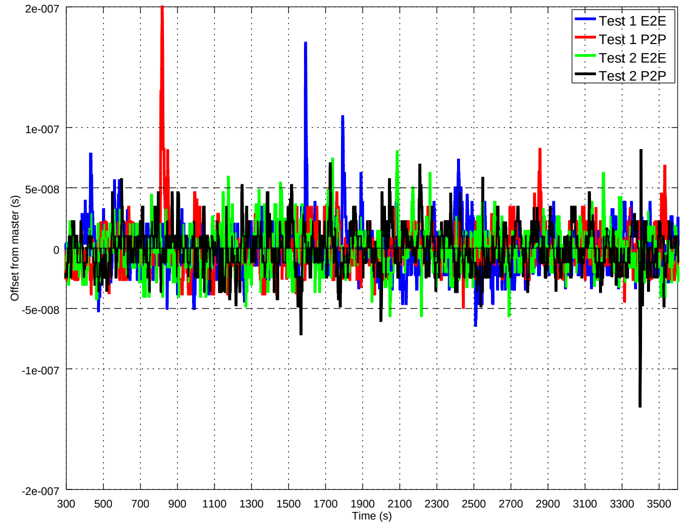
Figure 7.10 shows the time offset of the slave during the execution of 'Test 1' and 'Test 2' experiments in the initialization and the stable synchronization phases. The measurements were collected over one hour of PTP synchronization in both P2P and E2E modes of operation with and without MACsec protection. In all cases, the slave was synchronized below one microsecond in less than 30 seconds. It is worth mentioning that the integration of MACsec in 'Test 2' experiments did not have a negative impact on PTP performance, since synchronization accuracies were similar to those obtained in 'Test 1'. In fact, the standard deviation was still between 15 and 20 nanoseconds for all cases, as summarized in Table 7.3, which was a really good indicator of high precise synchronization. The probability density function of time offset from master is represented in Figure 7.11.

The last experiment made in the laboratory, named 'Test 3', consisted of installing an incorrect SAK in the master Zedboard after 400 seconds approximately, with the aim of modelling how the system behaves against an unsuccessful reauthentication process. The correct SAK is reinstalled 200 seconds later.

The time offset values over the time captured from the slave serial console are represented in Figure 7.12. The shaded area in this Figure represents the time interval while the slave lost the synchronization from master due to an incorrect installed SAK. As mentioned above, time offset values represented in the Figure were captured from the serial console and, since the 'PTPd' stack does not output them until synchronization in the slave is recovered, no values were obtained



(a) Initial synchronization phase



(b) Stable synchronization phase

Figure 7.10: Time offset oscillation during 'Test 1' and 'Test 2' experiments

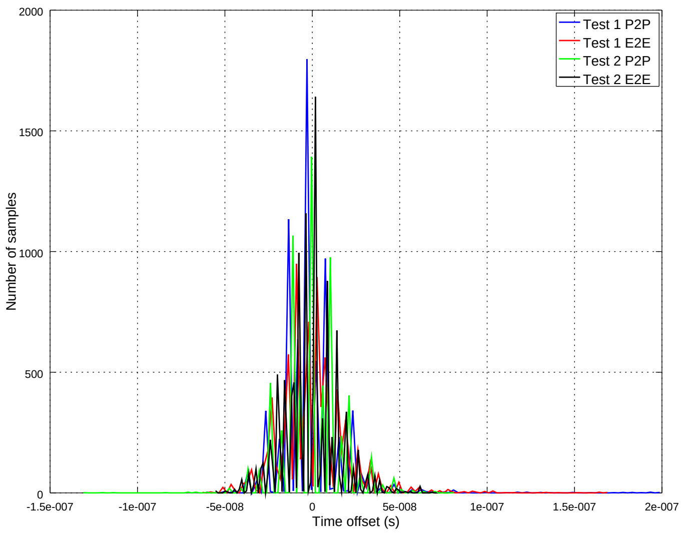


Figure 7.11: Time offset probability density function during 'Test 1' and 'Test 2' experiments

during this interval.

In order to overcome this problem, 'Test 3' was repeated and time offset values were directly stored on the hard disk drive of the Tektronix oscilloscope through the 'Action on Event' functionality, which was configured to save the delay measurement between two PPS signals on each trigger event.

Figure 7.13 depicts the results of the repeated experiment. In this graphic, the time offset from master measured with the oscilloscope reaches values higher than 30 microseconds. Actually, in less than two minutes, there were captured synchronization errors of more than one microsecond, which is the time synchronization accuracy allowed for substation communications.

7.3 Conclusions

This Chapter aims to validate the secure PTP SoC architecture proposed in Chapter 5. In order to analyse the effect of integrating MACsec hardware units in common PTP SoC architectures, two different topologies were designed, implemented and tested over the described experimental setup. The experimental

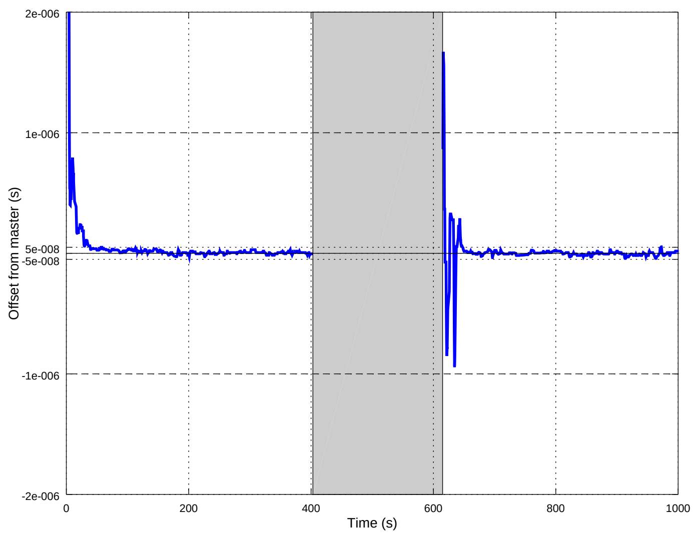


Figure 7.12: Time offset values during 'Test 3' experiment

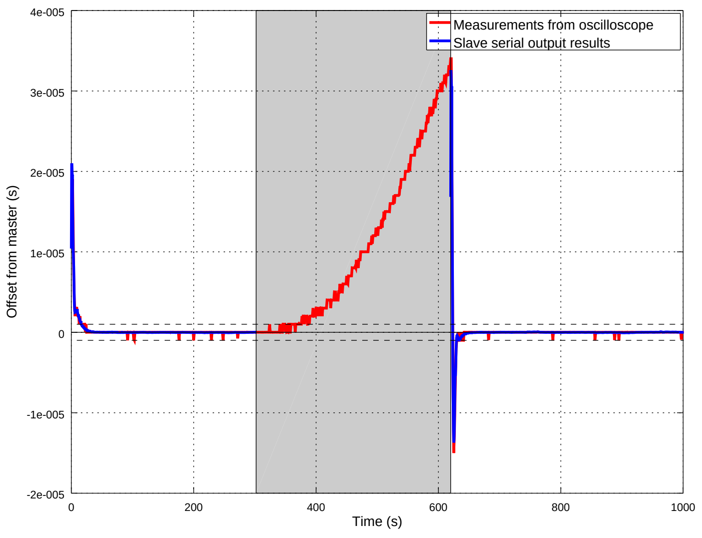


Figure 7.13: Time offset values during repeated 'Test 3' experiment

setup utilized two ZedBoards based on Xilinx Zynq FPGA. Case 1 topology consisted of an AXI Ethernet Subsystem IP plus an AXI DMA IP connected to the ARM processor through an AXI interconnect core combined with the Precise Time Basic IP core from SoC-e to provide IEEE 1588 support. In order to add MACsec capabilities, a MACsec IP core from Algotronix was included between the AXI Ethernet and the AXI DMA cores in the Case 2 topology, and specific AXI Stream interfaces were designed to allow the integration of the Algotronix IP.

Synchronization accuracies were measured with normal PTP and with PTP over MACsec by performing two basic experiments called 'Test 1' and 'Test 2'. Experimental results demonstrated the viability of using MACsec to protect PTP messages, since the standard variation of the time offset from master remains near 15 nanoseconds in all cases.

An additional experiment was performed using the Case 2 topology to study the behaviour of the PTP protocol under an authentication error. In the stable phase of the synchronization protocol, where the slave reaches synchronization accuracies below 50 nanoseconds, a false SAK was installed in the master. How fast the slave could lose synchronization was modelled by acquiring delay measurements between two PPS signals captured with the oscilloscope. In less than two minutes, the slave time offset from master was higher than one microsecond. In fact, time errors around 30 microseconds were observed after five minutes approximately.

Chapter 8

Conclusions and future work

8.1 Conclusions

In this work, the basis for securing communications in critical infrastructures like SASs was provided. In particular, precise synchronization protocol to be used in substations was demonstrated to be highly vulnerable. In this context, the challenge of integrating a hop-by-hop security solution into common SoC architectures without compromising PTP performance became the highest priority.

In the state of the art, two clearly parts were studied. First part analyses general aspects of cyber-security in substation communications, as well as PTP protocol operation and emerged security proposals. The interest of the research community regarding how to protect substation communications raised when the cyber-security of several industrial plants was compromised by some viruses and worms capable of taking control of the machines and turning them into failure. The security mechanisms specified in current versions of the standards present some inconsistencies. On the one hand, for IEC 61850 communications with stringent performance requirements such as GOOSE and SV, the security is defined in the IEC 62351-6, which mandates that RSA cryptography must be used to provide source authenticity. However, despite expensive processors with crypto accelerators were employed, execution times would exceed the maximum allowed transfer times. On the other hand, the optional PTP security extension, introduced in the standard in 2008, was based on old keyed hash algorithms that has also been demonstrated to be suboptimal due to latency and required resources. Therefore, it is highlighted the need for defining a new security framework which encompasses an end-to-end source authentication mechanism and a hop-by-hop group authentication and integrity protection scheme.

Since the work of P1588 Security SC has been mainly concentrated on end-to-end security mechanisms integrated in PTP, the utilization of external security solutions to provide hop-by-hop authentication and integrity services was detected as an interesting research field. Particularly, MACsec was the security standard that best fits in substation environments, since it protects communications at layer 2. In addition, the nature of MACsec intrinsically fulfils most of PTP security requirements because the corresponding security tag is verified and regenerated on each hop in the forwarding path, as required by TC functionality of intermediate nodes.

In the second part of the state of the art, a deep study about how to develop SoC architectures integrating precise time synchronization and MACsec was provided. Taking advantage of the experience of the research group with regard to FPGA-based systems and SoC designs, the research work explained in this manuscript was intentionally focused on the development of SoC architectures to provide secure synchronization in SASs with accuracies in the nanoseconds range. As a consequence, available hardware IP cores and open source software solutions to implement MACsec-aware PTP nodes were explored. In this sense, SoC-e and Flexibilis companies were discovered to be the main competitors in the PTP IP cores market nowadays and two open source PTP software solutions were identified. Regarding security, the utilization of MACsec-aware physical transceivers from Microsemi or the complete MACsec IP core from Algotronix is suggested, in addition to an open source software implementation of IEEE 802.1X found in the literature.

Addressing the findings in the state of the art, the design and security requirements were collected as the basis for defining the hybrid proposal: the integration of a security tag within PTP message, which provides source authentication using TESLA key chains, and the utilization of MACsec and IEEE 802.1X mechanisms to provide hop-by-hop group authentication ranging an entire PTP domain within a LAN. During the development of this thesis, efforts were focused on the external utilization of MACsec as stated above and, as a consequence, two proposals were presented as contributions. Firstly, a new authentication scheme to derive MACsec group keys using EAP methods was proposed for layer 2 networks, which includes a new key hierarchy and the associated EAPOL message exchange. Secondly, a MACsec-aware PTP SoC architecture was described for a single port ordinary clock and a multi-port hybrid clock, as well as for a doubled attached HSR node. After analysing the throughput and latency of these architectures, the viability of adding MACsec support to conventional PTP SoC architectures was theoretically demonstrated. Information from datasheets of some related IP cores was also used to estimate the area cost in modern System-on-Programmable-Chips such as the Xilinx Zynq platform.

Initial experiments without MACsec support were performed in order to demonstrate the viability of PTP SoC architectures using modern low-cost reconfigurable devices. Those experiments were executed on a variety of setups, ranging from basic experiments in the laboratory with few nodes to real experiments in an industrial plant over an HSR ring topology. As expected, synchronization accuracies in the nanoseconds range were achieved demonstrating the potential of hardware assisted P solutions. The validation part in this manuscript also includes the design, implementation and validation of a more complex SoC architecture which integrates MACsec support in conjunction with PTP hardware assistance. The synchronization accuracies obtained in the laboratory experiments when performing PTP over MACsec were nearly the same as when performing normal PTP over Ethernet without cyber-security. As a consequence, the utilization of low-cost reconfigurable devices to provide hop-by-hop authenticity and integrity protection in substation networks, which present stringent synchronization requirements and limited resources, is demonstrated to be an appealing solution for future deployments.

8.2 Main contributions

In this Section the main contributions of this thesis are listed. A short description of each point is given and a reference to the part of this manuscript, where the corresponding topic is discussed in greater detail. The order of the list is determined by the order the different contributions were presented in this manuscript and the importance of each point.

1. **General issues concerning cyber-security in substations.**

   The evolution of traditional SCADA systems to modern digital communication networks in substations lead power utilities to great opportunities but, the interconnection of station and process bus to external and public networks for remote monitoring and controlling increases the risk of cyber-attacks. In this thesis, a general introduction to standard protocols for substation communications and cyber-security requirements is included in Chapter 2, in which the challenging process of securing the recommended precise synchronization protocol is highlighted. In this sense, the need for deploying an hybrid solution that provides end-to-end source authentication and hop-by-hop group authentication is introduced.

As a result, a scientific article was published in the Renewable & Sustainable Energy Reviews journal (JCR: 6.798).

2. **Overview of cyber-security mechanisms for protecting PTP.**

   PTP networks, as defined in the second version of IEEE 1588 standard, rely on intermediate nodes participation in order to compute the propagation delay and compensate slaves time offset from master. PTP messages are modified on each hop due to TC functionality and, consequently, security checksums and authentication tags must be checked and regenerated each time the message traverse a network node. Chapter 3 gives an overview on PTP protocol operation, types of messages, devices and security features. The work done by the members of the P1588 Security SC is also summarized and related information was gathered into a congress paper that was presented at the ISPCS conference held in 2015.

3. **Secure PTP SoC architectures for programmable devices.**

   A complete state of the art about designing secure PTP SoC architectures is presented in Chapter 4. Concretely, a very detailed evaluation of open source software and most relevant commercial off-the-shelf IP cores for hardware assisted PTP solutions was included in this manuscript. A preliminary work regarding PTP SoC architectures for being implemented in FPGAs was published in proceedings of the DCIS conference and presented in a poster session in 2012. Additionally, it was also implemented a new TC architecture based on a shared memory where ingress timestamps were stored, which was accessed via wishbone bus, and results were presented at the ISIE conference in 2013.

4. **TimeWardenKey: a new key management scheme based on IEEE 802.1X authentication protocol for distributing MACsec group keys.**

   Over the years working on secure PTP SoC architectures, the idea of sharing a group CAK between PTP nodes within the same LAN to secure PTP traffic gained more and more importance. It emerged from the problem of deploying the traditional authentication scheme defined in IEEE 802.1X standard, which states the utilization of pairwise CAKs derived from EAP methods between a supplicant and the authenticator located in the peer PAE entity. As a matter of fact, both supplicant and authenticator state machines should be implemented in intermediate nodes. In order to simplify authentication message exchange and minimize CPU load due to execution of PAE state machines, the new conceptual authentication and key distribution scheme as described in Section 5.3 comprises a multi-host authenticator accessible after some hops from every PTP node acting as supplicant in the LAN. The new authentication message exchange and the new key hierarchy to allow the derivation and distribution of group CAKs from EAP methods is also proposed.

The utilization of group CAKs in PTP over Ethernet networks was firstly proposed within the conference paper presented at the ISPCS 2015. After that, it gradually evolved to a more mature proposal which was introduced to the P1588 Security SC in January 2016 for the first time.

5. **TimeWardenSoC: a new PTP SoC architecture integrating MACsec hop-by-hop security mechanisms.**

   With the aim of providing hop-by-hop authentication and integrity services in PTP networks, MACsec units must be integrated in conventional PTP SoC architectures. Besides hardware timestamping assistance, cryptographic algorithms must also be implemented in hardware so as to avoid throughput bottlenecks. Otherwise, if authentication tags were computed by software, PTP protocol performance could drastically get worse. In Section 5.4, the block diagrams of a single port PTP entity and a multiport intermediate node are described. Also, the particular case of an HSR node for substation environments is represented. The throughput and latency, as well as area cost, of these hardware-software architectures is thoroughly analysed to check the viability of implementing such a SoC architecture in modern programmable devices.

   Both the TimeWardenKey and the TimeWardenSoC theoretical concepts were introduced as described in Chapter 5 in a journal article, which is currently under review process by the IEEE Transactions on Industrial Electronics (JCR: 6.383).

6. **Experimental evaluation of PTP SoC architectures**

   Several experiments were executed to validate the single port and multiport PTP SoC architectures without MACsec, as described in Chapter 6. Zynq device from Xilinx was the selected platform to implement the hardware support in the reconfigurable logic part and a Linux host system was built in the processing system part in order to run the PTP software stack. Apart from experiments in the laboratory, the architecture was also validated in an interoperability test held in Bilbao in 2014 and in a real industrial plant in Ermua in 2015. Synchronization accuracies in the range of nanoseconds were achieved in all the described setups.

   These outstanding results allowed the presentation of a poster at the IS-

PCS 2014 first and, secondly, the publication of an article at the IEEE Transactions on Smart Grids journal (JCR: 3.19).

7. **Experimental evaluation of PTP over MACsec**

   Two new architectures were implemented to validate the operation of PTP over MACsec using the TimeWardenSoC architecture by performing three different tests. Chapter 7 outlines the test setup description and the experimental results obtained during the three tests. Once more, Xilinx Zynq device was the targeted platform to implement the hardware-software design. This design included some developed hardware modules needed for interfacing the commercial off-the-shelf employed MACsec and IEEE 1588 IP cores with several Xilinx soft-cores that formed the Ethernet Subsystem. In the experiments, the time offset from master measured with the oscilloscope demonstrated mean synchronization error near to zero, with standard deviations below 20 nanoseconds even when PTP traffic was protected by MACsec authentication and integrity mechanisms.

   An article regarding the validation of the proposed SoC architecture was submitted for review at the IEEE Transactions on Industrial Electronics journal (JCR: 6.383), as commented above.

8.3 Scientific publications in the context of this work

This Section presents all scientific publications, which formed part of this work. The articles are separated into publications in scientific journals and conference publications.

Journal publications

J1) **N. Moreira**, E. Molina, J. Lázaro, E. Jacob, A. Astarloa. *"Cyber-security in Substation Automation Systems"*, Renewable and Sustainable Energy Reviews, vol. 54, issue 1, 220-234, 2015. Impact Factor (JCR): 6.798.

J2) **N. Moreira**, J. Lázaro, U. Bidarte, J. Jiménez, A. Astarloa. *"On the Utilization of System-on-Chip Platforms to Achieve Nanosecond Synchronization Accuracies in Substation Automation Systems"*, IEEE Transactions on Smart Grid, vol. PP, issue 99, pages 1-11, 2016. Impact Factor (JCR): 3.19.

J3) **N. Moreira**, J. Lázaro, U. Bidarte, J. Jiménez, A. Astarloa. *"A New SoC Architecture to Secure IEEE 1588 Synchronization Protocol in SAS"*, IEEE Transactions on Industrial Electronics, under review. Impact Factor (JCR): 6.383.

J4) J. Lázaro, A. Astarloa, J.A. Araujo, **N. Moreira**, U. Bidarte. *"MACsec Layer 2 Security in HSR Rings in Substation Automation Systems"*, Energies, accepted for publication. Impact Factor (JCR): 2.077.

J5) E. Molina, E. Jacob, J. Matías, **N. Moreira**, A. Astarloa. *"Using software defined networking to manage and control IEC 61850-based systems"*, Computers & Electrical Engineering, vol. 43, pages 142-154, 2015. Impact Factor (JCR): 1.084.

J6) E. Molina, E. Jacob, J. Matías, **N. Moreira**, A. Astarloa. *"Availability Improvement of Layer 2 Seamless Networks Using OpenFlow"*, The Scientific World Journal, vol. 2015, pages 1-14, 2015. Impact Factor (SJR): 0.315.

Conference publications

C1) **N. Moreira**, A. Astarloa, J. Lázaro, J.A. Araujo, A. García. *"Implementation of Precise Time Synchronization in FPGAs: State of the Art"*, Conference on Design of Circuits and Integrated Systems (DCIS), pages 496-497, Avignon (France), 2012.

C2) **N. Moreira**, A. Astarloa, J. Lázaro, A. García , E. Ormaetxea. *"IEEE 1588 Transparent Clock Architecture for FPGA-based Network Devices"*, IEEE International Symposium on Industrial Electronics (ISIE), pages 1-6, Taipei (Taiwan), 2013.

C3) **N. Moreira**, A. Astarloa, U. Kretzschmar. *"SHA-3 based Message Authentication Codes to Secure IEEE 1588 Synchronization Systems"*, Conference of the IEEE Industrial Electronics Society (IECON), pages 2323-2328, Vienna (Austria), 2013.

C4) J.A. Araujo, J. Lázaro, A. Astarloa, **N. Moreira**, A. García. *"Memory Requirements Analysis for PRP and HSR Hardware Implementations on FPGAs"*, Conference of the IEEE Industrial Electronics Society (IECON), pages 2297-2302, Vienna (Austria), 2013.

C5) J.A. Araujo, J. Lázaro, A. Astarloa, A. Zuloaga, **N. Moreira**. *"Duplicate and Circulating Frames Discard Methods for PRP and HSR (IEC62439-3)"*, Conference of the IEEE Industrial Electronics Society (IECON), pages

4451-4456, Vienna (Austria), 2013.

C6) **N. Moreira**, A. Astarloa, U. Kretzschmar, J. Lázaro, E. Molina. *"Securing IEEE 1588 Messages with Message Authentication Codes based on the Keccak Cryptographic Algorithm Implemented in FPGAs"*, IEEE International Symposium on Industrial Electronics (ISIE), pages 1899-1904, Istanbul (Turkey), 2014.

C7) U. Kretzschmar, J. Gómez-Cornejo, **N. Moreira**, U. Bidarte, A. Astarloa. *"A Versatile FPGA Demonstration Platform for Academic Use"*, Tecnologías Aplicadas a la Enseñanza de la Electrónica (TAEE), pages 1-6, Bilbao (Spain), 2014.

C8) **N. Moreira**, J. Lázaro, A. Astarloa, A. García, S. Salas. *"Nanosecond Accuracy using SoC Platforms"*, International IEEE Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS), page 19, Austin (Texas, USA), 2014.

C9) A. Astarloa, J. Lázaro, U. Bidarte, J.A. Araujo, **N. Moreira**. *"FPGA Implemented Cut-Through vs Store-and-Forward Switches for Reliable Ethernet Networks"*, Conference on Design of Circuits and Integrated Systems (DCIS), pages 1-6, Madrid (Spain), 2014.

C10) A. Astarloa, **N. Moreira**, J. Lázaro, M. Urbina, A. García. *"1588-aware High-Availability Cyber-Physical Production Systems"*, International IEEE Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS), pages 25-30, Beijing (China), 2015.

C11) **N. Moreira**, J. Lázaro, J. Jiménez, M. Idirin, A. Astarloa. *"Security Mechanisms to protect IEEE 1588 Synchronization: State of the Art and Trends"*, International IEEE Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS), pages 115-120, Beijing (China), 2015.

C12) A. Astarloa, **N. Moreira**, U. Bidarte, M. Urbina, D. Modroño. *"FPGA based Nodes for Sub-microsecond Synchronization of Cyber-Physical Production Systems on High Availability Ring Networks"*, International Conference on Reconfigurable Computing and FPGAs (ReConFig), pages 1-6, Cancun (Mexico), 2015.

8.4 Future work

This Section proposes some lines of research in order to give continuity to the work presented in this thesis. These lines are the following:

- **Development of the TimeWardenKey proposal.**

  Although one of the contributions of this thesis is precisely the new key management scheme called TimeWardenKey, it was only theoretically presented. A development process consisting of the definition of new PAE components and states machines according to the proposed EAPOL frames addressing and EAPOL message exchange to distribute MACsec group CAKs as described in Chapter 5 should be carried out. Additionally, it could be interesting to study the application of such a group key management mechanism to establish and distribute keys to provide source authentication through the Security TLV, like initial TESLA keys, instead of using external asymmetric cryptographic based authentication mechanisms. The enhancements on the TimeWardenKey scheme could be proposed to the P1588 Security SC to be considered in future versions of the standard.

- **Validation of the TimeWardenKey proposal.**

  The experimental work presented in this manuscript was focused on the validation of the proposed TimeWardenSoC using Xilinx Zynq device and static pre-configured SAKs due to the expertise of the author on designing FPGA-based SoC systems. Nevertheless, with the aim of achieving a full authenticated PTP system for SASs, which would minimally impact substation communications performance, also MACsec group keys should be established and distributed through TimeWardenKey scheme proposed in Chapter 5. For this purpose, future efforts could be focused on studying and implementing the required modifications on current versions of 'hostapd' and 'wpa_supplicant' open source applications to support TimeWardenKey.

- **Validation of the multiport version of TimeWardenSoC proposal.**

  Another important branch of future research would be expanding the implemented TimeWardenSoC architecture for an ordinary clock to a multiport architecture including Transparent Clock functionality. This is another vital step forward a full authenticated PTP system, since intermediate nodes with Transparent Clock functionality are required in most deployed substation network topologies. Since the TC functionality of the switch IP core is responsible for modifying the *correctionField* of PTP messages, MACsec

ICV should be check on its ingress and egress ports and, accordingly, multiple MACsec IP cores might be integrated at [G]MII interfaces instead of the AXI stream interface employed in Chapter 7. Moreover, for implementations with more than two ports, the utilization of Zynq devices larger than ZC7Z020 should be considered so as to overcome the problem of limited area identified in Chapter 5.

- **Utilization of more complex and real test scenarios.**

  After having implemented the TimeWardenSoC with multiport capability and the TimeWardenKey as a set of modifications on available IEEE 802.1X software, new experimental setups could be configured in the laboratory. The definition of these new test setups would include intermediate nodes comprising more complex network topologies and they would allow the execution of more sophisticated experiments to validate the cyber-security proposal. Finally, the adaptation of such test setups to real industrial scenarios like the validation experiments described in Chapter 6 in conjunction with dynamic group MACsec keys management could represent a turning point in the integration of cyber-security mechanisms in industrial communications.

8.5 Acknowledgements

This work has been supported by the Ministerio de Economía y Competitividad of Spain within the project TEC2014-53785-R and it has been carried out inside the Research and Education Unit UFI11/16 of the UPV/EHU and partially supported by the Basque Government within the fund for research groups of the Basque university system IT394-10 and within the project TFactory ER-2014/0016. Also, FEDER funds and UPV/EHU PhD scholarship funding are acknowledged.

Appendix A

MACsec-based PTP security proposal details

A.1 Security and design requirements of PTP networks in SASs

R1: Source authentication. In the PTP power profile only multicast addressing is specified for PTP messages, so a multicast key management scheme for distributing a group key is needed. However, source authentication is not guaranteed if multicast PTP messages are protected using traditional symmetric cryptography and group keys, because the destination node cannot verify that the received frame was sent by an authorized node. For example, an attacker could highjack an intermediate node not authorized to act as a master and start sending malicious *Announce* or *Sync* messages to align slaves to a false reference time. In order to overcome this problem, a new Security TLV containing an ICV computed by the sender with a TESLA key chain must be defined.

R2: Group authentication. Second version of the IEEE 1588 standard introduced the concept of TC and, consequently, most PTP messages need to be modified by TC functionalities within intermediate nodes. By possessing a group key, intermediate nodes demonstrate they were authenticated against an authenticator and they belong to the group of nodes authorized to modify PTP packets. In this sense the MKA protocol must be used to verify the possession of a group CAK and, after that, establish the session SAKs.

R3: **Hop-by-hop integrity.** Besides group authentication, hop-by-hop integrity is also needed because all PTP messages, except from *Announce* messages, must be verified and regenerated on each hop. MACsec encapsulation is employed, which appends an ICV at the end of the frame. This ICV is computed using pairwise SAKs that are previously negotiated using the MKA protocol.

R4: **End-to-end integrity.** The end-to-end integrity protection is provided with the ICV contained in the Security TLV. This ICV must protect fields of PTP messages that are not modified by intermediate nodes and it is computed using symmetric cryptographic algorithms like the HMAC-SHA256 or the AES-GMAC-128.

R5: **Replay protection.** The replay check is another security mechanism to protect against replay of old packets by an attacker, which generally consist of an ascending packet counter integrated in every message and it is check in the receiver. MACsec SecTAG contains a packet number field to provide replay protection on each link. End-to-end replay protection is also provided with a sequence number in the Security TLV, for example, to avoid injection of old packets with false *correctionField* by an internal attacker that has previously highjacked an intermediate node.

R5: **Unicast key management.** A unicast key management scheme would highly likely be required to provide authentication and authorization services in early stages of authentication protocols. The utilization of external protocols like TLS or IKE could resolve the problem of unicast key distribution, as it was defined at the unicast operation of NTS. Nodes are normally authenticated against a key distribution centre using asymmetric cryptography and, as a result, they obtain a unicast key. IEEE 802.1X-2010 standard also uses EAP methods based on asymmetric cryptography in the first stage of the authentication procedure to stablish pairwise CAKs.

R6: **Multicast key management.** The multicast key management scheme must consist of two clearly separated parts: PTP master and slaves must share one TESLA key for computing the ICV in the Security TLV and the group CAK key must be distributed to all PTP nodes for computing the MACsec ICV. The former may be distributed inband from master to slaves or using a unicast key previously negotiated with a key distribution centre. The latter is distributed from a local key server to all PTP nodes using a modified IEEE 802.1X scheme to transport group CAKs over EAPOL messages.

R7: Sub-microsecond accuracy. The power profile specifies that sub-microsecond synchronization must be achieved within substation environments. The only manner to achieve synchronization accuracies in the range of nanoseconds is utilizing hardware assisted IEEE 1588 solutions, in addition to TC functionality in intermediate nodes. The proposed SoC architecture integrate MACsec hardware units and IEEE 1588 IP cores.

R8: Avoid negative impact on time-critical SAS communications. The hop-by-hop security solution must not negatively affect the performance of time-critical communications in substations, like the transmission of GOOSE and SV messages. These time-critical messages must be transmitted without SecTAG-ICV fields and MACsec crypto units must forward them through the uncontrolled port of the SecY. Also, cut-trough switching in intermediate nodes must be implemented to reduce propagation delays due to congestion conditions in the network.

R9: Avoid throughput bottlenecks. Special attention must be paid to the implementation of cryptography when designing the SoC architecture, in order to avoid throughput bottlenecks. Protecting PTP messages in software could drastically decrease PTP protocol performance, due to CPU resource utilization by cryptographic algorithms. Therefore, MACsec encryption units must be implemented in hardware and specific crypto accelerator cores[1] should be utilized to compute the ICV contained in the Security TLV, as well as to protect MKA frames used to negotiate MACsec key material.

R10: Reduce the probability of external attackers. The utilization of an external hop-by-hop security solution, like MACsec, reduces considerably the risk of external attackers in PTP networks. Since an external attacker does not know the cryptographic keys, it is only exposed to protected traffic with no possibility of neither tampering PTP messages nor generate new traffic it cannot and replayed messages are discarded on destination by the replay protection mechanism. However, these replayed messages can affect network performance and induce congestion conditions if intermediate nodes are saturated with false authentication requests. Moreover, packet delay attacks cannot be eliminated and interception and removal of packets is still possible, which could cause a reduction on the accuracy of the protocol. Additional redundancy techniques should also be included for

[1]This crypto cores are hardware modules that implement efficiently the cryptographic algorithms and they can usually be accessed through the Linux crypto API, for example.

mitigating the risk of delay manipulation by MITM attackers, which is out of the scope of this thesis.

R11: Reduce the probability of internal attackers. Internal attacker can also be minimized but never eliminated. An attacker that highjacks an intermediate node could have access to cryptographic keys and, consequently, intercept a PTP message at some point in the forwarding path inside the node and modify maliciously the *correctionField*, causing the slave to be aligned to a false reference time. If switching and TC functionalities are completely implemented in hardware, an internal injector attacker could only retransmit PTP messages received through the interlink port[2], which are discarded in the destination because of the end-to-end replay protection. End nodes could also become internal attackers if they are highjacked. For instance, a malicious software could intercept PTP messages before security mechanisms being applied in transmission, or after verifying the security checksums in reception. Secure and reliable programming techniques should be employed when implementing the PTP stack, so as to reduce software vulnerabilities.

[2]The interlink is commonly known as the communication port between the internal Ethernet switch and the processor network interface.

Appendix B

PTP over MACsec validation design details

B.1 Hardware implementation on Xilinx Zynq device

The schematics of the ‘Case 1’ and ‘Case 2’ designs implemented with Vivado are captured in Figures B.1 and B.2. Most interesting parts of these block diagrams have been enlarged in Figures B.3, B.4 and B.5.

In both topologies, the AXI DMA scatter/gather, MM2S and S2MM ports were connected to the AXI-HP interface through an AXI interconnect. When implementing ‘Case 1’ topology, another two AXI interconnets were required to connect AXI4-Lite ports from AXI DMA, AXI Ethernet and PTB IP cores to PS AXI-GP master interfaces. Moreover, a Concat IP was used for concatenating all PL interrupts and connect them to the PS General Interrupt Controller (GIC) [163].

In ‘Case 2’ topology, an additional AXI interconnect IP is required in order to connect the AXI4-Lite port of the AXI Stream MACsec IP for configuration purposes.

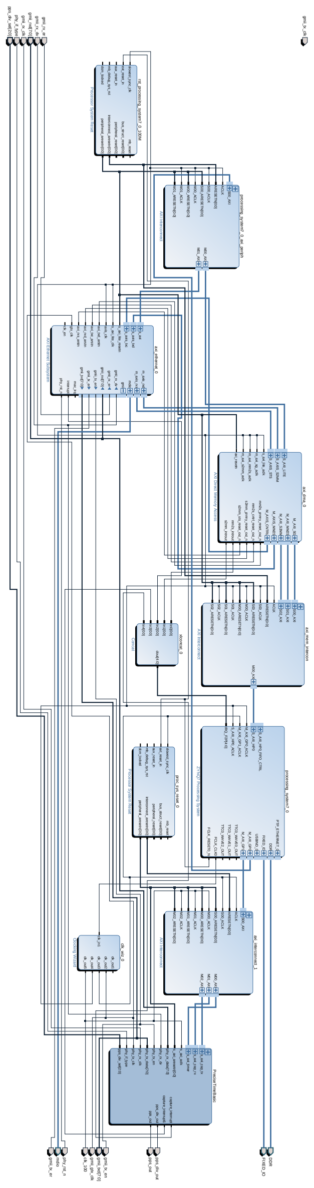


Figure B.1: Vivado block diagram of 'Case 1' topology implemented design

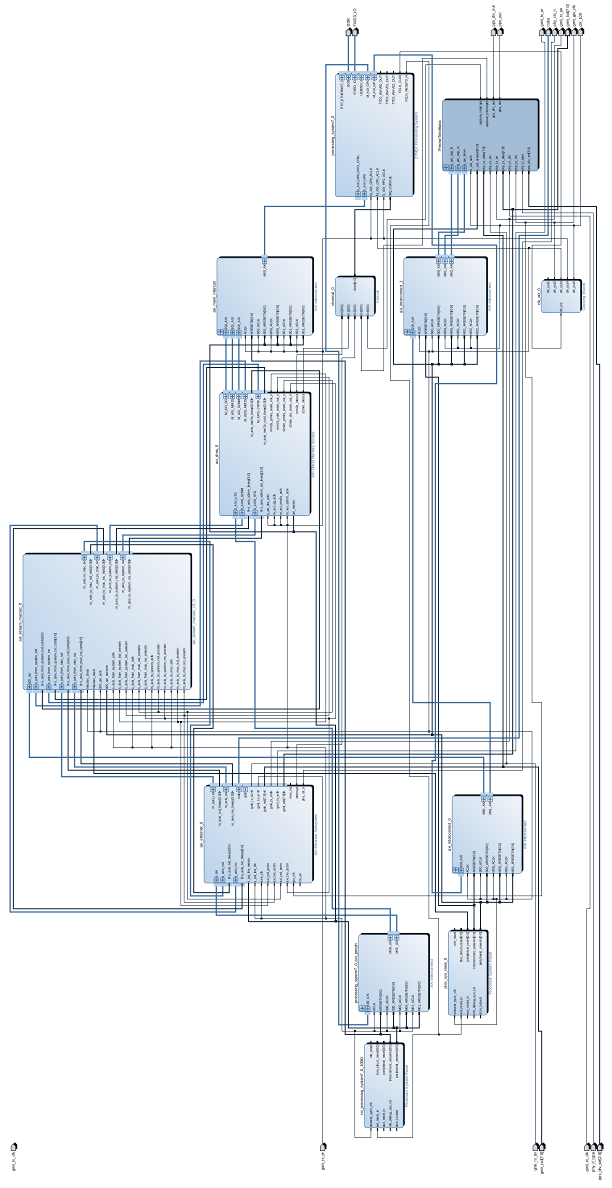
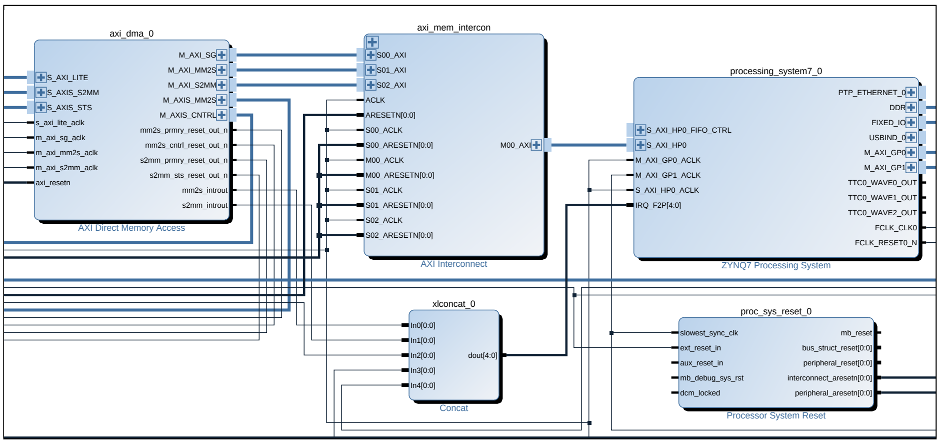
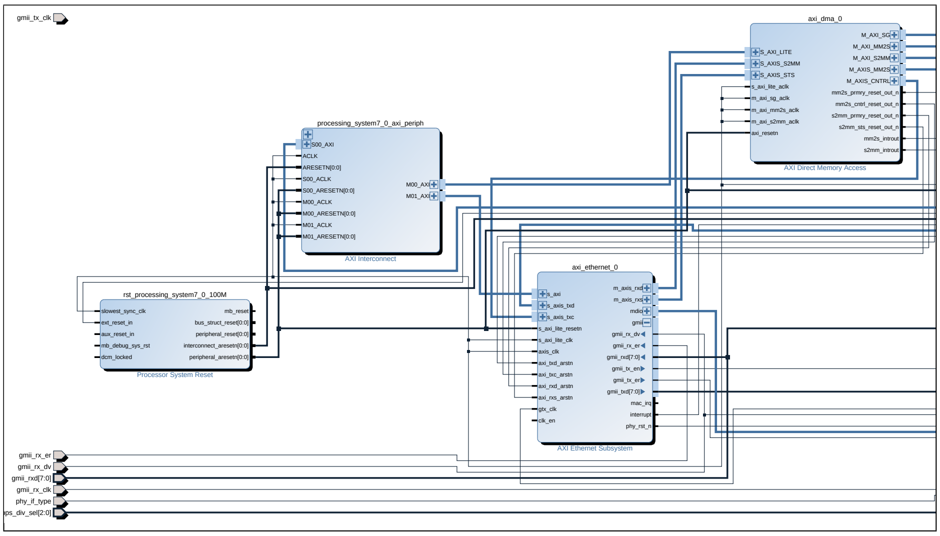


Figure B.2: Vivado block diagram of 'Case 2' topology implemented design



(a) Case 1 AXI DMA connections



(b) Case 1 AXI Ethernet connections

Figure B.3: Vivado block diagrams of AXI DMA and AXI Ethernet Subsystems of 'Case 1' topology

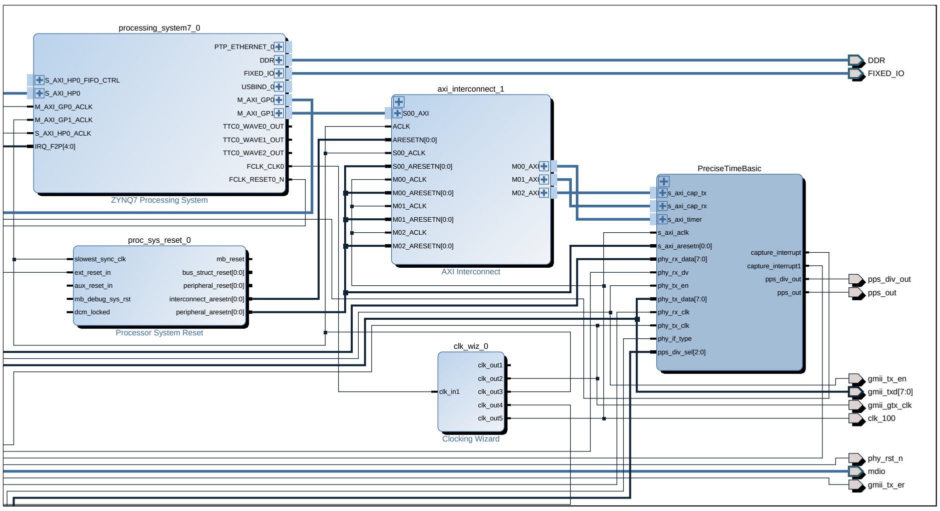


Figure B.4: Vivado block diagram of PTB core of 'Case 1' topology

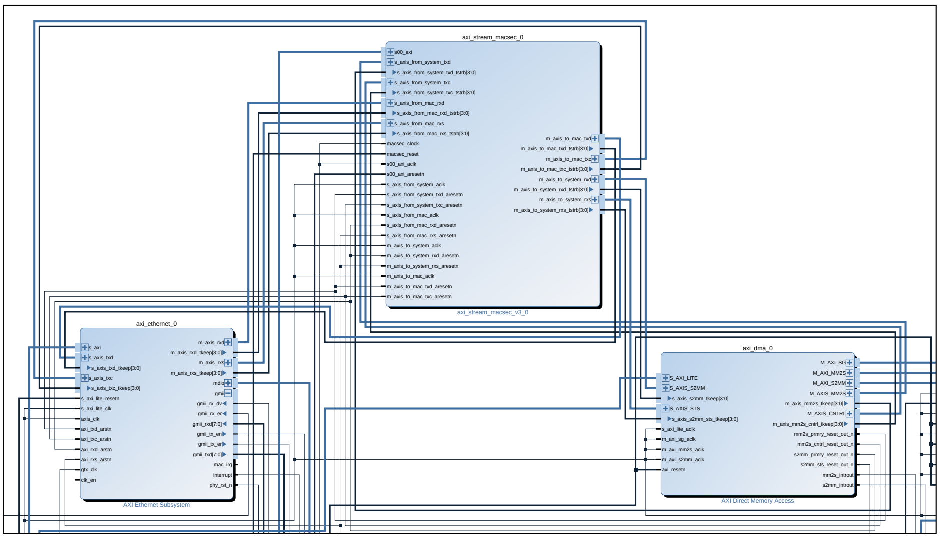


Figure B.5: Vivado block diagrams of AXI Stream MACsec of 'Case 2' topology

B.2 State machine diagrams of the implemented AXI Stream MACsec IP core

In order to validate the proposed SoC architecture using Xilinx Zynq device, the selected MACsec IP core had to be integrated between the AXI Ethernet Subsystem and the AXI DMA. Different state machines were designed in VHDL to implement the required MACsec 'sandwich' responsible for adapting interfaces and the resulted IP was named AXI Stream MACsec IP, as represented in Figure 7.5.

FSM RX number 1 in Figure 7.5 manages AXI stream slave interface and writes incoming Ethernet frames to the corresponding FIFO buffer. Thus, after receiving six control words, as specified in [143], this FSM writes a new 64-bit data block after receiving two followed 32-bit data words from the AXI stream interface. Finally, it saves the number of written blocks and waits until the Algotronix MACsec core has processed the whole frame to notify the AXI stream master interface that it is ready to receive a new frame. In Figure B.6, the state diagram of the FSM RX number 1 is drawn.

FSM number 2 reads FIFO output and provides two 64-bit data words to Algotronix core every 10 or 14 clock cycles, which is the time it takes to process a 128-bit data block with AES-GCM-128 or AES-GCM-256 algorithms. It also generates flow control signals as required by the Algotronix core, which basically are first word, final word and length of the final word. The state diagram of the FSM RX number 2 is shown in Figure B.7.

Figures B.8 and B.9 represent the state diagrams of the two remainder state machines within the receive channel. FSM RX number 3 is responsible for storing in the corresponding FIFO 64-bit output data blocks from the Algotronix core, while it prepares the control words that are required by the AXI stream interface to the AXI DMA. In addition, it causes the buffer to be emptied when it detects that a packet has to be dropped due to a false ICV for example. Otherwise, if the packet is successfully verified, it generates the initialization signals for FSM RX number 4 during last states. This number 4 FSM finally generates the AXI stream control and data signals, which are provided to the AXI DMA core in data words of 32 bits.

In the transmission channel, FSM TX number 1 and FSM TX number 4 are practically identical to FSM RX 1 and FSM RX 4 and the state diagrams are equal to that represented in Figures B.6 and B.9. The state diagram of the FSM TX number 2 is the state diagram of a FSM RX number 2 but duplicated in order to send frames through the controlled or uncontrolled port depending on

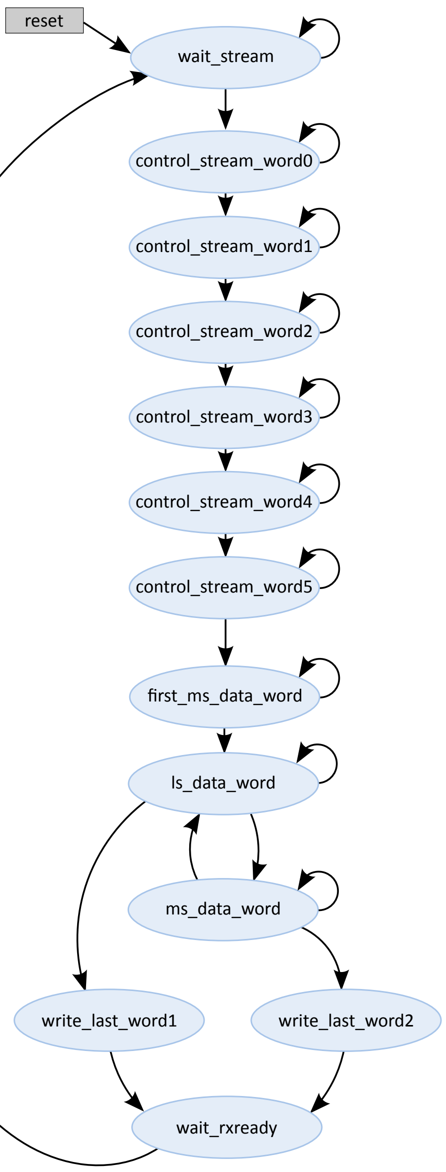


Figure B.6: State diagram of the FSM RX 1

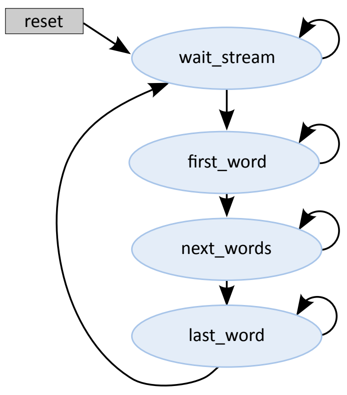


Figure B.7: State diagram of the FSM RX 2

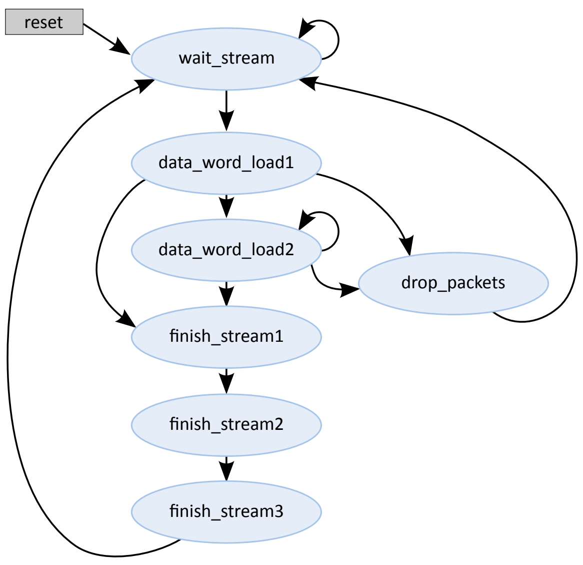


Figure B.8: State diagram of the FSM RX 3

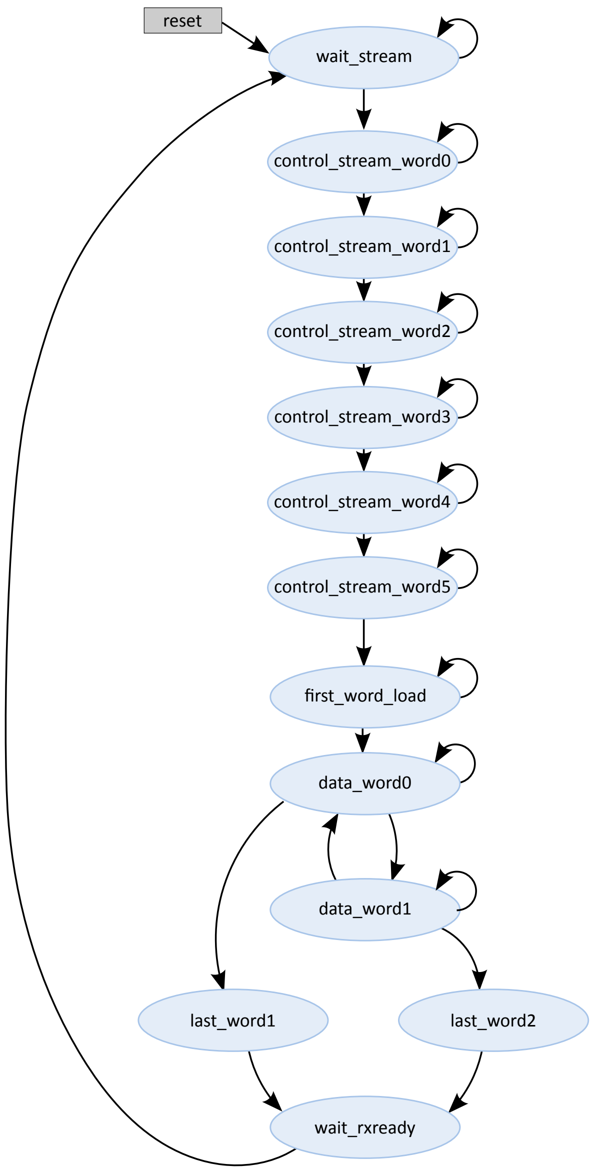


Figure B.9: State diagram of the FSM RX 4

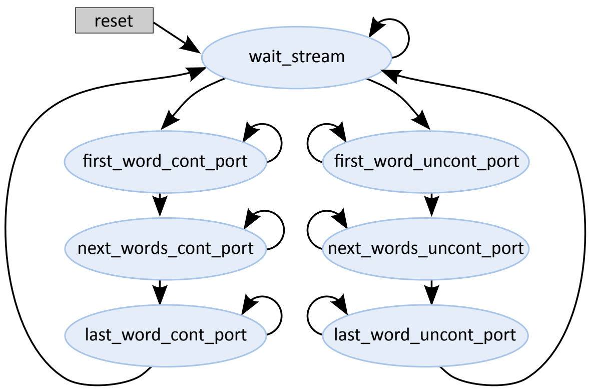


Figure B.10: State diagram of the FSM TX 2

the value of the Ethertype field, as it can be seen in Figure B.10. Thus, only PTP over Ethernet frames, whose Ethertype is 0x88F7 in hexadecimal, are transmitted through the controlled port in this case.

The state diagram of the FSM TX number 3 is represented in Figure B.11. It was simplified from that of the FSM RX 3 by removing replicated *data_word_load* and *finish_stream* states shown in Figure B.8. These states were only needed in the receive channel to gather AXI stream control information required by the AXI DMA.

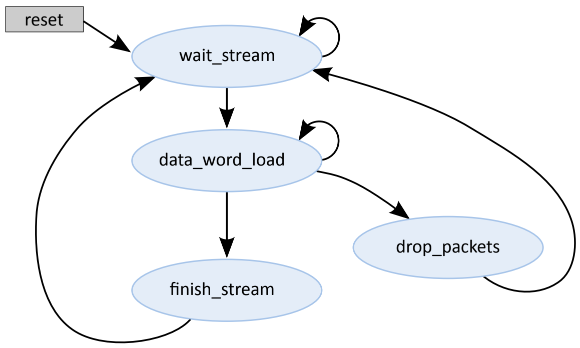


Figure B.11: State diagram of the FSM TX 3

Bibliography

[1] G. Ericsson, "Cyber security and power system communication - essential parts of a smart grid infrastructure," *IEEE Transactions on Power Delivery*, vol. 25, no. 3, pp. 1501–1507, July 2010.

[2] K. Poulsen, "Slammer worm crashed Ohio nuke plant network," August 2003. [Online]. Available: http://www.securityfocus.com/news/6767

[3] D. Kushner, "The real story of stuxnet," *IEEE Spectrum*, vol. 50, no. 3, pp. 48–53, March 2013.

[4] F. Cleveland, "IEC TC57 WG15: IEC 62351 security standards for the power system information infrastructure," International Electrotechnical Commission, White Paper, June 2012. [Online]. Available: http://xanthus-consulting.com/Publications/documents/IEC%20_TC57_WG15_White_Paper.pdf

[5] J. Hoyos, M. Dehus, and T. X. Brown, "Exploiting the GOOSE protocol: a practical attack on cyber-infrastructure," in *Proceedings of the IEEE Globecom Workshops*, December 2012, pp. 1508–1513.

[6] "The smartgrids european technology platform," October 2014. [Online]. Available: http://www.smartgrids.eu/ETPSmartGrids

[7] M. Kanabar, I. Voloh, and D. McGinn, "Reviewing smart grid standards for protection, control, and monitoring applications," in *Proceedings of IEEE PES Innovative Smart Grid Technologies (ISGT)*, January 2012, pp. 1–8.

[8] "ABB review, IEC 61850," August 2010. [Online]. Available: https://library.e.abb.com/public/a56430e1e7c06fdfc12577a00043ab8b/3BSE063756_en_ABB_Review_Special_Report_IEC_61850.pdf

[9] K.-P. Brand, V. Lohmann, and W. Wimmer, *Substation Automation Handbook.* Utility Automation Consulting Lohmann Bremgarten, 2003.

[10] T. Sidhu and P. Gangadharan, "Control and automation of power system substation using IEC 61850 communication," in *Proceedings of IEEE Conference on Control Applications*, August 2005, pp. 1331–1336.

[11] "IEC 61850-8-1 ed2.0 standard, Communicaiton networks and systems for power utility automation - Part 8-1: Specific Communication Service Mapping (SCSM) - Mappings to MMS (ISO 9506-1 and ISO 9506-2) and to ISO/IEC 8802-3," June 2011.

[12] "IEC 61850-9-2 ed2.0 standard, Communication networks and systems for power utility automation - Part 9-2: Specific Communication Service Mapping (SCSM) - Sampled values over ISO/IEC 8802-3," September 2011.

[13] "ISO 9506-1:2003 ed2 standard, Industrial automation systems - Manufacturing Message Specification," August 2003.

[14] "IEC 61850-1 ed2.0 standard, Communication networks and systems for power utility automation - Part 1: Introduction and overview," March 2013.

[15] S. Fuloria, R. Anderson, K. McGrath, K. Hansen, and F. Alvarez, "The protection of substation communications," in *Proceedings of SCADA Security Scientific Symposium*, January 2010.

[16] "Introduction to waterfall unidirectional security gateways: True unidirectionality, true security," August 2011. [Online]. Available: http://waterfall-security.ca/resources/wf-intro-v2.pdf

[17] "IEC/TS 62351-1 ed1.0 standard, Power systems management and associated information exchange - Data and communications security - Part 1: Communication network and system security - Introduction to security issues," May 2007.

[18] "IEC/TS 62351-6 ed1.0 standard, Power systems management and associated information exchange - Data and communication security - Part 6: Security for IEC 61850," June 2007.

[19] S. Fries and R. Falk, "Security considerations for multicast communication in power systems," *International Journal On Advances in Security*, vol. 6, no. 3 & 4, pp. 111–121, 2013.

[20] B. Weis, S. Rowles, and T. Hardjono, "RFC 6407 The group domain of interpretation," October 2011.

[21] T. Dierks and C. Allen, "RFC 2246 The TLS protocol version 1.0," January 1999.

[22] "Final report on the August 14th blackout in the United States and Canada: Causes and recommendations," April 2004. [Online]. Available: https://reports.energy.gov/BlackoutFinal-Web.pdf

[23] F. Steinhauser, C. Riesch, and M. Rudigier, "IEEE 1588 for time synchronization of devices in the electric power industry," in *Proceedings of the IEEE International Symposium on Precision Clock Synchronization for Measurement Control and Communication (ISPCS)*, October 2010, pp. 1–6.

[24] "IRIG Standard 200-04, Overview of IRIG-B time code standard," 2011. [Online]. Available: http://metis.ipfn.ist.utl.pt/@api/deki/files/184/=TN-102_IRIG-B.pdf

[25] "IRIG Standard 200-04, IRIG serial time code formats," September 2004. [Online]. Available: http://www.irigb.com/pdf/wp-irig-200-04.pdf

[26] J. Eidson, *Measurement, Control and Communication using IEEE 1588.* Springer Science & Business Media, 2006.

[27] "Framework and roadmap for smart grid interoperability standards," NIST Special Publication 1108, January 2010. [Online]. Available: http://www.nist.gov/public_affairs/releases/upload/smartgrid_interoperability_final.pdf

[28] "IEEE 1588-2008 Standard for a precision clock synchronization protocol for networked measurement and control systems," July 2008.

[29] "IEEE C37.238-2011 Standard profile for use of IEEE 1588 precision time protocol in power system applications," July 2011.

[30] "IEC 61850-90-4 ed1.0 standard, Communicaiton networks and systems for power utility automation - Part 90-4: Network engineering guidelines," August 2013.

[31] A. Treytl, G. Gaderer, P. Loschmidt, and N. Kerö, "Investigations on security aspects in clock synchronized industrial ethernet," in *Proceedings of the Precise Time and Time Interval (PTTI) Systems and Applications Meeting*, December 2006, pp. 232–240.

[32] A. Treytl, G. Gaderer, B. Hirschler, and R. Cohen, "Traps and pitfalls in secure clock synchronization," in *Proceedings of the IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS)*, October 2007, pp. 18–24.

[33] T. Mizrahi, "Time synchronization security using IPsec and MACsec," in *Proceedings of the IEEE International Symposium on Precision Clock*

Synchronization for Measurement Control and Communication (ISPCS), September 2011, pp. 38–43.

[34] H. Kirrmann, K. Weber, O. Kleineberg, and H. Weibel, "Seamless and low-cost redundancy for substation automation systems (high availability seamless redundancy, HSR)," in *Proceedings of the IEEE Power and Energy Society General Meeting*, July 2011, pp. 1–7.

[35] "IEC 62439-3 ed2.0 standard, Industrial communication networks - High availability automation networks - Part 3: Parallel Redundancy Protocol (PRP) and High-availability Seamless Redundancy (HSR)," July 2012.

[36] J. Araujo, J. Lazaro, A. Astarloa, A. Zuloaga, and N. Moreira, "Duplicate and circulating frames discard methods for PRP and HSR (IEC62439-3)," in *Proceedings of the IEEE Annual Conference on Industrial Electronics Society (IECON)*, November 2013, pp. 4451–4456.

[37] C. De Dominicis, P. Ferrari, A. Flammini, S. Rinaldi, and M. Quarantelli, "On the use of IEEE 1588 in existing IEC 61850-based SASs: current behavior and future challenges," *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 9, pp. 3070–3081, September 2011.

[38] A. Abdul, G. Ng, and P. Lupas, "Integration of HSR and IEEE1588 over Ethernet networks," in *Proceedings of the IEEE International Symposium on Precision Clock Synchronization for Measurement Control and Communication (ISPCS)*, September 2010, pp. 77–82.

[39] "IEC 62439-3 ed3.0 standard, Industrial communication networks - High availability automation networks - Part 3: Parallel Redundancy Protocol (PRP) and High-availability Seamless Redundancy (HSR)," March 2016.

[40] "IEC 61850-9-3 ed1.0 standard, Communication networks and systems for power utility automation - Part 9-3: Precision time protocol profile for power utility automation," May 2016.

[41] H. Kirrmann, C. Honegger, D. Ilie, and I. Sotiropoulos, "Performance of a full-hardware PTP implementation for an IEC 62439-3 redundant IEC 61850 substation automation network," in *Proceedings of the IEEE International Symposium on Precision Clock Synchronization for Measurement Control and Communication (ISPCS)*, September 2012, pp. 1–6.

[42] T. Hardjono and L. R. Dondeti, *Multicast and group security.* Artech House, 2003.

[43] P. Sáiz, "Modelo de establecimiento de sesiones seguras a nivel de enlace

entre estaciones finales en redes ethernet," Ph.D. dissertation, University of the Basque Country UPV/EHU, July 2007.

[44] "IEC 61588 ed2.0 standard, Precision clock synchronization protocol for networked measurement and control systems," February 2009.

[45] H. Weibel, "Technology update on IEEE 1588: The second edition of the high precision clock synchronization protocol," in *Proceedings of Embedded World Conference*, March 2009.

[46] H. Krawczyk, R. Canetti, and M. Bellare, "RFC 2104 HMAC: Keyed-hashing for message authentication," February 1997.

[47] C. Madson and R. Glenn, "RFC 2404 The use of HMAC-SHA-1-96 within ESP and AH," November 1998.

[48] NIST, *Secure Hash Standard (SHS)*, Information Technology Laboratory Std. FIPS PUB 180-4, August 2015.

[49] ——, *The Keyed-Hash Message Authentication Code (HMAC)*, Information Technology Laboratory Std. FIPS PUB 198-1, July 2012.

[50] A. Treytl and B. Hirschler, "Practical application of 1588 security," in *Proceedings of the IEEE International Symposium on Precision Clock Synchronization for Measurement Control and Communication (ISPCS)*, September 2008, pp. 37–43.

[51] ——, "Security flaws and workarounds for IEEE 1588 (transparent) clocks," in *Proceedings of the IEEE International Symposium on Precision Clock Synchronization for Measurement Control and Communication (ISPCS)*, October 2009, pp. 1–6.

[52] C. Onal and H. Kirrmann, "Security improvements for IEEE 1588 Annex K: Implementation and comparison of authentication codes," in *Proceedings of the IEEE International Symposium on Precision Clock Synchronization for Measurement Control and Communication (ISPCS)*, September 2012, pp. 1–6.

[53] N. Moreira, A. Astarloa, U. Kretzschmar, J. Lázaro, and E. Molina, "Securing IEEE 1588 messages with message authentication codes based on the KECCAK cryptographic algorithm implemented in FPGAs," in *Proceedings of the IEEE International Symposium on Industrial Electronics (ISIE)*, June 2014, pp. 1899–1904.

[54] A. Treytl and B. Hirschler, "Securing IEEE 1588 by IPsec tunnels - An analysis," in *Proceedings of the IEEE International Symposium on Preci-*

sion Clock Synchronization for Measurement Control and Communication (ISPCS), October 2010, pp. 83–90.

[55] T. Mizrahi, “Security requirements of time synchronization protocols in packet switched networks,” IETF RFC 7384, TICTOC WG, October 2014.

[56] ——, “A game theoretic analysis of delay attacks against time synchronization protocols,” in *Proceedings of the International IEEE Symposium on Precision Clock Synchronization for Measurement Control and Communication (ISPCS)*, September 2012, pp. 1–6.

[57] ——, “IEEE 1588 security encapsulation proposal: IPsec-based security TLV,” P1588 Working Group Document, April 2014.

[58] K. G. Paterson, “A cryptographic tour of the IPsec standards,” *Information Security Technical Report*, vol. 11, no. 2, pp. 72–81, January 2006.

[59] C. Kaufman, P. Hoffman, Y. Nir, P. Eronen, and T. Kivinen, “Internet key exchange protocol version 2 (IKEv2),” IETF RFC 7296, October 2014.

[60] S. Fries, “Lessons learned from securing IEC 61850 GOOSE/SV,” P1588 Working Group Presentation, April 2014. [Online]. Available: https://ieee-sa.centraldesktop.com/1588/file/31142043/

[61] ——, “Proposal for IEEE 1588v3 security: definition of a security TLV,” P1588 Working Group Document, May 2014.

[62] “GDOI reference implementation primer,” Cisco Systems - Linux Documentation, October 2003. [Online]. Available: http://ftp.unpad.ac.id/orari/library/library-sw-hw/linux-1/sip/vocal/source/gdoi/

[63] “IEEE 802.1AE-2006 standard for local metropolitan area networks: Media Access Control (MAC) security,” June 2006.

[64] L. Ellegaard, “PTP security using MACsec,” P1588 Working Group Presentation, August 2014. [Online]. Available: https://ieee-sa.centraldesktop.com/1588/file/33390811/

[65] D. Sibold, S. Roettger, and K. Teichel, “Network Time Security,” Internet Draft, IETF NTP Working Group, March 2015. [Online]. Available: https://tools.ietf.org/html/draft-ietf-ntp-network-time-security-08

[66] D. Sibold, “Status of Network Time Security (NTS): Applicability for PTP,” P1588 Working Group Presentation, January 2015. [Online]. Available: https://ieee-sa.centraldesktop.com/1588/file/37027677/

[67] A. Perrig, D. Song, R. Canetti, J. Tygar, , and B. Briscoe, "Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast source authentication transform introduction," IETF RFC 4082, June 2005. [Online]. Available: http://www.rfc-editor.org/info/rfc4082

[68] B. Dickerson, "Observations on secure PTP," P1588 Working Group Presentation, March 2014. [Online]. Available: https://ieee-sa.centraldesktop.com/1588/file/30928204/

[69] "Security SC standing document v5," P1588 Working Group Document, April 2015. [Online]. Available: https://ieee-sa.centraldesktop.com/1588/file/39406935/

[70] "Intel® 82576EB gigabit ethernet controller: Datasheet rev. 2.63," Intel Corporation Product Datasheet, December 2011. [Online]. Available: http://www.intel.la/content/www/xl/es/embedded/products/networking/82576eb-gigabit-ethernet-controller-datasheet.html

[71] "DP83640 precision PHYTER - IEEE 1588 precision time protocol transceiver," Texas Instruments Product Datasheet, April 2015. [Online]. Available: http://www.ti.com.cn/cn/lit/ds/symlink/dp83640.pdf

[72] A. García, M. Idirin, G. Corradi, and J. Heighton, "Sub-microsecond synchronization of LAN interconnected systems using Xilinx FPGAs," in *Proceedings of Embedded World Conference*, January 2012.

[73] N. Moreira, A. Astarloa, J. Lazaro, A. Garcia, and E. Ormaetxea, "IEEE 1588 transparent clock architecture for FPGA-based network devices," in *Proceedings of the IEEE International Symposium on Industrial Electronics (ISIE)*, May 2013.

[74] J. Eidson, A. Fernandez, B. Hamilton, J. Naous, and D. Vook, "Spider transparent clock," in *Proceedings of the IEEE International Symposium on Precision Clock Synchronization for Measurement Control and Communication (ISPCS)*, 2008, pp. 7–11.

[75] F.-J. Götz, R. Knoerzer, and S. Schüler, "Methods and devices for sending transmission-time or reception-time information for a transmitted or received message," US Patent 8,332,867, December 2012. [Online]. Available: https://www.google.ch/patents/US20090013330

[76] "Timekeeper client/server software," FSMLabs Product Webpage, June 2016. [Online]. Available: http://www.fsmlabs.com/timekeeper/

[77] "Domain time II," Greyware Automation Products, Inc. Product

Webpage, June 2016. [Online]. Available: https://www.greyware.com/software/domaintime/

[78] W. Owczarek, “Precision time protocol daemon,” SourceForge Project, June 2016. [Online]. Available: https://sourceforge.net/projects/ptpd/

[79] F. Winquist and O. Abdilameer, “Time synchronization system, investigation and implementation proposal,” Master’s thesis, Faculty of Engineering, LTH, Lund University, August 2015.

[80] “RTS IEEE 1588 network stack,” Real Time Systems GmbH Product Webpage, June 2016. [Online]. Available: https://www.real-time-systems.com/ieee_1588/index.php

[81] “IEEE 1588 PTP protocol software,” IXXAT Product Webpage, June 2016. [Online]. Available: http://www.ixxat.com/products/products-industrial/time-syncronization/ieee-1588-ptp-protocol-software

[82] “XR7 PTP,” Flexibilis Product Webpage, June 2016. [Online]. Available: http://www.flexibilis.com/products/xr7-ptp/

[83] “syn1588 PTP stack,” Oregano Systems Product Webpage, June 2016. [Online]. Available: http://www.oreganosystems.at/?page_id=85

[84] R. Cochran, “The linux PTP project,” SourceForge Project, June 2016. [Online]. Available: http://linuxptp.sourceforge.net/

[85] “PTP/IEEE 1588 protocol software,” Zurich University Product Webpage, June 2016. [Online]. Available: https://www.zhaw.ch/en/engineering/institutes-centres/ines/produkte-und-dienstleistungen/ptp-ieee-1588/ptp-software/

[86] K. Correll, N. Barendt, and M. Branicky, “Design considerations for software only implementations of the IEEE 1588 precision time protocol,” in *Proceedings of the Conference on IEEE-1588 Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, October 2005.

[87] P. Ohly, “hw-2008-11-07-lci-2008: PTPd 1.0.0 with patches for hardware time stamping,” GitHub Project, January 2009. [Online]. Available: https://github.com/pohly/ptpd/releases/tag/hw-2008-11-07-lci-2008

[88] P. Ohly, D. N. Lombard, and K. B. Stanton, “Hardware assisted precision time protocol. Design and case study,” in *Proceedings of LCI International Conference on High-Performance Clustered Computing. Urbana, IL, USA: Linux Cluster Institute*, 2008, pp. 121–131.

[89] R. Cochran and C. Marinescu, "Design and implementation of a PTP clock infrastructure for the linux kernel," in *Proceedings of the IEEE International Symposium on Precision Clock Synchronization for Measurement Control and Communication (ISPCS)*, September 2010, pp. 116–121.

[90] P. Ohly, "Precision time protocol - temporary fork with support for hardware timestamping," GitHub Project, February 2009. [Online]. Available: https://github.com/pohly/ptpd

[91] R. Cochran, C. Marinescu, and C. Riesch, "Synchronizing the linux system time to a PTP hardware clock," in *Proceedings of the IEEE International Symposium onPrecision Clock Synchronization for Measurement Control and Communication (ISPCS)*, September 2011, pp. 87–92.

[92] "F28M36x Concerto™ microcontrollers," Texas Instruments Product Datasheet, October 2015. [Online]. Available: http://www.ti.com/lit/ds/symlink/f28m36h53c2.pdf

[93] "Intel® IXP43X product line of network processors," Intel Corporation Product Webpage, December 2008. [Online]. Available: http://www.intel.com/content/www/us/en/intelligent-systems/previous-generation/ixp43x-product-line-network-processors-datasheet.html

[94] Atmel, "SAM4E series," Atmel Product Datasheet, March 2016. [Online]. Available: http://www.atmel.com/Images/Atmel-11157-32-bit-Cortex-M4-Microcontroller-SAM4E16-SAM4E8_Datasheet.pdf

[95] X. Jiang, "High Availability Seamless Ring Protocol Implementation in FPGA," Master's thesis, Swiss Federal Institute of Technology ETH Zurich, February 2009.

[96] J. Araujo, J. Lázaro, A. Astarloa, A. Zuloaga, and J. Garate, "PRP and HSR for high availability networks in power utility automation: A method for redundant frames discarding," *IEEE Transactions on Smart Grid*, vol. PP, no. 99, pp. 1–8, 2015.

[97] "PreciseTimeBasic: IEEE 1588-2008 PTPv2 IP core," System on Chip Engineering Product Webpage, June 2016. [Online]. Available: http://soc-e.com/products/precisetimebasic-ieee-1588-2008-v2-ptp-ip-core/

[98] "1588Tiny: IEEE 1588 v2 slave only hard IP core," System on Chip Engineering Product Webpage, June 2016. [Online]. Available: http://soc-e.com/products/1588tiny-ieee-1588-v2-slave-only-hard-ip-core/

[99] "MES – Managed Ethernet Switch IP core," System on Chip

Engineering Product Webpage, June 2016. [Online]. Available: http://soc-e.com/mes-managed-ethernet-switch-ip-core/

[100] "UES - Unmanaged Ethernet Switch IP core," System on Chip Engineering Product Webpage, June 2016. [Online]. Available: http://soc-e.com/products/reliable-ethernet/unmanagedethernet-switch-ip-core/

[101] "HSR-PRP Switch IP core," System on Chip Engineering Product Webpage, June 2016. [Online]. Available: http://soc-e.com/products/hsr-prp-switch-ip-core-all-hardware-low-latency-switch-for-fpgas/

[102] "Advanced Flexibilix Ethernet Controller (AFEC)," Flexibilis Product Webpage, June 2016. [Online]. Available: http://www.flexibilis.com/products/afec/

[103] "Flexibilis Ethernet Switch (FES)," Flexibilis Product Webpage, June 2016. [Online]. Available: http://www.flexibilis.com/products/fes/

[104] "Flexibilis Redundant Switch (FRS)," Flexibilis Product Webpage, June 2016. [Online]. Available: http://www.flexibilis.com/products/frs/

[105] "Flexibilis Deterministic Switch (FDS)," Flexibilis Product Webpage, June 2016. [Online]. Available: http://www.flexibilis.com/products/deterministic-switch/

[106] "IEEE 802.1X-2010 standard, Port-based Network Access Control," IEEE Computer Society, February 2010.

[107] "NIST SP 800-108, Recommendation for key derivation using pseudorandom functions," NIST Special Publication, October 2009.

[108] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, and H. Levkowetz, "Extensible Authentication Protocol (EAP)," IETF RFC 3748, June 2004.

[109] F. Rodríguez-Henríquez, A. D. Pérez, and Ç. K. Koç, *Cryptographic algorithms on reconfigurable hardware.* Springer, 2006.

[110] C. Paar and J. Pelzl, *Understanding Crytography.* Springer, 2010.

[111] J.-P. Thibault, "Deigning MACsec in," Ethernet Technology Summit Conference Presentation, April 2013. [Online]. Available: http://www.ethernetsummit.com/English/Collaterals/Proceedings/2013/20130404_A202_Thibault.pdf

[112] "Open1x - ieee 802.1x open source implementation," Open1X Project Webpage, June 2016. [Online]. Available: http://open1x.sourceforge.net/

[113] J. Malinen, “hostapd: IEEE 802.11 AP, IEEE 802.1X/WPA/WPA2/EAP/RADIUS authenticator,” Project Webpage, June 2016. [Online]. Available: https://w1.fi/hostapd/

[114] ——, “Linux WPA/WPA2/IEEE 802.1X supplicant,” Project Webpage, June 2016. [Online]. Available: https://w1.fi/wpa_supplicant/

[115] “The FreeRADIUS Project,” FreeRADIUS Project Webpage, June 2016. [Online]. Available: http://freeradius.org/

[116] J. Malinen, “Developers’ documentation for wpa_supplicant and hostapd,” September 2015. [Online]. Available: http://w1.fi/wpa_supplicant/devel/index.html

[117] “IEEE 802.11i-2004 Standard for information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Amendment 6: Medium Access Control (MAC) Security Enhancements,” July 2004.

[118] “IEEE 802.1X-2004 standard, Port-based Network Access Control,” December 2004.

[119] J. Vollbrecht, P. Eronen, N. Petroni, and Y. Ohba, “State machines for Extensible Authentication Protocol (EAP) peer and authenticator,” IETF RFC 4137, August 2005.

[120] “GCM/AES MACsec (IEEE 802.1AE) and FC-SP Cores GCM1/GCM2/GCM3,” IP Cores Product Webpage, June 2016. [Online]. Available: http://www.ipcores.com/MACsec-802.1AE-AES-GCM-Core.htm

[121] “AES-GCM cores,” Helion Product Webpage, June 2016. [Online]. Available: http://www.heliontech.com/aes_gcm.htm

[122] “Scalable AES-GCM/GMAC/CTR IP core,” Barco Silex Product Webpage, June 2016. [Online]. Available: http://www.barco-silex.com/ip-cores/encryption-engine/BA415

[123] “AES-GCM cores,” Algotronix Product Webpage, June 2016. [Online]. Available: http://www.algotronix.com/engineering/aes_gcm.html

[124] “Ethernet PHYs - Physical layer ICs for gigabit Ethernet metworking equipment,” Microsemi Product Webpage, June 2016. [Online]. Available: http://www.microsemi.com/products/ethernet-solutions/ethernet-phys

[125] M. Nuss, "Securing Ethernet networks," Ethernet Technology Summit Conference Presentation, April 2014. [Online]. Available: http://www.ethernetsummit.com/English/Collaterals/Proceedings/2014/20140430_A102_Nuss.pdf

[126] S. Singer, "High throughput, low latency Ethernet security with MACsec," Ethernet Technology Summit Conference Presentation, April 2014. [Online]. Available: http://www.ethernetsummit.com/English/Collaterals/Proceedings/2014/20140430_A102_Singer.pdf

[127] "MACsec cores," Algotronix Product Webpage, June 2016. [Online]. Available: http://www.algotronix-store.com/category_s/24.htm

[128] "Synopsys expands security solutions with acquisition of Elliptic Technologies," Synopsys News Release, June 2015. [Online]. Available: http://news.synopsys.com/2015-06-29-Synopsys-Expands-Security-Solutions-with-Acquisition-of-Elliptic-Technologies

[129] B. Branscomb, "Physical layer processing of timestamps and MAC security," Patent Request, May 2013. [Online]. Available: https://www.google.com/patents/US20130114601

[130] "Timing over MACsec," Microsemi Document Portal, November 2015. [Online]. Available: http://www.microsemi.com/document-portal/doc_view/135617-timing-over-macsec

[131] "Multiple SecY IEEE 802.1ae MACSEC for 1 Gbit Ethernet," Xilinx Intellectual Property Library, June 2016. [Online]. Available: http://www.xilinx.com/products/intellectual-property/1-4pilr2.html

[132] P. Dillien and T. Kean, "MACsec IP improves data center security," *XCell Journal*, vol. 90, pp. 28–33, 2015.

[133] J. McGhee and M. Goraj, "Smart high voltage substation based on IEC 61850 process bus and IEEE 1588 time synchronization," in *Proceedings of the IEEE International Conference on Smart Grid Communications (SmartGridComm)*, October 2010.

[134] D. McGrew and J. Viega, "The galois/counter mode of operation (GCM)," January 2004, submission to NIST modes of operation process. [Online]. Available: http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/proposedmodes/gcm/gcm-spec.pdf

[135] K. Gaj and P. Chodowiec, *FPGA and ASIC implementations of AES*. Springer, 2009, pp. 235–294.

[136] C. Benvenuti, *Understanding Linux network internals.* "O'Reilly Media, Inc.", 2006.

[137] "DMA v7.1," Xilinx LogiCORE IP Product Guide, November 2015.

[138] U. Cherukupally, "Zynq-7000 AP SoC real time interrupt latency reference design and demo tech tip," Xilinx Wiki Tech Tip, September 2014. [Online]. Available: http://www.wiki.xilinx.com/Zynq-7000+AP+SoC+-+RealTime+-+InterruptLatency+Reference+Design+and+Demo+Tech+Tip

[139] C. Herber, A. Saeed, and A. Herkersdorf, "Design and evaluation of a low-latency AVB Ethernet endpoint based on ARM SoC," in *Proceedings of the IEEE International Conference on Embedded Software and Systems (ICESS)*, August 2015, pp. 1128–1134.

[140] A. Astarloa, J. Lázaro, U. Bidarte, J. A. Araujo, and N. Moreira, "FPGA implemented cut-through vs store-and-forward switches for reliable ethernet networks," in *Proceedings of the Conference on Design of Circuits and Integrated Circuits (DCIS)*, November 2014, pp. 1–6.

[141] P. Emmerich, D. Raumer, A. Beifuß, L. Erlacher, F. Wohlfart, T. M. Runge, S. Gallenmüller, and G. Carle, "Optimizing latency and CPU load in packet processing systems," in *Proceedings of the International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*, July 2015, pp. 1–8.

[142] "Xilinx UG858 (v1.10), Zynq-7000 all programmable SoC technical reference manual," February 2015. [Online]. Available: http://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf

[143] "AXI Ethernet subsystem v6.2," Xilinx Product Guide, October 2014.

[144] "Zynq-7000 all programmable SoC overview," Xilinx Product Specification, January 2016.

[145] "Zedboard hardware user's guide," AVNET, January 2014. [Online]. Available: http://zedboard.org/sites/default/files/documentations/ZedBoard_HW_UG_v2_2.pdf

[146] *TB-FMCL-GLAN-B hardware user guide*, Inrevium, June 2012.

[147] "The Linux PTP project," SourceForge Project Webpage. [Online]. Available: http://linuxptp.sourceforge.net/

[148] "Linux with HDMI video output on the ZED and ZC702, ZC706 boards," Analog Devices Wiki, December 2012. [On-

line]. Available: https://wiki.analog.com/resources/eval/user-guides/ad-fmcomms1-ebz/software/linux/zynq

[149] U. Cherukupally, "Zynq-7000 AP SoC IEEE1588 v2 precision time protocol tech tip," Xilinx Wiki Tech Tip, 2014. [Online]. Available: http://www.wiki.xilinx.com/Zynq-7000+AP+SoC+IEEE1588+v2+Precision+Time+Protocol+Tech+Tip

[150] "PTP hardware clock infrastructure for Linux," Kernel Documentation, May 2015. [Online]. Available: https://www.kernel.org/doc/Documentation/ptp/ptp.txt

[151] "Real Time Clock (RTC) drivers for Linux," Kernel Documentation, March 2016. [Online]. Available: https://www.kernel.org/doc/Documentation/rtc.txt

[152] "RTC class/drivers configuration," Kernel Configuration File. [Online]. Available: http://lxr.free-electrons.com/source/drivers/rtc/Kconfig?v=3.13

[153] "GPIO user space app," Xilinx Wiki, March 2013. [Online]. Available: http://www.wiki.xilinx.com/GPIO+User+Space+App

[154] "Linux GPIO driver," Xilinx Wiki, August 2015. [Online]. Available: http://www.wiki.xilinx.com/Linux+GPIO+Driver

[155] "Precise time basic ZYNQ edition: IEEE 1588-2008 v2 PTP IP core," System on Chip Engineering Product Webpage, System-on-Chip Engineering (SoC-e), June 2016. [Online]. Available: http://soc-e.com/products/precisetimebasic-zynq-edition-ieee-1588-2008-v2-ptp-ip-core/

[156] *LANTIME M600/MRS/PTP multi reference source NTP server with PTP option*, Meinberg Radio Clocks GmbH and Co. KG, June 2015. [Online]. Available: https://www.meinbergglobal.com/download/docs/manuals/english/ltos_6-16.pdf

[157] "IEC 61850 and IEC 62439 technical interoperability test," CIGRE Pre-Test Document, July 2014. [Online]. Available: http://www.ucaiug.org/Meetings/CIGRE_2014/Bilbao%20pretest%20documents/iec61850_iec62439_TECHNICAL_INTEROPERABILITY_TEST_BILBAO_140718.pdf

[158] "ZC702 evaluation board for the Zynq-7000 XC7Z020 all programmable SoC," Xilinx User Guide, September 2015, uG850.

[159] "CPPS-Gate40 sensor," System on Chip Engineering Product Webpage, June 2016. [Online]. Available: http://soc-e.com/intelligent-g/

[160] X. Corp., "PS and PL Ethernet performance and jumbo frame support with PL Ethernet in the Zynq-7000 AP SoC," Application Note, December 2015. [Online]. Available: http://www.xilinx.com/support/documentation/application_notes/xapp1082-zynq-eth.pdf

[161] ——, "Leveraging data-mover IPs for data movement in Zynq-7000 AP SoC Systems," White paper, January 2015. [Online]. Available: http://www.xilinx.com/support/documentation/application_notes/xapp1082-zynq-eth.pdf

[162] "Linux user mode pseudo driver," Xilinx Wiki, April 2013. [Online]. Available: http://www.wiki.xilinx.com/Linux+User+Mode+Pseudo+Driver

[163] X. Corp., "LogiCORE IP Concat v2.1," Xilinx Product Brief, April 2016, pB041.