eman ta zabal zazu

Universidad del País Vasco
Euskal Herriko Unibertsitatea

Escuela Técnica Superior de Ingeniería de Bilbao

Departamento de Tecnología Electrónica

TESIS DOCTORAL

System-on-Programmable-Chip Architecture to Secure Real Time Traffic in the Smart Grid

Autor: Mikel Rodriguez Enriquez

Directores: Dr. Jesús Lázaro Arrotegui
Dr. Armando Astarloa Cuéllar

Bilbao, Octubre 2021

Agradecimientos

Me gustaría aprovechar estas líneas para dedicar un especial agradecimiento a aquellas personas que me han ayudado durante el desarrollo de esta tesis y sin las cuales no habría sido posible.

En primer lugar, me gustaría dar las gracias a mis dos directores de tesis Jesús Lázaro y Armando Astarloa por ayudarme a establecer la dirección del trabajo de investigación realizado. Los resultados obtenidos son de gran interés para un sector, el eléctrico, que cada vez está cobrando una mayor relevancia. Además, me gustaría agradecerles su paciencia e insistencia en aquellos momentos en los que no conseguía avanzar con la investigación tan rápido como me hubiera gustado.

En segundo lugar, me gustaría agradecer a la empresa System-on-Chip engineering (SoC-e) la oportunidad de poder realizar el desarrollo de la tesis en un entorno industrial. Los conocimientos adquiridos durante este tiempo han sido de gran utilidad para incorporarlos como aportaciones de la tesis y han sido de valor para mi crecimiento profesional y personal. Dentro de SoC-e quisiera dar las gracias a todos mis compañeros que en algún momento me hayan ayudado con sus conocimientos. En especial, a Mikel Idirin y Mar González por su disponibilidad para ayudarme con cualquier problema, consulta, duda o trámite, tanto personal como relacionado con la tesis.

Finalmente, dar las gracias a mi familia y amigos por haberme apoyado en todo momento y haber comprendido y aceptado aquellos momentos en los que no he estado tan presente como me hubiera gustado.

En Bilbao, a 19 de Octubre de 2021

Resumen

Las redes de generación y distribución de energía tradicionales hacían uso de sistemas propietarios de comunicaciones que presentaban grabes problemas de interoperabilidad entre dispositivos de distintos proveedores. A fin de estandarizar las comunicaciones, protocolos y modelos de datos que permitiesen alcanzar los requisitos de flexibilidad, robustez y eficiencia de las redes eléctricas de nueva generación, denominadas Smart Grid, el organismo internacional International Electrotechnical Commission (IEC) introdujo la familia de estándares IEC 61850. Los requisitos de acceso y control remoto de los sistemas Substation Automation System (SAS) hicieron necesaria la interconexión de las redes de comunicaciones privadas de las subestaciones a las redes públicas. De esta forma, con las nuevas posibilidades que introdujeron los nuevos sistemas de comunicaciones digitales, también se introdujeron nuevas vulnerabilidades que debían ser atendidas para garantizar la operativa de este tipo de infraestructuras.

En las últimas décadas han sido numerosos los ciberataques sufridos a nivel mundial por infraestructuras de generación y distribución de energía eléctrica. Atendiendo a estos acontecimientos, se publica el estándar de seguridad IEC 62351-6 para la protección de las comunicaciones IEC 61850 en SAS. Tras someterse al análisis de la comunidad investigadora y la industria, se publican los primeros resultados de las pruebas de implementación del estándar IEC 62351-6 en dispositivos SAS. Los resultados muestran como la protección de las comunicaciones de supervisión, control y transmisión de medidas de corriente y tensión eléctrica entre dispositivos Intelligent Electronic Devices (IEDs) suponen un reto por resolver. Dichas comunicaciones se realizan mediante mensajes Generic Object Oriented Substation Events (GOOSE) y Sampled Values (SV) con estrictos requisitos temporales. Por lo tanto, la protección de este tipo de comunicaciones en SAS supone una prioridad para garantizar el correcto funcionamiento de las infraestructuras eléctricas y que sin embargo está aún por resolver.

En la presente tesis, se pretende proponer una solución que permita proteger

los mensajes IEC 61850 GOOSE y SV sin comprometer sus estrictos requisitos temporales y cuya implementación sea viable en dispositivos SAS. Para ello, se aborda dicho problema mediante la definición de una arquitectura digital modular que, gracias a la aceleración hardware de los procesos criptográficos, sea capaz de proporcionar integridad, autenticidad y confidencialidad a las comunicaciones con requisitos de tiempo real en SAS.

En primer lugar, se presentará el estado actual de la tecnología de comunicaciones y ciberseguridad en subestaciones eléctricas. En particular, se describirá la problemática que supone la protección de los mensajes GOOSE y SV. A continuación, se realizará un análisis en profundidad de las soluciones existentes para la protección de este tipo de comunicaciones y se mostrará como no existe ninguna propuesta que sea capaz de llevar a cabo esta tarea. En segundo lugar, como solución al problema, se propondrá una arquitectura hardware flexible y adaptable. Además, como elemento habilitador de la arquitectura se presentará un Intellectual Property core (IP) Advanced Encryption Standard (AES) Galois/Counter Mode (AES-GCM) con baja latencia, alta capacidad de procesamiento y con capacidad de configuración del balance de uso de recursos lógicos y rendimiento.

Para finalizar, se realizará una validación de la arquitectura hardware propuesta en laboratorio mediante técnicas de simulación y pruebas en hardware real. Los resultados obtenidos serán comparados con las últimas soluciones disponibles en la literatura. Adicionalmente, se evaluará la viabilidad de implementar la solución propuesta en dispositivos SAS.

Abstract

Traditional power generation and distribution systems used to make use of proprietary communication systems that avoided interoperability among devices from different vendors. To standardize the communication systems, protocols, and data models that enabled achieving the requisites in terms of flexibility, sturdiness, and efficiency of the new generation electric facilities, commonly named Smart Grid, the international organism International Electrotechnical Commission (IEC) introduced the family of standards IEC 61850. Remote access and control requisites of modern Substation Automation System (SAS) generated the need of interconnecting substation private networks to publicly available networks. Therefore, with the new possibilities introduced by modern digital communication systems, new cybersecurity threads were also introduced that needed to be addressed to guarantee the operative of this type of facilities.

There have been several cyber-attacks targeted at power generation and distribution facilities in the last decade all over the world. To address these events, the security standard IEC 62351 was published to protect IEC 61850 communications in SAS. Shortly, the analysis carried out by the research and industrial community of the security mechanisms defined in IEC 62351 standard was published. Results show that the protection of the communications for control, supervision, and transmissión of measured sampled current and voltage values among Intelligent Electronic Devices (IEDs) is a challenge that needs to be solved. These types of communications make use of Generic Object Oriented Substation Events (GOOSE) and Sampled Values (SV) messages with stringent timing requisites. Thus, protection of these types of communications in SAS supposes a priority to guarantee proper behavior of power generation and distribution facilities.

This thesis aims to propose a solution to protect IEC 61850 GOOSE and SV messages without compromising their stringent timing requirements and suitable for its usage in SAS. This challenge is carried out through the definition of a digital and modular architecture that, making use of hardware acceleration of

cryptographic processes, can provide integrity, authenticity, and confidentiality to the communications with stringent timing requirements in SAS.

First, a detailed analysis of the current state of the art in communication and cybersecurity technologies for substations will be carried out. Specifically, the problems to protect communications based on GOOSE and SV messages will be discussed. Next, current solutions to protect these types of communications will be studied, showing that there is not any solution in the literature able to fulfill this task. Secondly, as the solution to the problem presented, a flexible and adaptable hardware architecture will be proposed. Furthermore, as an enabling element, an Intellectual Property core (IP) Advanced Encryption Standard (AES) Galois/Counter Mode (AES-GCM) with low latency, high processing power, and configurable to select the balance between area usage of the System-on-Programmable-Chip (SoPC) and the performance provided will be presented.

Finally, the proposed architecture will be validated in the laboratory employing simulation techniques and specific tests over real hardware. Results will be compared with the latest solutions available in the literature. Additionally, the viability to implement the proposed solution in SAS will be analyzed.

Índice general

Índice de figuras

Índice de cuadros

Lista de Acrónimos

AAA	*Authentication, Authorization and Accounting*
AAD	*Additional Authenticated Data*
ACSI	*Abstract Communication Service Interface*
AES	*Advanced Encryption Standard*
AES-GCM	*AES Galois/Counter Mode*
AES-GMAC	*AES Galois Message Authentication Code*
AMBA	*Advanced Microcontroller Bus Architecture*
APDU	*Application Protocol Data Unit*
API	*Application Programming Interface*
APPID	*Application Identifier*
ARM	*Advanced RISC Machine*
ASIC	*Application-Specific Integrated Circuit*
ASN.1	*Abstract Syntax Notation One*
AXI	*Advanced eXtensible Interface*
AXI-S	*Advanced eXtensible Interface Stream*
BER	*Basic Encoding Rules*
BRAM	*Block Random Access Memory*
CBC	*Cipher Block Chaining*
CBC-MAC	*CBC Message Authentication Code*
CCM	*Cipher block chaining - message authentication code*
CDC	*Common Data Classes*
CLI	*Command Line Interface*
CMAC	*Cipher-based Message Authentication Code*
CMIP	*Common Management Information Protocol*
CP	*Controlled Port*
CPPS	*Cyber-Physical Production Systems*
CPU	*Central Processing Unit*
CRC	*Cyclic Redundancy Check*

CTs/VTs	*Current and Voltage Transformers*
DDR	*Double Data Rate memory*
DMA	*Direct Memory Access*
DNP3	*Distributed Network Protocol version 3*
DoS	*Denial of Service*
DSP	*Digital Signal Processor*
DUT	*Device Under Test*
E&M	*Encrypt and MAC*
EAX	*Encrypt-then-Authenticate-then-translate*
ECDSA	*Elliptic Curve Digital Signature Algorithms*
EPDU	*Extended Protocol Data Unit*
EtM	*Encrypt-then-MAC*
FCS	*Frame Check Sequence*
FF	*Flip-Flop*
FIFO	*First In First Out*
FMC	*FPGA Mezzanine Card*
FPGA	*Field Programmable Gate Arrays*
FSM	*Finite State Machine*
GCM	*Galois Counter Mode*
GDOI	*Group Domain of Interpretation*
GF	*Galois Field*
GMAC	*Gigabit Media Access Controller*
GMII	*Gigabit Medium Independent Interface*
GOOSE	*Generic Object Oriented Substation Events*
GPIO	*General Purpose Input Output*
GSE	*Generic Substation Events*
GSSE	*Generic Substation State Events*
HMAC	*keyed-Hash Message Authentication Code*
HMI	*Human Machine Interface*
HPS	*HSR-PRP Switch*
HSR	*High-availability Redundancy*
IEC	*International Electrotechnical Commission*
IDS	*Intrusion Detection Systems*
IED	*Intelligent Electronic Device*
IEEE	*Institute of Electrical and Electronics Engineering*
IETF	*Internet Engineering Task Force*
IFG	*Inter Frame Gap*
IO	*Input/Output*
IP	*Intellectual Property core*
IPsec	*Internet Protocol security*

IPv4	*Internet Protocol version 4*
IPv6	*Internet Protocol version 6*
IT	*Information Technology*
IV	*Initialization Vector*
KDC	*Key Distribution Center*
KOA	*Karatsuba-Ofman Algorithm*
LAN	*Local Area Network*
LLC	*Logical Link Control protocol*
LUT	*Look-Up Table FPGA component*
MAC	*Media Access Control*
MACsec	*Media Access Control Security*
MII	*Media Independent Interface*
MITM	*Man-In-The-Middle attack*
MMS	*Manufacturing Message Specification*
MPDU	*MAC Protocol Data Unit*
MtE	*MAC then Encrypt*
MTU	*Master Terminal Unit*
MU	*Merging Unit*
NAT	*Network Address Translation*
NIC	*Network Interface Card*
NIST	*National Institute of Standards and Technology*
NTP	*Network Time Protocol*
OMAC	*One-key MAC*
OSI	*Open Systems Interconnection*
OS	*Operating System*
OT	*Operational Technology*
PC	*Personal Computer*
PDU	*Protocol Data Unit*
PHY	*Physical Interface Transceiver*
PKI	*Public Key Infrastructure*
PL	*Zynq Programmable Logic section*
PN	*Packet Number*
PPS	*Pulse Per Second signal*
PRP	*Parallel Redundancy Protocol*
PS	*Zynq Processing System section*
PSS	*Probabilistic Signature Scheme*
PTP	*Precision Time Protocol*
PTB	*Precise Time Basic*
RBAC	*Role-Based Access Control*
RGMII	*Reduced Gigabit Media Independent Interface*

RFC	*Request For Comments*
RMII	*Reduced Media Independent Interface*
RSA	*Rivest, Shamir and Adleman*
RSASSA-PKCS1-v15	*RSA-Probabilistic Signature Scheme based on Signature Scheme with Appendix*
RSASSA-PSS	*RSA Probabilistic Signature Scheme*
RSTP	*Rapid Spanning Tree Protocol*
RTU	*Remote Terminal Unit*
SAS	*Substation Automation System*
SCADA	*Supervisory Control And Data Acquisition*
SCL	*Substation Configuration Language*
SCSM	*Specific Communication Service Mapping*
SHA	*Secure Hash Algorithm*
SHA-256	*Secure Hash Algorithm-256*
SNMP	*Simple Network Management Protocol*
SoC	*System-on-Chip*
SoC-e	*System-on-Chip engineering*
SoPC	*System-on-Programmable-Chip*
stNum	*Sequence Number Value*
SV	*Sampled Values*
TC	*Transparent Clock*
TCP/IP	*Transmission Control Protocol/Internet Protocol*
TLS	*Transport Layer Security*
ToCK	*Time of Current Key*
TSN	*Time Sensitive Networking*
TtNK	*Time to Next Key*
UART	*Universal Asynchronous Receiver-Transmitter*
UCA	*Utility Communication Architecture*
UES	*Unmanaged Ethernet Switch*
UDP	*User Datagram Protocol*
UP	*Uncontrolled Port*
VHDL	*VHSIC Hardware Description Language*
VLAN	*Virtual Local Area Network*
VPN	*Virtual Private Network*
WAN	*Wide Area Network*
WB	*Wishbone Bus*
XML	*EXtensible Mark-Up Language*

Capítulo 1

Introducción

1.1 Descripción del problema

La constante evolución de las tecnologías de la información y comunicación ha traído consigo la transformación de las redes de generación y distribución de energía tradicionales en las llamadas redes inteligentes o Smart Grids. Estos sistemas de nueva generación deben cumplir con estrictos requisitos en materia de fiabilidad, flexibilidad y eficiencia que les permitan adaptarse a las necesidades del siglo XXI. A fin de asegurar la interoperabilidad entre dispositivos de los distintos fabricantes en Substation Automation System (SAS), el organismo International Electrotechnical Commission (IEC) definió el conjunto de estándares IEC 61850 para estandarizar las comunicaciones y los modelos de datos. Los sistemas de comunicaciones en SAS, son un elemento clave para posibilitar una mayor eficiencia y robustez de los procesos de generación y distribución de energía eléctrica. Adicionalmente, los antiguos sistemas de seguridad para SAS basados en soluciones propietarias y secretas, dan paso al uso de algoritmos criptográficos y estándares de ciberseguridad públicos [3]. De esta forma, en la actualidad, la robustez de los sistemas de seguridad se basa en la complejidad de los algoritmos y soluciones utilizadas, y no en la falta de información a cerca de los métodos y técnicas utilizados.

Por lo tanto, a pesar de que el uso de nuevos sistemas de comunicaciones trae consigo nuevas posibilidades, también es necesario contemplar las vulnerabilidades que emergen del uso de nuevas tecnologías. En las últimas dos décadas varios organismos internacionales como el IEC o el Institute of Electrical and Electronics Engineering (IEEE) han puesto grandes esfuerzos en definir medidas de seguridad

que permitan mitigar las vulnerabilidades presentes en los sistemas de comunicaciones en SAS [3]. La familia de estándares IEC 62351 presenta una serie de medidas y extensiones de seguridad que permiten proporcionar ciberseguridad a todas las comunicaciones definidas en IEC 61850.

IEC 62351 propone reutilizar varios de los mecanismos y protocolos de seguridad empleados habitualmente en las redes Information Technology (IT) para proteger las comunicaciones de niveles 3 de capa Open Systems Interconnection (OSI) y superiores. Estos sistemas, tales como Transport Layer Security (TLS), Internet Protocol security (IPsec) o certificados X.509 han sido ampliamente validados y utilizados en comunicaciones sin requisitos de tiempo real. Sin embargo, en el ámbito de las comunicaciones en SAS, hay un conjunto de mensajes de automatización y control que comunican los equipos de subestación y que cuentan con estrictos requisitos de tiempo real para su generación, comunicación y procesamiento. Estos mensajes de denominan en IEC 61850 Generic Object Oriented Substation Events (GOOSE) y Sampled Values (SV). Los mensajes GOOSE permiten intercambiar mensajes de forma rápida entre los dispositivos inteligentes de la subestación, indicando por ejemplo información sobre el estado de los interruptores, alarmas o la temperatura de los transformadores. Los mensajes SV distribuyen por los SAS la información digitalizada de las tensiones y corrientes trifásicas medidas en diversos puntos de la infraestructura. Los equipos de automatización y control deben hacer uso de la información de los mensajes GOOSE y SV dentro de unas ventanas de tiempo concretas.

El estándar IEC 62351-6 define una extensión de seguridad para proteger los mensajes IEC 61850 GOOSE y SV. Estos mensajes de capa 2 en el modelo OSI cuenta con estrictos requisitos temporales para su generación, distribución y procesamiento por la red de la subestación eléctrica. Tras un exhaustivo estudio y análisis por parte de la comunidad investigadora, se determina que las medidas de seguridad y en especial los algoritmos cripotgráficos que se habían definido en la versión inicial del estándar IEC 62351-6 no resultan adecuados para poder proteger los mensajes GOOSE y SV sin afectar a los requisitos temporales establecidos en el estándar IEC 61850. Atendiendo a las evidencias ofrecidas por los investigadores, el IEC presenta una revisión del estándar IEC 62351-6:2020 con el fin de abordar los problemas de la versión anterior. El principal cambio de la nueva versión del estándar es la sustitución de los algoritmos de criptografía de clave asimétrica por algoritmos criptográficos de clave simétrica. Los algoritmos criptográficos de criptografía de clave simétrica proporcionan un mayor rendimiento y eficiencia en comparación con algoritmos de criptografía de clave asimétrica y permiten abordar soluciones innovadoras de aceleración del proceso criptográfico mediante soluciones hardware ad-hoc.

Haciendo uso de la nueva revisión del estándar IEC 62351-6, la comunidad investigadora y la industria han presentado algunas soluciones para tratar de proteger mensajes GOOSE y SV. Sin embargo, los resultados demuestran que proporcionar integridad, autenticidad y confidencialidad a este tipo de mensajes sin comprometer los requisitos temporales asociados sigue siendo un reto por resolver.

En esta tesis se aborda el reto de proporcionar integridad, autenticidad y confidencialidad de forma simultánea a mensajes IEC 61850 GOOSE y SV sin comprometer sus estrictos requisitos temporales y haciendo uso de la extensión de seguridad IEC 62351-6. Para ello, se propone definir y desarrollar una novedosa arquitectura hardware que sea capaz de afrontar los retos definidos de rendimiento, baja y constante latencia e industrialización. En este sentido, cabe destacar el que es el objetivo de esta investigacion obtener una arquitectura flexible y escalable que pueda ser implementada en dispositivos System-on-Programmable-Chip (SoPC) de nueva generación de forma que sea viable la validación y transferencia de los resultados obtenidos en la investigación.

1.2 Objetivos

El objetivo de esta tesis definir y desarrollar una novedosa arquitectura hardware capaz de afrontar los retos definidos de rendimiento, baja y constante latencia e industrialización para la protección de los mensajes IEC 61850 GOOSE y SV de acuerdo al estándar IEC 62351-6. Para ello, durante el desarrollo de esta tesis se abordarán los siguientes objetivos parciales como medio de consecución del objetivo principal:

- Analizar el estado del arte de las tecnologías y estándares de comunicaciones en subestaciones eléctricas, así como la evolución de los estándares de ciberseguridad para proteger los mensajes con requisitos de tiempo real en SAS.
- Realizar un estudio de las implementaciones hardware de los algoritmos criptográficos propuestos en la literatura para proporcionar autenticidad, integridad y confidencialidad a las comunicaciones. El estudio incluye el análisis de los requisitos que garanticen la viabilidad del uso de comunicaciones seguras en dispositivos SAS.
- Proponer una arquitectura hardware flexible para SoPC que permita proteger los mensajes GOOSE y SV sin comprometer los estrictos requisitos temporales asociados a dichos mensajes y de acuerdo a la extensión de seguridad IEC 62351-6.

- Presentar un Intellectual Property core (IP) que implemente el algoritmo Advanced Encryption Standard (AES) Galois/Counter Mode (AES-GCM), sea adaptable y configurable y esté específicamente diseñado para su uso en dispositivos SAS. Esta implementación del algoritmo AES-GCM actúa como elemento habilitador para proporcionar integridad, autenticación y confidencialidad de los datos.
- Validar la arquitectura hardware propuesta, la compatibilidad con los estándares de comunicaciones y ciberseguridad para SAS, así como el cumplimiento de los requisitos de rendimiento, latencia y uso de recursos lógicos establecidos que garanticen la viabilidad de la solución.

1.3 Organización

Este documento se encuentra dividido en cuatro partes, estando algunas de ellas divididas en varios capítulos:

- **Estado del arte.** En el Capítulo 2 se realiza un análisis de la evolución de las comunicaciones en el entorno de las subestaciones eléctricas. Se introducen los ataques y vulnerabilidades encontradas por la comunidad investigadora y se presentan las necesidades en materia de ciberseguridad para la protección de los mensajes IEC 61850 con estrictos requisitos temporales. Posteriormente, en el Capítulo 3 se hace una descripción detallada de los algoritmos criptográficos designados para la protección de mensajes con requisitos de tiempo real en SAS. En esta sección se analiza la capacidad de afrontar los retos identificados para la protección del tráfico SAS de tiempo real por parte de las soluciones identificadas en el estado del arte. Como conclusión de este análisis cabe destacar que ninguna de las alternativas identificadas es capaz de resolver satisfactoriamente la problemática planteada.
- **Contribución.** Tras analizar el estado del arte, y concluir que que ninguna de las alternativas identificadas es capaz de resolver satisfactoriamente la problemática planteada, en el Capítulo 4, se propone una arquitectura hardware de alto rendimiento que afronte todos los retos identificados. Esta arquitectura permitirá la protección de un flujo de datos bidireccional de alto ancho de banda de forma determinista e incorporando una mínima latencia a los mensajes. La arquitectura hardware del algoritmo criptográfico AES-GCM integrado en la solución, también será una contribución específica del trabajo de investigación realizado. Esta arquitectura será una solución flexible en cuanto a la relación entre los recursos hardware nece-

sarios para su implementación en el silicio y a la latencia requerida para el procesamiento de los datos. Esta flexibilidad incluirá mecanismos para generar automáticamente estructuras hardware criptográficas con distintos niveles de reutilización en el proceso de cálculo de la lógica implementada, permitiendo abordar la generación de circuitos electrónicos con diferentes capacidades según las necesidades del equipo final de subestación destinatario del mecanismo de protección.

- **Validación.** En el Capítulo 5 se demuestra la viabilidad de la arquitectura hardware presentada. Para ello, se describen una serie de pruebas basadas en simulación y experimentos reales que permiten comprobar el cumplimiento con los requisitos establecidos. Los resultados obtenidos se evalúan y comparan respecto al estado del arte en materia de ciberseguridad para dispositivos SAS.
- **Conclusiones.** Finalmente, en el Capítulo 6 se realiza un resumen del documento y se proporciona una visión de las lineas de investigación futuras basadas en los descubrimientos científicos de este trabajo.

Capítulo 2

Ciberseguridad en SAS

2.1 Evolución de las Comunicaciones

Las subestaciones son los nodos que forman parte de la red de distribución eléctrica y permiten la distribución de electricidad en diferentes puntos geográficos. La automatización de subestaciones se compone de SAS que permiten a un operador de red eléctrica monitorizar, controlar y coordinar los componentes de distribución eléctrica instalados en la subestación de forma remota [4]. En los últimos años, los avances en materia electrónica y tecnologías de la información han traído consigo cambios significativos en la forma en la que se operan estas infraestructuras.

Tradicionalmente, las infraestructuras de distribución de energía se basaban en sistemas Supervisory Control And Data Acquisition (SCADA). Estos, a su vez estaban compuestos por una Master Terminal Unit (MTU) la cual se encontraba generalmente localizada en el centro de control de la subestación, así como múltiples Remote Terminal Unit (RTU) distribuidas a lo largo de las instalaciones de la subestación. Las RTUs tenían como principal objetivo la recolección de datos y el control de conmutadores a través de comandos básicos. Por otro lado, la MTU se encargaba de comprobar el estado de las RTUs, así como gestionar los comandos recibidos de los dispositivos Human Machine Interface (HMI). El uso de HMI permitía a los operadores de la subestación comprobar el estado general de la subestación al mismo tiempo que les daba la posibilidad de realizar tareas de control de la misma o aplicar medidas de protección adicionales.

De forma general, las RTUs tenían la capacidad de proporcionar nuevos datos

a la MTU con una periodicidad en el orden de los segundos. Estos datos, eran transmitidos haciendo uso de distintas tecnologías que fueron evolucionando con el paso de los años.

En la década de los años 80 la llegada de SAS basados en software conectados a través de enlaces serie en vez de cables paralelos de cobre rígido tuvo una gran aceptación y supuso una evolución notable. Sin embargo, a pesar de su gran éxito, este tipo de sistemas se basaban o bien en protocolos de comunicaciones propietarios de cada fabricante, o en estándares de comunicaciones derivados de otros dominios de aplicación, como por ejemplo el protocolo de red distribuida Distributed Network Protocol version 3 (DNP3) o IEC 60870 [5]. Así, la falta de uniformidad y estandarización de las comunicaciones hacía imposible la interoperabilidad entre dispositivos de distintos fabricantes y, en algunos casos, incluso entre diferentes versiones de dispositivos de un mismo fabricante. En la mayoría de los casos la única solución a estos problemas residía en realizar costosas adaptaciones de protocolos. Por otro lado, la conexión de los SAS a los transformadores y conmutadores de la subestación seguía siendo de forma analógica, lo que introducía otra capa de complejidad adicional a la hora de contar con una solución interoperable.

Tuvieron que pasar más de 20 años hasta que de forma coordinada fabricantes de dispositivos, distribuidores y operadores de subestaciones reclamaran una solución a los crecientes problemas de interoperabilidad. El objetivo era conseguir un estándar de comunicaciones específico para subestaciones eléctricas que diera fin a los problemas anteriormente descritos. Además, dada la alta velocidad de desarrollo tecnológico, los autores del estándar debían abstraerse en la medida de lo posible de utilizar tecnologías específicas en la definición del estándar, garantizando así su independencia de los avances tecnológicos. Así, a inicios de los años 90 y durante más de una década la IEC comenzó la definición del conjunto de estándares IEC 61850. Finalmente, en el año 2004 se publicó la primera versión del estándar, la cual especificaba los requisitos de comunicaciones, las características funcionales, la estructura de datos de los dispositivos, la nomenclatura para los modelos de datos, y la forma en que las aplicaciones interaccionan y controlan los dispositivos.

La publicación del estándar IEC 61850 supuso una revolución en la operativa de las subestaciones, así como la forma en la que los SAS funcionan y se comunican entre sí. A fin de satisfacer las necesidades demandadas por el sector, el estándar tuvo como requisitos fundamentales resolver los problemas de interoperabildiad, libertad arquitectural y estabilidad a largo plazo [6].

A nivel de interoperabilidad el estándar debía ser capaz de soportar todas las funciones y aplicaciones asociadas al dominio de una subestación eléctrica. Por

lo tanto, además de integrar funciones de protección, automatización, control y, monitorización, también era necesario que el estándar soportase diversas funciones tales como la sincronización temporal o el control de versiones. Este tipo de funcionalidades son llevadas a cabo por los Intelligent Electronic Devices (IEDs) mediante implementaciones software de las mismas. Así, el concepto de interoperabilidad en SAS supone que los IEDs de distintos fabricantes, o diferentes versiones de un mismo IED, deben ser capaces de intercambiar y utilizar información en tiempo real sin la necesidad de utilizar ningún tipo de conversión de protocolos o incluso la interpretación humana de la información [6]. Sin embargo, es necesario diferenciar interoperabilidad frente a intercambiabilidad. Si los IEDs además de interoperables fueran intercambiables, las funciones y configuraciones de los mismos deberían estar estandarizadas, lo cual supondría un freno a futuras evoluciones de los dispositivos. Por otro lado, la libertad de arquitectura proporciona la posibilidad de adoptar cualquier estrategia de operación que sea más favorable para cada caso de uso concreto. Para ello, es necesario que el estándar soporte la asignación de un número arbitrario de funcionalidades a cada dispositivo, así como arquitecturas de sistemas tanto de tipo centralizado como descentralizado.

Dado que de media la vida útil del equipamiento principal de una subestación ronda los 40 años, es habitual que ciertos SAS deban ser sustituidos en varias ocasiones durante este periodo de tiempo. Además, en ciertas ocasiones también es necesario actualizar, mejorar o ampliar las funcionalidades y prestaciones de la subestación a fin de que esta sea capaz de atender futuras necesidades. Sin embargo, a pesar de dichas modificaciones, es necesario que la interoperabilidad se mantenga inalterada y que el estándar sea capaz de acomodar requisitos futuros. Esta necesidad no solo se aplica a los dispositivos que conforman la subestación si no que también afecta a las tecnologías empleadas en lo largo de la propia infraestructura.

Por lo tanto, la introducción del estándar IEC 61850 ha supuesto un cambio drástico en la estructura y funcionamiento de las subestaciones modernas. Tal y como muestra la Figura 2.1, las subestaciones basadas en IEC 61850 normalmente están divididas en tres niveles lógicos.

- **Nivel de proceso:** los dispositivos a nivel de proceso son aquellos que permiten tomar medidas en distintos puntos de la subestación, así como actuar de forma física para modificar el funcionamiento de la misma.
- **Nivel de bahía:** el bus de proceso permite interconectar aquellos dispositivos que se encuentren a nivel de bahía o proceso. Por otro lado, los dispositivos a nivel de bahía son aquellos que permiten controlar y monitorizar el funcionamiento y los datos proporcionados por los dispositivos a

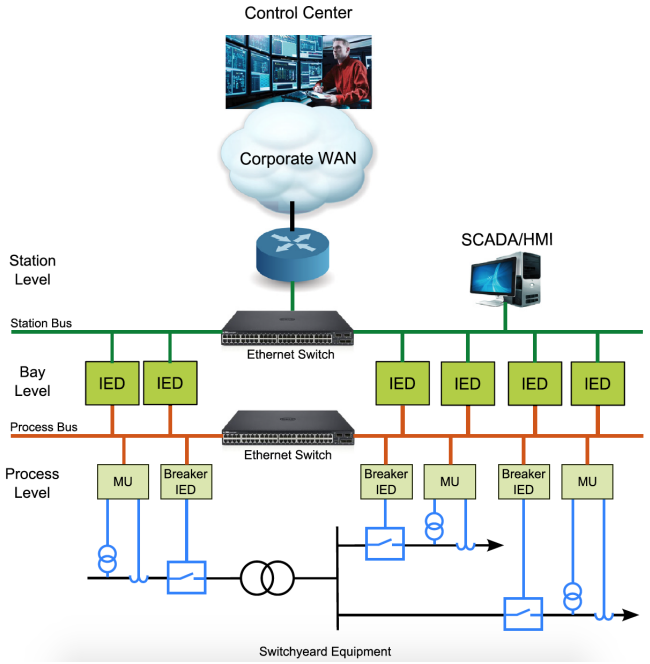

Figura 2.1: Arquitectura de Subestación Basada en IEC 61850

nivel de proceso.

- **Nivel de estación:** el bus a nivel de estación se utiliza para interconectar los dispositivos a nivel de estación y bahía. Este cuenta con dispositivos a nivel de estación que permiten configurar y monitorizar el funcionamiento de la subestación tanto de forma local como de forma remota.

Los SAS están basados en IEDs que son utilizados para monitorizar y controlar el equipamiento de potencia de la subestación, como por ejemplo transformadores o conmutadores de alto voltaje. Los IEDs se componen de microprocesadores, dispositivos electrónicos programables e interfaces de comunicaciones que les permiten procesar los datos y tomar decisiones en tiempo real siguiendo una estructura mucho mas descentralizada en comparación con con las RTUs antiguamente utilizadas. El protocolo de comunicaciones elegido para SAS ha sido Ethernet. Gracias a este esfuerzo de estandarización, los dispositivos que componen la red de la subestación son capaces de recibir mensajes de supervisión o enviar comandos de control haciendo uso de la familia de estándares IEC 61850. Esta toma

de decisiones distribuida, así como la comunicación entre dispositivos tiene como principales beneficios el aumento de la eficiencia de los sistemas de potencia de la subestación, además de un incremento de la robustez de la subestación [4].

2.2 IEC 61850

Las redes de distribución inteligentes Smart Grid en general y los SAS en particular, son considerados infraestructuras críticas tanto por organizaciones como por gobiernos [7–10]. A fin de garantizar su correcto funcionamiento y evitar fallos de suministro eléctrico, este tipo de infraestructuras deben cumplir estrictos requisitos en materia de fiabilidad, flexibilidad, eficiencia e interoperabilidad. Concretamente, las comunicaciones juegan un papel determinante en los apartados de fiabilidad e interoperabilidad. La IEC ha desarrollado el conjunto de estándares IEC 61850 con el objetivo de solucionar los problemas de interoperabilidad entre dispositivos de distintos fabricantes así como estandarizar las comunicaciones de los sistemas SAS [11, 12].

A pesar de que inicialmente el estándar IEC 61850 estaba orientado a las comunicaciones dentro del entorno de una subestación, finalmente se extendió al conjunto de la red de distribución de energía eléctrica, lo que incluye las comunicaciones entre subestaciones, plantas de generación de energía, etc. La Figura 2.2 muestra los diferentes usos del IEC 61850 en la cadena de distribución de energía.

Tal y como se muestra en la Tabla 2.1 y Tabla 2.2, el estándar IEC 61850 está dividido en 10 secciones que definen todos los aspectos relacionados con las comunicaciones en el entorno de generación y distribución de energía eléctrica. Las secciones 3, 4 y 5 del documento especifican los requisitos funcionales que deben seguir las comunicaciones dentro del entorno de una subestación eléctrica. Estos, se utilizan para identificar los servicios, modelos de datos o protocolos específicos de cada aplicación. Puesto que las subestaciones modernas se componen de un gran número de dispositivos que proporcionan un elevado número de funciones y servicios específicos, resulta de vital importancia establecer un protocolo de configuración que permita automatizar y simplificar el proceso de configuración de dichos elementos. Para ello, la sección 6 del estándar presenta un Substation Configuration Language (SCL) basado en el lenguaje EXtensible Mark-Up Language (XML) que especifica las funcionalidades y relación entre dispositivos para cada elemento SAS dentro de la subestación.

Puesto que la vida útil media de una subestación eléctrica es de unos 30-40 años, la independencia respecto a tecnologías de comunicaciones específicas supuso uno de los principales requisitos del proceso de estandarización. Por ello, el estándar

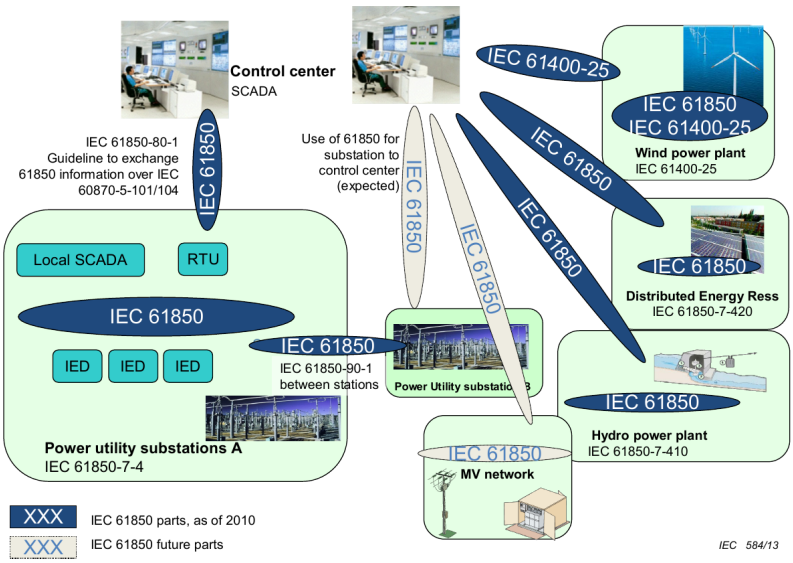

Figura 2.2: Usos del estándar IEC 61850 en la red de distribución de energía eléctrica [1]

se centra en la definición de modelos de datos y objetos en vez de tecnologías específicas. La definición de los servicios ofrecidos dentro de la subestación se detalla en la sección 7.2 del estándar. Por otro lado, en la sección 7.3 se introduce el concepto de Common Data Classes (CDC) para definir bloques de datos genéricos que sirvan como base sobre la que crear modelos de datos complejos. Así, la definición de los objetos de datos, nodos lógicos, se describe en la sección 7.4 del documento. En concreto, de acuerdo al estándar IEC 61850, cualquier dispositivo que intercambie datos dentro de la red de comunicaciones de una subestación eléctrica haciendo uso de servicios estandarizados debe ser considerado como un objeto o nodo lógico [13]. El mapeo de los servicios y objetos que conforman la red de la subestación en mensajes de tipo Manufacturing Message Specification (MMS) se describe en la sección 8.1 del estándar. Finalmente, las secciones 9.1 y 9.2 del documento describen la forma en la que se establecen las comunicaciones para intercambiar mensajes de tipo SV. Estas, pueden estar basadas a su vez en enlaces unidireccionales punto a punto o enlaces bidireccionales utilizando redes Ethernet.

IEC 61850 utiliza el modelo OSI para estructurar las comunicaciones. Este modelo sigue una estructura dividida en capas en la que cada capa desempeña una función claramente definida y que proporciona servicios a las capas adyacentes. En la

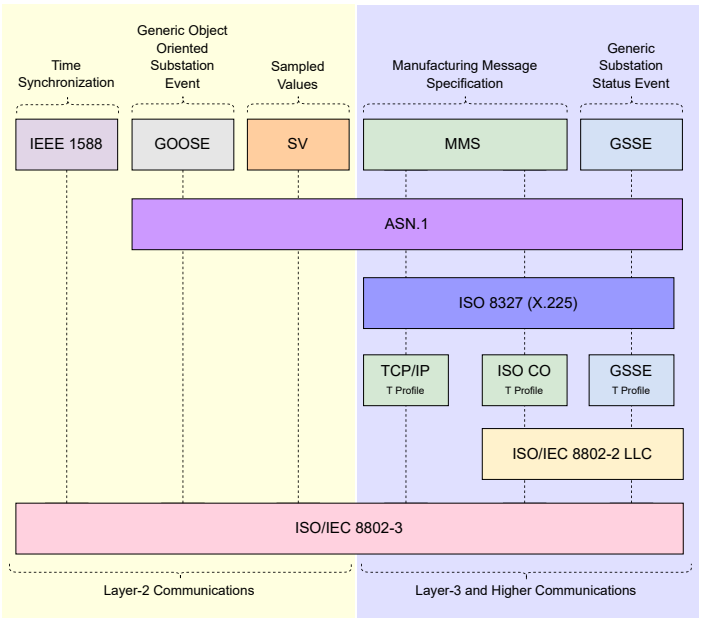

Figura 2.3: Modelos de comunicación en IEC 61850

Figura 2.3 se muestran los 5 tipos de comunicaciones que se definen en el estándar para proporcionar los distintos servicios definidos en IEC 61850. Estas se pueden dividir en 2 grupos principales:

- Mensajes de capa 2 en el modelo OSI que cuentan con requisitos de tiempo real.
- Mensajes de capa 3 o superiores en el modelo OSI que no cuentan con requisitos de tiempo real.

Tabla 2.1: Secciones del estándar IEC 61850 (Parte 1)

IEC 61850	Título	Tipo	Versión	Fecha
Sección 1	Introduction and Overview	S	ed1.0	06/2020
Sección 2	Glossary	S	ed2.0	04/2019
Sección 3	General requirements	S	ed2.0	12/2013
Sección 4	System and project management	S	ed2.0	04/2011
Sección 5	Communication requirements for functions and device models	S	ed2.0	01/2013
Sección 6	Configuration description language for communication in electrical substations related to IEDs	IS	ed2.0	12/2009
Sección 7-1	Basic communication structure - Principles and models	IS	ed2.0	07/2011
Sección 7-2	Basic communication structure - Abstract Communication Service Interface (ACSI)	IS	ed2.0	08/2010
Sección 7-3	Basic communication structure - Common data classes	IS	ed2.0	12/2010
Sección 7-4	Basic communication structure - Compatible logical node classes and data classes	IS	ed2.0	03/2010
Sección 7-410	Basic communication structure - Hydroelectric power plants - Communication for monitoring and control	I S	ed2.0	10/2012
Sección 7-420	Basic communication structure - Distributed energy resources logical nodes	IS	ed1.0	03/2009
Sección 7-5	IEC 61850 modelling concepts	TR	ed1.0	04/2021
Sección 7-500	Basic communication structure - Use of logical nodes for modeling application functions and related concepts and guidelines for substations	TR	ed1.0	07/2017
Sección 7-510	Basic communication structure - Hydroelectric power plants - Modelling concepts and guidelines	IS	ed1.0	03/2012
Sección 7-6	Guideline for definition of Basic Application Profiles (BAPs) using IEC 61850	TR	ed1.0	01/2019
Sección 7-7	Machine-processable format of IEC 61850-related data models for tools	TS	ed1.0	03/2018
Sección 8-1	Specific Communication Service Mapping (SCSM) - Mappings to MMS (ISO 9506-1 and ISO 9506-2) and to ISO/IEC 8802-3	IS	ed2.0	06/2011
Sección 8-2	Specific communication service mapping (SCSM) - Mapping to Extensible Messaging Presence Protocol (XMPP)	IS	ed1.0	12/2018
Sección 9-2	Specific Communication Service Mapping (SCSM) - Sampled values over ISO/IEC 8802-3	IS	ed2.0	09/2011
Sección 9-3	Precision time protocol profile for power utility automation	IS	ed1.0	05/2016
Sección 10	Conformance testing	IS	ed2.0	12/2012
Sección 80-1	Guideline to exchanging information from a CDC-based data model using IEC 60870-5-101 or IEC 60870-5-104	TS	ed2.0	07/2016
Sección 80-3	Mapping to web protocols - Requirements and technical choices	TR	ed1.0	11/2015
Sección 80-4	Translation from the COSEM object model (IEC 62056) to the IEC 61850 data model	TS	ed1.0	03/2016
Sección 90-1	Use of IEC 61850 for the communication between substations	TR	ed1.0	03/2010
Sección 90-2	Using IEC 61850 for communication between substations and control centres	TR	ed1.0	02/2016
Sección 90-3	Using IEC 61850 for condition monitoring diagnosis and analysis	TR	ed1.0	05/2016
Sección 90-4	Network engineering guidelines	TR	ed2.0	05/2020
Sección 90-5	Use of IEC 61850 to transmit synchrophasor information according to IEEE C37.118	TR	ed1.0	05/2012
Sección 90-6	Use of IEC 61850 for Distribution Automation Systems	TR	ed1.0	09/2018

Tabla 2.2: Secciones del estándar IEC 61850 (Parte 2)

IEC 61850	Título	Tipo	Versión	Fecha
Sección 90-7	Object models for power converters in Distributed Energy Resources (DER) systems	TR	ed1.0	02/2013
Sección 90-8	Object model for E-mobility	TR	ed1.0	04/2016
Sección 90-9	Use of IEC 61850 for Electrical Energy Storage Systems	TR	ed1.0	09/2020
Sección 90-10	Models for scheduling	TR	ed1.0	10/2017
Sección 90-11	Methodologies for modelling of logics for IEC 61850 based applications	TR	ed1.0	09/2020
Sección 90-12	Wide area network engineering guidelines	TR	ed1.0	07/2015
Sección 90-13	Deterministic networking technologies	TR	ed1.0	02/2021
Sección 90-16	Requirements of system management for Smart Energy Automation	TR	ed1.0	06/2021
Sección 90-17	Using IEC 61850 to transmit power quality data	TR	ed1.0	05/2017

IS: International Standard
TR: Technical Report
TS: Technical Specification

2.2.1 Mensajes con requisitos de tiempo real

Dentro del conjunto de comunicaciones que existen dentro de las redes de las subestaciones eléctricas modernas, existen diferentes tipos de mensajes que permiten establecer comunicaciones entre dispositivos de acuerdo a las necesidades específicas de cada caso. En concreto, se utilizan mensajes de capa 2 en la pila de protocolos OSI para proporcionar servicios que requieran la utilización de mensajes con estrictos requisitos de tiempo real [2]. En concreto, para los SAS más exigentes, este tipo de tramas de comunicaciones deben generarse, transmitirse, recibirse y ser procesadas por el destinatario en menos de 3 ms. Así, el estándar IEC 61850 define que los mensajes que cuenten con requisitos de tiempo real deben estar mapeados directamente a la capa de nivel de enlace (capa 2 en el modelo OSI). Gracias a este enfoque se consigue un formato de tramas de comunicaciones simples que requieren menos carga computacional que otras alternativas diseñadas en base a protocolos de niveles superiores en el modelo OSI. La Tabla 2.3 muestra los requisitos temporales de los distintos tipos de mensajes especificados en el estándar IEC 61850 en función de cada aplicación.

Dentro del grupo de mensajes con requisitos de tiempo real, los mensajes de tipo Generic Substation Events (GSE) se utilizan para proporcionar un mecanismo que permita compartir datos de eventos y estado entre SAS. Los mensajes de tipo GSE pueden dividirse a su vez en dos protocolos de transmisión de eventos/estado: mensajes Generic Substation State Events (GSSE) y mensajes GOOSE. Los mensajes de tipo GOOSE son aquellos que son generados y recibidos por los IEDs siguiendo un modelo multidifusión de tipo publicador/suscriptor.

Tabla 2.3: Requisitos temporales IEC 61850 [2]

Aplicación	Tipo de Mensaje	Requisito Temporal
Mensajes Rápidos	GOOSE	$\leq 3\,\text{ms}$
	GSSE	$\leq 10\,\text{ms}$
Datos sin Procesar	SV	$\leq 3\,\text{ms}$
Mensajes de Velocidad Media	MMS	$\leq 100\,\text{ms}$
Mensjaes Lentos	MMS	$\leq 500\,\text{ms}$
Sincronización Temporal	IEEE 1588	$> 500\,\text{ms}$
Transferencia de Archivos	MMS	$> 500\,\text{ms}$

Una de las principales características de este tipo de mensajes es que no utilizan ningún tipo de mecanismo para confirmar la correcta recepción de los mensajes. Por lo tanto, a fin de mejorar la robustez y proporcionar cierto nivel de redundancia, los mensajes GOOSE se retransmiten varias veces. Para ello, los suscriptores de los mensajes deben detectar la recepción de mensajes repetidos y descartarlos [14]. Este tipo de mensajes utilizan el formato de sintaxis Abstract Syntax Notation One (ASN.1) para definir y codificar la estructura del cuerpo de los mensajes. Así, los mensajes GOOSE generalmente se utilizan para transmitir datos binarios que representan el estado de dispositivos mediante indicadores o alertan de situaciones anómalas a través de alarmas. Algunos ejemplos del uso de mensajes GOOSE son la señalización de un fallo en un conmutador de potencia o el estado de las lineas de distribución de energía eléctrica.

Por otro lado, los mensajes de tipo SV permiten realizar la transmisión de muestras digitales de valores instantáneos de corriente y tensión eléctrica a través de una red Ethernet. Los datos se envían como flujos de datos continuos formados por tramas Ethernet de capa 2 en el modelo OSI. Este tipo de mensajes también se codifican utilizando la norma ASN.1. Puesto que se tratan de mensajes con requisitos de tiempo real, estos también deben ser recibidos y procesados por el destinatario en menos de 3 ms desde su creación. Dentro de la subestación eléctrica, son las Merging Unit (MU) las encargadas de realizar las medidas de corriente y tensión eléctrica en distintos puntos de la misma. Estas medidas, son encapsuladas y transmitidas en forma de mensaje SV. Al igual que ocurría con los mensajes GOOSE, el modelo de transmisión utilizado para este tipo de mensajes es el de publicador/suscriptor. Por lo tanto, en función de la configuración establecida, los IEDs pueden suscribirse a los flujos de cada MU que necesiten para así monitorizar el estado de la subestación.

2.2.2 Mensajes sin requisitos de tiempo real

Aquellas comunicaciones que se basan en protocolos de capa 3 o capas superiores del modelo OSI no son consideradas de tiempo real. Esto es debido a que la sobrecarga de datos y procesamiento introducida por las capas superiores impide que sea posible garantizar que los mensajes sean generados, transmitidos, recibidos y procesados dentro de las ventanas temporales definidas para los procesos máx exigentes. Los mensajes GSSE se utilizan para intercambiar cadenas de datos binarios con información de estado. Esto resulta adecuado para dispositivos con capacidad de computación reducida que necesiten tiempos de respuesta rápidos. Al igual que ocurría con los mensajes GOOSE, los mensajes GSSE se basan en el modelo publicador/suscriptor. Por otro lado, los mensajes MMS se utilizan para intercambiar datos e información de supervisión y control entre los dispositivos de la red de la subestación. Estos, se basan en el modelo cliente/servidor. En cuanto a la estructura, tanto los mensajes GSSE como los mensajes MMS se codifican utilizando ASN.1. Además, ya que se trata de comunicaciones orientadas a la conexión, como protocolo de sesión este tipo de mensajes utilizan el protocolo X.225. Este proporciona servicios que permitan coordinar las comunicaciones entre aplicaciones locales y remotas, así como establecer, gestionar y finalizar las comunicaciones.

Adicionalmente, tal y como se muestra en la Tabla 2.3, existen mensajes de sincronización temporal que no deben cumplir con requisitos de tiempo real. Este tipo de mensajes se basan en el protocolo denominado IEEE 1588 que proporciona sincronización temporal en el orden de los nanosegundos a través de una red de comunicaciones Ethernet. El motivo por el cual este tipo de mensajes no tiene impuestos requisitos de tiempo real es que el propio protocolo cuenta con mecanismos para corregir los retardos introducidos por la red y los dispositivos que la componen. Para ello, es necesario que todos los dispositivos de la red, incluyendo los switches o routers de la misma, sean compatibles con IEEE 1588. Este tipo de equipos tienen la capacidad de calcular el retardo introducido en la transmisión de las tramas IEEE 1588 y de actualizar consecuentemente un campo de la misma para añadir el retardo introducido en cada switch por el que pase la trama. Este protocolo utiliza un mecanismo maestro/esclavo en el que los esclavos se sincronizan con respecto a la hora del maestro en el orden de las decenas o centenas de nanosegundos. Habitualmente los mensajes de sincronización IEEE 1588 utilizan la capa 2 en el modelo OSI aunque el estándar también permite que estos se encapsulen utilizando protocolos de capas superiores, tales como User Datagram Protocol (UDP),Internet Protocol version 4 (IPv4) o Internet Protocol version 6 (IPv6).

2.3 Vulnerabilidades en SAS

El estándar IEC 61850 define los protocolos de comunicación utilizados en el entorno de la generación y distribución de energía eléctrica, a la vez que abre las puertas de la estandarización y la digitalización. La integración progresiva de los servicios Operational Technology (OT) con los servicios IT trae consigo la introducción de nuevas vulnerabilidades.

El intercambio de información entre dispositivos se realiza a través de la transmisión de tramas Ethernet. Estas, pueden ser fácilmente alteradas, grabadas o reproducidas por dispositivos indeseados [15–17]. En IEC 61850 no se especifica ningún mecanismo relacionado con la ciberseguridad.

En cambio, todo lo relacionado en materia de ciberseguridad fue definido posteriormente por la familia de estándares IEC 62351. Sin embargo, puesto que la primera versión del estándar IEC 62351 fue publicado varios años después de que la primera versión del estándar IEC 61850 estuviera disponible, una gran cantidad de dispositivos, tales como IEDs no cuentan con soporte del estándar IEC 62351 [18]. Por lo tanto, este hecho ha generado la necesidad de combinar soluciones alternativas en materia de ciberseguridad con aquellos mecanismos especificados en el estándar IEC 62351 para proteger las subestaciones actuales y futuras que estén basadas en IEC 61850. Como consecuencia, tal y como se analiza en [19], la necesidad de que una buena parte de los mecanismos de seguridad definidos en IEC 62351 cuenten con retro compatibilidad con dispositivos que no implementen las medidas de seguridad especificadas en el estándar, ha hecho que ciertos protocolos y estándares descritos en IEC 62351 no alcancen los niveles de seguridad que hubiera sido posible si se hubiera tomado un enfoque mas ambicioso.

2.3.1 Vulnerabilidades de alto nivel

Desde la definición de la primera versión del estándar IEC 61850 y la implementación de las primeras redes de distribución de energía basadas en el mismo, varios ataques a instalaciones reales han probado la falta de protección en materia de ciberseguridad de algunos de los protocolos, dispositivos o las propias infraestructuras. Por ejemplo, el virus denominado Slammer infectó parte de los sistemas de control de una planta nuclear, derivando en un eventual fallo de la misma. Este ciberataque fue capaz de saltarse las medidas de seguridad con las que contaba la planta nuclear y fue capaz de deshabilitar el sistema de supervisión de seguridad durante casi 5 horas [20]. De forma similar, varias plantas industriales en Irán se vieron afectadas por el ataque de un virus llamado Stuxnet. Entre ellas, se encontraba una planta nuclear que utilizaba programas de control industrial del

fabricante Siemens basados en el sistema operativo Microsoft Windows [21]. Esto supuso un hecho relevante ya que Stuxnet ha sido el primer ciberataque conocido a un sistema SCADA. Este virus fue utilizado para obtener información a cerca del funcionamiento de los sistemas de la planta. Esta información fue posteriormente utilizada para tomar el control de ciertos dispositivos conectados a la red de la planta y provocar el fallo de los mismos.

Los ataques descritos son ataques de alto nivel que afectan a sistemas, dispositivos y protocolos sin requisitos de tiempo real, a diferencia de lo que ocurre con los mensajes de tipo GOOSE y SV. Los autores en [22] proponen distintos mecanismos para combatir los ataques de este tipo. Entre ellos destaca la inclusión de software Anti-Malware o el uso de parches de seguridad para aplicaciones software. Adicionalmente, a fin de proteger las comunicaciones basadas en IEC 61850 sin requisitos de tiempo real, en [23] se propone hacer uso de protocolos de comunicaciones seguros tales como IPsec, Media Access Control Security (MACsec) o TLS. Por otro lado, en [24] se propone hacer uso de la tecnología Virtual Private Network (VPN) y de otro tipo de protocolos de comunicaciones basadas en túneles para proteger las comunicaciones entre subestaciones.

2.3.2 Vulnerabilidades en mensajes GOOSE y SV

En la actualidad, la comunidad investigadora a descubierto y publicado la descripción de varias vulnerabilidades que afectan a los mensajes IEC 61850 de tipo GOOSE y SV. En [25] se describe un ataque de suplantación de identidad a través del cual es posible enviar mensajes GOOSE a un receptor en la red de la subestación. Para ello, después de grabar tráfico GOOSE legítimo, el atacante utiliza dichos mensajes en los que modifica ciertos campos como el valor de algunas variables o del campo de control de secuencia Sequence Number Value (stNum). Los mensajes fraudulentos enviados por el atacante son correctamente validados por el receptor siempre y cuando la marca temporal de los mismos no exceda los 2 minutos y el campo de control stNum sea superior al del último mensaje recibido que haya sido marcado como válido. De forma similar, los autores de [26] y [27] evalúan el impacto de los ataques basados en la suplantación de identidad frente a sistemas basados en IEC 61850.

La falta de mecanismos de seguridad en el estándar IEC 61850 en forma de autenticación o cifrado de los mensajes abre la puerta a que un atacante pueda ejecutar ataques basados en inyectar mensajes en la red. Tal y como se describe en [28] y [29], un atacante podría capturar mensajes enviados por un dispositivo legítimo de la red de una subestación, lo cual le daría la información necesaria con la que generar mensajes fraudulentos. Estos podrían ser utilizados por el atacante para forzar estados de error en dispositivos o incluso colapsar el funcionamiento de la

subestación. A diferencia de los ataques de suplantación de identidad anteriormente descritos, en este caso es necesario es necesario conocer el valor del campo stNum y dirección Media Access Control (MAC) de los mensajes.

En [30] se evalua el uso de modificaciones al campo stNum de los mensajes GOOSE para realizar ataques de tipo Denial of Service (DoS). En este caso, el atacante envía mensajes GOOSE fijando un valor de $(2^{32} - 1)$ en el campo stNum de dichos mensajes. Este valor es el máximo que el campo stNum puede tomar antes de que haya un desbordamiento y fuerza a que si el receptor recibe mensajes GOOSE legítimos con un valor de stNum inferior, estos sean descartados.

Por otro lado, tanto en [31] como en [32] se estudia en profundidad el uso de ataques de tipo Flooding. Este tipo de ataques consisten en saturar la red de comunicaciones. Para ello, es posible utilizar mensajes legítimos que hayan sido enviados previamente y enviarlos de forma masiva con el único propósito de consumir parte de los recursos de comunicaciones o procesamiento del receptor. Esto impediría que los mensajes legítimos fueran recibidos por el receptor o que estos fueran procesados a tiempo. Hussain et al. [33] han analizado las distintas vulnerabilidades en materia de ciberseguridad, así como los posibles ataques que pueden llevarse a cabo para comprometer los mensajes IEC 61850. En concreto, los mensajes GOOSE y SV son susceptibles a vulnerabilidades basadas en la integridad de los mensajes o disponibilidad de servicio. Estas vulnerabilidades pueden ser explotadas utilizando ataques basados en repetición de mensajes, violación de integridad, DoS, manipulación de datos o inyección de mensajes fraudulentos. Adicionalmente, puesto que la mayoría de los ataques previamente descritos se basan en conocer parte del contenido de los mensajes GOOSE y SV, estos podían prevenirse si se utilizasen mecanismos de seguridad basados en la confidencialidad de los mensajes.

Tal y como se ha demostrado en el análisis de los ciberataques a SAS previamente descritos, estos tienen la capacidad de tener un impacto en su comportamiento. Así, es posible provocar problemas de rendimiento o el fallo total de los sistemas de comunicaciones de la subestación, dejándola inutilizada. A continuación se recoge el impacto que tienen los ciberataques mas comunes en el entorno de la generación y distribución de energía eléctrica [34]:

- Interrupción del sistema de monitorización: los IEDs no son capaces de recibir los datos de las MUs u otros IEDs, los datos recibidos están corruptos o los IEDs no son capaces de procesar los datos recibidos de forma normal.
- DoS sobre el sistema de control: los IEDs no son capaces de enviar comandos de control a las protecciones eléctricas o el funcionamiento de los mismos

ha sido modificado por un atacante.

- Interrupción de las comunicaciones de las protecciones: las protecciones eléctricas no son capaces de recibir comandos de control de los IEDs.
- Comportamiento erróneo de las protecciones: las protecciones eléctricas reciben mensajes de control fraudulentos o no operan de forma adecuada a los mensajes recibidos.
- Interrupción de la red de comunicaciones: los dispositivos en la red de comunicaciones de la subestación no son capaces de comunicarse entre sí, lo que supone una situación de alto riesgo para la integridad de la subestación.

2.4 Seguridad en mensajes GOOSE y SV

IEC 62351 es la familia de estándares que define los requisitos y medidas de seguridad que deben cumplirse para proteger las comunicaciones basadas en IEC 61850 para la generación y distribución de energía eléctrica. IEC 62351 está compuesto por 13 partes, las cuales están recogidas en la Tabla 2.4.

La sección 3 del estándar define los requisitos de seguridad para cualquier perfil basado en los protocolos Transmission Control Protocol/Internet Protocol (TCP/IP). Para ello se proporciona autenticidad, confidencialidad e integridad de los mensajes haciendo uso del protocolo TLS. Además, en esta sección también se definen los parámetros de seguridad asociados a TLS. La sección 4 del documento proporciona seguridad para aquellos perfiles basados en comunicaciones MMS. En este caso también se ha uso de TLS. Sin embargo, su uso se limita exclusivamente a autentificar las entidades que forman parte de una comunicación. El estándar permite la coexistencia de comunicaciones MMS seguras y no seguras. En la sección 5 se definen los mecanismos de seguridad para comunicaciones serie, tales como IEC 60870-5-101. Este tipo de comunicaciones se caracterizan por tener una tasa de transferencia de datos reducida o ser utilizadas en dispositivos con poca capacidad de computación. Por lo tanto, el estándar define una serie de mecanismos de seguridad específicamente diseñados que contemplan las características y restricciones de este tipo de comunicaciones.

Por otro lado, la sección 7 del documento se centra en la administración segura de la red de la subestación y los equipos conectados a la misma. Para ello, se propone monitorizar el estado de la red y sus sistema a traés de los protocolos Common Management Information Protocol (CMIP) y Simple Network Management Protocol (SNMP). En la sección 8 se define el control de acceso de usuarios y dispositivos automatizados a los datos de los sistemas de generación y distri-

Tabla 2.4: Secciones del estándar IEC 62351

IEC 62351	Título	Tipo	Versión	Fecha
Sección 1	Communication network and system security - Introduction to security issues	TS	ed1.0	05/2007
Sección 2	Glossary of terms	TS	ed1.0	08/2008
Sección 3	Communication network and system security - Profiles including TCP/IP	IS	ed1.0	10/2014
Sección 4	Profiles including MMS and derivatives	IS	ed1.0	11/2018
Sección 5	Security for IEC 60870-5 and derivatives	TS	ed2.0	04/2013
Sección 6	Security for IEC 61850	IS	ed1.0	10/2020
Sección 7	Network and System Management (NSM) data object models	IS	ed1.0	07/2017
Sección 8	Role-based access control for power system management	IS	ed1.0	04/2020
Sección 9	Cyber security key management for power system equipment	IS	ed1.0	05/2017
Sección 10	Security architecture guidelines	TR	ed1.0	10/2012
Sección 11	Security for XML documents	IS	ed1.0	09/2016
Sección 12	Resilience and security recommendations for power systems with distributed energy resources (DER) cyber-physical systems	TR	ed1.0	04/2016
Sección 13	Guidelines on security topics to be covered in standards and specifications	TR	ed1.0	08/2016
Sección 90-1	Guidelines for handling role-based access control in power systems	TR	ed1.0	01/2018
Sección 90-2	Deep packet inspection of encrypted communications	TR	ed1.0	09/2018
Sección 90-3	Guidelines for network and system management	TR	ed1.0	03/2021
Sección 100-1	Conformance test cases for IEC TS 62351-5 and IEC TS 60870-5-7	TS	ed1.0	11/2018
Sección 100-3	Conformance test cases for the IEC 62351-3, the secure communication extension for profiles including TCP/IP	TS	ed1.0	01/2020

bución de energía haciendo uso de la metodología Role-Based Access Control (RBAC). La sección 10 describe las directivas que se deben seguir en la definición de las arquitecturas de los sistemas de generación y distribución de energía. Estas deben permitir la integración de dichos sistemas en la arquitectura global de la subestación. En la sección 11 se describen los mecanismos de seguridad asociados a la protección de archivos XML. El principal objetivo del documento es proporcionar autenticación de los documentos y evitar así la manipulación de los mismos por parte de un atacante o entidad no autorizada.

La protección de los mensajes de capa 2 en el modelo OSI GOOSE y SV con requisitos de tiempo real se recoge en la sección 6 del documento. Así mismo, la sección 9 del estándar define como generar, distribuir, revocar y administrar tanto certificados digitales como las claves criptográficas utilizadas para proteger las comunicaciones basadas en IEC 61850. La Figura 2.4 relaciona las diferentes secciones del estándar IEC 62351 con las distintas capas de seguridad en las que se divide la protección de las comunicaciones basadas en IEC 61850.

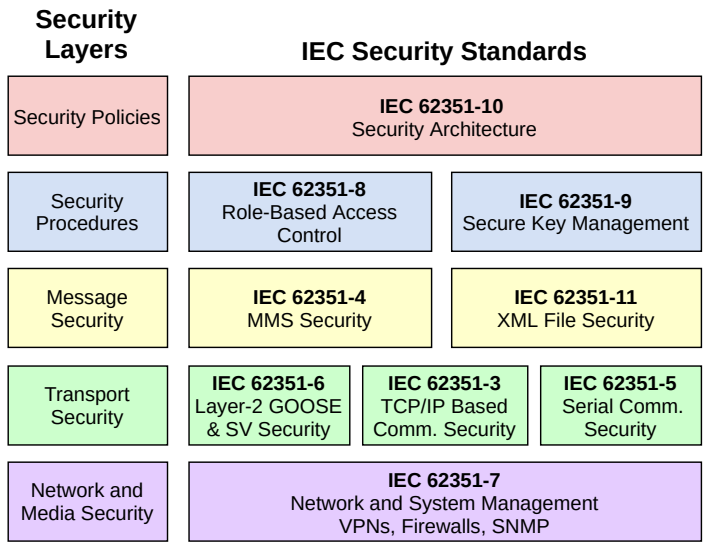

Figura 2.4: Capas de seguridad en IEC 62351

2.4.1 IEC 62351-6

Con el objetivo de poder corregir las vulnerabilidades y combatir los ataques frente a comunicaciones basadas en mensajes GOOSE y SV presentados en la sección 2.3.2 del presente documento, el IEC publicó la sección 6 del estándar IEC 62351. Este define los mecanismos de seguridad para comunicaciones punto a punto y mensajes de capa 2 con requisitos de tiempo real [35]. En vez de utilizar mecanismos o protocolos de seguridad de propósito general como TLS, se ha diseñado una solución específica que garantice que los mensajes con requisitos de tiempo real sean entregados dentro de los límites temporales establecidos para este tipo de mensajes. En concreto, se define la autenticación de los mensajes como obligatoria. Por otro lado, debido a la carga computacional introducida por el cifrado de los mensajes, esta se define como opcional. La autenticación de los mensajes se lleva a cabo haciendo uso de una función criptográfica de tipo Hash. Este tipo de funciones garantizan que los datos no hayan sido modificados por un agente externo, asegurando la integridad y autenticidad de los mismos.

La autenticación de los mensajes impide que un posible atacante pudiera modificar mensajes generados por una fuente legítima o generar sus propios mensajes falsos. Sin embargo, puesto que el cifrado que asegura la confidencialidad de los datos solamente es un requisito definido como opcional en el estándar, un atacan-

te podría conectarse a la red de la subestación y capturar los mensajes GOOSE y SV que se envíen por la red de comunicaciones de la misma. Esta información indicaría al atacante el estado y comportamiento de los dispositivos de la subestación. A su vez, dicha información podría ser utilizada para explotar posibles vulnerabilidades en otros dispositivos. Por lo tanto, la confidencialidad de los datos resulta una característica deseable a fin de garantizar un alto nivel de seguridad en las comunicaciones basadas en IEC 61850. En cambio, proporcionar confidencialidad a los mensajes GOOSE y SV supone un reto tecnológico, ya que es necesario que la carga computacional extra introducida por esta operación sea lo suficientemente pequeña como para que no se comprometa el estricto requisito temporal definido por el estándar para este tipo de mensajes.

De acuerdo con el análisis realizado en [36] y tal como se muestra en la Figura 2.5, se necesitan 2,6 ms para que un mensaje IEC 61850 GOOSE o SV sea procesado extremo a extremo. Este tiempo no incluye ningún mecanismo de seguridad IEC 62351-6.

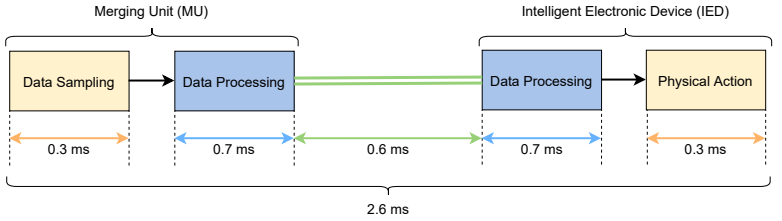

Figura 2.5: Tiempo de procesamiento de mesnajes GOOSE y SV sin mecanismos de seguridad

En primer lugar, el transmisor del mensaje emplea 0,3 ms en realizar el muestreo de los datos a enviar. Adicionalmente, este emplea otros 0,7 ms adicionales en la ejecución de la pila software IEC 61850 que se encarga de generar y transmitir el mensaje. A continuación, se emplean 0,6 ms en que el mensaje sea transmitido hasta los destinatarios del mensaje a través de la red de comunicaciones de la subestación. Una vez el mensaje llega al destinatario del mismo, este emplea 0,7 ms en ejecutar la pila software IEC 61850 para procesar los datos recibidos y determinar las acciones correspondientes en cada caso. Finalmente, el receptor del mensaje emplea 0,3 ms en llevar a cabo la acción física asociada al procesamiento de los datos recibidos, como por ejemplo el cierre de un circuito protección de la subestación.

Por lo tanto, a fin de que el uso de los mecanismos de seguridad definidos en IEC 62351-6 sea viable y teniendo en consideración el requisito de 3 ms impuesto por IEC 61850 para el tiempo de entrega de mensajes GOOSE y SV, es necesario que

estos se lleven a cabo en un tiempo inferior a 0,4 ms. Así, el cálculo de la firma digital por parte del transmisor del mensaje y su correspondiente comprobación por parte del receptor del mismo, debe llevarse a cabo en menos de 0,4 ms para no comprometer el tiempo de entrega de 3 ms de este tipo de mensajes. Adicionalmente, en caso de proporcionar confidencialidad a los mensajes, es necesario que las operaciones de encriptar y desencriptar los datos, junto a la generación y comprobación de la firma digital no superen los 0,4 ms.

2.4.2 Extensión de seguridad IEC 62351-6

El estándar IEC 62351-6 presenta una extensión sobre las tramas Ethernet de los mensajes IEC 61850 GOOSE y SV. La Figura 2.6 muestra una comparativa entre una trama Ethernet IEC 61850 y una trama Ethernet con la extensión de seguridad IEC 62351-6. Tal y como se puede apreciar, esta extensión de seguridad se encuentra localizada al final de la trama Ethernet y tiene un tamaño de 58 bytes. La localización de la extensión de seguridad al final de la trama Ethernet IEC 61850 hace compatible el uso de mensajes GOOSE y SV protegidos mediante IEC 62351-6, con mensajes que no hayan sido protegidos. Además, en la Figura 2.6 también se muestran los campos de la trama IEC 62351-6 que deben ser autenticados y cuales son encriptados de manera opcional. Adicionalmente se muestran los campos que componen la extensión de seguridad, así como su estructura y su tamaño.

En el estándar IEC 62351-6 se especifica que la cabecera Ethernet, así como la cabecera Virtual Local Area Network (VLAN) deben mantenerse inalteradas. Con lo cual, los datos que componen la misma no son autenticados o encriptados. Es necesario que estos campos permanezcan en claro puesto que son utilizados por elementos de la red de comunicaciones de la subestación tales como switches Ethernet. Estos dispositivos de comunicaciones necesitan conocer, y en algunos casos modificar, la dirección MAC origen y destino de la trama Ethernet, así como la información VLAN de la misma.

Por otro lado, a continuación en la trama Ethernet se encuentra el campo Extended Protocol Data Unit (EPDU). Este sirve para identificar el tipo de mensaje IEC 61850 y se encuentra compuesto por el Ethertype, el campo Application Identifier (APPID), longitud del campo Application Protocol Data Unit (APDU), longitud de la extensión de seguridad y el Cyclic Redundancy Check (CRC) del EPDU. En una trama GOOSE o SV sin extensión de seguridad, los bytes que se utilizan para indicar la longitud de la propia extensión de seguridad, así como el CRC del EPDU están marcados como reservados y no tienen ningún uso asociado. A su vez, todos los campos que forman parte del EPDU se utilizan para realizar el cálculo de la firma digital.

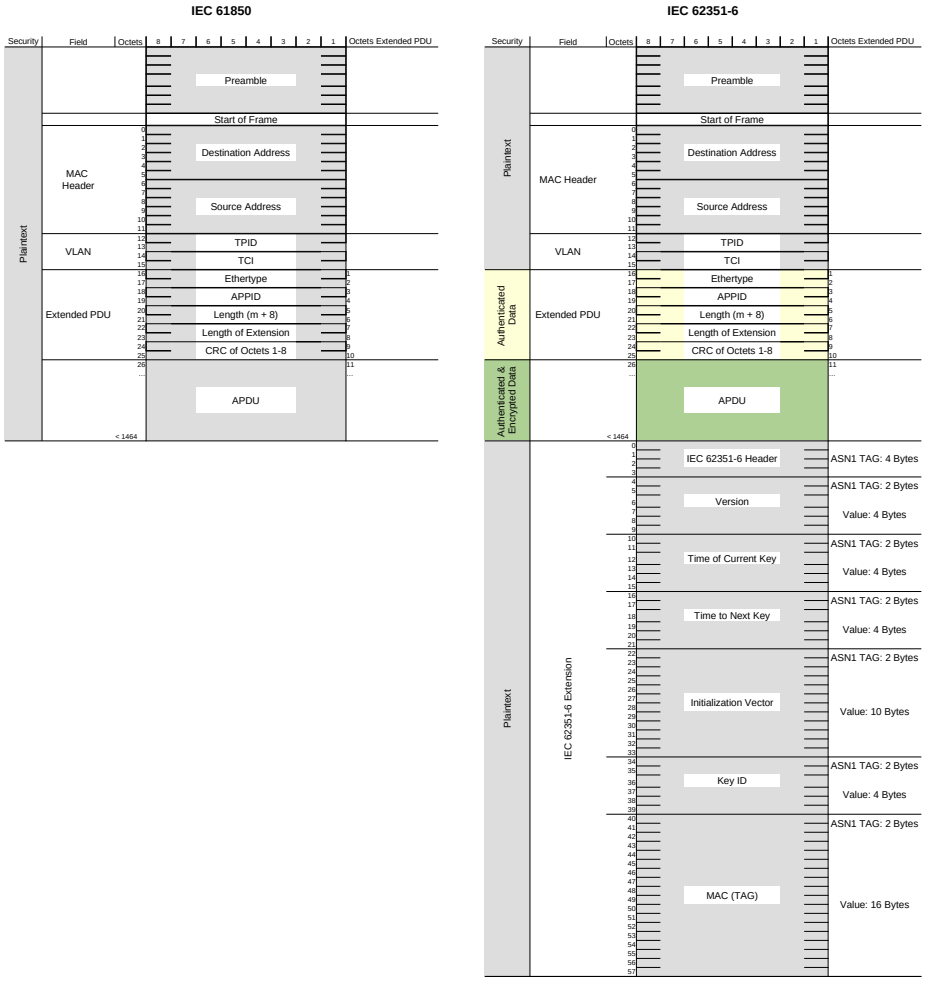

Figura 2.6: Comparación entre trama IEC 61850 sin extensión de seguridad y con extensión de seguridad IEC 62351-6

El campo APDU de los mensajes GOOSE y SV contiene toda la información a nivel de aplicación. Este se encuentra codificado utilizando una estructura dinámica ASN.1. Su tratamiento varía en función del tipo de protección que se aplique a los mensajes. Si solamente se hace uso de autenticación, el contenido de este campo debe ser utilizado para el cálculo de la firma digital. Sin embargo, si también se hace uso de cifrado, después de que los datos sean utilizados para el cálculo de la firma digital, estos serán encriptados a fin de ocultar su contenido a aquellos elementos de la red de la subestación que no conozcan las claves criptográficas utilizadas.

A continuación del APDU es donde se añade la extensión de seguridad IEC 62351-6. Al igual que la codificación de la propia APDU, la extensión de seguridad se encuentra codificada utilizando ASN.1 Basic Encoding Rules (BER) y cuenta con los siguientes campos:

- Cabecera: marca el inicio de la extensión de seguridad IEC 62351-6. Este campo tiene un tamaño de 4 bytes y se componen exclusivamente de la cabecera ASN.1.
- Versión: indica el número de versión del estándar que se ha utilizado para realizar la protección del mensaje. Por defecto este campo está fijado a un valor de 1. Este campo tiene un tamaño de 6 bytes divididos en 2 bytes de la cabecera ASN.1 y 4 bytes para la codificación del número de versión.
- Time of Current Key (ToCK): representa el número de segundos desde el 1 de enero de 1970 para indicar de forma inequívoca el instante temporal en el que la clave criptográfica utilizada para la protección del mensaje fue marcada como válida para su uso. Este campo tiene un tamaño de 6 bytes divididos en 2 bytes de la cabecera ASN.1 y 4 bytes para la codificación del número de segundos.
- Time to Next Key (TtNK): representa el número de segundos restantes hasta que la siguiente clave criptográfica sea marcada como válida para su uso. Este campo tiene un tamaño de 6 bytes divididos en 2 bytes de la cabecera ASN.1 y 4 bytes para la codificación del número de segundos.
- Initialization Vector (IV): se trata de un campo opcional requerido por algunos algoritmos criptográficos de generación de firma digital o cifrado, como por ejemplo AES Galois Message Authentication Code (AES-GMAC) o AES-GCM. Se trata de un número aleatorio utilizado por dichos algoritmos a fin de evitar que al utilizar los mismos datos en sucesivas ocasiones para generar la firma digital o encriptarlos, el resultado obtenido sea el mismo. Por lo tanto, es necesario asegurarse que cada mensaje que sea protegido utilizando una misma clave criptográfica utilice un valor de IV

distinto. Este campo tiene un tamaño de 12 bytes divididos en 2 bytes de la cabecera ASN.1 y 10 bytes para la codificación del valor del IV.

- Identificador de clave: proporciona al receptor del mensaje un identificador con el que conocer la clave criptográfica que ha sido utilizada por el generador del mensaje para realizar el cálculo de la firma digital o encriptar el propio mensaje. Su valor es asignado por el Key Distribution Center (KDC) en el proceso de distribución de claves de acuerdo al estándar IEC 62351-9. Este campo tiene un tamaño de 6 bytes divididos en 2 bytes de la cabecera ASN.1 y 4 bytes para la codificación del identificador de clave.
- Firma digital: se trata del propio valor de la firma digital que autentica los datos desde el Ethertype hasta el final del APDU. Este campo tiene un tamaño de 18 bytes divididos en 2 bytes de la cabecera ASN.1 y 16 bytes para la codificación del identificador de clave.

2.4.3 Evolución de IEC 62351-6

Desde que en el año 2007 se publicase la primera versión del estándar IEC 62351-6, han sido numerosas las publicaciones que han estudiado y analizado la viabilidad de los mecanismos de seguridad descritos en el documento. En [37] se presenta un análisis a cerca de los impactos en el rendimiento introducidos por el estándar IEC 62351-6. En él, los autores describen como la criptografía de clave asimétrica, tal y como se había configurado en esta versión del estándar, no es una opción viable para proteger comunicaciones de tiempo real. El análisis muestra como el impacto temporal de este tipo de criptografía es mayor para los dispositivos con capacidad de procesamiento limitada, tales como IEDs y MUs. En sus pruebas los autores generan firmas digitales haciendo uso del algoritmo Rivest, Shamir and Adleman (RSA), tal y como se especifica en el estándar. Para ello utilizan una implementación software del mismo, obteniendo como resultado un tiempo mínimo de 1,5 ms para generar la firma digital. En sus pruebas este tiempo se ve incrementado hasta los 4 ms cuando se utiliza la longitud de clave recomendada por el estándar, que es de 1024 bits. Por lo tanto, sus resultados demuestran que no es viable utilizar implementaciones del algoritmo RSA para proteger mensajes GOOSE y SV en los SAS más estrictos, que exigen la generación, transmisión y procesamiento de los mensajes en un tiempo inferior a 3 ms. Por otro lado, en [38] los autores presentan una pruebas similares en las que hacen uso de algunos de los procesadores más potentes disponibles en su momento para evaluar si sería posible hacer uso de RSA para la protección de mensajes GOOSE y SV. Los resultados de las pruebas confirman que aun usando Central Processing Units (CPUs) de alto rendimiento el tiempo necesario para efectuar el cálculo de la firma digital utilizando RSA está en el orden de los milisegundos.

Como alternativa al uso del algoritmo criptográfico RSA, en [39] se propone el uso del algoritmo de firma digital RSA Probabilistic Signature Scheme (RSASSA-PSS), el cual está basado en la Request For Comments (RFC) 3447 del Internet Engineering Task Force (IETF). Puesto que este algoritmo requiere de una capacidad de procesamiento inferior a RSA, se espera que su uso mejore los resultados obtenidos con respecto a implementaciones basadas en RSA. A pesar de que los resultados muestran una mejora en el tiempo necesario para calcular la firma digital de los mensajes GOOSE y SV, este sigue siendo demasiado alto como para garantizar que los mensajes sean entregados en un tiempo inferior a 3 ms tal y como se establece en el estándar IEC 61850.

A raíz de recibir y analizar los resultados obtenidos por la comunidad científica e investigadora a cerca de los problemas temporales derivados de generar las firmas digitales de los mensajes GOOSE y SV utilizando el algoritmo criptográfico RSA y ciertas variantes del mismo, el IEC comenzó a trabajar en una nueva versión del estándar IEC 62351-6 que hiciera frente a dichos problemas. Finalmente, en 2020 se publicó la nueva versión del estándar IEC 62351-6:2020 [40]. El principal cambio de está versión fue el remplazo del algoritmo criptográfico RSA para la generación de las firmas digitales de los mensajes GOOSE y SV. Así, tal y como se muestra en la Tabla 2.5 se introdujeron los algoritmos Secure Hash Algorithm-256 (SHA-256), AES-GMAC y AES-GCM para encriptar y autenticar los mensajes.

Tabla 2.5: Algoritmos criptográficos en IEC 62351-6:2020

Algoritmos	**Bits**	**Uso**	**Seguridad**
SHA-256	80	Obligatorio	Autenticación
SHA-256	128	Obligatorio	Autenticación
SHA-256	256	Obligatorio	Autenticación
AES-GMAC	64	Obligatorio	Autenticación
AES-GMAC	128	Obligatorio	Autenticación
AES-GCM	64	Opcional	Autenticación y Cifrado
AES-GCM	128	Opcional	Autenticación y Cifrado

Tanto SHA-256, AES-GMAC como AES-GCM son algoritmos criptográficos de clave simétrica. Este tipo de criptografía reduce considerablemente la carga computacional requerida con respecto a algoritmos basados en criptografía de clave asimétrica, como RSA. Por lo tanto, la introducción de estos cambios por parte del IEC pretendía habilitar protección de mensajes GOOSE y SV sin comprometer el tiempo de entrega de los mismos. Sin embargo, los algoritmos de criptografía de clave simétrica trasladan parte de la complejidad al sistema de distribución de claves definido en IEC 62351-9, ya que es necesario que los extremos de una comunicación protegida mediante criptografía de clave simétrica

compartan una clave común secreta [41].

La nueva revisión del estándar IEC incluye el uso del algoritmo AES-GCM para proporcionar tanto autenticación de los mensajes como cifrado. Este algoritmo ha demostrado ser extremadamente eficiente cuando es implementado en hardware, consiguiendo procesar altos volúmenes de datos con una baja latencia [42–44]. Sin embargo, conseguir encriptar los mensajes GOOSE y SV sin afectar a su tiempo de entrega máximo de 3 ms sigue siendo un reto tecnológico. Por tanto, el estándar IEC 62351-6:2020 sigue especificando como opcional la confidencialidad de los mensajes.

2.4.4 Estado del arte en AES-GCM para SAS

Los dispositivos SAS, especialmente las MUs generan de forma periódica flujos de tráfico multicast a tasa de transferencia de información elevada. Esta varía en función de varios factores de configuración tales como la frecuencia de funcionamiento de la red eléctrica, la frecuencia de muestreo o el número de muestras enviadas en cada mensaje SV que genera la MU.

En configuraciones SAS estándar es habitual que este tipo de dispositivos generen mensajes multicast con una periodicidad de 250 µs. Por lo tanto, teniendo en cuenta la naturaleza de los mensajes generados y el número de dispositivos que conforman la red, la tasa de transferencia agregada de todos los dispositivos de la red se encuentra en el rango de los cientos de Mbps. A continuación se recogen los requisitos que debe satisfacer la arquitectura AES-GCM para garantizar la implementación del estándar IEC 62351-6 en SAS:

- Baja latencia: puesto que los mensajes GOOSE y SV tienen un tiempo de entrega máximo de 3 ms, es necesario que el mecanismo de seguridad aplicado a los mensajes se complete en un tiempo en el orden de los microsegundos. El tiempo empleado para realizar la generación y comprobación de la firma digital, así como encriptar y desencriptar los datos debe ser inferior a los 0,4 ms.
- Uso de recursos eficiente: la capacidad de procesamiento de los IEDs y MUs es limitada. Por lo tanto, es necesario que la protección de los mensajes GOOSE y SV no haga uso de muchos recursos lógicos de la Field Programmable Gate Arrays (FPGA) o trabaje a frecuencias de reloj elevadas.
- Adaptabilidad: es necesario que la capacidad de procesamiento criptográfico sea adaptable en función del tipo de dispositivo SAS y el caso de uso concreto.

- Bidireccionalidad: la arquitectura AES-GCM debe ser capaz de tanto encriptar como desencriptar los mensajes enviados y recibidos respectivamente.
- Independencia de claves: tal y como se define en IEC 62351-9, las claves de los dispositivos en SAS se cambia de forma dinámica. Por lo tanto, la arquitectura AES-GCM debe ser independiente de las claves a utilizar.
- Tamaño de datos variable: es necesario que la arquitectura AES-GCM propuesta pueda procesar tramas Ethernet de cualquier tamaño sin que esto además afecte a su rendimiento.

El estándar AES es uno de los algoritmos criptográficos de clave simétrica más extendidos [45, 46]. Este algoritmo consiste en un bloque criptográfico que es capaz de procesar bloques de datos de 128 bits utilizando tamaños de claves de 128, 192 y 256 bits. Por otro lado, tal y como se define en [47], Galois Counter Mode (GCM) es un modo de uso del algoritmo AES que usa una función de Hash universal sobre el campo binario Galois Field (GF) para proporcionar integridad, autenticidad y confidencialidad de los datos. Este algoritmo resulta de especial interés puesto que es posible realizar implementaciones para FPGAs que alcancen un nivel de rendimiento y eficiencia altos haciendo uso de técnicas de paralelización y procesamiento secuencial [48, 49].

Es posible encontrar en la literatura múltiples propuestas a cerca de arquitecturas de los algoritmos AES y AES-GCM para FPGAs. En [50] se define una arquitectura específicamente diseñada para proporcionar confidencialidad a datos de alta velocidad que cuenten con requisitos de tiempo real. Los resultados de implementación muestran que esta arquitectura es capaz de alcanzar tasas de procesamiento de datos de 66,1 Gbps o más cuando se utilizan múltiples instancias del IP en paralelo. Sin embargo, estos resultados dependen de la capacidad de que la frecuencia de reloj del sistema sea del orden de los 500 Mhz, lo cual limita el uso de esta propuesta a dispositivos de alto rendimiento. De forma similar, en [51] se proponen dos arquitecturas de alto rendimiento que permiten autenticar datos utilizando AES-GCM. En este caso, debido al uso masivos de etapas de registros en las arquitecturas, es posible alcanzar frecuencias de reloj elevadas que permiten conseguir tasas de procesamiento de datos de 40 Gbps. Sin embargo, esta arquitectura tiene un alto uso de recursos de la FPGA y reduce su rendimiento si es necesario procesar grandes volúmenes de datos de tamaño reducido.

En [52] se describe una arquitectura del algoritmo AES-GCM que sea capaz de encriptar y desencriptar flujos de datos de alta velocidad. Sin embargo, esta arquitectura se basa en el uso de un algoritmo software que haga un pre-procesamiento

de la expansión de claves AES, reduciendo así el uso de recursos FPGA y aumentando la eficiencia. Este enfoque es viable en usos en los que las claves tengan un periodo de renovación elevado y estas se usen para proteger todos los datos, como por ejemplo ocurre en las VPNs. Por lo tanto, dada la naturaleza de los datos transmitidos en SAS y la periodicidad de renovación de claves establecida en IEC 62351-9, esta solución no resulta viable.

En [53] se presenta una arquitectura AES-GCM de alto rendimiento. Los resultados muestran que se puede conseguir un rendimiento de hasta 100 Gbps cuando se combinan 4 instancias del IP. Internamente la arquitectura propuesta utiliza un procesamiento secuencial de cuatro etapas que permite utilizar altas frecuencias de reloj. Por contra, esta arquitectura requiere del uso de 4 instancias del IP propuesto para procesar cualquier tipo y tasa de datos. Esto hace que el uso de recursos sea elevado incluso en entornos en los que no sea necesario alcanzar niveles de rendimiento tan elevados.

La Tabla 2.6 recoge las arquitecturas AES-GCM para FPGA analizadas y evalúa el cumplimiento de los requisitos establecidos para la implementación de IEC 62351-6 en SAS. Como se puede ver, ninguna de las arquitecturas es capaz de satisfacer los requisitos establecidos. La arquitectura propuesta en [50] no es capaz de cumplir con los requisitos de uso de recursos eficiente y adaptabilidad a cada caso de uso. El rendimiento de esta arquitectura excede las necesidades de dispositivos SAS. Por contra, el uso de recursos FPGA asociado a este elevado rendimiento hace que su implementación no sea viable en SAS.

Tabla 2.6: Estado del arte en AES-GCM

Requisitos	**[50]**	**[51]**	**[52]**	**[53]**
Integridad y autenticidad	X	X	X	X
Confidencialidad	X	-	X	X
Baja latencia	X	-	-	-
Uso de recursos FPGA eficiente	-	-	-	-
Adaptabilidad	-	-	-	-
Bidireccionalidad	X	-	X	X
Independencia de claves	X	X	-	X
Tamaño de datos variable	X	-	X	X

Por lo tanto, después de analizar los requisitos que debe satisfacer la arquitectura del algoritmo criptográfico AES-GCM para hacer viable su implementación en SAS, se puede determinar que no se ha encontrado en la literatura ninguna alternativa que cumpla con los requisitos establecidos.

2.4.5 Estado del arte en IEC 62351-6

La mayoría de propuestas existentes en la literatura no han abordado el reto tecnológico que supone proporcionar de forma combinada integridad, autenticidad y confidencialidad de los mensajes IEC 61850 GOOSE y SV de acuerdo al estándar IEC 62351-6. La mayor parte de las soluciones propuestas solo proporcionan integridad y autenticidad de los datos, pero no integridad. A pesar de que el estándar IEC 62351-6 solamente fija como opcional proporcionar confidencialidad de los mensajes, tal y como se ha analizado en secciones anteriores del presente documento, muchas de las vulnerabilidades en SAS pueden mitigarse si se dota de esta característica a las comunicaciones.

En [54] los autores presentan una metodología basada en un desarrollo software para proporcionar integridad y autenticidad de los mensajes GOOSE y SV. Para ello hacen uso del algoritmo de generación de firma criptográfica RSA-Probabilistic Signature Scheme based on Signature Scheme with Appendix (RSASSA-PKCS1-v15). Los resultados muestran que a pesar de haber utilizados procesadores modernos de alto rendimiento, RSASSA-PKCS1-v15 es demasiado lento para ser usado en aplicaciones con requisitos de tiempo real como GOOSE y SV. En este trabajo, los autores también describen un mecanismo de protección basado en el algoritmo criptográfico SHA-256. Sin embargo, no se proporcionan resultados de rendimiento para el cálculo de la firma digital cuando se emplea dicho algoritmo.

Adicionalmente, en [37] y [38] se hace uso del algoritmo RSA para calcular la firma digital de los mensajes GOOSE y SV. Los autores hacen uso de distintas familias de procesadores para analizar como afecta la capacidad de procesamiento de cada procesador en el tiempo necesario para calcular la firma digital. Los resultados muestran que solamente la familia de procesadores Intel Xeon es capaz de generar la firma digital en menos de 1 ms. Finalmente, los autores de este trabajo también llevan a cabo un análisis del uso de los algoritmo AES-GMAC para generar la firma digital de los mensajes GOOSE y SV. En este caso, los resultados muestran como este algoritmo de firma digital de criptografía de clave simétrica permite que dispositivos con capacidad de procesamiento reducida, como Raspberry Pi 2 o BeagleBone, sean capaces de realizar el cálculo de la firma digital en unos pocos microsegundos.

Por otro lado, en [55] los autores utilizan un procesador Intel Celeron con 4 GB de memoria RAM para evaluar el rendimiento de la autenticación de mensajes en dispositivos de bajo rendimiento cuando se utilizan los algoritmos AES-GMAC y SHA-256. Los resultados de este análisis muestran que el retardo extremo a extremo para la entrega de mensajes GOOSE autentificados con dicho algoritmo está por debajo del requisito de 3 ms aun en los peores casos.

En [56] los autores presentan su solución para proporcionar integridad, autenticidad y confidencialidad a los mensajes GOOSE y SV. De acuerdo a su análisis, el estándar IEC 62351-6 no especifica ningún método para proporcionar confidencialidad. Sin embargo, tal y como se muestra en la Tabla 2.5, el estándar define el algoritmo AES-GCM para autenticar y encriptar los mensajes. Además, en dicho trabajo, los autores proponen 3 métodos derivados del IEC 62351-6 que hacen uso de los algoritmos criptográficos AES y SHA-256 para proteger de tres formas distintas las comunicaciones IEC 61850 con requisitos de tiempo real. El primero de los métodos, el cual llaman Encrypt-then-MAC (EtM), encripta el APDU de los mensajes GOOSE haciendo uso de AES. Después, se utilizan los datos encriptados para calcular la firma digital utilizando SHA-256. Este procedimiento requiere que en el proceso de transmisión primero se encripten los datos y después se realice el cálculo de la firma digital. Sin embargo, en el proceso de recepción los datos no pueden ser desencriptados hasta que el mensaje se haya recibido completamente y este haya sido autenticado. Como consecuencia, el tiempo necesario para proteger los mensajes puede crecer considerablemente.

El segundo método descrito en [56] es denominado Encrypt and MAC (E&M) por los autores. El procedimiento de este método consiste en encriptar el APDU del mensaje GOOSE y calcular la firma digital sobre el APDU original del mensaje, esto es, el APDU sin encriptar. El problema asociado a este sistema, el cual también afecta al método EtM, es que puesto que el APDU está encriptado con AES, las partes del mismo que tienen un valor fijo siempre producirán el mismo resultado si se utiliza la misma clave. Por lo tanto, un atacante podría utilizar esta información para obtener información de los mensajes o incluso obtener la propia clave AES y tener así acceso a los mensajes en claro.

Finalmente, en el tercer método denominado MAC then Encrypt (MtE), se realiza el cálculo de la firma digital sobre el APDU del mensaje GOOSE para posteriormente encriptar tanto el propio APDU como la extensión de seguridad. Este procedimiento hace que desencriptar el mensaje sea tarea imposible para el receptor ya que la extensión de seguridad IEC 62361-6 es donde se almacena la información criptográfica necesaria para poder desencriptar le mensaje. Por lo tanto, este campo de los mensajes IEC 61850 con extensión de seguridad IEC 62351-6 siempre debe estar sin encriptar. Para solventar este problema, los autores proponen utilizar algunos bits de las cabeceras de los mensajes GOOSE y SV para especificar el método utilizado de los 3 propuestos. Ya que los tres soluciones propuestas son modificaciones ad-hoc del estándar IEC 62351-6, no es posible garantizar la interoperabilidad entre dispositivos de distintos fabricantes, lo cual es una de las principales razones por las que se introdujo el estándar IEC 61850.

La Tabla 2.7 recoge las características principales de las distintas propuestas presentes en la literatura para proteger comunicaciones IEC 61850. La mayor parte de las propuestas utilizan algoritmos criptográficos de firma digital basados en RSA. Como se ha analizado previamente, este algoritmo, al utilizar criptografía de clave asimétrica, requiere de una capacidad de procesamiento elevada. Por lo tanto, los resultados obtenidos al proteger mensajes GOOSE y SV no satisfacen el requisito de transferencia de los mensajes en menos de 3 ms. Por otro lado, existen dos propuestas que utilizan los algoritmos criptográficos de firma digital SHA-256 y AES-GMAC definidos en la última revisión del estándar IEC 62351-6. Estos algoritmos, al utilizar criptografía de clave simétrica, tienen unos requisitos de capacidad de procesamiento muy inferiores respecto a RSA. Esto tiene como consecuencia que utilizando procesadores de alto rendimiento sea posible generar la firma digital de los mensajes sin afectar al tiempo de transferencia máximo de 3 ms. Sin embargo, las propuestas analizadas en [55] solo proporcionan integridad y autenticidad y no confidencialidad.

Tabla 2.7: Estado del arte en IEC 62351-6

Propuesta	Algoritmo	Seguridad	Tipo	Latencia Máxima	Rendimiento Máximo	Determinista	Cumple IEC 62351-6
[37]	RSA	Firma digital	SW	4000 µs	-	No	No (Tiempo)
[37]	RSA	Firma digital	HW	1917 µs	-	Si	No (Tiempo)
[38]	RSA	Firma digital	SW	6000 µs	-	No	No (Tiempo)
[39]	RSASSA-	Firma digital	SW	942 µs	-	No	No (Tiempo)
[39]	PKCS1-v1_5	Firma digital	SW	3560 µs	-	No	No (Tiempo)
[55]	SHA-256	Firma digital	SW	12,7 µs	-	No	Si (Solo firma digital)
[55]	AES-GMAC	Firma digital	SW	5,4 µs	-	No	Si (Solo firma digital)
[56]	EtM (AES y SHA-256)	Firma digital y cifrado	SW	242 µs	-	No	No (Formato)
[56]	E&M (AES y SHA-256)	Firma digital y cifrado	SW	235 µs	-	No	No (Formato)
[56]	MtE (AES y SHA-256)	Firma digital y cifrado	SW	284 µs	-	No	No (Formato)

Las propuestas en la literatura que proporcionan simultáneamente integridad, autenticidad y confidencialidad se basan en procedimientos propietarios. Estos si que consiguen obtener una latencia máxima que no afecte al tiempo de entrega de los mensajes. Sin embargo, ninguna de ellas cumple el estándar IEC 62351-6 por lo que se pierde la interoperabilidad entre dispositivos IEC 61850. Además, en ninguna de las propuestas analizadas (tanto para solo autenticar como autenticar y encriptar los mensajes GOOSE y SV) se especifica como se ven afectados los resultados obtenidos en función del número de mensajes a procesar. Además, los autores tampoco especifican la tasa de datos que cada una de las propuesta es capaz de procesar ni si los resultados de latencia obtenidos se ven afectados en función de la cantidad de mensajes a procesar. Por lo tanto, se puede determinar

que no existe ninguna propuesta en la literatura que sea capaz de proporcionar integridad, autenticidad y confidencialidad a las comunicaciones IEC 61850 con requisitos de tiempo real GOOSE y SV de acuerdo al estándar IEC 62351-6.

Capítulo 3

Criptografía de clave simétrica

3.1 Introducción

Tal y como se ha analizado en el Capítulo 2, en la primera versión del estándar IEC 62351-6 se hacia uso de algoritmos criptográficos basados en criptografía de clave asimétrica para proporcionar autenticidad, integridad y confidencialidad a los mensajes IEC 61850 GOOSE y SV. En concreto, el estándar IEC 62351-6 definía el uso del algoritmo RSA y solamente establecía la firma digital como requisito obligatorio, dejando el cifrado de los datos como opcional. Sin embargo, debido a los estrictos requisitos temporales establecidos para la distribución y procesamiento de este tipo de mensajes, la viabilidad del uso del algoritmo RSA y cualquier otro basado en criptografía de clave asimétrica quedaba en entredicho.

Después de múltiples análisis por parte de la comunidad científica e investigadora, se pudo determinar como el uso de algoritmos de criptografía de clave asimétrica, en concreto RSA, no permitían satisfacer los estrictos requisitos temporales establecidos para los mensajes GOOSE y SV del estándar IEC 61850. Como respuesta a los resultados obtenidos, el IEC lanzó una nueva revisión del estándar IEC 62351-6:2020. En esta nueva revisión se sustituía el uso de algoritmos de clave asimétrica por algoritmos criptográficos de clave simétrica. Estos, se caracterizan por tener unos requisitos de capacidad de procesamiento sensiblemente inferiores a los establecidos por los algoritmos de criptografía de clave asimétrica. Sin embargo, como contra partida, la complejidad del sistema de distribución de

claves se incrementa cuando se hace uso de algoritmos de criptografía de clave simétrica. La definición del sistema de distribución de claves se recoge en el estándar IEC 62351-9.

El algoritmo AES [57] es uno de los algoritmos de criptografía de clave simétrica más extendidos. AES permite cifrar datos a alta velocidad y con una baja latencia, especialmente cuando se hace uso de este algoritmo en modo contador [58]. El modo contador es uno de los diversos modos de usos que existen del algoritmo AES. Los modos de uso definen que datos se utilizan como datos de entrada del algoritmo y la forma en la que los resultados obtenidos se utilizan para generar los datos encriptados [59]. Sin embargo, AES no proporciona autenticidad e integridad de los datos. Puesto que no existía ningún algoritmo criptográfico que fuera capaz de proporcionar autenticidad e integridad de los datos a las velocidades que AES proporcionaba confidencialidad, surgió la necesidad de definir un algoritmo basado en AES que proporcionase dichas características a través del cálculo de la firma digital .

Así, GCM es un modo de funcionamiento de cifrado de bloque de datos basado en el algoritmo AES en modo contador, que utiliza Hashing universal sobre un campo de Galois binario para proporcionar cifrado y autenticación de los datos. Su estructura permite diseñar arquitecturas hardware que sean capaces de alcanzar altas tasas de transferencia de datos a la vez que se consiguen unos valores de latencia reducidos. Para ello, se establece como requisitos que el algoritmo AES-GCM [60] sea capaz de admitir implementaciones del mismo basadas en técnicas de paralelización y procesamiento segmentado de los datos. Otros modos de autenticación basados en AES no cumplen estas dos características fundamentales. El modo de uso Cipher Block Chaining (CBC) Message Authentication Code (CBC-MAC) del algoritmo AES, así como el resto de modos que lo utilizan para proporcionar autenticidad de los datos, como Cipher block chaining - message authentication code (CCM) [61], Encrypt-then-Authenticate-then-translate (EAX) [62] y One-key MAC (OMAC) [63] no pueden ser paralelizados o procesados de forma segmentada. Por lo tanto, estos algoritmos no resultan viables para procesar elevados flujos de datos con una baja latencia.

Consecuentemente, el algoritmo criptográfico elegido en la última versión del estándar IEC 62351-6:2020 es AES-GCM. Dicho algoritmo puede ser utilizado tanto para proporcionar exclusivamente autenticidad e integridad a los mensajes, como para proporcionar autenticidad, integridad y confidencialidad de forma simultánea, lo cual hace su uso idóneo para SAS. Además, a pesar de que el estándar sigue estableciendo la confidencialidad como un requisito opcional, el uso del algoritmo AES-GCM abre la puerta a su uso cuando se utilizan implementaciones hardware de alto rendimiento del mismo. Este hecho, posibilitaría

proporcionar una protección integral de los mensajes GOOSE y SV sin afectar a los requisitos temporales establecidos en el estándar IEC 61850 para este tipo de mensajes. Sin embargo, después de realizar en el Capítulo 2 un análisis del estado del arte en materia de implementaciones AES-GCM para SAS, se pudo determinar que en la literatura no existe una propuesta que cumpla los requisitos establecidos para hacer viable la autenticación y cifrado de mensajes GOOSE y SV.

En este capítulo se proporciona un análisis detallado de los algoritmos criptográfico AES y AES-GCM. Para ello, se realiza una descripción de los fundamentos matemáticos en los que dichos algoritmos se basan, así como las transformaciones de datos y operaciones que los componen.

3.2 Fundamentos matemáticos del algoritmo AES

Antes de realizar la descripción del algoritmo criptográfico AES o sus operaciones fundamentales, es necesario realizar una descripción sobre la composición, estructura y organización de los datos. Además, AES se fundamenta en una serie de conceptos y operaciones matemáticas que serán descritas a continuación y servirán como base de las transformaciones de los datos que se realizan durante la ejecución del algoritmo.

Las entradas y salidas del algoritmo AES consisten en secuencias de datos de 128 bits. Generalmente se refiere a estas secuencias de datos como bloques, siendo el número de bits que componen dichos bloques la longitud de los mismos. Por otro lado, la clave criptográfica para el algoritmo AES está compuesta por secuencias de 128, 192 o 256 bits. Así, tanto el tamaño de los datos de entrada y salida, como el tamaño de las claves criptográficas a utilizar suponen una característica fija y esencial de este algoritmo.

La unidad de procesamiento básico del algoritmo AES es el byte, siendo este una secuencia de 8 bits que son tratados como una única entidad. De esta forma, las entradas, salidas y claves criptográficas son secuencias de bits que se procesan como arrays de bytes. Estos se crean mediante la división de de las secuencias de bits en grupos de 8 bits contiguos que conforman arrays de bytes. Los bits que conforman un byte son interpretados como elementos de campo finito para el cual se utiliza una representación polinómica de los mismos expresada en la Ecuación 3.1. Los elementos de campo finito pueden ser sumados o multiplicados, sin embargo, la forma en la que se llevan a cabo estas operaciones difiere de la usada para números convencionales.

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x^1 + b_0x^0 = \sum_{i=0}^{7} b_i x^i \qquad (3.1)$$

Las operaciones internas del algoritmo AES se llevan a cabo utilizando arrays de bytes bidimensionales llamados State. Por lo tanto, el State se compone de 4 filas de bytes, cada una teniendo un número Nb de bytes y siendo Nb la longitud de bloque dividida entre 32, es decir: $Nb = 128/32 = 4$. De esta forma, un byte que forme parte del State $s[r, c]$ cuenta con dos índices, siendo r el número de fila y teniendo un rango $0 \leq r < 4$, mientras que c define su número de columna y tiene un rango $0 \leq c < Nb$. Por lo tanto, cuando un bloque de datos de entrada se carga en el State, la organización de los bytes que componen el State sigue la relación que se define en la Ecuación 3.2. En cambio, cuando la salida del State es copiada a un bloque de datos de salida, la organización de los bytes sigue la Ecuación 3.3.

$$s[r, c] = in[r + 4c] \qquad \text{for } 0 \leq r < 4 \text{ and } 0 \leq c < Nb \qquad (3.2)$$

$$out[r + 4c] = s[r, c] \qquad \text{for } 0 \leq r < 4 \text{ and } 0 \leq c < Nb \qquad (3.3)$$

Una vez definida la estructura y organización básica de los datos en el algoritmo AES, es necesario describir los conceptos matemáticos sobre los que se fundamentan las operaciones y tratamiento de los datos descritos en AES.

La suma de dos elementos pertenecientes a un campo finito se lleva a cabo mediante la "suma"de los coeficientes de las potencias correspondientes a los polinomios de los dos elementos a ser sumados. Esta suma se realiza mediante la operación lógica XOR, la cual es denominada mediante el símbolo $\oplus$. Consecuentemente, la resta de polinomios se lleva a cabo de forma idéntica a la suma. Por otro lado, en la representación polinómica, la multiplicación en $GF(2^8)$, la cual es denominada mediante el símbolo $\bullet$, se corresponde con la multiplicación de polinomios y la realización del cálculo del módulo del resultado respecto al polinomio irreducible de grado 8. El cálculo del módulo de un polinomio se representa mediante la expresión $m(x)$. Cabe destacar que un polinomio resulta irreducible solamente si sus divisores son uno y el propio polinomio. En el algoritmo AES, el polinomio irreducible es el representado en la Ecuación 3.4.

$$m(x) = x^8 + x^4 + x^3 + x + 1 \qquad (3.4)$$

Así, la reducción modular utilizando $m(x)$ asegura que el resultado de la multiplicación de dos polinomios sea un polinomio binario cuyo grado sea inferior a 8. Conseguir un polinomio de grado inferior a 8 permite que este pueda ser representado mediante un byte. A diferencia de la suma, no existe una operación simplificada a nivel de byte que se corresponda con la multiplicación descrita. Además, para cualquier polinomio binario $b(x)$ que sea distinto de 0 y de grado inferior a 8, el inverso de la multiplicación de $b(x)$, denominado $b^{-1}(x)$, cumple el uso del algoritmo Euclidiano extendido [64] para realizar el cálculo de los polinomios $a(x)$ y $c(x)$ de tal forma que

$$b(x)a(x) + m(x)c(x) = 1 \quad \Rightarrow \quad a(x) \bullet b(x) \bmod m(x) = 1 \tag{3.5}$$

Como consecuencia de la Ecuación 3.5 se obtiene que

$$b^{-1}(x) = a(x) \bmod m(x) \tag{3.6}$$

Teniendo en cuenta el conjunto de los 256 valores posibles que puede tomar un byte y haciendo uso de la suma basada en operaciones XOR y la multiplicación previamente descrita, se puede determinar que el conjunto de los valores obtenidos tiene una estructura de campo finito $GF(2^8)$. Además, para cualquier $a(x)$, $b(x)$ y $c(x)$ que pertenezcan al campo finito, se cumple que

$$a(x) \bullet (b(x) + c(x)) = a(x) \bullet b(x) + a(x) \bullet c(x) \tag{3.7}$$

En el caso de multiplicar el polinomio definido en la Ecuación 3.1 con el polinomio x, el resultado obtenido sería

$$b_7x^8 + b_6x^7 + b_5x^6 + b_4x^5 + b_3x^4 + b_2x^3 + b_1x^2 + b_0x \tag{3.8}$$

De esta forma, el resultado de la operación $x \bullet b(x)$ se obtiene de la reducción del resultado obtenido en la Ecuación 3.8. Dicha reducción se realiza mediante el cálculo del módulo de dicho resultado con $m(x)$, tal y como se define en la Ecuación 3.4. En caso de que $b_7 = 0$, el resultado ya se encontraría en su forma reducida. Por el contrario, si $b_7 = 1$, la reducción se llevaría a cabo realizando la resta del polinomio irreducible $m(x)$, o lo que es la mismo, realizando la operación XOR $m(x)$. Por lo tanto, se puede resumir que una multiplicación por x puede ser implementada a nivel de byte como un desplazamiento a la izquierda de los bits y la posterior operación lógica XOR del resultado del desplazamiento con

el valor hexadecimal $1b$. En el algoritmo AES esta operación toma el nombre de $xtime()$. Además, es posible realizar multiplicaciones de potencias superiores de x mediante el uso recursivo del la operación $xtime()$.

Por otro lado, es posible definir polinomios de cuatro términos que tengan coeficientes que sean elementos de campo finito. La diferencia con las operaciones previamente descritas, es que en este caso, los propios coeficientes son elementos de campo finito, es decir bytes, en vez de bits. Además, en este caso, el polinomio irreducible no es el descrito en la Ecuación 3.4, si no que es el polinomio $x^4 + 1$.

En las ecuaciones 3.9 y 3.10 se muestran dos palabras con las formas $[a_0, a_1, a_2, a_3]$ y $[b_0, b_1, b_2, b_3]$ respectivamente.

$$a(x) = a_3x^3 + a_2x^2 + a_1x + a_0 \tag{3.9}$$

$$b(x) = b_3x^3 + b_2x^2 + b_1x + b_0 \tag{3.10}$$

La suma de los dos polinomios anteriores se lleva a cabo mediante la suma de sus coeficientes de campo finito. Esta suma se corresponde con la operación XOR de los bytes correspondientes de cada una de las palabras, tal y como se muestra en la Ecuación 3.11.

$$a(x) + b(x) = (a_3 \oplus b_3)x^3 + (a_2 \oplus b_2)x^2 + (a_1 \oplus b_1)x + (a_0 \oplus b_0) \tag{3.11}$$

Adicionalmente, la multiplicación de este tipo de elementos se lleva a cabo en dos etapas. En la primera etapa se realiza la expansión algebraica de la multiplicación polinómica $c(x) = a(x) \bullet b(x)$, dando como resultado

$$c(x) = c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0 \tag{3.12}$$

Siendo cada uno de los elementos de $c(x)$ los expresados en la Ecuación 3.13. Puesto que el resultado $c(x)$ no representa una palabra de cuatro bytes, resulta necesario realizar un segunda etapa de reducción. De esta forma, el polinomio $c(x)$ es reducido mediante el cálculo del módulo respecto a un polinomio de grado 4. Dicha operación tiene como resultado un polinomio de grado inferior a 4, el cual si que puede ser representado mediante una palabra de cuatro bytes. En el algoritmo AES dicho polinomio es $x^4 + 1$.

$$
\begin{aligned}
c_0 &= a_0 \bullet b_0 \\
c_1 &= a_1 \bullet b_0 \oplus a_0 \bullet b_1 \\
c_2 &= a_2 \bullet b_0 \oplus a_1 \bullet b_1 \oplus a_0 \bullet b_2 \\
c_3 &= a_3 \bullet b_0 \oplus a_2 \bullet b_1 \oplus a_1 \bullet b_2 \oplus a_0 \bullet b_3 \\
c_4 &= a_3 \bullet b_1 \oplus a_2 \bullet b_2 \oplus a_1 \bullet b_3 \\
c_5 &= a_3 \bullet b_2 \oplus a_2 \bullet b_3 \\
c_6 &= a_3 \bullet b_3
\end{aligned}
\tag{3.13}
$$

El cálculo del producto modular $a(x) \otimes b(x)$ viene dado por el polinomio de 4 términos expresado en la Ecuación 3.14. Además, en el caso de que $a(x)$ sea un polinomio fijo, el resultado de la multiplicación modular de la Ecuación 3.14 puede representarse de forma matricial, tal y como se muestra en la Ecuación 3.15.

$$
d(x) = d_3x^3 + d_2x^2 + d_1x + d_0 \tag{3.14}
$$

$$
d(x) = \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} \tag{3.15}
$$

Tal y como se puede ver, muchas de las operaciones matemáticas en el algoritmo AES se basan en las sumas o restas de elementos de campo finito mediante la operación lógica XOR. Además, las multiplicaciones tanto de bits como de bytes son reducidas mediante el cálculo del módulo del resultado respecto al polinomio irreducible, a fin de que el resultado final pueda ser representado con palabras de bytes, según corresponda.

3.3 Descripción del algoritmo AES

En el algoritmo AES la longitud de los bloques de datos de entrada, salida y el State es de 128 bits. Por lo tanto, $Nb = 4$, siendo así el State una matriz de $4x4 = 16$ bytes. Por otro lado, el tamaño de clave K utilizado en el algoritmo AES puede ser de 128, 192 o 256 bits. En este caso, la longitud del tamaño de la clave se representa mediante Nk, teniendo este los posibles valores $Nk = 4, 6, 8$. En el algoritmo AES se definen un conjunto de operaciones que deben ser ejecutadas en sucesivas rondas durante la ejecución del algoritmo. El número de rondas a ejecutar es dependiente del tamaño de clave utilizado y del tamaño de bloque de

los datos. Puesto que Nb tiene un valor fijo de $Nb = 4$ y que la longitud de clave tiene un valor variable $Nk = 4, 6, 8$, el número de rondas AES viene determinado por el parámetro Nr. Los posibles valores del mismo vienen expresados en la tabla 3.1.

Tabla 3.1: Número de rondas AES en función del tamaño de clave utilizado

Tamaño de clave	Nb	Nk	Nr
128 bits	4	4	10
192 bits	4	6	12
256 bits	4	8	14

Las operaciones fundamentales en las que se basa el algoritmo AES son: cifrado, cifrado inverso y expansión de clave [57]. El cifrado consiste en una serie de transformaciones sobre los bytes que componen el bloque de datos a ser procesado. De esta forma, el resultado de las transformaciones aplicadas durante la operación de cifrado da como lugar a los datos encriptados. De forma contraria, el cifrado inverso utiliza el inverso de las transformaciones que se han aplicado durante la operación de cifrado. Así, al aplicar el cifrado inverso sobre un bloque de datos encriptado, el resultado obtenido es el bloque de datos en claro antes de aplicar el cifrado. Finalmente, la operación de expansión de clave es utilizado para, a partir de la clave original, generar las palabras necesarias que permitan utilizar estas en cada una de las rondas que componen las operaciones de cifrado y cifrado inverso. Las transformaciones fundamentales, y sus inversos, en las que se basan cada una de las rondas AES son las siguientes:

- *AddRoundKey*: utiliza los datos obtenidos como resultado de realizar la expansión de clave para añadirlos al State.
- *SubBytes*: se trata de realizar la sustitución de los bytes que componen un bloque de datos utilizando para ello una tabla de sustitución denominada S-box.
- *ShiftRows*: consiste en realizar una serie permutaciones sobre las filas de datos que componen el State.
- *MixColumns*: se basa en realizar una mezcla de los datos dentro de cada una de las columnas que componen el State.

3.3.1 Expansión de clave

En el algoritmo AES se define una rutina de expansión de la clave criptográfica que es utilizada para generar distintas palabras derivadas de la clave original, la

cual es denominada K. Estas subclaves que son generadas en una serie de rondas de expansión de clave. Estas subclaves son a su vez utilizadas en cada una de las rondas que componen tanto el proceso de cifrado AES como el proceso de cifrado inverso AES. Así, el algoritmo AES necesita generar un número de subclaves $Nb(Nr + 1)$, siendo cada una de las subclaves una palabra de Nb palabras de datos. De esta forma, el número de subclaves generadas por el procedimiento de expansión de clave depende del tamaño de clave utilizado.

El procedimiento de expansión de clave AES toma la clave de entrada K y genera las diferentes subclaves de acuerdo a una serie de transformaciones que se definen a continuación.

- La transformación *SubWord* utiliza una entrada de 4 bytes sobre la que aplica a cada uno de los bytes la S-box definida en la Sección 3.3.3 para así generar una nueva palabra de salida de 4 bytes.
- La transformación *RotWord* es utilizada para utilizando una palabra de entrada de 4 bytes, aplicar una permutación cíclica sobre los bytes que la componen y generar así una nueva palabra de salida tal y como se puede observar en la Ecaución 3.16.

$$[a_0, a_1, a_2, a_3] \rightarrow \text{RotWord} \rightarrow [a_1, a_2, a_3, a_0] \tag{3.16}$$

- La constante *Rcon[i]* consiste en una serie de palabras de 4 bytes de valor constante y dependiente del número de ronda de expansión de clave. La Ecuación 3.17 describe el contenido del array de constantes *Rcon[i]*, siendo x^{i-1} potencias de 2 de x y teniendo a su vez x el valor hexadecimal $0x02$ en el campo $GF(2^8)$.

$$Rcon[i] = [x^{i-1}, 00, 00, 00] \qquad 1 \leq i < (Nr + 1) \tag{3.17}$$

El proceso de expansión de la clave AES se define en la Figura 3.1 a través de un diagrama de flujo. Tal y como se puede ver, las primeras Nk de la clave expandida se componen de la clave criptográfica original K. Las siguientes palabras $w[i]$ de la clave expandida se generan realizando la operación lógica XOR de la palabra previa $w[i-1]$ y la palabra $w[i-Nk]$ que está localizada en las Nk posiciones previas. En aquellas posiciones que sean múltiplo de Nk, se aplica una doble transformación sobre la palabra $w[i-1]$. Dicha doble transformación consiste en aplicar la transformación *RotWord* para realizar un desplazamiento cíclico de los

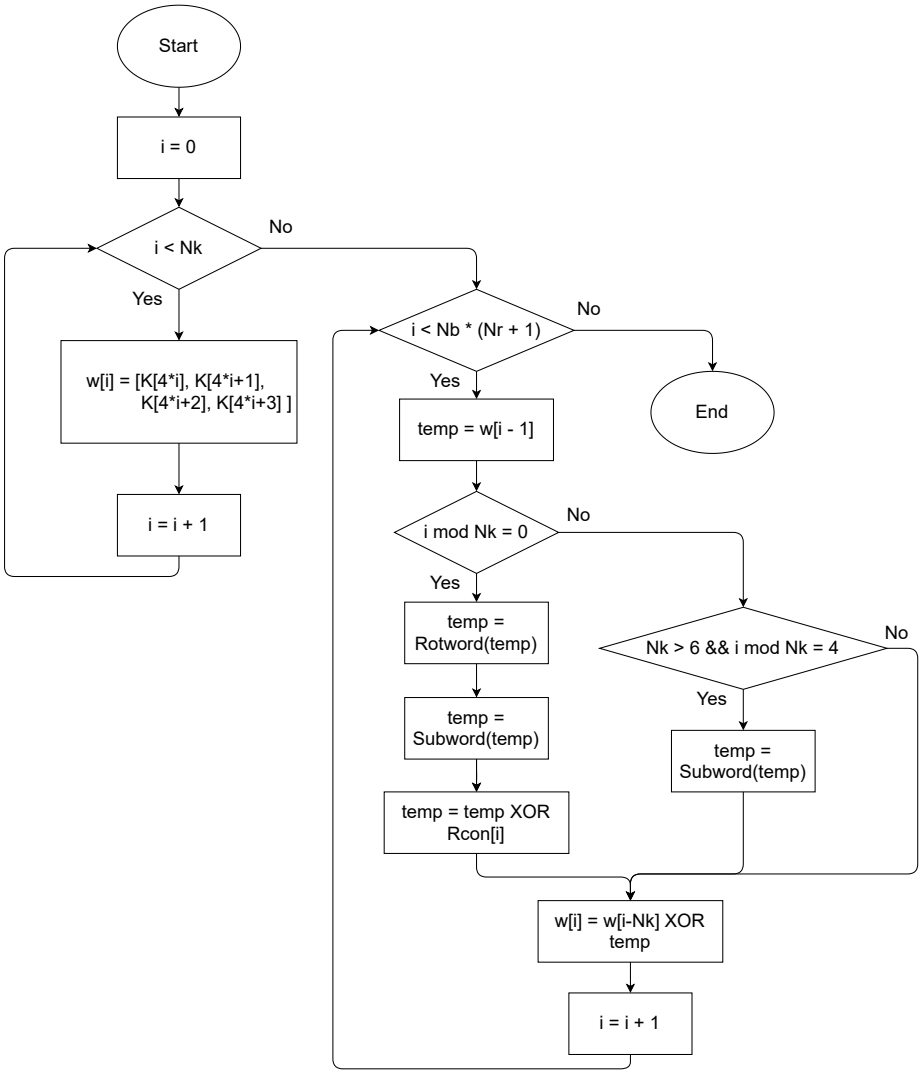

Figura 3.1: Rutina de expansión de clave en el algoritmo AES

bytes que componen la palabra. Al resultado de la transformación *RotWord* se le aplica la transformación *SubWord* para realizar una sustitución de los bytes que componen la palabra de acuerdo al contenido de la tabla S-box. Finalmente, al resultado de la transformación *SubWord* se le aplica la operación lógica XOR con la constante $Rcon[i]$. Además, cuando la clave criptográfica utilizada es de 256 bits, en el caso de que $Nk = 8$ y $i-4$ sea un múltiplo de Nk, antes de realizar la operación lógica XOR se aplicará la transformación *SubWord* sobre $w[i-1]$.

3.3.2 Transformación de datos *AddRoundKey*

La transformación de datos *AddRoundKey* consiste en añadir al State mediante la operación lógica XOR una de las subclaves generadas como resultado del proceso de expansión de clave AES. Cada una de las subclaves generadas consiste en Nb palabras, siendo cada una de ellas añadida a las columnas del State de acuerdo a la Ecuación 3.18, siendo w_i la subclave obtenida como resultado del proceso de expansión de clave y *round* el número de ronda AES.

$$[s'_{0,c}, s'_{1,c}, s'_{2,c}, s'_{3,c}] = [s_{0,c}, s_{1,c}, s_{2,c}, s_{3,c}] \oplus [w_{round*Nb+c}] \qquad \text{for } 0 \leq c < Nb \tag{3.18}$$

La Ecuación 3.19 muestra como al aplicar la transformación *AddRoundKey*, el contenido del State, denominado s, se modifica y pasa a tener el valor s'. La transformación *AddRoundKey* es utilizada tanto para el proceso de cifrado como cifrado inverso. Puesto que esta solamente consiste en realizar la operación lógica XOR entre el State y la subclave correspondiente, no existe una transformación inversa de la misma.

$$\begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} \rightarrow \text{AddRoundkey} \rightarrow \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix} \tag{3.19}$$

3.3.3 Transformación de datos *SubBytes*

La transformación de datos *SubBytes* es una sustitución de bytes no lineal que actúa de forma independiente sobre cada uno de los bytes que componen el State, haciendo uso de una tabla de sustitución denominada S-box, la cual se muestra en la Tabla 3.2. Esta tabla, tiene la propiedad fundamental de ser invertible, siendo su composición el resultado de las siguientes transformaciones:

1. Haciendo una asignación del elemento $0x00$ sobre sí mismo, se toma la multiplicación inversa en campo finito $GF(2^8)$ expresada en la Ecuación 3.5.
2. Se aplica la transformación afín descrita en la Ecuación 3.20. Para ello, b_i representa el bit i del byte, mientras que c_i es el bit i del byte c, que tiene un valor de $0x63$ expresado en hexadecimal.

$$b_i = b_i \oplus b_{(i+4)\text{mod}8} \oplus b_{(i+5)\text{mod}8} \oplus b_{(i+6)\text{mod}8} \oplus b_{(i+7)\text{mod}8} + c_i \tag{3.20}$$

Tabla 3.2: Tabla de sustitución S-box

		Y															
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**a**	**b**	**c**	**d**	**e**	**f**
X	**0**	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Por lo tanto, la transformación *SubBytes* consiste en realizar una permutación de los elementos que componen el State por el valor correspondiente almacenado en la tabla S-box. Para ello, es necesario recorrer cada uno de los 16 bytes que componen el State y utilizar su contenido como coordenadas X e Y en la tabla de sustitución S-box. Así, el valor original de cada uno de los bytes que componen el State s será sustituido por el valor correspondiente en la tabla de sustitución, lo que dará como resultado un nuevo valor de State denominado s'. A fin de ilustrar el procedimiento descrito, y suponiendo que el primer elemento de un State s sea $s_{0,0}$, cuyo valor hexadecimal sea $0x0a$, se utilizarían las coordenadas $X = 0$ e $Y = a$ para obtener el nuevo valor $s'_{0,0}$. En este caso, el nuevo valor sería $s'_{0,0} = 0x67$.

La Ecuación 3.21 muestra como dado un un State s compuesto por elementos $s_{r,c}$, como resultado de aplicar la transformación *SubBytes* se obtiene un State s'

cuyos elementos toman un valor de acuerdo a la tabla S-box.

$$\begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} \rightarrow \text{SubBytes} \rightarrow \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix} \tag{3.21}$$

Tabla 3.3: Tabla de sustitución S-box inversa

		Y															
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**a**	**b**	**c**	**d**	**e**	**f**
X	**0**	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	a0	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

La transformación inversa de *SubBytes* es denominada *InvSubBytes*. Esta, también consiste en realizar una permutación de los elementos que componen el State por el valor correspondiente almacenado en la tabla de sustitución. Sin embargo, en este caso la tabla de sustitución utilizada es la Tabla 3.3. El contenido de esta tabla, la cual es denominada S-box inversa, es calculado mediante el inverso de la transformación afín descrita en la Ecuación 3.20. La transformación *InvSubBytes* usa el contenido de cada uno de los elementos que componen el State como coordenadas de la tabla S-box inversa para obtener el nuevo valor de cada uno de los elementos del State.

La Ecuación 3.22 muestra como dado un un State s compuesto por elementos $s_{r,c}$, como resultado de aplicar la transformación *InvSubBytes* se obtiene un State s' cuyos elementos toman un valor de acuerdo a la tabla S-box. El objetivo de esta transformación es revertir la permutación de los elementos que componen el State realizada mediante la transformación *SubBytes*.

$$\begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} \rightarrow \text{InvSubBytes} \rightarrow \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix} \tag{3.22}$$

3.3.4 Transformación de datos *ShiftRows*

La transformación de datos *ShiftRows* consiste en realizar desplazamientos cíclicos de los elementos en las últimas tres filas $0 < r \leq 3$ que componen el State. Para ello, en cada una de las filas en las que sea realiza el desplazamiento se utiliza un índice de desplazamiento distinto. La Ecuación 3.23 define como se lleva a cabo la transformación *ShiftRows*.

$$s'_{r,c} = s_{r,(c+ \text{ shift(r,Nb)) mod Nb}} \qquad \text{for } 0 < r < 4 \text{ and } 0 \leq c < Nb \tag{3.23}$$

De esta manera, el valor de $shift(r, Nb)$ depende del número de fila r y del número de bloques Nb, el cual tiene un valor fijo en el algoritmo AES de 4. Así, $shift(r, Nb)$ toma los valores 1, 2 y 3 cuando los valores de fila r toman los valores 1, 2 y 3 respectivamente. Lo que se consigue con esta operación es realizar un desplazamiento de los bytes de cada fila del State a posiciones inferiores en la fila (valores inferiores de c), mientras que los bytes en posiciones inferiores son desplazados a posiciones superiores de la fila (valores superiores de c).

$$s = \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} \rightarrow \text{ShiftRows} \rightarrow s' = \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,1} & s_{1,2} & s_{1,3} & s_{1,0} \\ s_{2,2} & s_{2,3} & s_{2,0} & s_{2,1} \\ s_{3,3} & s_{3,0} & s_{3,1} & s_{3,2} \end{bmatrix} \tag{3.24}$$

Además, la Ecuación 3.24 muestra el posicionamiento de los elementos del State s antes de realizar la transformación *ShiftRows* y su posicionamiento en el State s' después de realizar la transformación.

De forma equivalente, la transformación inversa de *ShiftRows*, la cual es denominada *InvShiftRows*, realiza permutaciones del orden de los bytes que componen las tres últimas filas del State aplicando un índice de desplazamiento distinto en función del número de fila. La Ecuación 3.25 define como se lleva a cabo la

transformación *InvShiftRows*. Al igual que en la transformación *ShiftRows*, el valor de $shift(r, Nb)$ depende del número de fila r y del número de bloques Nb, tomando los valores 1, 2 y 3 cuando los valores de fila r toman los valores 1, 2 y 3 respectivamente.

$$s'_{r,(c+\text{ shift(r,Nb))} \bmod \text{Nb}} = s_{r,c} \qquad \text{for } 0 < r < 4 \text{ and } 0 \leq c < Nb \tag{3.25}$$

La Ecuación 3.26 muestra el posicionamiento de los elementos del State s antes de realizar la transformación *InvShiftRows* y su posicionamiento en el State s' después de realizar la transformación. Como se puede apreciar tanto en la Ecuación 3.25 como en la Ecuación 3.26, el objetivo de esta transformación es revertir las permutaciones de datos realizadas como consecuencia de ejecutar la transformación *ShiftRows*.

$$s = \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} \rightarrow \text{InvShiftRows} \rightarrow s' = \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,3} & s_{1,0} & s_{1,1} & s_{1,2} \\ s_{2,2} & s_{2,3} & s_{2,0} & s_{2,1} \\ s_{3,1} & s_{3,2} & s_{3,3} & s_{3,0} \end{bmatrix} \tag{3.26}$$

3.3.5 Transformación de datos *MixColumns*

La transformación *MixColumns* realiza operaciones en cada una de las columnas del State. Para ello, cada una de las columnas es tratada como un polinomio de cuatro términos sobre $GF(2^8)$ al que se le aplica una multiplicación modular con el polinomio fijo $a(x)$ descrito a continuación

$$s'(x) = a(x) \otimes s(x) \quad \text{where} \quad a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\} \tag{3.27}$$

Como resultado de la multiplicación descrita en la Ecuación 3.27, cada uno de los 4 bytes, denominado $s_{r,c}$, que componen cada una de las columnas del State es reemplazado por los siguientes elementos

$$
\begin{aligned}
s'_{0,c} &= (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c} \\
s'_{1,c} &= s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c} \\
s'_{2,c} &= s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c}) \\
s'_{3,c} &= (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \otimes s_{2,c} \oplus (\{02\} \bullet s_{3,c})
\end{aligned}
\tag{3.28}
$$

Como se puede observar en la Ecuación 3.28, todas las operaciones que se deben llevar a cabo para realizar la transformación *MixColumns* consisten en aplicar a los bytes de cada columna del State operaciones XOR entre los elementos que componen la columna y multiplicaciones por las constantes $0x02$ y $0x03$.

$$
\begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} \rightarrow \text{MixColumns} \rightarrow \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix} \tag{3.29}
$$

Por otro lado, la transformación inversa de *MixColumns* es denominada *InvMixColumns*. Al igual que en el resto de transformaciones inversas, el objetivo de esta transformación es revertir las modificaciones del State que se han obtenido como producto de realizar la transformación *MixColumns*. Para ello, la transformación *InvMixColumns* realiza operaciones sobre cada una de las columnas del State. De esta forma, cada una de las columnas es tratada como un polinomio de cuatro términos sobre $GF(2^8)$ al que se le aplica una multiplicación modular con el polinomio fijo $a^{-1}(x)$ descrito a continuación

$$
s'(x) = a^{-1}(x) \otimes s(x) \quad \text{where} \quad a^{-1}(x) = \{0b\}x^3 + \{0d\}x^2 + \{09\}x + \{0e\} \tag{3.30}
$$

Como resultado de la multiplicación descrita en la Ecuación 3.30, cada uno de los 4 bytes, denominado $s_{r,c}$, que componen cada una de las columnas del State es reemplazado por los siguientes elementos

$$
\begin{aligned}
s'_{0,c} &= (\{0e\} \bullet s_{0,c}) \oplus (\{0b\} \bullet s_{1,c}) \oplus (\{0d\} \bullet s_{2,c}) \oplus (\{09\} \bullet s_{3,c}) \\
s'_{1,c} &= (\{09\} \bullet s_{0,c}) \oplus (\{0e\} \bullet s_{1,c}) \oplus (\{0b\} \bullet s_{2,c}) \oplus (\{0d\} \bullet s_{3,c}) \\
s'_{2,c} &= (\{0d\} \bullet s_{0,c}) \oplus (\{09\} \bullet s_{1,c}) \oplus (\{0e\} \bullet s_{2,c}) \oplus (\{0b\} \bullet s_{3,c}) \\
s'_{3,c} &= (\{0b\} \bullet s_{0,c}) \oplus (\{0d\} \bullet s_{1,c}) \oplus (\{09\} \bullet s_{2,c}) \oplus (\{0e\} \bullet s_{3,c})
\end{aligned}
\tag{3.31}
$$

En este caso, tal y como se muestra en la Ecuación 3.31, las transformaciones que se aplican a cada uno de los cuatro bytes que componen cada una de las columnas del State, consiste en realizar operaciones XOR entre los elementos de la columna y multiplicaciones por las constantes $0x0b$, $0x0d$, $0x09$ $0x0e$.

$$\begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} \rightarrow \text{MixColumns} \rightarrow \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix} \tag{3.32}$$

3.3.6 Cifrado y cifrado inverso

Tal y como se ha descrito anteriormente, el proceso de cifrado AES consiste en una serie de rondas en las cuales se aplican las transformaciones AES. Inicialmente, el bloque de datos de entrada se copia al State, pasando posteriormente a realizar la transformación *AddRoundKey* entre el contenido del State y la primera subclave generada en el proceso de expansión de clave AES.

Posteriormente, durante un número de rondas AES, el cual depende del tamaño de clave AES utilizado, se realizarán una serie de transformaciones del contenido del State. Cada una de estas rondas se compone de las mismas transformaciones. Sin embargo, la ronda final AES, denominada $(Nr - 1)$ difiere de las rondas anteriores. En esta ronda, además de hacer una serie de transformaciones distintas a las de las rondas previas, se proporciona el resultado final del State como el resultado de cifrado AES. La Figura 3.2 muestra el diagrama de flujo asociado al proceso de cifrado AES, es decir, los datos encriptados.

Al igual que ocurre con las transformaciones AES que conforman cada una de las rondas AES, el cifrado inverso tiene como objetivo principal recuperar los datos en claro que han sido cifrados a través de una serie de transformaciones inversas. Por lo tanto, el cifrado inverso sigue la misma estructura que el proceso de cifrado pero utilizando las transformaciones inversas AES.

Inicialmente, el bloque de datos de entrada se copia al State, pasando posteriormente a realizar la transformación *AddRoundKey* entre el contenido del State y la primera subclave generada en el proceso de expansión de clave AES. Posteriormente, durante un número de rondas AES, el cual depende del tamaño de clave AES utilizado, se realizarán una serie de transformaciones del contenido del State. Cada una de estas rondas se compone de las mismas transformaciones. Sin embargo, la ronda final AES, denominada $(Nr - 1)$ difiere de las rondas anteriores. En esta ronda, además de hacer una serie de transformaciones distintas a las

de las rondas previas, se proporciona el resultado final del State como el resultado de cifrado inverso AES, es decir, los datos en claro. La Figura 3.3 muestra el diagrama de flujo asociado al proceso de cifrado inverso AES.

3.4 Fundamentos matemáticos del algoritmo AES-GCM

En el algoritmo AES-GCM se definen dos operaciones fundamentales: cifrado autenticado y cifrado inverso autenticado [60]. Ambas operaciones serán descritas en detalle en la Sección 3.5. Sin embargo, antes de ello es necesario describir los elementos básicos y fundamentos matemáticos en los que se basa este algoritmo criptográfico.

En primer lugar, cabe destacar que el algoritmo AES-GCM está dividido en dos partes claramente diferenciadas. Por un lado está el uso del algoritmo AES en modo contador que es utilizado para proporcionar confidencialidad de los datos. Los fundamentos matemáticos y funcionamiento de dicho algoritmo han sido descritos en las Secciones 3.2 y 3.3 respectivamente. Por otro lado, se encuentra el algoritmo GCM que es utilizado para proporcionar autenticidad e integridad de los datos.

El algoritmo AES-GCM consta de las siguientes entradas:

- Clave criptográfica, denominada K en el algoritmo. Esta puede tomar una longitud de 128, 192 y 256 bits. Las posibles longitudes de claves admitidas vienen determinadas por el algoritmo de cifrado de bloque de datos utilizado, en este caso se trata del algoritmo AES. Esta entrada es utilizada tanto en el proceso de cifrado como en el proceso de cifrado inverso.
- IV, denominado IV en el algoritmo. La longitud de datos del vector de inicialización puede ser de entre 2^{64} y 1 bit. El IV tiene como requisito fundamental tomar valores distintos cada vez que se encripta un bloque de datos utilizando una misma clave criptográfica K. En los casos en los que la eficiencia del algoritmo sea un aspecto relevante, tal y como lo es en SAS, se recomienda en uso de un IV de 96 bits. Esta entrada es utilizada tanto en el proceso de cifrado como en el proceso de cifrado inverso.
- Additional Authenticated Data (AAD), denominado A en el algoritmo. Se trata de una entrada opcional que solamente es utilizada cuando se necesita proporcionar autenticidad e integridad pero no confidencialidad a los datos. Esta entrada solamente es utilizada en el proceso de cifrado.

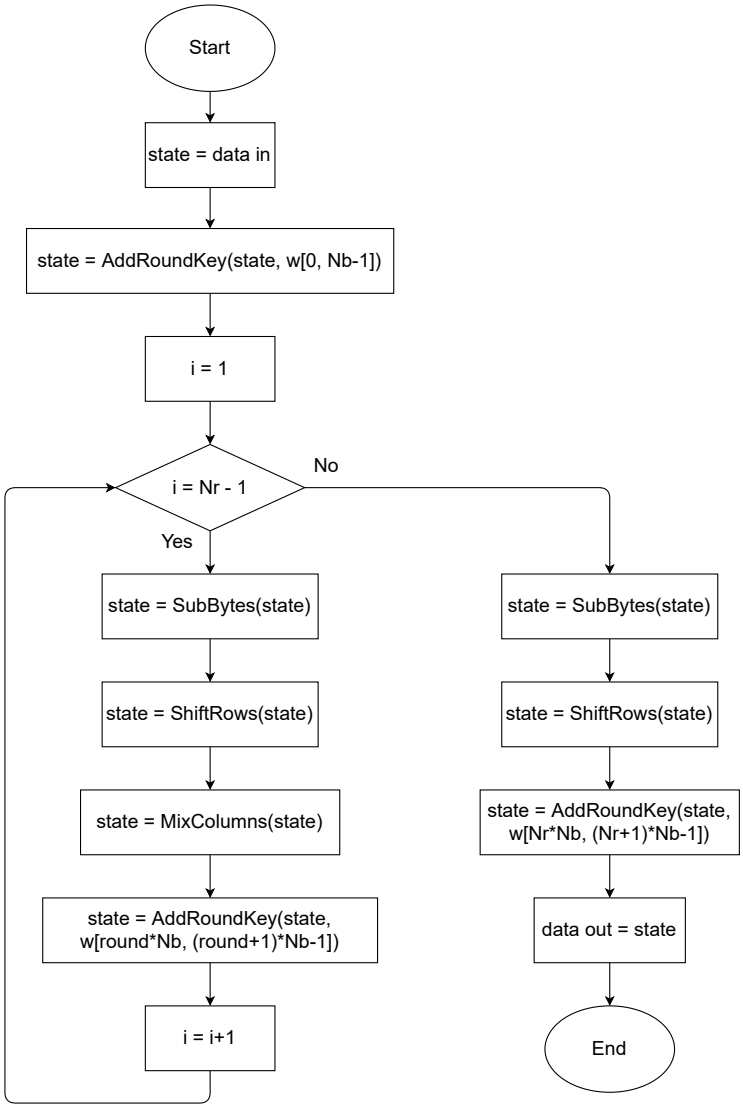

Figura 3.2: Proceso de cifrado de bloque de datos en el algoritmo AES

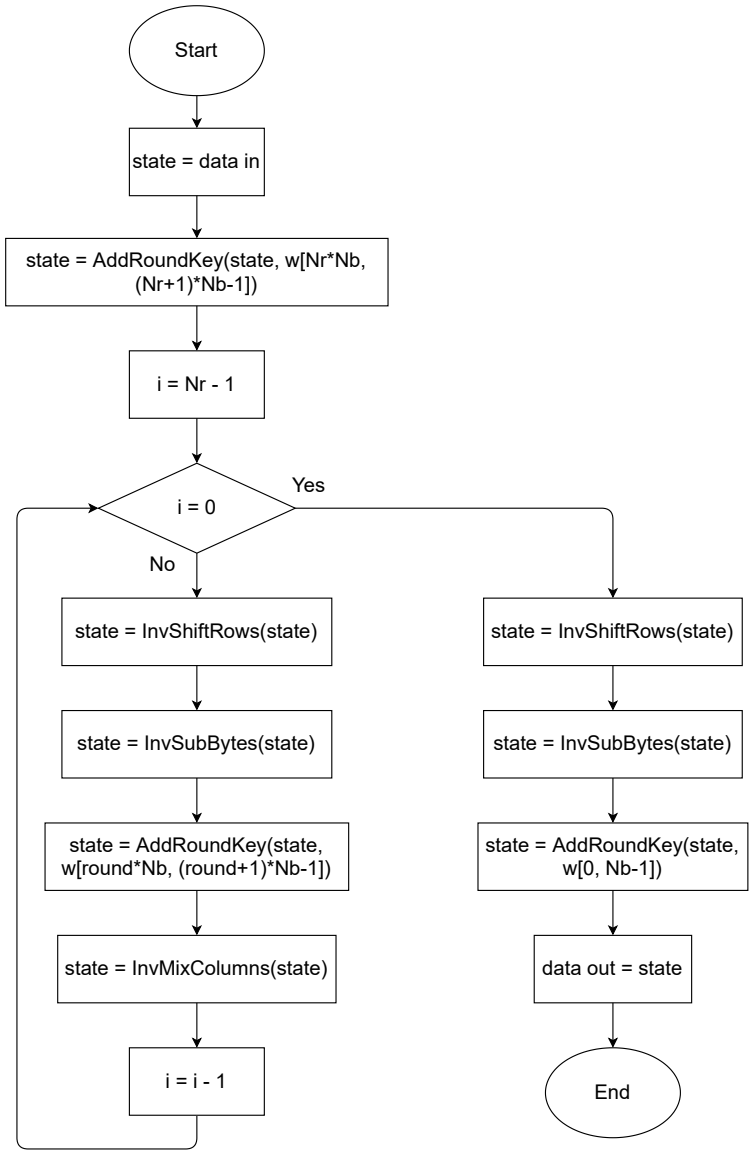

Figura 3.3: Proceso de cifrado inverso de bloque de datos en el algoritmo AES

Por otro lado, el algoritmo AES-GCM consta de las siguientes salidas:

- Firma digital, denominada T en el algoritmo. Esta se compone de 128 bits de datos que representan el valor de la firma digital. La principal característica de la firma digital es que el cambio de 1 bit de alguna de las entradas utilizadas para su cálculo conlleva que al menos el 50 % de los bits que componen la firma digital cambien. Tanto cuando se utiliza el algoritmo en modo cifrado como cuando se utiliza en modo cifrado inverso, una de las salidas proporcionadas es T.

Finalmente, el algoritmo AES-GCM consta de las siguientes entradas/salidas:

- Datos encriptados, denominados C en el algoritmo. Cuando se utiliza el algoritmo en modo cifrado, uno de los dos resultados proporcionados es C. Se trata de los datos que son proporcionados como salida del algoritmo y están protegidos mediante autenticidad, integridad y confidencialidad. Adicionalmente, estos datos también pueden ser una entrada del algoritmo cuando este se utiliza en modo cifrado inverso. De esta forma, se utiliza C para obtener el texto en claro P y realizar el cálculo de la firma digital.
- Texto en claro, denominado P en el algoritmo. Cuando se utiliza el algoritmo en modo cifrado, se trata de una entrada opcional que solamente es utilizada cuando se necesita proporcionar confidencialidad a los datos. En el algoritmo AES-GCM todos los datos que son cifrados también son autenticados. Sin embargo, es posible que algunos datos solamente estén autenticados y no cifrados. Adicionalmente, cuando se utiliza el algoritmo en modo cifrado inverso, se trata de una salida opcional que solamente es utilizada cuando en los datos procesados existían datos cifrados.

En el algoritmo AES-GCM el texto en claro T consiste en una secuencia de n bloques de datos. El último de los bloques de datos tiene una longitud u, mientras que la longitud del resto de bloques de datos es 128 bits. Cada uno de los bloques de datos, así como la longitud de los mismos se encuentran representados en las siguientes ecuaciones

$$len(T) = (n-1) \cdot 128 + u \qquad \text{where } 1 \leq u \leq 128 \tag{3.33}$$

$$T = P_1, P_2, \ldots, P_{n-1}, P_n^* \qquad \begin{array}{l} \text{where } len(P_i) = 128 \text{ and } 1 \leq i < n \\ \text{and } 1 \leq len(P_n^*) \leq 128 \end{array} \tag{3.34}$$

De forma similar a las Ecuaciones 3.33 y 3.34, la representación de los datos de C y A sigue la misma estructura. Los datos encriptados C, se componen

de la secuencia de bloques de datos: $C_1, C_2, \ldots, C_{n-1}, C_n^*$. Igualmente, los datos solamente autenticados A, se componen de la secuencia de bloques de datos: $A_1, A_2, \ldots, A_{m-1}, A_m^*$.

El algoritmo AES-GCM se basa en dos modos de funcionamiento principales, también denominados funciones principales. El primero consiste en el cifrado de bloque, para el cual se utiliza el algoritmo AES en modo contador. El segundo se basa en la multiplicación de elementos pertenecientes al campo $GF(2^{128})$. El cifrado de los datos X con la clave K utilizando el algoritmo criptográfico AES es representado mediante: $E(K, X)$. En cambio, la multiplicación de dos elementos X e Y pertenecientes al campo $GF(2^{128})$ se representa mediante: $X \cdot Y$. Adicionalmente, la suma de los elementos X e Y se puede representar como: $X \oplus Y$. Además, existen una serie de expresiones y funciones auxiliares que forman parte de la definición del algoritmo AES-GCM y que son recogidas a continuación:

- La función $len()$ devuelve como resultado la longitud de su argumento.
- La expresión 0^l describe un bloque de datos de longitud l bits cuyo valor sea 0.
- La expresión $A||B$ describe la concatenación de dos bloques de datos A y B.
- La función $MSB_t(S)$ devuelve como resultados los t bits más significativos del argumento S.
- La expresión $\{\}$ define un bloque de datos nulo, es decir, un bloque de datos cuya longitud sea 0.
- La función $GHASH(H, A, C)$ se utiliza para calcular la firma digital que sirve para autenticar los datos. Su definición de lleva a cabo en la Sección 3.5.

Una vez definidas las entradas y salidas del algoritmo AES-GCM, así como las estructuras de datos con las que este algoritmo trabaja, es posible introducir los fundamentos matemáticos que definen las operaciones en las que se basa el algoritmo.

En el algoritmo GCM se definen elementos pertenecientes al campo finito $GF(2)$. Un campo finito viene determinado por la definición de las operaciones suma y multiplicación. Para ello, estas operaciones deben seguir las propiedades algebraicas que son esperables de la suma y multiplicación de dos elementos, como son la conmutabilidad, distributividad, y asociatividad. Así, utilizando una representación polinómica, la multiplicación de los elementos X e Y consiste en multiplicar ambos elementos de 128 bits y dividir el resultado de 256 bits entre

el polinomio de campo. El resto obtenido de la anterior división representa el resultado de 128 bits. Por lo tanto, el polinomio de campo tiene una importancia relevante, puesto que en función de su valor, define la representación del campo. En el algoritmo criptográfico GCM se utiliza el siguiente polinomio de campo

$$f = 1 + \alpha + \alpha^2 + \alpha^7 + \alpha^{128} \tag{3.35}$$

Por otro lado, la suma de dos elementos X e Y consiste en realizar la suma de ambos polinomios. Puesto que cada coeficiente se suma de forma independiente y estos pertenecen al campo $GF(2)$, esta operación se puede representar con la operación lógica XOR tal y como ocurría en el caso del algoritmo AES. Además, en este caso no es necesario realizar ninguna operación de reducción. De forma similar, la resta de dos elementos X e Y es equivalente a la operación de suma, es decir, se puede llevar a cabo mediante la operación lógica XOR de ambos elementos. Esto es así debido a que se trata de una de las propiedades del campo $GF(2)$.

La Ecuación 3.36 muestra como se lleva a cabo la multiplicación de dos polinomios. A fin de describir el proceso de multiplicación, se utiliza el elemento de campo α que tiene la siguiente propiedad: $\alpha_1 = 1$. Como se puede apreciar en dicha ecuación, esta consiste en realizar un desplazamiento de los índices de los mismos. En el caso de que $x_{127} = 0$, se puede determinar que el polinomio resultante de la multiplicación es un polinomio de grado 127. En caso contrario, si el polinomio resultante es de grado 128 o superior, es necesario dividir el polinomio resultante entre el polinomio de campo f definido en la Ecuación 3.35.

$$\alpha \cdot (x_0 + x_1\alpha^1 + \ldots + x_{127}\alpha^{127}) = x_0\alpha + x_1\alpha^2 + \ldots + x_{127}\alpha^{128} \tag{3.36}$$

En el caso de la Ecuación 3.36, puesto que el resultado es un polinomio de grado 128, es necesario realizar la división entre el polinomio de campo f a fin de encontrar el cociente q y el resto r de tal forma que se cumpla lo siguiente

$$\alpha^{128} + a = q \cdot f + r \quad \text{where} \quad a = a_0 + a_1\alpha + \ldots + a_{127}\alpha^{127} \tag{3.37}$$

Hay que tener en cuenta que en la Ecuación 3.41 el resto tiene grado 127. Así es posible resolver dicha ecuación para el caso en el que el cociente toma el valor $q = 1$, siendo por lo tanto el valor del resto:

$$r = \alpha^{128} + a - f = a + 1 + \alpha + \alpha^2 + \alpha^7 \tag{3.38}$$

Puesto que se trata de una suma sobre el campo $GF(2)$, el elemento de mayor índice de f se cancela. Por lo tanto, teniendo en cuenta que la suma y la resta son equivalentes en el campo $GF(2)$, la Ecuación 3.38 puede completarse realizando la suma entre los elementos de f y a.

Para poder completar la operación $Y = X \cdot \alpha$, es necesario combinar los pasos descritos en las Ecuaciones 3.36 y 3.38, consistentes en realizar el desplazamiento de los coeficientes y la suma de los elementos de f en el caso de que X tenga un grado superior a 127. Esta operación se describe en la Figura 3.4

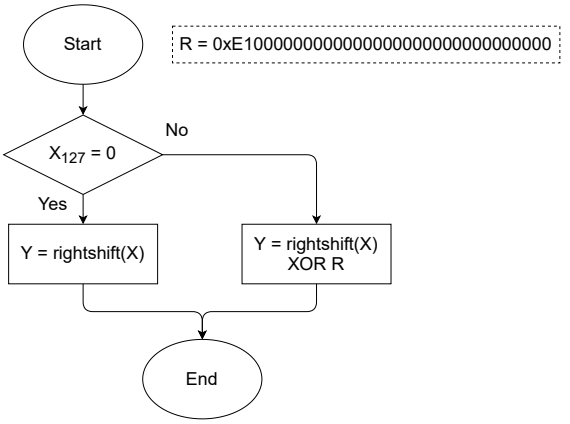

Figura 3.4: Multiplicación entre un elemento de campo $GF(2)$ y el elemento α

Para poder realizar la multiplicación de dos elementos de campo arbitrarios X e Y, es necesario poder expresar Y en términos del elemento α, de esta forma, es posible realizar la multiplicación utilizando la operación descrita en la Figura 3.4. La Figura 3.5 muestra como se lleva a cabo la operación de multiplicar dos elementos de campo arbitrarios X e Y en base a expresar el elemento Y en términos del elemento α.

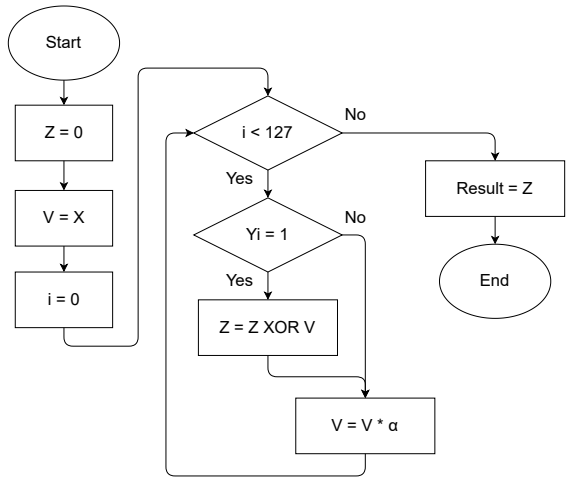

Figura 3.5: Multiplicación entre dos elementos arbitrarios de campo $GF(2)$

3.5 Descripción del algoritmo AES-GCM

La descripción del algoritmo criptográfico AES-GCM se recoge en tres operaciones claramente diferenciadas: cifrado y autenticación, cifrado inverso y autenticación y la función GHASH basada en multiplicaciones en el campo $GF(2^{128})$. En esta sección se realizará un análisis detallado de dichas operaciones.

3.5.1 Cifrado y autenticación

El cifrado autenticado de los datos en el algoritmo AES-GCM se puede especificar utilizando la Ecuación 3.39. En ella se describe como se generan cada uno de los elementos internos de los que se compone el algoritmo.

$$\begin{aligned} &H = E(K, 0^{128}) \\ &Y_0 = IV||0^{31}1 \\ &Y_i = incr(Y_{i-1}) \qquad \text{for } i = 1, \ldots, n-1 \\ &C_i = P_i \oplus E(K, Y_i) \qquad \text{for } i = 1, \ldots, n-1 \\ &C_n^* = P_n^* \oplus MSB_u(E(K, Y_n)) \\ &T = MSB_t(GHASH(H, A, C) \oplus E(K, Y_0)) \end{aligned} \tag{3.39}$$

El elemento H se obtiene como resultado de encriptar un bloque de datos de 128 bits de valor 0 con la clave K. Dicho valor de H es utilizado para el cálculo de la firma digital. El elemento Y se utiliza como entrada de la función $E()$, generando como resultado los datos encriptados mediante AES y utilizando la clave K. A su vez, el elemento Y es el resultado de concatenar el IV con un contador de 32 bits que se incrementa con cada bloque de datos procesado y cuyo valor inicial es igual a 1. De esta forma, se puede determinar como en el algoritmo AES-GCM los datos que se cifran utilizando el algoritmo AES no son los bloques de datos que pertenecen al texto en claro a encriptar, si no que lo que se cifra es el vector resultante de concatenar el IV y un contador. Es por esto que se denomina que en AES-GCM se utiliza el algoritmo AES en modo contador.

El resultado de $E(K, Y_i)$ se suma, mediante el uso de la operación lógica XOR, al texto en claro que debe ser encriptado, generándose así el elemento C_i. Finalmente, para generar el último bloque de datos encriptado C_n^*, se toma del resultado de $E(K, Y_n)$ el mismo número de bits que P_n^* para así poder realizar la suma de ambos vectores. El cálculo de la firma digital T se hace mediante la operación lógica XOR entre el resultado de encriptar el valor inicial del contador con el resultado de la función $GHASH()$. Puesto que la longitud t de la firma digital es 128 bits, en este caso se puede obviar el uso de la función $MSB_t(S)$.

3.5.2 Cifrado inverso y autenticación

El cifrado inverso autenticado de los datos en el algoritmo AES-GCM se puede especificar utilizando la Ecuación 3.40. En ella se describe como se generan cada uno de los elementos internos de los que se compone el algoritmo.

$$\begin{aligned} H &= E(K, 0^{128}) \\ Y_0 &= IV||0^{31}1 \\ Y_i &= incr(Y_{i-1}) \qquad \text{for } i = 1, \ldots, n-1 \\ P_i &= C_i \oplus E(K, Y_i) \qquad \text{for } i = 1, \ldots, n-1 \\ P_n^* &= C_n^* \oplus MSB_u(E(K, Y_n)) \\ T' &= MSB_t(GHASH(H, A, C) \oplus E(K, Y_0)) \end{aligned} \tag{3.40}$$

Realizando una comparación entre las Ecuaciones 3.39 y 3.40 se puede observar como ambas son equivalentes teniendo en cuenta ciertas consideraciones. En primer lugar, al realizar el cálculo del cifrado inverso, el algoritmo pasa a tener como entradas los datos encriptados en vez del texto en claro, siendo este una salida del algoritmo. Así, se puede observar como el algoritmo AES-GCM desencripta

los datos encriptando el vector resultante de concatenar el vector de inicialización con el contador de bloque de datos. El resultado de esta operación se utiliza para realizar la suma con los datos encriptados mediante la operación lógica XOR. De esta forma, como resultado de dicha operación se obtienen el texto en claro.

Cabe destacar como al igual que en el caso del cifrado AES-GCM, en el cifrado inverso AES-GCM se hace mediante el uso de la función $E()$. Es decir, en el algoritmo AES-GCM siempre se utiliza el algoritmo AES para cifrar datos y nunca se utiliza el crifado inverso AES. Esta característica reduce el uso de recursos necesarios para implementar el algoritmo AES-GCM ya que no es necesario contar con una implementación completa del algoritmo AES.

3.5.3 Función GHASH

La multiplicación resulta un elemento fundamental de la función *GHASH* utilizada para realizar el cálculo de la firma digital. Tal y como se puede apreciar en la Figura 3.6, en el algoritmo GCM una multiplicación de dos elementos pertenecientes al campo $GF(2^{128})$ se lleva a cabo mediante comparaciones, desplazamientos de bits y operaciones lógicas XOR. Esta operación es el resultante de combinar las operaciones descritas en las Figuras 3.4 y 3.5.

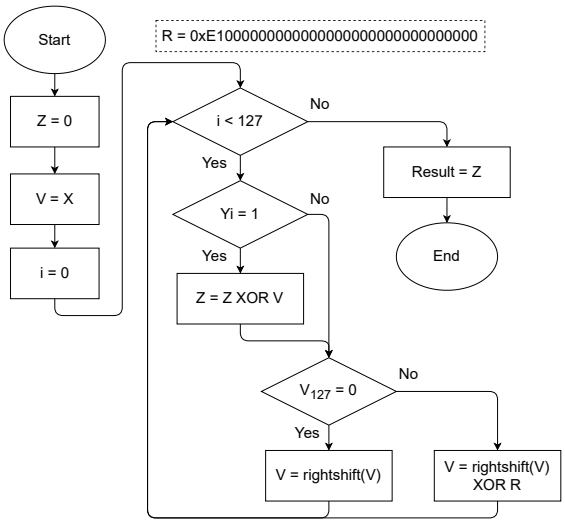

Figura 3.6: Multiplicación en $GF(2^{128})$ en el algoritmo GCM

Para que la operación de multiplicación descrita en la Figura 3.6 sea válida, se debe cumplir que

$$Z = X \cdot Y \quad \text{where} \quad X, Y \text{ and } Z \in GF(2^{128}) \tag{3.41}$$

De esta forma, la Ecuación 3.42 define la función *GHASH*. Para ello, es necesario tener en consideración que $GHASH(H, A, C) = X_{m+n+1}$, siendo m el número de bloques de datos solamente autenticados y n el número de bloques de datos encriptados C. Adicionalmente, los elementos u y v representan el tamaño del último bloque de datos solamente autenticados y el tamaño del último de bloque de datos encriptado respectivamente.

$$\begin{array}{ll}
0 & \text{for } i = 0 \\
(X_{i-1} \oplus A_i) \cdot H & \text{for } i = 1, \ldots, m-1 \\
(X_{m-1} \oplus (A_m^*) || 0^{128-v}) \cdot H & \text{for } i = m \\
(X_{i-1} \oplus C_i) \cdot H & \text{for } i = m+1, \ldots, m+n-1 \\
(X_{m+n-1} \oplus (C_m^*) || 0^{128-u}) \cdot H & \text{for } i = m+n \\
(X_{m+n} \oplus (len(A) || len(C)) \cdot H & \text{for } i = m+n+1
\end{array} \tag{3.42}$$

Por lo tanto, se puede resumir que la función *GHASH* consiste en realizar de forma iterativa la suma del anterior firma digital calculada con el bloque de datos solamente autenticados o el bloque de datos encriptados según corresponda. El resultado de estas operaciones es multiplicado por el elemento H utilizando la multiplicación en $GF(2^{128})$. Finalmente, una vez procesados todos los bloques de datos, se realiza la suma de la firma digital con la concatenación de la longitud de los datos solamente autenticados y la longitud de los datos encriptados. El resultado obtenido de esta operación se multiplica por el elemento H dando como resultado la salida de la función *GHASH*.

Capítulo 4

Arquitectura SoPC para proteger mensajes GOOSE y SV en SAS

4.1 Introducción

Después de realizar un análisis del estado del arte en materia de ciberseguridad para SAS en el Capítulo 2, se pudo determinar como la protección de las comunicaciones IEC 61850 con estrictos requisitos temporales sigue suponiendo un reto tecnológico por resolver. Atendiendo a los resultados obtenidos por la comunidad investigadora, el IEC identificó la necesidad de presentar una revisión del estándar IEC 62351-6:2020 que hiciera viable la protección de los mensajes GOOSE y SV de acuerdo a los requisitos temporales establecidos en el estándar IEC 61850. Gracias a la nueva revisión del estándar IEC 62351-6 y el uso de criptografía de clave simétrica mediante el algoritmo AES-GCM analizado en el Capítulo 3, se abre la puerta a que la protección de mensajes GOOSE y SV sea viable.

Se han encontrado propuestas en la literatura que permiten proporcionar integridad y autenticidad a los mensajes IEC 61850 con requisitos de tiempo real. Sin embargo, no se ha encontrado ninguna propuesta que se capaz de proporcionar de forma simultánea integridad, autenticidad y confidencialidad a los mensajes GOOSE y SV de acuerdo al estándar IEC 62351-6:2020 y sin comprometer los estrictos requisitos temporales establecidos por el estándar IEC 61850 para este

tipo de mensajes.

Consecuentemente, el objetivo principal de esta tesis es proponer una arquitectura hardware para FPGA y SoPC que permita encriptar y autenticar los mensajes IEC 61850 GOOSE y SV con requisitos de tiempo real sin comprometer el tiempo máximo de 3 ms definido en el estándar para la generación, transmisión, recepción y procesamiento de este tipo de mensajes.

La Tabla 4.1 muestra los requisitos que la arquitectura presentada debe cumplir para garantizar su viabilidad. A si mismo, la tabla presenta una descripción de cómo la arquitectura propuesta hace frente a los requisitos establecidos.

Tabla 4.1: Requisitos de la arquitectura SoPC IEC 62351-6 para SAS

Número	**Descripción**	**Propuesta**
R1	Baja latencia: el proceso de autenticación y encriptación de los mensajes no debe comprometer los estrictos requisitos temporales establecidos en IEC 61850.	La arquitectura hace uso de la paralelización de los distintos procesos que componen la protección de los mensajes GOOSE y SV, dando lugar a una baja latencia.
R2	Retardo constante: el tiempo empleado en encriptar y autenticar un mensaje de un tamaño dado debe ser constante a fin de que el retardo introducido por el sistema sea predecible.	La arquitectura hardware propuesta, basándose en un diseño paralelizado y segmentado permite que el retardo introducido sea constante.
R3	Uso de recursos eficiente: la capacidad de procesamiento de los dispositivos SASs es limitada.	La arquitectura propuesta ha sido diseñada haciendo uso de técnicas avanzadas de paralelización y diseño segmentado que permiten conseguir una alta eficiencia.
R4	Rendimiento: los dispositivos SAS pueden soportar tasas de transferencia de datos de hasta 1 Gbps.	La arquitectura propuesta tiene una capacidad de procesamiento de hasta 10 Gbps y cuenta con parámetros configurables que permiten adaptar el rendimiento a cada caso de uso.
R5	Independencia de los datos a procesar: dependiendo del caso de uso concreto y de la arquitectura de la subestación eléctrica, los mensajes GOOSE y SV pueden tener un tamaño variable.	La arquitectura propuesta es independiente del tamaño de los datos a procesar y su rendimiento no está condicionado a este hecho.
R6	Independencia de las claves a procesar: las claves criptográficas utilizadas en el estándar IEC 62351-6 cambian de forma periódica.	La arquitectura propuesta es independiente de las claves criptográficas a utilizar. No es necesario conocer dichas claves o realizar un preprocesamiento de las mismas.

Continua en la siguiente página

Tabla 4.1 – *Continuación de la página anterior*

Número	Descripción	Propuesta
R7	Adaptabilidad para futuras actualizaciones: el estándar IEC 62351-6 está en constante evolución.	La arquitectura propuesta se basa en un diseño modular que permite realizar mejoras y modificaciones. Esto permite soportar tanto futuras revisiones del estándar IEC 62351-6 o nuevos protocolos, como modificaciones para proteger otro tipo de mensajes con requisitos de tiempo real
R8	Adaptabilidad para generar automáticamente arquitecturas hardware con distintas necesidades de rendimiento y uso de recursos lógicos	Se propone una arquitectura AES-GCM ad-hoc que permite el ajuste de rendimiento y uso de recursos lógicos. Esto supone una mejora industrial que permite generar soluciones para muchos tipos de equipos.

4.2 Definición de la arquitectura

La arquitectura presentada está dividida en seis secciones principales, las cuales están representadas en la Figura 4.1 con las letras A-F. Cada una de las secciones lleva a cabo una de las tareas que componen el proceso de cifrar y autenticar los mensajes GOOSE y SV. Una de las características principales de la arquitectura presentada es que sigue un diseño en espejo. Esto quiere decir que los módulos y procesos que se utilizan para proteger los mensajes GOOSE y SV son los inversos a los módulos que se utilizan para desproteger dichos mensajes. Esto permite que, al tener dos flujos de procesamiento de datos independientes para encriptar o desencriptar los mensajes, sea posible soportar el procesamiento de los datos a la velocidad máxima de la línea de 1 Gbps. A continuación se presentan las secciones que componen la arquitectura:

- Sección A: se denomina Port Interface a la sección que se encarga de realizar la funciones relativas a la capa 2 en el modelo OSI y proporcionar una capa de abstracción respecto a los distintos interfaces existentes.
- Sección B: se encarga de analizar los mensajes entrantes y extraer la información necesaria poder llevar a cabo la protección de los mensajes IEC 61850.
- Sección C: recoge los datos criptográficos de los mensajes a ser procesados y se los proporciona de forma correspondiente al motor criptográfico.
- Sección D: se trata de del motor criptográfico utilizado para la protección de los mensajes GOOSE y SV a través del cálculo de la firma digital y el

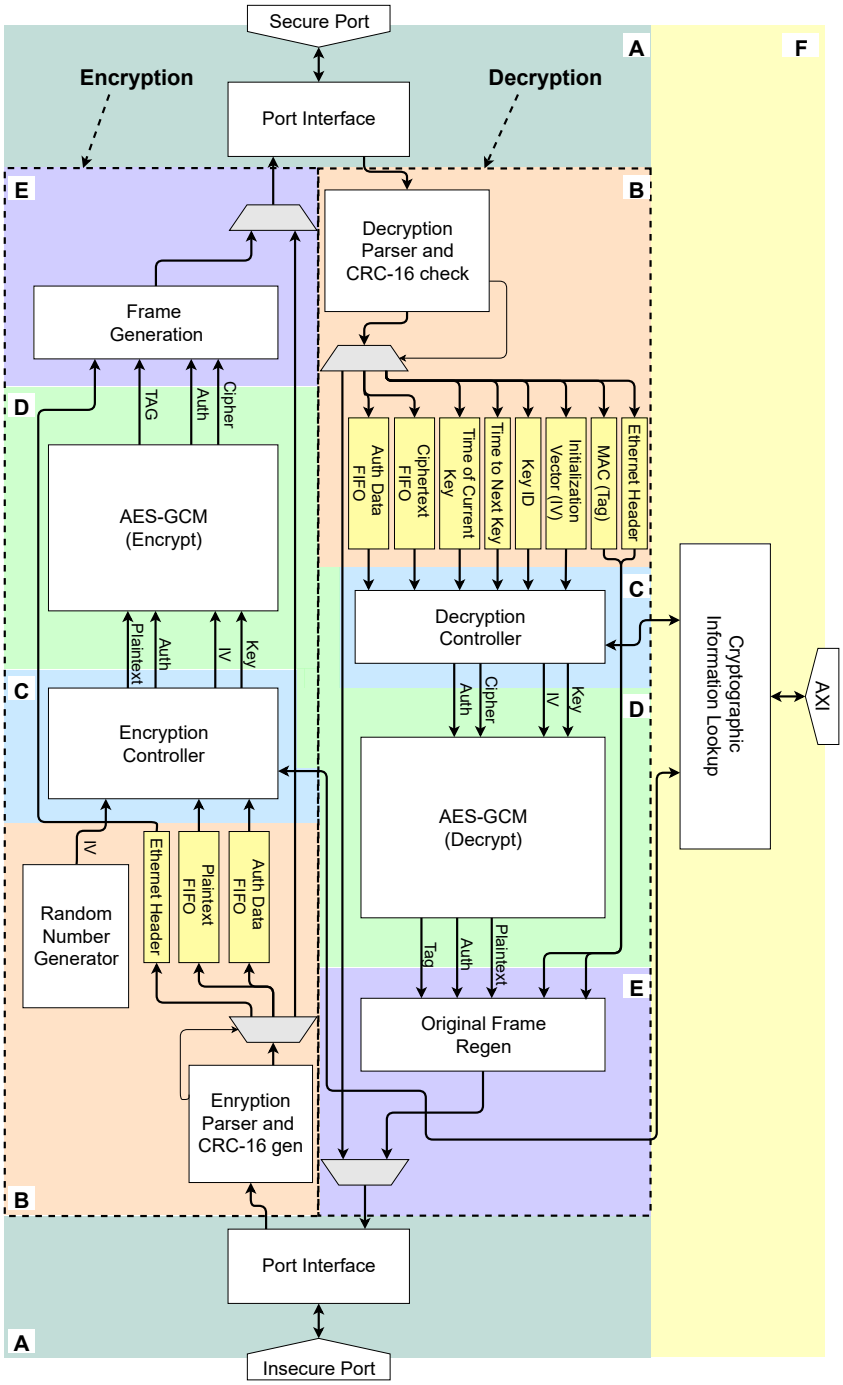

Figura 4.1: Arquitectura SoPC IEC 62351-6 para SAS

cifrado del mensaje.

- Sección E: obtiene la salida del motor criptográfico y ciertos valores extraídos del mensaje para generar tanto los mensajes protegidos IEC 62351-6, como los mensajes desprotegidos IEC 61850, según corresponda.
- Sección F: almacena la información criptográfica correspondiente a cada flujo de mensajes que deba ser procesado.

A parte de la sección F, la arquitectura propuesta consta de cinco secciones que se encuentran duplicadas. La primera de las instancias de cada una de las secciones es utilizada para el procesamiento de mensajes desprotegidos que necesitan ser protegidos. En cambio, la segunda instancia se utiliza para el procesamiento de mensajes IEC 61850 que deben ser protegidos. Todas las secciones de la arquitectura, así como los módulos que la componen han sido implementados utilizando el interfaz de comunicaciones punto a punto Advanced eXtensible Interface Stream (AXI-S). El interfaz AXI-S es parte de la Advanced Microcontroller Bus Architecture (AMBA) desarrollada por Advanced RISC Machine (ARM) y define un estándar de código abierto para la interconexión y gestión de bloques funcionales en un System-on-Chip (SoC) [65]. De esta forma, el uso de un interfaz de comunicaciones común y estándar facilita el proceso de modificar o reemplazar alguna de las secciones que componen la arquitectura si fuera necesario.

4.2.1 Port Interface

El módulo Port Interface, representado con la letra A en la Figura 4.2, implementa varios tipos de interfaces utilizados para las comunicaciones entre la capa MAC y la capa Physical Interface (PHY). El objetivo principal de este módulo es proporcionar una capa de abstracción entre los interfaces utilizados en la capa PHY y el interfaz AXI-S utilizado de forma interna en la arquitectura. De esta forma, la complejidad del resto de módulos que componen la arquitectura se ve reducida, puesto que estos son agnósticos al tipo de interfaz utilizado en el puerto físico. Entre los tipos de interfaces soportados destacan:

- Interfaz Media Independent Interface (MII), con soporte de velocidades de entre 10 Mbps y 100 Mbps.
- Interfaz Reduced Media Independent Interface (RMII), con soporte de velocidades de entre 10 Mbps y 100 Mbps.
- Interfaz Gigabit Medium Independent Interface (GMII), con soporte de velocidades de entre 10 Mbps y 1 Gbps.

- Interfaz Reduced Gigabit Media Independent Interface (RGMII), con soporte de velocidades de entre 10 Mbps y 1 Gbps.

Adicionalmente, este módulo también es utilizado para realizar el cálculo del código CRC tanto de los mensajes entrantes como de los salientes. El código CRC permite detectar errores de paridad de los mensajes recibidos. Cuando los mensajes son enviados a través del medio físico, estos pueden verse expuesto a perturbaciones que pueden hacer que el contenido del mensaje se corrompa. De esta forma, cuando se detecta un error de paridad de un mensaje recibido, este es descartado antes de ser procesado para así evitar el procesamiento innecesario de mensajes erróneos. De forma similar, este módulo también se encarga de generar el código CRC de los mensajes salientes a fin de que el receptor de los mensajes sea capaz de detectar posibles errores de paridad en su recepción.

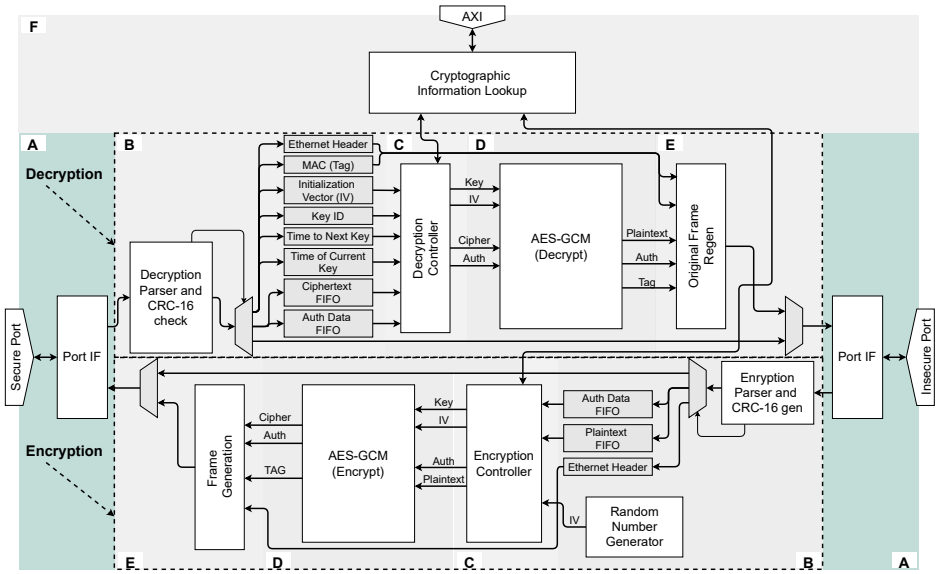

Figura 4.2: Módulo Port Interface de la arquitectura SoPC IEC 62351-6

Finalmente, este módulo se encarga de gestionar tanto el preámbulo como el espaciado entre mensajes definido en el estándar Ethernet IEEE 802.3 [66]. En transmisión, el Port Interface introduce tanto el espaciado entre mensajes como el preámbulo antes de realizar la transmisión de un nuevo mensaje. En cambio, en recepción, el Port Interface elimina el preámbulo de los mensajes para evitar que este sea interpretado como datos válidos. El preámbulo es una secuencia conocida que se envía precediendo a la transmisión de un nuevo mensaje para que

el receptor pueda sincronizarse con el transmisor. Por otro lado, la transmisión de mensajes de forma consecutiva debe respetar un espaciado temporal entre mensajes que varía en función de la velocidad del interfaz de comunicaciones.

4.2.2 Analizador de mensajes

El módulo analizador de mensajes, denominado con la letra B en la Figura 4.3, es utilizado para realizar la inspección de los mensajes recibidos y extraer aquellos campos de información que sean necesarios para su posterior procesamiento. En primer lugar, cada mensaje entrante es analizado para determinar si se trata de un mensaje IEC 61850 GOOSE o SV. Esta tarea se lleva a cabo a través de una avanzada máquina de estados que es capaz de decodificar estructuras ASN.1. La peculiaridad de los mensajes codificados mediante ASN.1, como lo son los mensajes IEC 61850 GOOSE y SV, es que la posición en la que se encuentra cada uno de los campos del mensaje puede ser variable dependiendo de la información que se incluya y el tamaño de la misma. Esto es debido a que este esquema de codificación se apoya en una estructura basada en una jerarquía de etiquetas para organizar el contenido de los mensajes y dividir los diferentes campos que los componen.

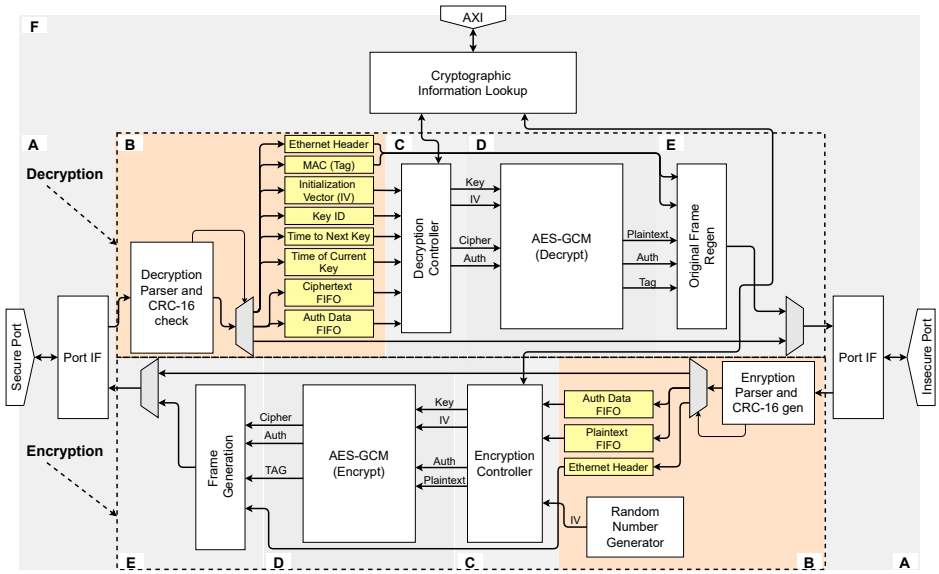

Figura 4.3: Módulo analizador de mensajes de la arquitectura SoPC IEC 62351-6

En el caso de que el mensaje sea de tipo GOOSE o SV, si estos se encuentran protegidos mediante el estándar IEC 62351-6, el analizador procederá a extraer la información de los mensajes que será utilizada para su posterior descifrado o verificación. Por el contrario, si el mensaje GOOSE o SV no se encuentra protegido, el analizador procederá a extraer la información de los mensajes que será utilizada para su posterior cifrado, cálculo de la firma digital y generación de la extensión de seguridad IEC 62351-6. Finalmente, en el caso de que el mensaje no sea de tipo GOOSE o SV, este será marcado por el analizador para ser transmitido sin ser procesado.

A parte de detectar la naturaleza de los mensajes, el analizador también es capaz de detectar errores en el formato o el contenido de los mismos. Así, en el caso de que los mensajes no cuenten con una estructura de datos ASN.1 correcta, el analizador procederá a descartar este tipo de mensajes.

En el caso de que el mensaje necesite ser protegido, el analizador divide la información extraída de cada mensaje en tres tipos claramente diferenciados de acuerdo al estándar IEC 62351-6:

- Datos que no necesitan tener un procesamiento criptográfico: cabecera Ethernet.
- Datos que deben ser autenticados pero no encriptados: el campo EPDU de los mensajes GOOSE y SV.
- Datos que deben ser autenticados y encriptados: el campo APDU de los mensajes GOOSE y SV.

La Figura 4.4 muestra los campos de los mensajes GOOSE y SV y el tipo de procesamiento criptográfico que se realiza de cada uno de ellos de acuerdo al estándar IEC 62351-6.

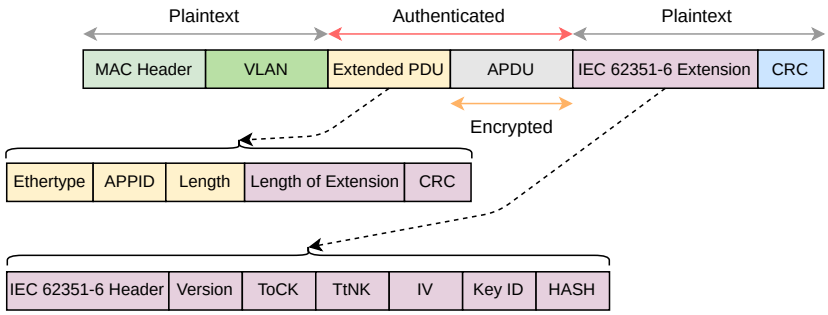

Figura 4.4: Campos de los mensajes GOOSE y SV protegidos mediante IEC 62351-6

Por otro lado, en el caso de que el mensaje recibido sea de tipo GOOSE o SV y ya se encuentre protegido de acuerdo al estándar IEC 62351-6, el analizador deberá extraer, a parte de los campos previamente mencionados, los campos que componen la extensión de seguridad IEC 62351-6 localizada en la parte final de los mensajes. Toda la información extraída y clasificada por el analizador es almacenada en estructuras de almacenamiento temporal de tipo First In First Out (FIFO) para posteriormente pasar dicha información al módulo que se encarga del control del motor criptográfico y al módulo que se encarga de regenerar los mensajes para su transmisión.

De forma adicional, en el caso del analizador empleado para el flujo de mensajes que deben ser encriptados, se incluye un módulo encargado de realizar la generación de números aleatorios. Los números aleatorios generados, requeridos por el motor criptográfico, se generan para cada nuevo mensaje. En concreto, se genera un número aleatorio de 64 bits haciendo uso de una estructura lógica denominada Ring Oscillator [67–69]. Este tipo de estructuras aprovechan la incertidumbre derivada de las imperfecciones en el proceso de fabricación de ciertos componentes físicos del SoPC para generar un resultado que no es predecible y variable dependiendo de factores físicos externos al dispositivo, como la temperatura o el ruido electromagnético.

La Figura 4.5 muestra la estructura lógica en la que se basa un Ring Oscillator para la generación de bits aleatorios. Realizando agrupaciones de estas estructuras es posible generar números aleatorios del tamaño necesario. Un Ring Oscillator consta de dos partes claramente diferenciadas. Por un lado se encuentran una serie de puertas lógicas conectadas en serie y cuyo resultado se retroalimenta al sistema. Puesto que este tipo de elementos son puramente combinacionales, cada uno de ellos introduce un retardo individual y no predecible. Por otro lado, se encuentra una lógica basada en dos Flip-Flops que se encarga de realizar el muestreo de los datos, siendo la salida del segundo Flip-Flop un bit de valor aleatorio.

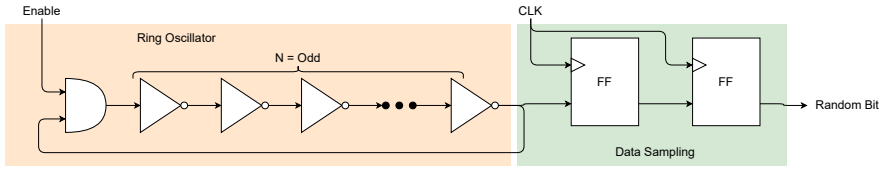

Figura 4.5: Estructura lógica para la generación de números aleatorios

4.2.3 Controlador del motor criptográfico

El módulo controlador del motor criptográfico, denominado con la letra C en la Figura 4.6, actúa como un intermediario entre el módulo analizador de los mensajes (sección B), el motor de búsqueda de información criptográfica (sección F) y el motor criptográfico (sección D). De esta forma, se obtienen todos los datos proporcionados por el módulo analizador de mensajes y se genera una petición de búsqueda de la información criptográfica asociada al conjunto de datos del mensaje.

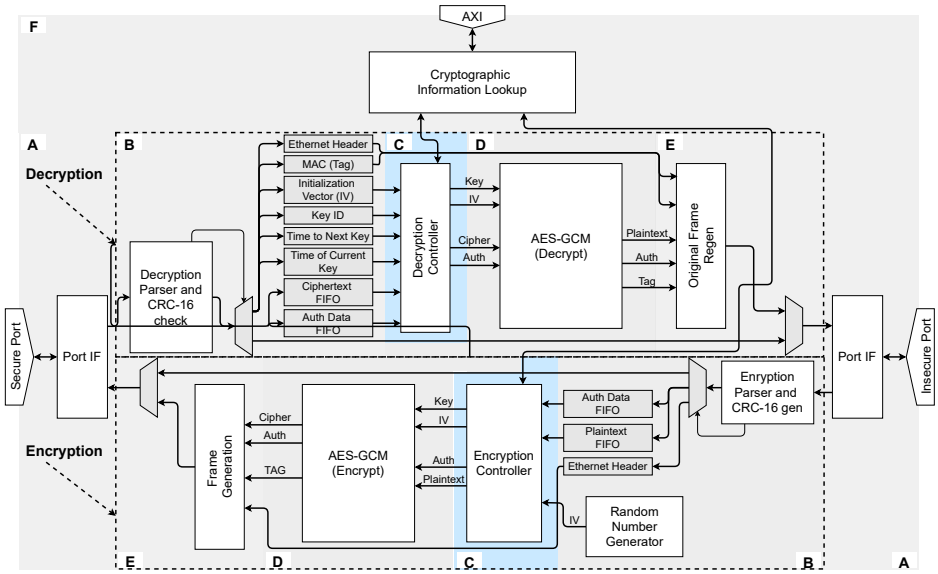

Figura 4.6: Módulo controlador del motor criptográfico de la arquitectura SoPC IEC 62351-6

En el caso de que el motor de búsqueda haya sido configurado con una clave criptográfica para el conjunto de datos del mensaje, esta será proporcionada al controlador del motor criptográfico. En caso contrario, el mensaje será propagado hasta el módulo generador de mensajes (sección E) sin realizar ningún tipo de procesamiento criptográfico del mensaje.

De entre toda la información criptográfica recibida del motor de búsqueda para cada mensaje, el controlador del motor criptográfico proporciona al motor criptográfico los siguientes datos:

- Clave criptográfica con la que encriptar y autenticar o desencriptar y au-

tenticar los datos. Tiene una longitud de 128 bits.

- Vector de inicialización requerido por el algoritmo criptográfico utilizado, AES-GCM. El vector de inicialización se compone de 96 bits divididos en dos partes. En el caso de que se encripte un mensaje, la primera parte está compuesta por 64 bits que se corresponden al número aleatorio generado en el analizador de mensajes. La segunda parte se compone de 32 bits que se corresponden al número de secuencia que representa el número de mensajes encriptados con la clave criptográfica utilizada. En el caso de que se trate del proceso de desencriptar el mensaje, el vector de inicialización ha sido obtenido íntegramente de la extensión de seguridad IEC 62351-6.
- Datos que solamente deben ser autenticados. El tamaño de estos datos es variable en función de cada mensaje GOOSE y SV.
- Datos que deben ser autenticados y encriptados. El tamaño de estos datos es variable en función de cada mensaje GOOSE y SV.

En el caso del proceso de encriptar los mensajes, adicionalmente el controlador criptográfico debe hacer llegar al módulo generador de mensajes parte de la información criptográfica recibida del motor de búsqueda de información criptográfica:

- Identificador de clave, denominado KeyID. Se trata de un identificador de clave inequívoco asignado a cada una de las claves durante el proceso de distribución de las mismas.
- Tiempo actual de clave, denominado ToCK. Se trata de la marca temporal que identifica el instante en el que la clave fue marcada como válida para su uso.
- Tiempo hasta la siguiente clave, denominado TtNK. Se trata del número de segundos restantes hasta que la clave actual sea sustituida por una nueva clave.
- Vector de inicialización, denominado IV. Se trata del IV utilizado para proteger el mensaje y que permite al receptor del mismo inicializar el motor criptográfico para poder realizar el proceso de desencriptar y autenticar el mensaje.

4.2.4 Motor criptográfico AES-GCM

De acuerdo al análisis detallado del algoritmo AES-GCM realizado en el Capítulo 3, se ha diseñado un motor criptográfico AES-GCM específicamente para su uso en SAS, denominado con la letra D en la Figura 4.7. Para ello, la arquitectura

del motor criptográfico se ha diseñado teniendo en cuenta los requisitos de rendimiento y latencia para el procesamiento de mensajes GOOSE y SV al mismo tiempo que se minimiza el uso de recursos FPGA o SoPC para garantizar su viabilidad en dispositivos SAS. Consecuentemente, el motor criptográfico AES-GCM propuesto tiene la capacidad de dimensionar y adaptar su arquitectura interna en función de diversos parámetros en tiempo de síntesis. Estos parámetros de configuración permiten seleccionar el balance adecuado entre rendimiento y uso de recursos lógicos para cada tipo de aplicación y dispositivos en SAS. La Figura 4.8 muestra la arquitectura interna del motor criptográfico desarrollado.

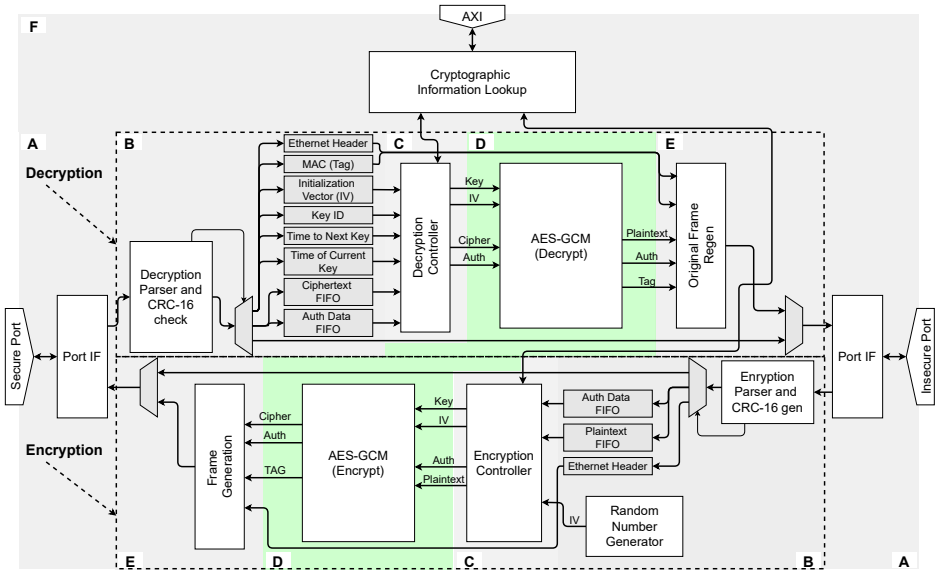

Figura 4.7: Motor criptográfico de la arquitectura SoPC IEC 62351-6

La arquitectura propuesta se basa en dos módulos principales y está dividida en cuatro fases lógicas. Por un lado se encuentra el motor AES que proporciona confidencialidad. Por otro lado, se encuentra el motor que realiza las multiplicaciones en GF para proporcionar autenticidad y confidencialidad. A pesar de que estos dos módulos aparecen múltiples veces en la Figura 4.8, solamente se encuentran instanciados una única vez en la arquitectura. Así, la Figura 4.8 representa las operaciones y las entradas y salidas de estos módulos durante cada una de las fases lógicas:

1. Inicialización. Se utiliza el motor AES para encriptar un bloque de datos de 128 bits de valor cero. El resultado de la operación es utilizado para el

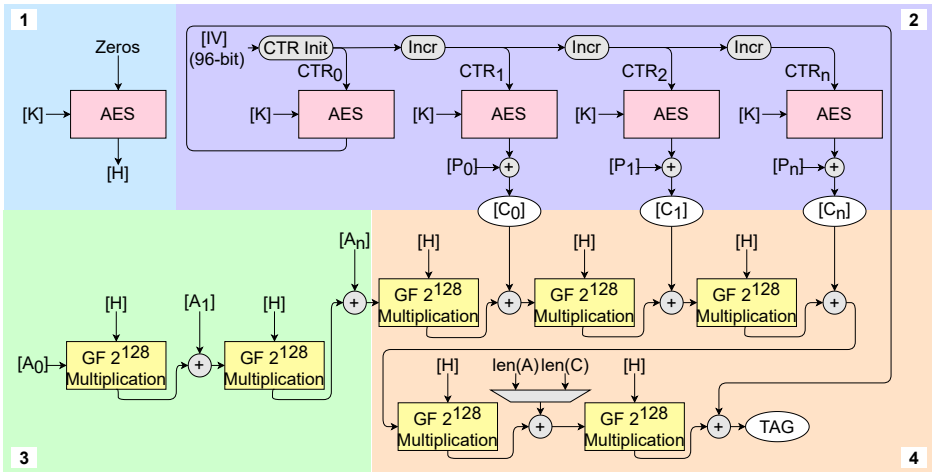

Figura 4.8: Arquitectura interna del motor criptográfico AES-GCM

posterior cálculo de la firma digital.

2. Generación de los datos encriptados. Se utiliza el motor AES para encriptar un bloque de datos de 128 bits compuesto por 96 bits del vector de inicialización y 32 bits correspondientes a un contador que se incrementa por cada bloque de datos a encriptar. El primero de los resultados de este proceso se almacena y reserva para su posterior uso en el proceso de autenticación. El resto de resultados obtenidos se suma a los datos a encriptar mediante la operación lógica XOR, obteniendo así los datos encriptados.

3. Autenticación. El motor de multiplicación en GF utiliza el resultado obtenido en el proceso de inicialiazación junto el primer bloque de datos que solamente deba ser autenticado. En el caso de que haya más datos que deban ser solamente autenticados, el resultado de la multiplicación es añadido al siguiente bloque de datos mediante la operación lógica XOR. Este resultado se utiliza como entrada del motor de multiplicación en GF.

4. Autenticación de los datos encriptados. El resultado del proceso de autenticación se suma mediante la operación lógica XOR al primer bloque de datos encriptado. Este resultado es utilizado como entrada del motor de multiplicación GF. Esta operación se repite de forma recurrente hasta haber procesado todos los datos encriptados. El resultado obtenido de la multiplicación en GF del último bloque de datos encriptados se añade mediante la operación lógica XOR a un bloque de datos de 128 bits compuesta

por 64 bits que representan la longitud de los datos solamente autenticados y 64 bits que representan la longitud de los datos encriptados y autenticados. El resultados de esta operación se usa como entrada del motor de multiplicación en GF para finalmente obtener como salida la firma digital.

La diferencia principal entre el proceso de encriptar y desencriptar los datos reside en el orden en el que los datos son procesados. En el proceso de encriptar los datos, estos se encriptan y después se autentican. Sin embargo, en el proceso de desencriptar los datos, estos primero se autentican y después se desencriptan. El motor AES-GCM presentado tiene la capacidad de procesar un bloque de datos de 128 bits cada ciclo de reloj, lo cual representa un flujo de datos de 16 Gbps si se emplea un reloj de 125 MHz de frecuencia.

La Figura 4.9 muestra la arquitectura interna del motor AES utilizado. Esta se basa en dos secciones claramente diferenciadas. La primera, denominada con el número 1 en la figura, se encarga de realizar la expansión de clave AES. Así, la arquitectura propuesta genera una de las etapas que conforman el proceso de expansión de clave en cada ciclo de reloj. Este hecho, a su vez, garantiza que las claves expandidas se encuentran perfectamente sincronizadas con cada una de las rondas AES en las que las claves generadas son utilizadas.

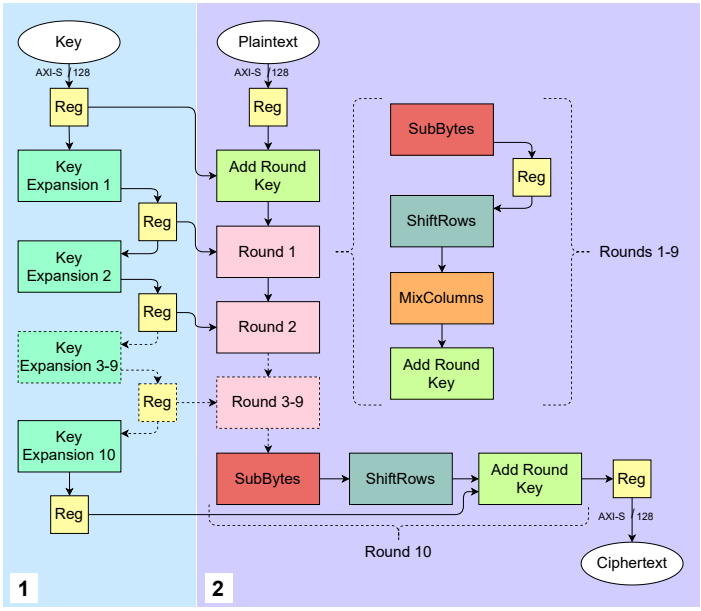

Figura 4.9: Arquitectura interna del motor criptográfico AES

Por otro lado, la segunda sección de la arquitectura del motor AES, denominada con el número 2 en la Figura 4.9, se encarga de llevar a cabo cada una de las rondas AES mediante el uso de las transformaciones de datos que componen el algoritmo criptográfico. La arquitectura presentada permite procesar un bloque de datos de 128 bits cada ciclo de reloj. Por lo tanto, por cada ciclo de reloj los datos avanzan a la siguiente ronda AES mientras que el resto de rondas procesan otro bloque de datos distinto. De esta forma, la latencia del motor AES es igual al número de rondas AES definidas en el algoritmo.

Las multiplicaciones en GF son operaciones complejas que requieren de un elevado número de recursos de la FPGA o el SoPC. El algoritmo Karatsuba-Ofman Algorithm (KOA), el cual ha sido analizado de forma extensa en la literatura [70–73], propone un método basado en una simplificación matemática que permite realizar las multiplicaciones en GF utilizando un número inferior de recursos lógicos si se compara con la multiplicación cuadrática clásica. En [74] se describen los fundamentos matemáticos para implementar una multiplicación en campos finitos mediante el uso del algoritmo KOA. La Ecuación 4.1 muestra la descripción del algoritmo KOA.

$$\begin{aligned} D_0 &= A_l \cdot B_l \\ D_{01} &= (A_h + A_l) \cdot (B_h + B_l) \\ D_1 &= A_h \cdot B_h \\ D &= A \cdot B = D_0 + x^{\frac{m}{2}} \cdot (D_0 + D_{01} + D_1) + x^m \cdot D_1 \end{aligned} \tag{4.1}$$

Tal y como se puede ver, el algoritmo define que la multiplicación de dos elementos A y B se puede dividir en tres operaciones D_0, D_{01} y D_1, siendo cada una de ellas de $m/2$ bits. Para ello, se determina que A_l y B_l representan los $m/2$ bits menos significativos de los elementos A y B respectivamente. Por otro lado A_h y B_h representan los $m/2$ bits más significativos de los elementos A y B. Así, haciendo un uso recursivo del algoritmo se consigue una reducción el número de bits de los elementos A y B hasta que estos tengan una anchura de 2 bits.

La Figura 4.10 muestra la arquitectura del motor de multiplicación en GF. Su arquitectura se encuentra dividida en las siguientes cuatro etapas:

1. División de los datos. Los datos de entrada A y B son divididos en A_h, A_l y B_h, B_l respectivamente de acuerdo al algoritmo KOA.
2. Multiplicación de los datos. Los datos divididos en la etapa anterior, junto con el resultado de realizar las operaciones $A_l \oplus A_h$ y $B_l \oplus B_h$ son usados como entradas del módulo de sub-multiplicación. Este módulo consiste en

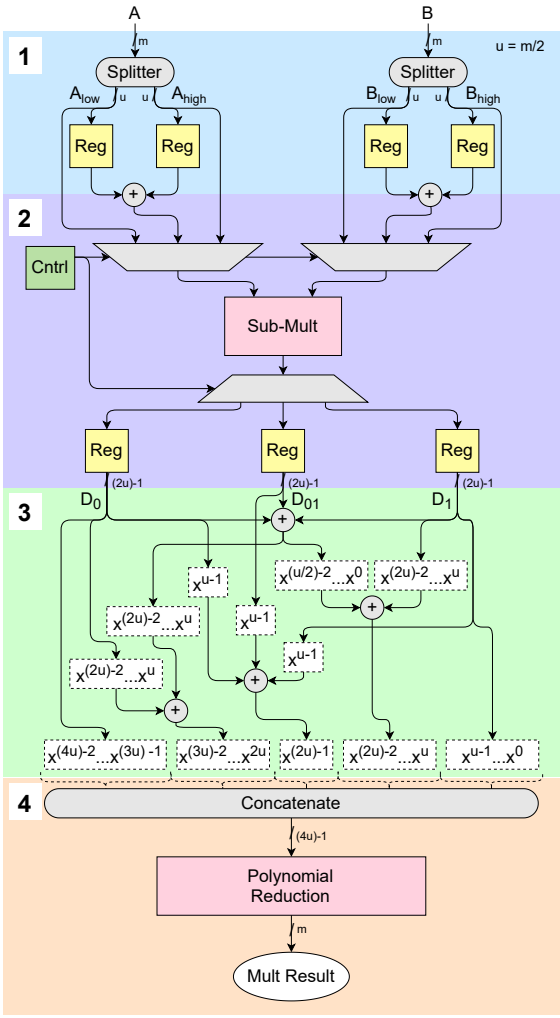

Figura 4.10: Arquitectura interna del motor de multiplicación en GF utilizando el algoritmo KOA

realizar instancias recursivas del motor de multiplicación en GF a fin de reducir la anchura de los datos a multiplicar. En la instancia del motor de multiplicación de menor nivel en la jerarquía es donde se implementan las 3 unidades lógicas que se encargan de calcular los valores D_0, D_{01} y D_1.

3. Alineación de los datos. Esta etapa consiste en una serie de operaciones lógicas XOR que se realizan sobre las salidas de datos D_0, D_{01} y D_1 para producir el resultado de la multiplicación, denominado D en la Ecuación 4.1. Hay que tener en cuenta que la anchura de datos del resultado D es de $2m - 1$ bits, siendo m el número de bits de los datos de entrada.

4. Reducción polinómica. Consiste en una serie de operaciones lógicas XOR que forman la reducción de datos del resultado D, pasando este a tener una anchura de datos de m bits.

4.2.5 Generador de mensajes

El módulo generador de mensajes, denominado con la letra E en la Figura 4.11, se encarga de generar los mensajes encriptados o desencriptados juntando las datos procedentes de las distintas secciones de la arquitectura.

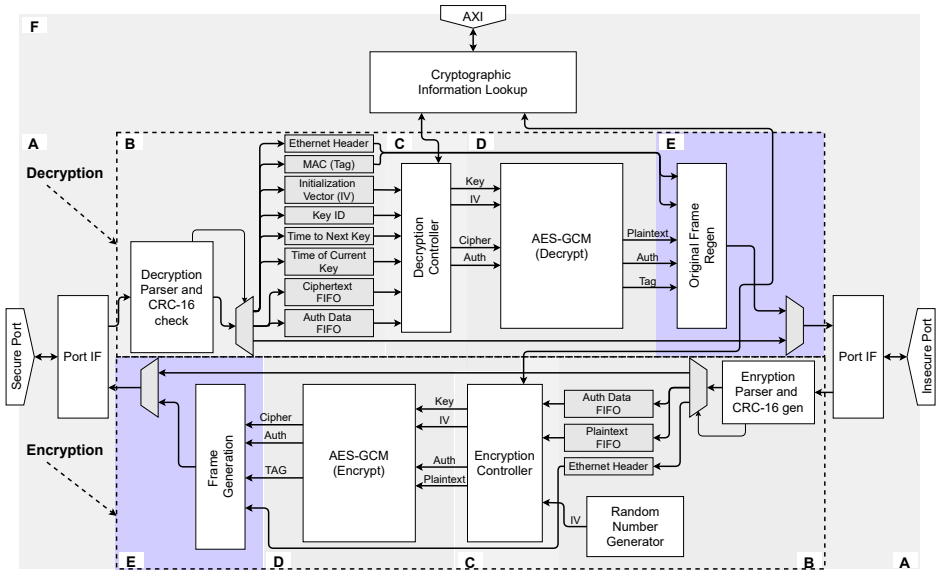

Figura 4.11: Módulo generador de mensajes de la arquitectura SoPC IEC 62351-6

Para ello, este módulo recibe datos del módulo analizador de mensajes (sección B), módulo controlador del motor criptográfico (sección C) y del motor criptográfico (sección E). Los datos recibidos de cada una de las secciones son:

- Analizador de mensajes: proporciona la cabecera Ethernet del mensaje re-

cibido y que no recibe procesamiento criptográfico. Adicionalmente, en el proceso de desencriptar los mensajes, también se proporciona la firma digital extraída de la extensión de seguridad IEC 62351-6.

- Controlador del motor criptográfico: en el caso de que se trate del proceso de encriptar los mensajes, se proporciona la información recibida del motor de búsqueda criptográfica (sección F) para que sea incluida en la extensión de seguridad IEC 62351-6. Esta información se compone de ToCK, TtNK, KeyID e IV.
- Motor criptográfico: proporciona los datos autenticados, así como los datos encriptados o desencriptados según el proceso del que se trate. También se proporciona la firma digital calculada.

En el caso del proceso de desencriptar los mensajes, este módulo también se encarga de realizar la comprobación de la firma digital. Se utiliza la firma digital criptográfica calculada por el motor criptográfico y se compara con la firma digital recibida en el mensaje GOOSE o SV protegido mediante la extensión de seguridad IEC 62351-6. En caso de que ambas firmas digitales sean idénticas, el mensaje es regenerado y transmitido a la salida. Por el contrario, si las firmas digitales no coinciden, los datos son descartados y por lo tanto el mensaje no es regenerado como medida de seguridad.

4.2.6 Motor de búsqueda de información criptográfica

El motor de búsqueda criptográfica, denominado con la letra F en la Figura 4.12, se encarga de múltiples funciones relacionadas con la configuración y la monitorización del estado del IP, así como el almacenamiento de la información criptográfica para la protección de los flujo de datos.

En primer lugar, este módulo implementa un interfaz de tipo Advanced eXtensible Interface (AXI) para proporcionar el acceso de lectura y escritura a los registros estadísticos y de configuración. A través de los registros estadísticos es posible obtener diferentes parámetros y métricas a cerca del funcionamiento del IP, como el número de mensajes recibidos, mensajes procesados, mensajes erróneos, etc. Además, los registros de configuración permiten almacenar o borrar la información criptográfica asociada a un flujo de datos que debe ser procesado. La información almacenada para cada flujo de datos es:

- Dataset ID. Identificador del flujo de datos. Se trata de un campo presente en los mensajes GOOSE y SV. Se trata de un campo de hasta 128 Bytes.
- KeyID. Identificador de clave asignado por el KDC en el proceso de distribución de clave. Se trata de un campo de 32 bits.

- ToCK. Marca temporal por la cual la clave actual ha sido marcada como válida. Se trata de un campo de 32 bits.
- TtNK. Segundos restantes hasta que se produzca un cambio de clave. Se trata de un campo de 32 bits.
- Key 0. Clave criptográfica actual. Se trata de un campo de 128 bits.
- Key 1. Clave criptográfica siguiente. Se trata de un campo de 128 bits.

Dicha información se almacena utilizando los elementos físicos con los que cuentan las FPGAs y SoPCs para el almacenamiento de grandes cantidades de información, denominados Block Random Access Memorys (BRAMs). Una BRAM es una memoria de doble puerto que puede ser escrita y leída de forma simultánea por sus dos puertos A y B. El motor de búsqueda criptográfico propuesto hace uso del puerto de escritura A y de los puertos de lectura B. De esta forma, el puerto A se utiliza para almacenar nuevos parámetros criptográficos en la memoria a través del interfaz de configuración. No se proporciona la posibilidad de leer los datos almacenados como medida de seguridad, evitando así accesos indeseados a la información.

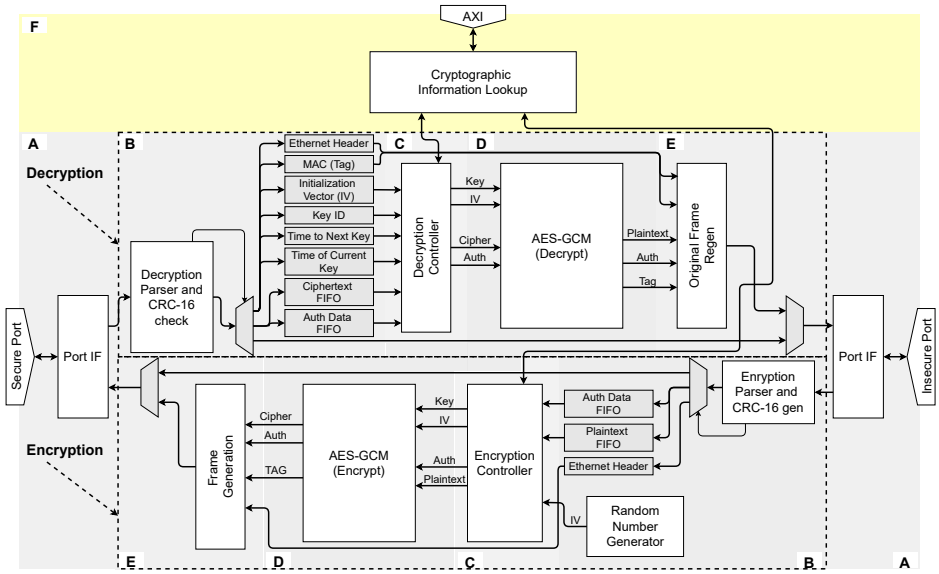

Figura 4.12: Motor de búsqueda criptográfica de la arquitectura SoPC IEC 62351-6

Por otro lado, el puerto B de la memoria se utiliza para proporcionar el acceso a los datos criptográficos en los procesos de búsqueda para encriptar y desen-

criptar los mensajes. Tal y como se puede ver en la Figura 4.13, la información criptográfica asociada a un flujo de datos se encuentra duplicada para permitir que ambos procesos puedan operar de forma simultánea. De esta forma, cada vez que se recibe un mensaje GOOSE o SV el módulo analizador de mensajes obtiene el campo Dataset ID y el módulo controlador del motor criptográfico lanza la petición de búsqueda al motor de búsqueda. El proceso de búsqueda consiste en ir leyendo de forma simultánea de todas las memorias la información asociada al identificador de flujos de datos y compararla con información extraída del mensaje. Si se encuentra la información criptográfica asociada, esta es devuelta al módulo controlador del motor criptográfico. En caso contrario, se señaliza al módulo controlador del motor criptográfico que no se ha encontrado la información solicitada y el mensaje es propagado a la salida sin que este sea procesado.

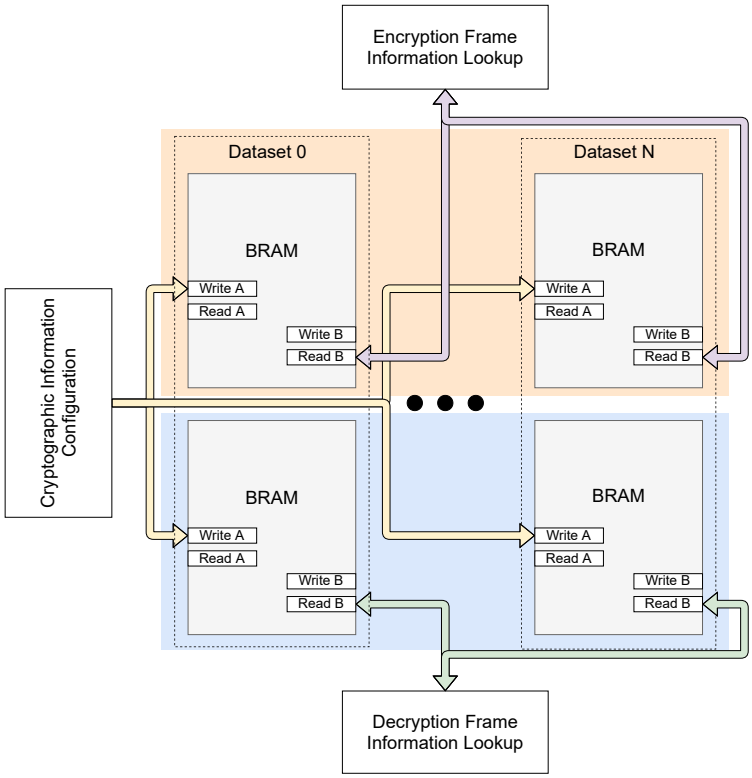

Figura 4.13: Almacenamiento y búsqueda de la información criptográfica

Capítulo 5

Validación de la arquitectura SoPC para la protección de mensajes GOOSE y SV

5.1 Introducción

La arquitectura hardware propuesta para proteger mensajes GOOSE y SV descrita en el Capítulo 4 ha sido evaluada y validada mediante experimentos realizados tanto en entornos de simulación como en pruebas sobre dispositivos físicos reales.

El reto principal al que la arquitectura propuesta hace frente es conseguir proporcionar confidencialidad, autenticidad e integridad a los mensajes GOOSE y SV sin comprometer los estrictos tiempos de entrega definidos en el estándar IEC 61850 para este tipo de mensajes. Por lo tanto, gran parte del proceso de validación se ha centrado en evaluar el rendimiento temporal que la arquitectura propuesta consigue. Para ello, a fin de medir la latencia introducida, se proponen dos entornos de pruebas. Por un lado, se han realizado mediciones temporales utilizando un entorno de pruebas basado en la simulación de la arquitectura. Este tipo de experimentos permiten medir el rendimiento temporal de la arquitectura con un nivel de precisión de un ciclo de reloj.

Por otro lado, utilizando un entorno de pruebas basado en dispositivos hardware,

se han realizado pruebas equivalentes a las realizadas en el entorno de simulación que permiten evaluar la correlación de los resultados obtenidos en simulación con el comportamiento real en hardware. Realizar mediciones temporales de alta precisión en hardware supone un reto en sí mismo. Consecuentemente, se propone una arquitectura para SoPC que permita realizar las mediciones temporales necesarias con una precisión del orden de los nanosegundos. Finalmente, también se han realizado pruebas utilizando una implementación real de la arquitectura propuesta para realizar la validación funcional de la misma, así como evaluar el uso de recursos lógicos consumidos y la viabilidad de su uso en dispositivos SAS.

5.2 Pruebas de validación

5.2.1 Validación temporal basada en simulación

Con el objetivo de evaluar la latencia introducida por la arquitectura propuesta, se ha diseñado un entorno de pruebas basado en simulación. Dicho entorno de pruebas se ha creado utilizando la herramienta Vivado en su versión 2018.3 [75] y consta de dos instancias *Sascrypt_0* y *Sascrypt_1* del IP IEC 62351-6 que implementa la arquitectura propuesta. La Figura 5.1 muestra dicho entorno de pruebas.

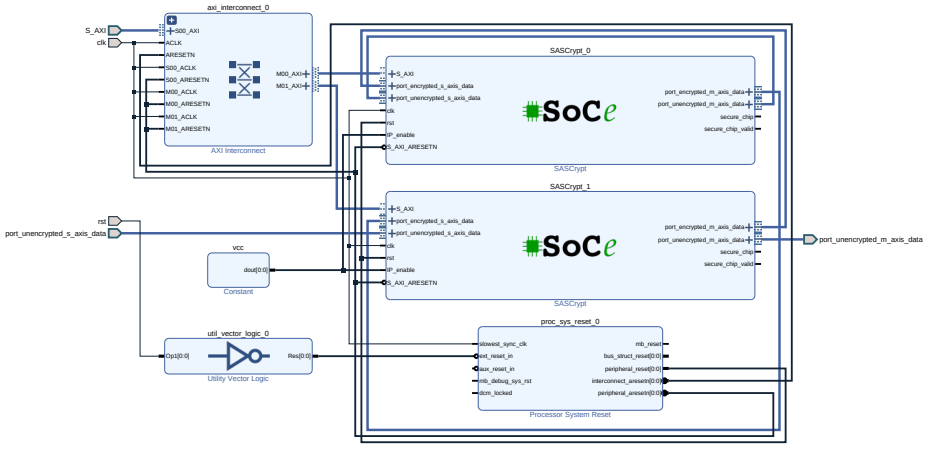

Figura 5.1: Entorno de pruebas basado en simulación utilizando la herramienta Vivado 2018.3

Tal y como se puede ver en la Figura 5.1 ambas instancias del IP se encuentran conectadas en serie. A continuación se describe el flujo de datos y las operaciones principales de cada uno de los elementos que componen el entorno de pruebas:

1. El IP *Sascrypt_1* recibe mensajes GOOSE y SV sin extensión de seguridad por el puerto inseguro, denominado *port_unencrypted_s_axis_data*.
2. Los mensajes GOOSE y SV recibidos, son analizados, encriptados y autenticados, añadiendo debidamente la extensión de seguridad IEC 62351-6. Después de que los mensajes sean procesados por el IP *Sascrypt_1*, estos son transmitidos por el puerto seguro, denominado *port_encrypted_m_axis_data*.
3. Los mensajes protegidos por el IP *Sascrypt_1* son recibidos por el IP *Sascrypt_0* a través de su puerto seguro, denominado *port_encrypted_s_axis_data*. Así, los mensajes GOOSE y SV recibidos, son analizados, desencriptados y autenticados.
4. En caso de que la firma digital calculada para cada uno de los mensajes se corresponda con el valor contenido en la extensión de seguridad IEC 62351-6 del propio mensaje, el mensaje es marcado como válido. En caso contrario, el mensaje es marcado como inválido y descartado por el IP. Finalmente, los mensajes marcados como válidos por el IP *Sascrypt_0* son transmitidos por el puerto inseguro, denominado *port_unencrypted_m_axis_data*.
5. Los mensajes GOOSE y SV desprotegidos, son nuevamente recibidos por el propio IP *Sascrypt_0*. De esta forma, se repiten los pasos 1 a 4 cambiando el sentido del flujo de datos. Es decir, los mensajes se encriptan y autentican por el IP *Sascrypt_0* y se desencriptan y autentican por el IP *Sascrypt_1*.

El objetivo de esta prueba es poder evaluar que la arquitectura propuesta tiene una latencia lo suficientemente baja como para no comprometer el tiempo máximo de 3 ms establecido por el estándar IEC 61850 para la generación, transmisión y procesamiento de los mensajes GOOSE y SV. En concreto, en el Capítulo 2 se establece que el tiempo asociado a los procesos de protección y desprotección de los mensajes GOOSE y SV no debe exceder el tiempo máximo de 0,4 ms.

El análisis de los resultados obtenidos en la simulación se realiza mediante el estudio de la ventana que muestra la forma de onda de cada una de las señales internas y externas de todos los elementos que componen el entorno de pruebas. Adicionalmente, la configuración de los IPs *Sascrypt_0* y *Sascrypt_1* se realiza mediante la escritura de registros desde el propio entorno de pruebas, para lo que se utiliza el bus de datos AXI. La configuración de ambos IPs permite establecer la información criptográfica que será utilizada para realizar el cifrado y autenticación de los mensajes. Por otro lado, las medidas temporales de los valores de latencia se llevan a cabo detectando el tiempo transcurrido entre la entrada y salida del primer bloque de datos de los puertos seguros e inseguros de cada instancia de los IPs.

La prueba llevada a cabo para realizar la medición de los valores de latencia consiste en enviar los siguientes mensajes:

- Mensaje GOOSE sin extensión de seguridad IEC 62351-6 y con un tamaño de 159 bytes.
- Mensaje SV sin extensión de seguridad IEC 62351-6 y con un tamaño de 156 bytes.

La prueba se lleva a cabo cuatro veces, modificando el número de mensajes de cada tipo que son enviados. Consecuentemente, en cada una de las iteraciones de la prueba se envían: 1, 100, 500 y 5000 mensajes respectivamente. Adicionalmente, cada una de las iteraciones descritas se repite en cinco ocasiones, modificando el instante temporal en el que se transmiten los mensajes. Así, el objetivo es determinar que no existe ninguna relación entre el instante temporal en el que los IPs procesan los mensajes y los resultados obtenidos. Los mensajes se envían respetando el espaciado mínimo de 96 ns fijado por el estándar Ethernet para enlaces de 1 Gbps [66]. La Figura 5.2 muestra los resultados obtenidos en la prueba descrita.

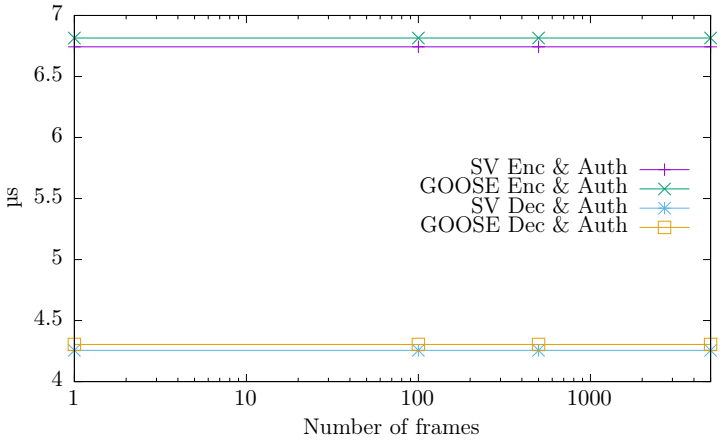

Figura 5.2: Mediciones de latencia temporal para la arquitectura IEC 62351-6 propuesta

Los resultados reflejados en la Figura 5.2 muestran que la arquitectura propuesta tiene un tiempo de respuesta determinista. Se puede observar como el proceso de encriptar y autenticar o desencriptar y autenticar un mensaje de un tamaño determinado, siempre se completa en un espacio temporal dado. En concreto, el proceso de encriptar y autenticar un mensaje de tipo GOOSE introduce una

latencia de 6,816 µs, mientras que la misma operación introduce una latencia de 6,744 µs cuando se trata de un mensaje SV. Por otro lado, el proceso de desencriptar y autenticar un mensaje de tipo GOOSE introduce una latencia de 4,304 µs, mientras que la misma operación introduce una latencia de 4,256 µs cuando se trata de un mensaje SV. Por lo tanto, estos resultados muestran que, para la peor situación posible, el cual es encriptar y autenticar un mensaje GOOSE, la latencia introducida por la arquitectura propuesta solamente representa el 1,5 % de los 0,4 ms disponibles para realizar la protección de los mensajes.

Además, los resultados obtenidos también muestran como la arquitectura propuesta es capaz de procesar los mensajes a velocidad de línea, soportando tasas de transferencia de 1 Gbps. Este hecho garantiza la robustez frente ataques de tipo DoS, ya que la arquitectura es capaz de recibir, filtrar y procesar cada uno de los mensajes, sin que la tasa de transferencia afecte al rendimiento.

5.2.2 Validación temporal basada en pruebas físicas

Una vez realizadas las medidas de latencia en el entorno de simulación, es necesario comprobar que es posible reproducir los resultados obtenidos en una implementación física. La Figura 5.3 muestra dicho entorno de pruebas.

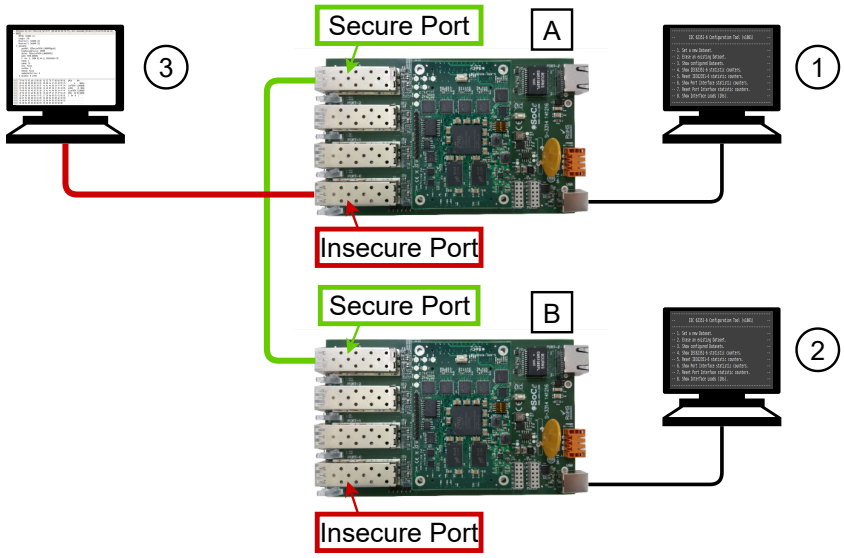

Figura 5.3: Entorno de pruebas para la validación temporal de la arquitectura

Para llevar a cabo la prueba, se han utilizado dos tarjetas de evaluación Smartzynq Brick [76] de la empresa System-on-Chip engineering (SoC-e). Estas tarjetas incluyen el SoPC Zynq 7020 de la empresa Xilinx [77]. La prueba, tal y como se puede observar en la Figura 5.3 está dividida en tres etapas que se recogen a continuación:

1. Se utiliza el Personal Computer (PC) para a través del interfaz de configuración Command Line Interface (CLI) configurar la información criptográfica del dispositivo A. Dicha información especifica que flujos de mensajes GOOSE y SV deben ser protegidos, así como los parámetros criptográficos asociados para encriptar, desencriptar y autenticar los mensajes. Una vez iniciada la prueba, a través del interfaz CLI se obtienen los valores de latencia medidos para el proceso de encriptar y autenticar los mensajes.
2. Se utiliza el PC para a través del interfaz de configuración CLI configurar la información criptográfica del dispositivo B. Una vez iniciada la prueba, a través del interfaz CLI se obtienen los valores de latencia medidos para el proceso de desencriptar y autenticar los mensajes.
3. Se utiliza el PC para generar los mensajes GOOSE y SV utilizados en la prueba y que son transmitidos al dispositivo A a través de su puerto inseguro. Los mensajes recibidos por el dispositivo A son protegidos con la extensión de seguridad IEC 62351-6 para ser después transmitidos al dispositivo B por su puerto seguro. El dispositivo B desencripta y autentica los mensajes recibidos para finalmente regenerar los mensajes originales sin protección IEC 62351-6.

Realizar mediciones temporales con una precisión en el orden de los nanosegundos supone un reto en sí mismo. Por ello, se ha propuesto un diseño FPGA específico que permita reproducir la prueba llevada a cabo en el entorno de simulación y realizar las medidas temporales asociadas. Este diseño FPGA ha sido generado utilizando la herramienta Vivado 2018.3. La Figura 5.4 muestra el diagrama de bloques del diseño FPGA propuesto para realizar las pruebas físicas y medidas de latencia.

Cada una de las dos tarjetas de evaluación utilizadas implementa el diseño FPGA de pruebas presentado en la Figura 5.4. El diagrama de bloques del diseño FPGA de pruebas se divide en dos partes claramente diferenciadas: Zynq Programmable Logic section (PL) y Zynq Processing System section (PS). A continuación se describen los elementos lógicos implementados en la PL y sus funcionalidades:

- IP SASCrypt que implementa la arquitectura hardware propuesta para proteger mensajes GOOSE y SV. Se trata del dispositvo Device Under Test (DUT) de la prueba, sobre el cual se pretenden realizar las medidas de

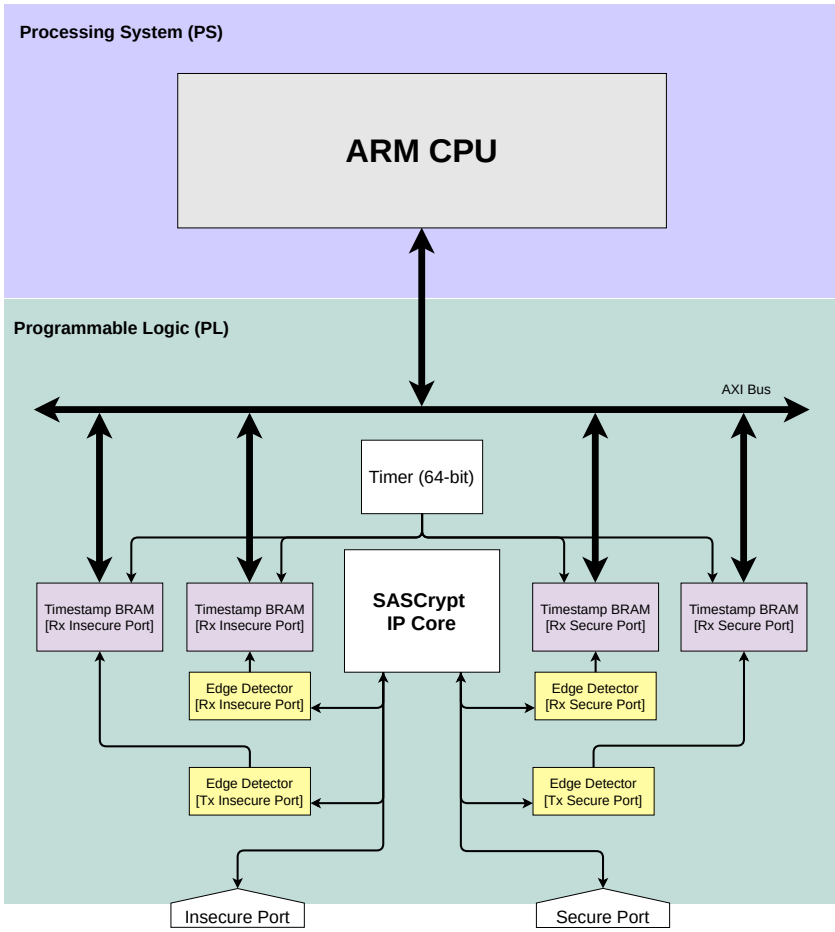

Figura 5.4: Arquitectura hardware para la medición de la latencia

latencia.

- Módulo detector de flancos que se utiliza para detectar el momento en el que se comienza la recepción o transmisión de un mensaje por alguno de los puertos del IP SASCrypt. Cuando se detecta un nuevo mensaje, el módulo detector de flancos genera una señal de disparo con una duración de un ciclo de reloj. En concreto, se cuenta con cuatro instancias del módulo detector de flancos, cada una de ellas asociada a la recepción y transmisión de mensajes por el puerto inseguro y seguro respectivamente.

- Módulo temporizador de 64 bits utilizado como referencia temporal para realizar las medidas de latencia de la prueba. Los 32 bits más significativos del temporizador representan el número de segundos desde que se encendió el equipo, mientras que los 32 bits menos significativos representan el tiempo transcurrido dentro de un segundo, expresado en nanosegundos. Puesto que se utiliza un reloj de una frecuencia de 125 MHz, la precisión del módulo temporizador es igual al periodo del reloj utilizado, 8 ns en este caso.
- Módulo generador de la marca temporal. Este módulo recibe el valor del módulo temporizador y la señal de disparo generada en el módulo detector de flancos. De esta forma, cuando el módulo detector de flancos señala un nuevo evento a través de la señal de disparo, el módulo generador de la marca temporal registra en dicho instante el valor de 64 bits procedente del módulo temporizador. En concreto, se cuenta con cuatro instancias del módulo generador de la marca temporal, cada una de ellas asociada a la recepción y transmisión de mensajes por el puerto inseguro y seguro respectivamente.
- Memoria de almacenamiento de marcas temporales. Esta módulo está basado en una memoria BRAM de doble puerto. Uno de los puertos es accedido por el módulo generador de la marca temporal, el cual escribe el valor temporal de 64 bits asociado a un nuevo evento. El otro puerto, es accedido mediante un interfaz AXI por la CPU que es parte de la PS del SoPC Zynq. En concreto, se cuenta con cuatro instancias del módulo de memoria de almacenamiento de marcas temporales, cada una de ellas asociada a la recepción y transmisión de mensajes por el puerto inseguro y seguro respectivamente.

Por otro lado, en la sección PS del SoPC Zynq se encuentra una CPU ARM de doble núcleo. Se ha desarrollado una aplicación software, la cual es ejecutada sobre dicha CPU, y que no hace uso de ningún sistema operativo. Las funciones principales de esta aplicación software son realizar la configuración del IP SASCrypt, así como realizar el cálculo de las medidas de latencia asociadas al proceso de encriptar y autenticar o desencriptar y autenticar los mensajes GOOSE y SV. La aplicación software desarrollada realiza las siguientes operaciones para llevar a cabo el cálculo de las medidas de latencia:

1. Se detecta la presencia de un nuevo dato en la memoria de almacenamiento de marcas temporales asociada a la recepción de mensajes por el puerto inseguro o seguro. Cuando este hecho ocurre, se realiza la lectura de la marca temporal de 64 bits.
2. Se detecta la presencia de un nuevo dato en la memoria de almacenamiento

de marcas temporales asociada a la transmisión de mensajes por el puerto inseguro o seguro, dependiendo de por qué puerto se hubiera detectado un nuevo dato en el punto anterior. Cuando este hecho ocurre, se realiza la lectura de la marca temporal de 64 bits.

3. Se realiza el cálculo de la latencia mediante la siguiente ecuación:

$$\begin{aligned} Latency_{enc} &= Timestamp_{(\text{tx secure port})} - Timestamp_{(\text{rx insecure port})} \\ Latency_{dec} &= Timestamp_{(\text{tx insecure port})} - Timestamp_{(\text{rx secure port})} \end{aligned} \tag{5.1}$$

4. Los resultados de las medidas de latencia son presentados utilizando un interfaz CLI.

La Figura 5.5 muestra las opciones de configuración que ofrece la aplicación software desarrollada a tal efecto. Además de añadir, modificar o eliminar nuevas entradas en los registros que almacenan la información criptográfica, el interfaz de configuración permite acceder a ciertos parámetros estadísticos que muestran información a cerca del estado o funcionamiento del IP.

```
------------------------------------------------------------
--        IEC 62351-6 Configuration Tool (v1801)          --
------------------------------------------------------------
-- 1. Set a new Dataset.                                   --
-- 2. Erase an existing Dataset.                           --
-- 3. Show configured Datasets.                            --
-- 4. Show IEC62351-6 statistic counters.                  --
-- 5. Reset IEC62351-6 statistic counters.                 --
-- 6. Show Port Interface statistic counters.              --
-- 7. Reset Port Interface statistic counters.             --
-- 8. Show Interface Loads (10s).                          --
------------------------------------------------------------
```

```
              ------------------------------------------------------------
              --              Configured Dataset List                   --
              ------------------------------------------------------------
> Dataset 0 Info:
       > Dataset: GEDeviceF650/LLN0$GOOSE1
       > Key ID: 0x00000001

       > Key 0: 0x112233445566778899aabbccddeeff
       > TOCK 0: 0x1000000 | 16777216s
       > TTNK 0: 0xb4 | 180m
```

Figura 5.5: Configuración a través del interfaz CLI

El acceso al interfaz de configuración CLI y a las lecturas de los parámetros de configuración se realiza mediante un PC conectado mediante un interfaz serie a

las tarjetas de evaluación Smartzynq Brick utilizadas en la prueba. Utilizando el entorno de pruebas descrito, se replica la prueba detallada en la Sección 5.2.1 para la medida de la latencia de la arquitectura propuesta. Los resultados obtenidos en el entorno de pruebas físico muestran una desviación de $\pm$ 8 ns (un ciclo de reloj), respecto a los resultados obtenidos en el entornos de simulación y representados en la Figura 5.2. Esta desviación en los resultados era esperada y se produce como consecuencia de los cambios de dominio de reloj necesarios para realizar las marcas temporales utilizadas para el cálculo de la medida de la latencia. Por lo tanto, teniendo esto en cuenta, se puede determinar que existe una correlación entre los resultados obtenidos en el entorno de simulación y en el entorno de pruebas físico.

5.2.3 Validación funcional basada en pruebas físicas

A fin de evaluar las capacidades funcionales de la arquitectura propuesta, se ha diseñado un entorno de pruebas basado en una implementación física. La Figura 5.6 muestra una descripción de los elementos utilizados para el desarrollo de la prueba.

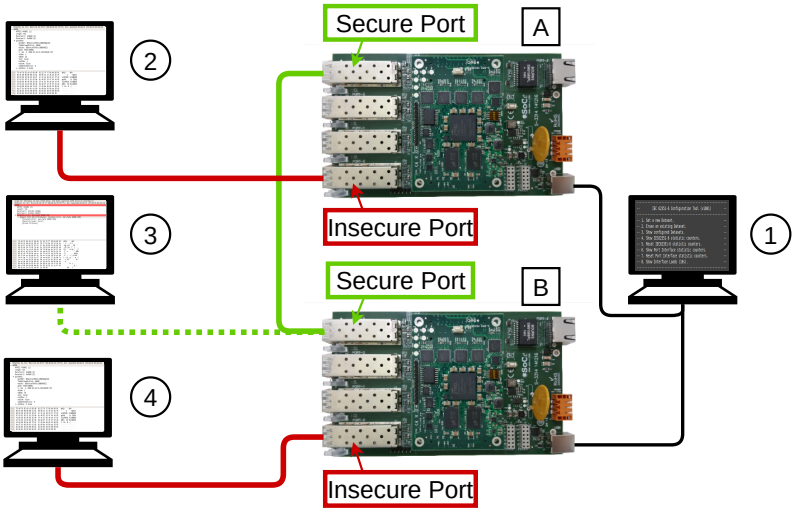

Figura 5.6: Entorno de pruebas para la validación funcional de la arquitectura

Los dispositivos A y B del entorno de pruebas son dos tarjetas de evaluación Smartzynq Brick de la empresa SoC-e. Cada una de las tarjetas cuenta en la PL del SoPC Zynq 7020 con una instancia del IP SASCrypt que implementa

la arquitectura propuesta para proteger los mensajes GOOSE y SV. Por otro lado, en la parte PS del SoPC Zynq 7020 el procesador ARM embebido ejecuta el software de configuración y monitorización que ha sido desarrollado para el desarrollo de las pruebas y que ha sido descrito en la Sección 5.2.2. Para ello, se utiliza un PC para tener acceso al interfaz de configuración CLI y realizar la configuración y monitorización de ambos dispositivos A y B. Adicionalmente, el PC también es utilizado para generar recibir y capturar los mensajes GOOSE y SV que serán utilizados durante la validación funcional de la arquitectura.

La prueba, tal y como se puede observar en la Figura 5.4 está dividida en cuatro etapas que se recogen a continuación:

1. Se utiliza el PC para a través del interfaz de configuración CLI configurar la información criptográfica de los dispositivos A y B. Dicha información especifica que flujos de mensajes GOOSE y SV deben ser protegidos, así como los parámetros criptográficos asociados para encriptar, desencriptar y autenticar los mensajes.

2. Se utiliza el PC para enviar al puerto inseguro del dispositivo A mensajes GOOSE y SV sin extensión de seguridad IEC 62351-6. El IP SASCrypt del dispositivo A analiza los mensajes recibidos y los protege de acuerdo a los parámetros criptográficos establecidos. Posteriormente, los mensajes GOOSE y SV protegidos son transmitidos a través del puerto seguro del dispositivo A.

3. Se utiliza el PC para capturar y analizar el flujo de mensajes GOOSE y SV transmitidos del dispositivo A al dispositivo B mediante sus puertos seguros.

4. El dispositivo B analiza los mensajes GOOSE y SV recibidos por su puerto seguro y realizar el procesamiento criptográfico asociado de acuerdo a los parámetros de configuración establecidos previamente. Finalmente, los mensajes que hayan sido correctamente autenticados son transmitidos sin extensión de seguridad IEC 62351-6 por el puerto inseguro del dispositivo B. Se utiliza el PC para realizar la captura y análisis de dichos mensajes.

En el desarrollo de la prueba se envían 50000 mensajes GOOSE con el identificador de flujo de datos IEC 61850: *GEDeviceF650/LLN0GOgcb01*. La Figura 5.7 muestra las lecturas realizadas a través del interfaz CLI de los registros estadísticos de los dispositivos A y B.

Tal y como se puede observar, el dispositivo A ha recibido 50000 mensajes a través de su puerto inseguro, este parámetro aparece como el número de tramas procesadas en la Figura 5.7. Además, también se puede ver en la figura como

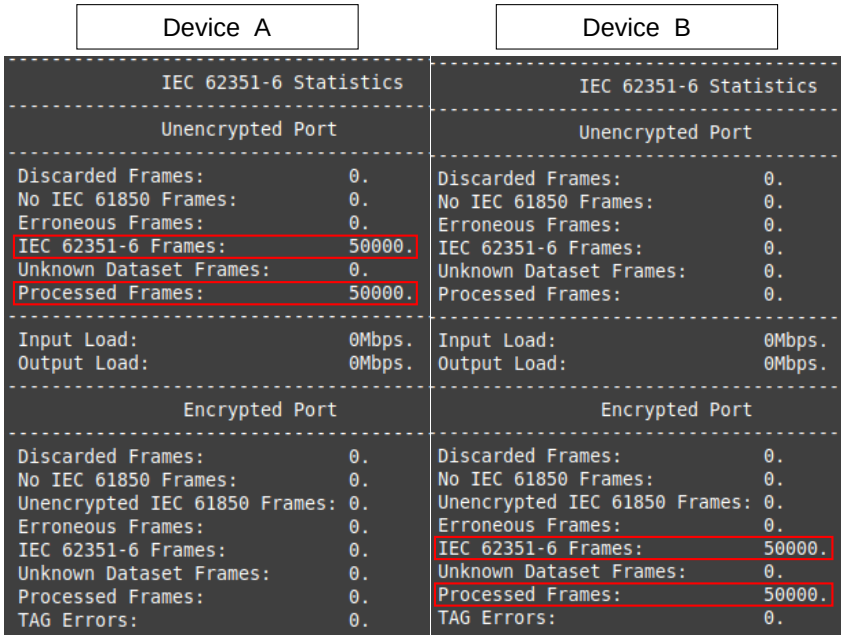

Figura 5.7: Resultado de las pruebas funcionales a través del interfaz CLI

todos los mensajes recibidos han sido identificados por el dispositivo A como mensajes IEC 61850 que han sido protegidos mediante la extensión de seguridad IEC 62351-6. Por otro lado, el dispositivo B ha recibido 50000 mensajes por su puerto seguro, estando este parámetro representado como el número de tramas procesadas. Adicionalmente, también se puede observar en la Figura 5.7 como todos los mensajes recibidos por el dispositivo B por su puerto seguro han sido identificados como mensajes IEC 61850 protegidos mediante la extensión de seguridad IEC 62351-6. Este hecho aparece representado como el número de mensajes IEC 62351-6 en la figura.

A fin de evaluar el procesamiento criptográfico realizado por el dispositivo A, se realiza un análisis de los mensajes capturados en cada una de las etapas de la prueba. En primer lugar, la Figura 5.8 muestra uno de los 50000 mensajes GOOSE que ha sido enviado durante la prueba y que no contiene la extensión de seguridad IEC 62351-6. Dicha captura se ha realizado en el puerto inseguro del dispositivo A mediante el software de análisis de de protocolos de red Wireshark [78].

La captura de la Figura 5.8 ha sido dividida en tres secciones. En primer lugar, en

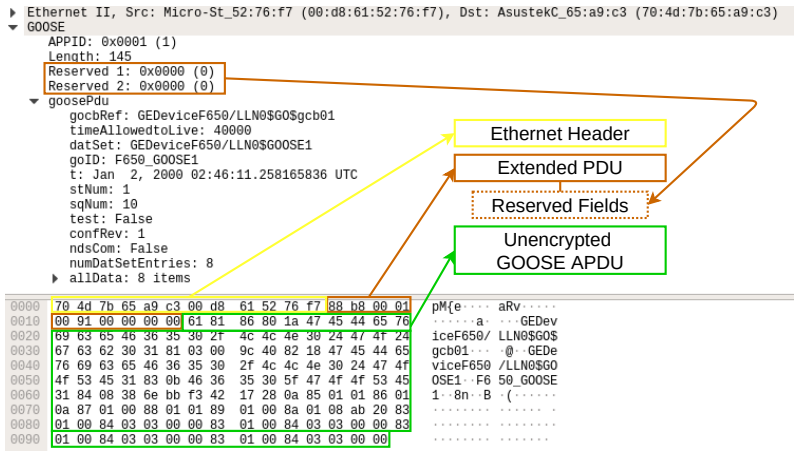

Figura 5.8: Captura de un mensaje GOOSE sin la extensión de seguridad IEC 62351-6

color amarillo, se han destacado los campos que componen la cabecera Ethernet. En color marrón, el campo EPDU contiene dos campos reservados y que serán utilizados posteriormente cuando el mensaje sea protegido mediante la extensión de seguridad IEC 62351-6. Finalmente, en color verde en la figura, se encuentra el campo GOOSE APDU, que representa el contenido de la trama GOOSE. Cabe destacar como, tal y como se puede ver en la Figura 5.8 el analizador de protocolos Wireshark es capaz de identificar los subcampos que componen el GOOSE APDU, como por ejemplo: *gocbRef* o *datSet*.

Por otro lado, en la Figura 5.9 se muestra el mismo mensaje que se había capturado en la Figura 5.8. Sin embargo, en este caso el mensaje ha sido capturado entre los puertos seguros de los dispositivos A y B. Por lo tanto, este mensaje ha sido procesado y protegido mediante la extensión de seguridad IEC 62351-6 por el dispositivo A.

Si se realiza una comparación entre el mensaje original y el mensaje protegido mediante la extensión de seguridad IEC 62351-6, se pueden identificar varias diferencias significativas. En primer lugar, se puede ver como el campo EPDU, en color marrón en las figuras, ahora es utilizado para indicar la longitud de la extensión de seguridad y el CRC de la propia cabecera. Adicionalmente, se puede ver como en la Figura 5.9 el campo GOOSE APDU, representado en color verde, está ahora encriptado. De hecho, el propio Wireshark es incapaz de identificar los subcampos que componen el GOOSE APDU y es por esto que el mensaje aparece marcado como erróneo. Finalmente, después del campo GOOSE APDU

se encuentra localizada la extensión de seguridad IEC 62351-6. Esta se compone de varios subcampos, siendo el primero de ellos la cabecera de la extensión de seguridad, representada en color morado en la figura y siendo el último de ellos el subcampo que contiene la firma digital del mensaje, representado en color rojo en la figura.

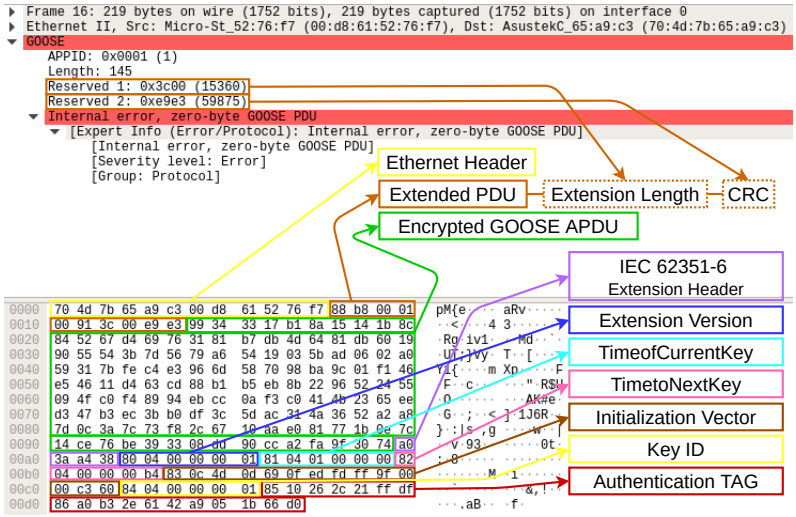

Figura 5.9: Captura de un mensaje GOOSE con la extensión de seguridad IEC 62351-6

Realizando un análisis del mismo mensaje capturado en el puerto inseguro del dispositivo B, se observa como el contenido del mismo es idéntico al mensaje original representado en la Figura 5.8. Por lo tanto, se puede concluir que los resultados de la prueba muestran como la arquitectura propuesta es capaz de tanto encriptar y autenticar como desencriptar y autenticar los mensajes IEC 61850 de acuerdo a la extensión de seguridad IEC 62351-6.

5.3 Uso de recursos lógicos

Con el fin de garantizar la viabilidad de la arquitectura propuesta en dispositivos SAS, es necesario evaluar el uso de recursos lógicos de la misma. Tal y como se analizó en el Capítulo 2, los dispositivos SAS utilizan habitualmente SoPCs con un número de recursos lógicos limitados. Además, es necesario tener en consideración que parte de los recursos lógicos del SoPC son utilizados para implementar

otras funcionalidades requeridas por las comunicaciones definidas en el estándar IEC 61850.

Realizando un análisis específico del uso de recursos lógicos de la arquitectura propuesta, se ha podido determinar como el motor criptográfico que implementa el algoritmo AES-GCM hace uso de una parte significativa de los recursos lógicos empleados por el conjunto de la arquitectura. En concreto, el análisis de la función GHASH empleada para el cálculo de la firma digital resulta relevante, pues supone entre un 27,77 % y un 8,21 % del total de la arquitectura propuesta dependiendo del algoritmo utilizado para realizar las multiplicaciones en campos de Galois. La Tabla 5.1 muestra el uso de recursos lógicos de la función GHASH dependiendo del algoritmo de multiplicación en campos de Galois utilizado.

Tabla 5.1: Uso de recursos de la función GHASH en el algoritmo AES-GCM

Algoritmo	LUTs	FFs	BRAMs	Recursos	Rendimiento
Cuadrático	8072	256	0	100 %	100 %
KOA	5752	384	0	73,68 %	100 %
KOA-2	4089	639	0	56,77 %	50 %
KOA-3	2627	640	0	39,23 %	33 %
KOA-4	2387	767	0	37,87 %	25 %

El motor criptográfico AES-GCM utilizado en la arquitectura propuesta puede hacer uso de dos algoritmos para realizar las multiplicaciones en campos de Galois utilizadas en la función GHASH. En la Tabla 5.1 se puede observar como el algoritmo KOA es capaz de conseguir el mismo rendimiento que el algoritmo Cuadrático utilizando únicamente un 73,68 % de los recursos lógicos. Los algoritmos KOA-2, KOA-3 y KOA-4, son implementaciones del algoritmo KOA que, mediante la multiplexación temporal de los elementos que implementan el algoritmo, consiguen realizar una reducción del uso de recursos lógicos a costa de una reducción del rendimiento. La multiplexación temporal de los recursos lógicos incrementa el número de ciclos de reloj empleados para el procesamiento de un bloque de datos de 128 bits. Se puede observar como la implementación KOA-4 consigue realizar una reducción del 63,13 % del uso de recursos lógicos mientras que el rendimiento obtenido es del 25 % respecto al conseguido con el algoritmo Cuadrático.

Sin embargo, es necesario tener en consideración que utilizando un reloj de 125 MHz el algoritmo Cuadrático, gracias a su capacidad de procesar un bloque de datos de 128 bits cada ciclo de reloj, es capaz de proporcionar un rendimiento de 16 Gbps. Por lo tanto, el algoritmo KOA-4, utilizando un reloj de igual frecuencia, es capaz de proporcionar un rendimiento de 4 Gbps, lo cual resulta

suficiente para soportar la tasa de transferencia máxima de 1 Gbps utilizada en SAS. Como contrapartida, bajo ciertas situaciones de alta ocupación de los recursos lógicos de la FPGA en los que el rutado de dichos elementos no puede ser óptimo, el algoritmo Cuadrático presenta unos requisitos temporales menos estrictos respecto a los establecidos por el algoritmo KOA. Por lo tanto, en este tipo de situaciones específicas el algoritmo Cuadrático puede tener cabida respecto al algoritmo KOA. Cabe destacar que se ha observado en las pruebas como implementaciones con un grado de multiplexación de los recursos lógicos del algoritmo KOA superiores a las presentadas en la Tabla 5.1, imponen unos requisitos temporales que hacen inviable su implementación en la mayoría de los casos.

El motor criptográfico AES-GCM desarrollado como parte de la arquitectura propuesta ha sido comparado frente a otras alternativas identificadas en la literatura. La tabla 5.2 compara el uso de recursos lógicos, el rendimiento y la eficiencia de los motores criptográficos analizados.

Tabla 5.2: Comparativa del uso de recursos entre arquitecturas del algoritmo criptográfico AES-GCM

Arquitectura	FPGA	Clave Variable	Slices	BRAMs	Rendimiento	Rendimiento/Slice
[52]	Virtex 5	No	2478	40	16 Gbps	6,46 Mbps
[74]	Virtex 5	Si	4533	41	6,74 Gbps	1,48 Mbps
[79]	Virtex 4	Si	13200	14	14 Gbps	1,07 Mbps
Este trabajo - KOA	Virtex 7	Si	5145	40	16 Gbps	3,11 Mbps
Este trabajo - KOA-4	Virtex 7	Si	2194	40	4 Gbps	1,81 Mbps
Este trabajo - KOA-4	Artix 7	Si	2300	40	4 Gbps	1,73 Mbps

Tal y como se puede ver en la Tabla 5.2, la propuesta realizada en [52] es capaz de procesar datos de 128 bits en cada ciclo de reloj. Por lo tanto, utilizando un reloj de 125 MHz el rendimiento asociado es de 16 Gbps. Sin embargo, cabe destacar que esta solución hace uso de claves criptográficas precompiladas. Este hecho tiene como consecuencia que, cada vez que se necesite utilizar una nueva clave criptográfica, es necesario realizar la expansión de clave AES de forma previa al procesamiento de los datos. Para ello, una pila software debe llevar a cabo dicha expansión de clave AES y configurar el valor en el IP. Este tipo de enfoque no tiene cabida en SAS, puesto que hay que tener en consideración que cada flujo de datos utiliza una clave criptográfica distinta y que el cambio de las claves criptográficas asociadas a cada flujo de datos tiene una periodicidad del orden de los minutos.

Por otro lado, las propuestas presentadas en [74] y [79] basan su excelente rendimiento en el uso de altas frecuencias de reloj. Para ello, es necesario hacer uso de FPGAs o SoPCs de alto rendimiento que sean capaces de funcionar a elevadas frecuencias de reloj.

Sin embargo, el motor criptográfico AES-GCM desarrollado en este trabajo y

que es utilizado en la arquitectura propuesta, no tiene ninguna de las limitaciones mencionadas. El rendimiento es independiente de las claves criptográficas utilizadas, así como del uso de familias de FPGAs o SoPCs de alto rendimiento. Adicionalmente, cabe destacar que la frecuencia de reloj utilizada es de 125 MHz, lo cual resulta un valor asumible en familias de FPGAs o SoPCs de bajo coste modernas. El motor criptográfico desarrollado, cuenta con distintos parámetros de configuración que permiten adaptar su rendimiento y uso de recursos lógicos en tiempo de síntesis. De esta forma, tal y como se puede ver en la Tabla 5.2, cuando se implementa el algoritmo KOA para la función GHASH, el rendimiento del motor criptográfico es de 16 Gbps, lo cual excede ampliamente el requisito de soportar el procesamiento de un flujo de datos de 1 Gbps. En cambio, cuando se hace uso del algoritmo KOA-4, el rendimiento del motor criptográfico es de 4 Gbps, mientras que el uso de recursos lógicos se ve reducido significativamente.

Por lo tanto, se puede determinar que el motor criptográfico desarrollado en este trabajo, presenta la mejor relación rendimiento/uso de recursos lógicos respecto al resto de alternativas analizadas. Sin embargo, a pesar de que la mayor eficiencia se consigue cuando se hace uso del algoritmo KOA, puesto que el rendimiento ofrecido excede los requisitos de los dispositivos SAS, el algoritmo KOA-4 resulta el idóneo para el uso en SAS, ya que se consigue un rendimiento suficiente con un uso de recursos lógicos mínimo.

Utilizando la herramienta software Vivado 2018.3 de Xilinx, se ha generado un diseño FPGA que implemente las tres funcionalidades principales que son necesarias para el funcionamiento seguro de los dispositivos SAS en el entorno de una subestación eléctrica: sincronización temporal, redundancia de las comunicaciones y ciberseguridad de los mensajes GOOSE y SV. Para proporcionar la sincronización temporal se ha utilizado el IP Precise Time Basic (PTB) [80]. Dicho IP implementa el estándar IEEE 1588, el cual utiliza mensajes de red para proporcionar sincronización temporal con una precisión del orden de nanosegundos utilizando una topología maestro-esclavo [81].

Por otro lado, para proporcionar redundancia de las comunicaciones se ha utilizado el IP HSR-PRP Switch (HPS) [82]. Dicho IP implementa el estándar IEC 62439-3, el cual define los protocolos High-availability Redundancy (HSR) y Parallel Redundancy Protocol (PRP) para proporcionar redundancia de red a través del envío de mensajes duplicados [83]. Finalmente, para proporcionar ciberseguridad de los mensajes GOOSE y SV se ha utilizado el IP SASCrypt que implementa la arquitectura propuesta en esta tesis. Cabe destacar que, puesto que la arquitectura propuesta presenta un alto grado de configurabilidad que permite adaptar su rendimiento y uso de recursos lógicos a las necesidades específicas de cada caso, el uso de recursos lógicos mostrados en la Tabla 5.3 es dependiente de la

configuración seleccionada. En este caso, se ha hecho uso del algoritmo KOA-4.

Tabla 5.3: Uso de recursos de cada una de las funcionalidades IEC 61850 en un dispositivo SoPC Xilinx Zynq-7020

Estándar	Funcionalidad	IP	LUTs	FFs	BRAMs
IEEE 1588	Sincronización Temporal	PTB [80]	2631 (4,95 %)	1191 (1,12 %)	0
IEC 62439-3	Redundancia	HPS [82]	13986 (26,29 %)	14431 (13,56 %)	41.5 (29,64 %)
IEC 62351-6	Ciberseguridad de mensajes GOOSE y SV	SASCrypt (Este Trabajo)	29071 (54,64 %)	29997 (28,19 %)	85.5 (61,07 %)
	Total		45688 (85,88 %)	45619 (42,88 %)	127 (90,71 %)

La Tabla 5.3 muestra como es posible integrar sincronización temporal, redundancia de las comunicaciones y ciberseguridad de los menesajes GOOSE y SV en un dispositivo SoPC Zynq 7020. Esta familia de SoPC de bajo coste resulta adecuada para su uso en SAS debido a su capacidad para proporcionar aceleración hardware a diversos protocolos, así como realizar procesamiento digital de señal. Al mismo tiempo, este SoPC integra un procesador ARM de doble núcleo y ofrece un consumo energético reducido.

5.4 Análisis de los resultados

Los resultados obtenidos en las pruebas de validación son evaluados y comparados respecto al estado del arte en materia de seguridad de mensajes GOOSE y SV mediante la extensión de seguridad IEC 62351-6. La Tabla 5.4 muestra los resultados de la comparación.

Las primeras cinco entradas de la Tabla 5.4 se componen de implementaciones software y hardware del cálculo de la firma digital para proporcionar autenticidad e integridad de los datos. Ninguna de estas propuestas hace uso de la confidencialidad de los mensajes. Cabe destacar que los autores de dichas propuestas no proporcionan los valores de transferencia de datos máximos que las soluciones presentadas son capaces de soportar. Por lo tanto, resulta imposible determinar si el rendimiento alcanzado es suficiente para su implementación en SAS o si los resultados de latencia obtenidos se ven afectados en función de la tasa de transferencia. Adicionalmente, ninguna de las propuestas basadas en una implementación software es capaz de proporcionar unos valores de latencia fijos y predecibles. Por lo tanto, los resultados obtenidos son dependientes de las tareas llevadas a cabo por el procesador.

Finalmente, se puede observar como las primeras cinco propuestas de la Tabla 5.4 son incapaces de cumplir con el requisito impuesto en el estándar IEC 61850 por el cual los mensajes GOOSE y SV deben ser generados, enviados, recibidos

Tabla 5.4: Comparativa del estado del arte en ciberseguridad de mensajes GOOSE y SV mediante la extensión de seguridad IEC 62351-6

Propuesta	Algoritmo	Seguridad	Tipo	Latencia Máxima	Rendimiento Máximo	Determinista	Cumple IEC 62351-6
[37]	RSA	Firma digital	SW	4000 µs	-	No	No (Tiempo)
[37]	RSA	Firma digital	HW	1917 µs	-	Si	No (Tiempo)
[38]	RSA	Firma digital	SW	6000 µs	-	No	No (Tiempo)
[39]	RSASSA-PKCS1-v15	Firma digital	SW	942 µs	-	No	No (Tiempo)
[39]	RSASSA-PKCS1-v15	Firma digital	SW	3560 µs	-	No	No (Tiempo)
[55]	SHA-256	Firma digital	SW	12,7 µs	-	No	Si (Solo firma digital)
[55]	AES-GMAC	Firma digital	SW	5,4 µs	-	No	Si (Solo firma digital)
[56]	EtM (AES & SHA-256)	Firma digital Cifrado	SW	242 µs	-	No	No (Formato)
[56]	E&M (AES & SHA-256)	Firma digital Cifrado	SW	235 µs	-	No	No (Formato)
[56]	MtE (AES & SHA-256)	Firma digital Cifrado	SW	284 µs	-	No	No (Formato)
Este trabajo	AES-GCM	Firma digital Cifrado	HW	6 µs	$>1\,\mathrm{Gbit\,s^{-1}}$	Si	Si

y procesados en menos de 3 ms. Cuando no se hace uso de la extensión de seguridad IEC 62351-6, la estimación es que 2,6 ms son empleados en la generación, procesamiento y distribución de los mensajes GOOSE y SV. Por lo tanto, solamente 0,4 ms quedarían disponibles para llevar a cabo tanto la protección como la desprotección de los mensajes. Consecuentemente, se concluye que ninguna de las propuestas presentadas en [37–39] es válida para realizar la protección de mensajes GOOSE y SV de acuerdo a los estándares IEC 61850 e IEC 62351-6.

En [56] se presenta la única propuesta identificada en la literatura que proporciona integridad, autenticidad y confidencialidad a los mensajes GOOSE y SV. Para ello, los autores describen tres métodos que se apoyan en el algoritmo AES para cifrar los mensajes y el algoritmo SHA-256 para realizar el cálculo de la firma digital. Los valores de latencia presentados para cualquiera de los tres métodos descritos son lo suficientemente bajos como para poder encriptar y autenticar los mensajes GOOSE y SV sin comprometer los estrictos requisitos temporales establecidos en el estándar IEC 61850. Sin embargo, los tres métodos propuestos se basan en modificaciones sobre el estándar IEC 62351-6. Por lo tanto, ninguno de los métodos garantiza la interoperabilidad con otras propuestas para la protección de mensajes GOOSE y SV.

Adicionalmente, puesto que los autores en [56] no proporcionan información a cerca del rendimiento alcanzado cuando se procesan altos flujos de datos, resulta imposible determinar si los métodos propuestos resultan adecuados para una

red de comunicaciones real. Se debe considerar que dependiendo de la topología específica de la subestación eléctrica, es posible alcanzar tasas de transferencia de datos de cientos de Mbps o incluso valores cercanos a 1 Gbps. Por lo tanto, es necesario garantizar que el rendimiento y valores de latencia permanecen estables independientemente de las tasas de transferencia de datos utilizadas.

Finalmente, puesto que los métodos propuestos en [56] hacen uso del algoritmo AES para proporcionar confidencialidad de los mensajes, se introduce una vulnerabilidad que podría ser utilizada por un atacante para comprometer las comunicaciones de la subestación. Cuando se hace uso del algoritmo AES una entrada del algoritmo siempre produce la misma salida. De esta forma, teniendo en cuenta que los mensajes GOOSE y SV cuentan con campos fijos que siempre contienen el mismo valor, esta información podría ser utilizada para determinar la clave criptográfica utilizada.

Por lo tanto, tal y como se puede observar en la Tabla 5.4, la arquitectura propuesta en la presente tesis, es la única propuesta respecto a todas las analizadas en la literatura que cumple con todos los requisitos establecidos para poder proteger los mensajes GOOSE y SV de acuerdo al estándar IEC 62351-6. La arquitectura propuesta presenta una latencia que representa el 1,5 % del tiempo que se estima disponible para proporcionar seguridad a los mensajes GOOSE y SV sin comprometer los estrictos requisitos temporales establecidos en el estándar IEC 61850. Además, esta latencia es constante y predecible independientemente de la tasa de transferencia de mensajes utilizada. La arquitectura propuesta es capaz de soportar tasas de transferencia de hasta 1 Gbps sin que el rendimiento temporal se vea comprometido. Finalmente, los valores de ocupación de recursos lógicos presentados muestran como es viable implementar sincronización temporal, redundancia de las comunicaciones y ciberseguridad de los mensajes GOOSE y SV haciendo uso de dispositivos SoPC de bajo coste adecuados para SAS.

Capítulo 6

Conclusiones y trabajo futuro

6.1 Conclusiones

En este trabajo se ha presentado una solución para poder proteger las comunicaciones IEC 61850 con requisitos de tiempo real en SAS. En concreto, se identificó que atendiendo los descubrimientos y aportaciones de la comunidad investigadora e industrial, proporcionar integridad, autenticidad y confidencialidad de forma simultánea a las comunicaciones GOOSE y SV suponía un reto por resolver dada la naturaleza y requisitos temporales de este tipo de mensajes. Por lo tanto, dada la relevancia de este tipo de comunicaciones, proponer una solución que fuera capaz de protegerlas ha sido el objetivo principal del trabajo de investigación realizado.

En el análisis del estado del arte se ha dividido en dos partes. En primer lugar, se ha llevado acabo un estudio de la evolución de las comunicaciones en los sistemas e infraestructuras de generación y distribución de energía eléctrica. En segundo lugar, se ha realizado un análisis de los mecanismos de seguridad definidos para dispositivos SAS, así como las vulnerabilidades a las que se encuentran expuestos los sistemas en SAS y los requisitos de seguridad para combatirlas.

Después de identificar los problemas y limitaciones de las soluciones propietarias utilizadas en los sistemas SCADA de las infraestructuras de generación y distribución de energía, se introduce la necesidad de unificar y estandarizar los sistemas de comunicaciones para poder atender los requisitos de flexibilidad, robustez y

eficiencia que demandan las subestaciones del siglo XXI. Consecuentemente, a finales de la década de los 90, los organismos internacionales publican la familia de estándares IEC 61850 que regula las comunicaciones, protocolos y modelos de datos en SAS. El estándar IEC 61850 introduce nuevas posibilidades y funcionalidades que habilitan la creación de las redes de generación y distribución de energía inteligentes, llamadas Smart Grids. Sin embargo, de la mano de las nuevas funcionalidades y de la interconexión de las redes privadas de las subestaciones eléctricas a las redes públicas externas, surgen nuevas vulnerabilidades que ponen en riesgo los sistemas SAS.

Después de analizar los eventos en forma de ciberataques que se han producido a distintas subestaciones a lo largo de todo el mundo en las últimas décadas, el IEC identificó la necesidad de publicar una familia de estándares que definiese los mecanismos de seguridad necesarios para proteger las comunicaciones en SAS. En IEC 61850, las comunicaciones para el control y monitorización de los IEDs se llevan a cabo a través de los mensajes GOOSE y SV. Dada su naturaleza, el estándar IEC 61850 define unos estrictos requisitos temporales para la generación, transmisión, recepción y procesamiento de este tipo de mensajes. Por lo tanto, su protección debe llevarse a cabo mediante mecanismos de seguridad específicos y no mediante el uso de protocolos de seguridad de propósito general. En concreto, el estándar IEC 62351-6 define los mecanismos de seguridad aplicados a los mensajes GOOSE y SV. Las investigaciones llevadas a cabo por la comunidad investigadora y la industria muestran que la primera versión del estándar IEC 62351-6, al hacer uso de algoritmos de criptografía de clave asimétrica, imposibilita la protección de los mensajes GOOSE y SV atendiendo sus estrictos requisitos temporales. Consecuentemente, el IEC publica una nueva revisión del estándar IEC 62351-6:2020 para atender las limitaciones identificadas por la comunidad investigadora.

La última versión del estándar IEC 62351-6 define el uso del algoritmo AES-GCM para proporcionar integridad, autenticidad y confidencialidad a los mensajes GOOSE y SV en SAS. En esta tesis se realiza un detallado análisis de las soluciones identificadas en el estado del arte para la protección de los mensajes GOOSE y SV, concluyendo que ninguna de ellas satisface todos los requisitos planteados para su protección y uso generalizado. Por tanto, el reto de proponer una solución para este tipo de comunicaciones estaba aun por resolver. La hipótesis de partida para la contribución presentada en esta tesis se basa en el análisis detallado realizado del algoritmo AES-GCM y presentado en el estado del arte de esta tesis. Sus fundamentos matemáticos, composición lógica, operaciones básicas y modos de funcionamiento permiten plantear una arquitectura hardware flexible y de alto rendimiento capaz de proteger este tipo de tráfico a la velocidad de comunicación en la línea. Idealmente, ofreciendo tiempos de latencias para

el procesamiento pequeños y constantes y que permita proporcionar integridad, autenticidad y confidencialidad de forma simultánea.

Atendiendo al análisis del estado del arte realizado, se definen los requisitos de seguridad y diseño que permitan llevar a cabo la protección de las comunicaciones con requisitos de tiempo real en dispositivos SAS.En esta tesis se propone el uso de una arquitectura hardware implementable en SoPC, llamada SASCrypt, para resolver el reto de proteger las comunicaciones GOOSE y SV. Esta arquitectura, cuenta con una estructura modular que le permite actualizarse a nuevas revisiones de estándar IEC 62351-6, ser utilizada para generar hardware con rendimientos de computación diferentes, utilizar nuevos algoritmos criptográficos o incluso soportar la protección de nuevos protocolos de comunicaciones con requisitos de tiempo real. Como elemento habilitador de la arquitectura, se define un IP AES-GCM flexible, adaptable, con baja latencia, respuesta determinista y con alta capacidad de procesamiento. El diseño del IP AES-GCM ha sido llevado a cabo teniendo los requisitos específicos en SAS e incluye una serie de mecanismos que permiten balancear el rendimiento y latencia ofrecidos respecto al uso de recursos lógicos del SoPC dependiendo de las necesidades de cada caso de uso concreto.

En la fase de validación se simula y se verifica en laboratorio mediante hardware real la arquitectura SASCrypt. El objetivo de las pruebas aplicadas es doble. Por un lado, verificar el cumplimiento de los requisitos definidos y por otro, evaluar el rendimiento en términos de latencia y recursos ofrecida por la misma con el fin de poder demostrar la superación del estado del arte en la materia. Mediante una prueba combinada en simulación y en un entorno de pruebas físico se evalúa la correlación de las mediciones de latencia obtenidas en ambos entornos. Los resultados muestran que la arquitectura propuesta es capaz de proteger los mensajes GOOSE y SV introduciendo una latencia que es ordenes de magnitud inferior al requisito establecido. Adicionalmente, se realizan pruebas funcionales que permitan comprobar el cumplimiento del estándar IEC 62351-6 y garanticen la interoperabilidad de la solución. Posteriormente, se evalúa el uso de recursos lógicos del SoPC de la arquitectura SAScrypt y se demuestra la viabilidad de la solución en dispositivos modernos de bajo coste adecuados para su uso en SAS. Finalmente, se realiza una comparación entre los resultados obtenidos en las pruebas y las últimas propuestas identificadas en la literatura para la protección de las comunicaciones GOOSE y SV. Los resultados muestran como SASCrypt es la única solución que es capaz de proteger las comunicaciones IEC 61850 de acuerdo al estándar de seguridad IEC 62351-6, sin comprometer los estrictos requisitos temporales asociados a los mensajes GOOSE y SV.

6.2 Contribuciones Principales

En esta sección se recoge un listado de las principales contribuciones de esta tesis. Para ello, se proporciona una breve descripción y una referencia a la sección del documento donde se realiza un desarrollo en profundidad. El orden de cada uno de los puntos de la lista viene determinado por el orden en el que cada una de las contribuciones aparecen en el documento.

1. **Análisis del estado del arte en seguridad para mensajes GOOSE y SV.**

   La evolución de los sistemas SCADA tradicionales en redes de comunicaciones digitales modernas, abrió la puerta a que los sistemas de generación y distribución de energía alcanzasen niveles de fiabilidad, flexibilidad y eficiencia sin precedentes. Sin embargo, la interconexión de las redes privadas de la subestaciones eléctricas a las redes públicas externas para habilitar su supervisión y control remoto, trajo consigo nuevos riesgos y vulnerabilidades en forma de ciberataques. Dentro de los distintos tipos de comunicaciones en SAS, los mensajes GOOSE y SV son utilizados por los dispositivos electrónicos inteligentes para intercambiar información de control y estado, así como las medidas digitalizadas de los valores de corriente y tensión eléctrica. Dada la importancia de este tipo de comunicaciones para el correcto funcionamiento de los sistemas de generación y distribución de energía, la protección de los mensajes GOOSE y SV toma especial importancia. Sin embargo, dados los estrictos requisitos temporales de estos mensajes, su protección supone un reto por resolver. En en Capítulo 2 de esta tesis, se evalúan las medidas de seguridad definidas por el estándar IEC 62351 para la protección de los mensajes IEC 61850 GOOSE y SV. Además, se analizan las propuestas presentadas por la comunidad investigadora y la industria para la protección de mensajes con requisitos de tiempo real en SAS. Los resultados muestran que en la actualidad no existe una solución que permita proporcionar de forma simultánea integridad, autenticidad y confidencialidad de las comunicaciones. Consecuentemente, el estudio de las vulnerabilidades en materia de ciberseguridad en SAS fue publicado en la revista científica Electronics en el año 2021.

2. **Estudio de algoritmos de criptografía de clave simétrica.**

   La primera versión del estándar de seguridad IEC 62351-6 hacía uso de algoritmos de criptografía de clave asimétrica para proteger las comunicaciones IEC 61850 GOOSE y SV. Sin embargo, tras atender las evidencias presentadas por la comunidad investigadora, que mostraban la inviabilidad de proteger estas comunicaciones sin comprometer sus requisitos temporales, el

IEC presentó una revisión del estándar IEC 62351-6:2020. El principal cambio introducido fue el uso de algoritmos de criptografía de clave simétrica, los cuales posibilitan acelerar los procesos criptográficos cuando se realizan avanzadas implementaciones hardware de los mismos. En en Capítulo 3 de esta tesis, se presenta el algoritmo AES-GCM, detallando sus fundamentos matemáticos, estructura, operaciones y funcionamiento lógico.

3. **SASCrypt: arquitectura hardware flexible para proteger mensajes GOOSE y SV en SAS.**

   A pesar de que se pueden encontrar propuestas en la literatura para proteger mensajes GOOSE y SV en SAS, ninguna de las soluciones identificadas cumplía con los requisitos de baja latencia, determinismo y alta capacidad de procesamiento de flujos de datos. En en Capítulo 4 de esta tesis, se presenta la arquitectura hardware SASCrypt para la protección de mensajes con requisitos de tiempo real en SAS. SASCrypt hace uso de avanzadas técnicas de diseño para, a través de una arquitectura flexible, proporcionar simultáneamente integridad, autenticidad y confidencialidad a los mensajes GOOSE y SV. EL diseño modular de la arquitectura propuesta permite soportar nuevos algoritmos criptográficos, proteger nuevos mensajes o actualizarse a futuras revisiones de los estándares de comunicaciones y seguridad en SAS. Como resultado, la arquitectura hardware propuesta fue publicada en la revista científica IEEE Access en el año 2021.

4. **IP AES-GCM adaptable, configurable y con una baja latencia, específicamente diseñado para su uso en SAS.**

   Tras realizar un estudio de los algoritmos de criptografía de clave simétrica definidos en la última revisión del estándar IEC 62351-6, en el Capítulo 4 de esta tesis se presenta una arquitectura de IP AES-GCM que sea adaptable y configurable, ofrezca latencia reducida y determinista para cada tamaño de trama a procesar. Consecuentemente, la solución propuesta permite adaptar la relación entre los recursos hardware necesarios para su implementación en el silicio y a la latencia requerida para el procesamiento de los datos, así como el volumen de datos que es capaz de procesar. La arquitectura de IP AES-GCM se detalló en un artículo de congreso que fue presentado en la conferencia DCIS en el año 2018.

5. **Validación experimental de la arquitectura SASCrypt.**

   La arquitectura SASCrypt que hace uso del IP AES-GCM presentado es validada mediante pruebas temporales y funcionales que son llevadas a cabo en el Capítulo 5 de esta tesis. Utilizando un banco de pruebas en el entorno de simulación de la herramienta Vivado, se realizan mediciones temporales

que permiten evaluar el retardo introducido por los procesos de protección de los mensajes GOOSE y SV. Los resultados muestran que la solución introduce una latencia determinista que es órdenes de magnitud inferior al requisito establecido. Mediante el uso de un entorno de pruebas específico para la medida de la latencia y haciendo uso de equipos de pruebas basados en la plataforma Zynq de Xilinx, se comprueba que existe una correlación entre las mediciones temporales realizadas en simulación y las mediciones temporales realizadas en pruebas físicas. Posteriormente, se evalúa el uso de recursos lógicos de la arquitectura SASCrypt y se demuestra la viabilidad de hacer un uso industrial de la solución propuesta en dispositivos SAS.

6.3 Publicaciones científicas

Esta sección presenta las publicaciones científicas que han sido llevadas a cabo durante el desarrollo de la tesis.

- M. Rodríguez, J. Lázaro, U. Bidarte, J. Jiménez and A. Astarloa. *"A Fixed-Latency Architecture to Secure GOOSE and Sampled Value Messages in Substation Systems"*, IEEE Access, vol. 9, pp. 51646-51658, 2021.
- Lázaro J, Astarloa A, Rodríguez M, Bidarte U, Jiménez J. *"A Survey on Vulnerabilities and Countermeasures in the Communications of the Smart Grid"*, Electronics, 10(16):1881, 2021.
- M. Urbina, A. Astarloa, J. Lázaro, U. Bidarte, I. Villalta and M. Rodriguez. *"Cyber-Physical Production System Gateway Based on a Programmable SoC Platform"*, IEEE Access, vol. 5, pp. 20408-20417, 2017.
- Urbina M, Moreira N, Rodriguez M, Acosta T, Lázaro J, Astarloa A. *"Secure Protocol and IP Core for Configuration of Networking Hardware IPs in the Smart Grid"*, Energies, 11(3):510, 2018.
- M. Rodríguez, A. Astarloa, J. Lázaro, U. Bidarte and J. Jiménez. *"System-on-Programmable-Chip AES-GCM implementation for wire-speed cryptography for SAS"*, 2018 Conference on Design of Circuits and Integrated Systems (DCIS), pp. 1-6, 2018.
- L. Muguira, J. Lázaro, S. Alonso, A. Astarloa and M. Rodriguez. *"Secure Critical Traffic of the Electric Sector over Time-Sensitive Networking"*, 2020 XXXV Conference on Design of Circuits and Integrated Systems (DCIS), pp. 1-6, 2020.
- A. Astarloa, M. Rodríguez, F. Duran, J. Jiménez and J. Lázaro. *"Synchro-*

nizing NTP Referenced SCADA Systems Interconnected by High-availability Networks", 2020 XXXV Conference on Design of Circuits and Integrated Systems (DCIS), pp. 1-6, 2020.

- J. Matias, M. Rodríguez, J. Garay and A. Astarloa. *"Solución para la Securización de Comunicaciones con Requisitos de Tiempo Real en Infraestructuras Críticas"*, Cuartas Jornadas Nacionales de Investigación en Ciberseguridad (JNIC), 2018.

6.4 Trabajo Futuro

En esta sección se presentan algunas lineas de investigación que darían continuidad al trabajo y los avances presentados en esta tesis:

- **Desarrollo del sistema de gestión y distribución de claves.**

  A pesar de que los algoritmos criptográficos de clave simétrica posibilitan conseguir acelerar los procesos criptográficos cuando se realizan avanzadas implementaciones hardware de los mismos, una parte de la complejidad se traslada al sistema de generación, gestión y distribución de claves. El estándar IEC 62351-9 define los elementos, protocolos, algoritmos y procesos que componen la cadena completa de gestión de claves criptográficas en SAS. Para poder realizar un uso industrial de la arquitectura SASCrypt en la red de una subestación eléctrica, es necesario que esta cuente con toda la infraestructura necesaria para que los dispositivos inteligentes que componen la red de comunicaciones puedan intercambiar las claves criptográficas de forma segura. Por lo tanto, sería interesante realizar un profundo estudio de los mecanismos de generación y distribución de claves definidos en IEC 62351-9, así como una propuesta para implementar dichas funciones en SAS.

- **Integración de la arquitectura hardware y un sistema de gestión de claves en un equipo final.**

  Puesto que en la última revisión del estándar IEC 62351-6:2020 se hace uso del algoritmo criptográfico AES-GCM, es necesario renovar las claves criptográficas que protegen las comunicaciones entre pares de equipos de forma periódica. Un uso prolongado de las claves criptográficas podría comprometer la seguridad de las comunicaciones. Para la validación de la arquitectura SASCrypt se han utilizado claves criptográficas estáticas. Sin embargo, la arquitectura propuesta ha sido diseñada para soportar una renovación periódica y dinámica de las claves criptográficas. Con el objetivo de contar

con una solución completa que pudiera ser utilizada en un uso industrial en SAS, sería necesario integrar en un equipo final la arquitectura SASCrypt junto con un sistema software de gestión de claves que hiciera de intermediario entre la arquitectura hardware y el sistema de distribución de claves dinámicas.

- **Validación de la solución completa en la red de una subestación.**

  Las contribuciones de esta tesis han sido validadas en entornos de simulación y dispositivos físicos en entornos de pruebas específicos que han permitido validar cada uno de los requisitos establecidos para garantizar la viabilidad de la solución en SAS. Sin embargo, sería interesante realizar una prueba completa de todos los elementos necesarios para proteger las comunicaciones IEC 61850 GOOSE y SV en la red de comunicaciones de una subestación. Para ello, sería necesario contar con un sistema de generación, gestión y distribución de claves criptográficas, así como dispositivos SAS que implementasen la arquitectura SASCrypt propuesta en esta tesis.

6.5 Agradecimientos

Este trabajo cuenta con el apoyo del Ministerio de Economía y Competitividad del Gobierno de España dentro del proyecto TEC2017-84011-R y fondos del Fondo Europeo de Desarrollo Regional (FEDER), el programa de Doctorados Industriales DI-15-07857 y el Departamento de Educación, Política Lingüística y Cultura del Gobierno Vasco dentro de los fondos asignados a los grupos de investigación del sistemas de universidades Vascas en el programa IT978-16.

Bibliografía

[1] IEC, "IEC 61850-1 ed2.0 Communicaiton networks and systems for power utility automation - Part 1: Introduction and overview," IEC, Technical, Mar. 2013. [Online]. Available: https://webstore.iec.ch/publication/6007

[2] International Electrotechnical Comission (IEC), "IEC 61850-6, Communication networks and systems for power utility automation - Part 6: Configuration description language for communication in power utility automation systems related to IEDs."

[3] G. N. Ericsson, "Cyber security and power system Communication—Essential parts of a smart grid infrastructure," *IEEE Trans. Power Delivery*, vol. 25, no. 3, pp. 1501–1507, Jul. 2010. [Online]. Available: https://doi.org/10.1109/tpwrd.2010.2046654

[4] K.-P. Brand, V. Lohmann, and W. Wimmer, *Substation automation handbook: Comprehensive description of substation automation and the coordination with network operation to obtain both performance and cost benefits by enabling enhanced power system management.* Bremgarten: Utility Automation Consulting Lohmann, 2003.

[5] P. Pruthvi, H. Bhuvaneswari, and L. Sudheendran, "Analysis of utility communication protocol IEC 61850 for substation automation systems," in *National Conference on Challenges in Research & Technology in the Coming Decades (CRT 2013)*. Ujire, India: Institution of Engineering and Technology, 2013, pp. 2.16–2.16. [Online]. Available: https://doi.org/10.1049/cp.2013.2504

[6] Abb, "ABB Review Special Report IEC 61850," ABB, Tech. Rep., 2010. [Online]. Available: https://library.e.abb.com/public/a56430e1e7c06fdfc12577a00043ab8b/3BSE063756_en_ABB_Review_Special_Report_IEC_61850.pdf

[7] I. Stellios, P. Kotzanikolaou, M. Psarakis, C. Alcaraz, and J. Lopez, "A survey of IoT-enabled cyberattacks: Assessing attack paths to critical infrastructures and services," *IEEE Commun. Surv. Tutorials*, vol. 20, no. 4, pp. 3453–3495, 2018. [Online]. Available: https://doi.org/10.1109/comst.2018.2855563

[8] S. Adepu, N. K. Kandasamy, J. Zhou, and A. Mathur, "Attacks on smart grid: Power supply interruption and malicious power generation," *Int. J. Inf. Secur.*, vol. 19, no. 2, pp. 189–211, Jul. 2019. [Online]. Available: https://doi.org/10.1007/s10207-019-00452-z

[9] T. Nguyen, S. Wang, M. Alhazmi, M. Nazemi, A. Estebsari, and P. Dehghanian, "Electric power grid resilience to cyber adversaries: State of the art," *#IEEE_O_ACC#*, vol. 8, pp. 87 592–87 608, 2020. [Online]. Available: https://doi.org/10.1109/access.2020.2993233

[10] R. Samikannu, V. Sampath Kumar, and J. Prasad, "A critical review of cyber security and cyber terrorism - threats to critical infrastructure in the energy sector," *IJCIS*, vol. 14, no. 2, p. 101, 2018. [Online]. Available: https://doi.org/10.1504/ijcis.2018.10013025

[11] R. P. Gupta, "Substation automation using IEC 61850 standard," in *Fifteenth National Power Systems Conference (NPSC), IIT Bombay*, Dec. 2008, pp. 462–466. [Online]. Available: http://www.krec.ir/Automation/Substation_Automation_Using_IEC61850_Standard.pdf

[12] R. Mackiewicz, "Overview of IEC 61850 and Benefits," in *2005/2006 PES TD*. Dallas, TX, USA: IEEE, 2006, pp. 376–383. [Online]. Available: http://ieeexplore.ieee.org/document/1668522/

[13] J. McGhee and M. Goraj, "Smart high voltage substation based on IEC 61850 process bus and IEEE 1588 time synchronization," in *2010 First IEEE International Conference on Smart Grid Communications*. IEEE, Oct. 2010, pp. 489–494. [Online]. Available: https://doi.org/10.1109/smartgrid.2010.5622092

[14] International Electrotechnical Comission (IEC), "IEC 61850-8-1, Communication 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."

[15] T. Bartman and K. Carson, "Securing communications for SCADA and critical industrial systems," in *2016 69th Annual Conference for Protective Relay Engineers (CPRE)*. IEEE, Apr. 2016. [Online]. Available: https://doi.org/10.1109/cpre.2016.7914914

[16] Y. Zhang, L. Wang, Y. Xiang, and C.-W. Ten, "Power system reliability evaluation with SCADA cybersecurity considerations," *IEEE Trans. Smart Grid*, vol. 6, no. 4, pp. 1707–1721, Jul. 2015. [Online]. Available: https://doi.org/10.1109/tsg.2015.2396994

[17] L. Briesemeister, S. Cheung, U. Lindqvist, and A. Valdes, "Detection, correlation, and visualization of attacks against critical infrastructure systems," in *2010 Eighth International Conference on Privacy, Security and Trust*. IEEE, Aug. 2010. [Online]. Available: https://doi.org/10.1109/pst.2010.5593242

[18] U. Carmo, D. H. Sadok, and J. Kelner, "IEC 61850 traffic analysis in electrical automation networks," in *2015 IEEE International Conference on Smart Grid Communications (SmartGridComm)*. IEEE, Nov. 2015. [Online]. Available: https://doi.org/10.1109/smartgridcomm.2015.7436344

[19] R. Schlegel, S. Obermeier, and J. Schneider, "A security evaluation of IEC 62351," *Journal of Information Security and Applications*, vol. 34, pp. 197–204, Jun. 2017. [Online]. Available: https://doi.org/10.1016/j.jisa.2016.05.007

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

[21] D. Kushner, "The real story of stuxnet," *IEEE Spectr.*, vol. 50, no. 3, pp. 48–53, Mar. 2013. [Online]. Available: https://doi.org/10.1109/mspec.2013.6471059

[22] A. Elgargouri, R. Virrankoski, and M. Elmusrati, "IEC 61850 based smart grid security," in *2015 IEEE International Conference on Industrial Technology (ICIT)*. IEEE, Mar. 2015. [Online]. Available: https://doi.org/10.1109/icit.2015.7125460

[23] N. R. Indukuri, "Layer 2 security for smart grid networks," in *2012 IEEE International Conference on Advanced Networks and Telecommunciations Systems (ANTS)*. IEEE, Dec. 2012. [Online]. Available: https://doi.org/10.1109/ants.2012.6524237

[24] C. Diago and A. Forshaw, "Cybersecurity for shared infrastructure substation networks with IEC 61850 GOOSE and sampled values," *J. eng.*, vol. 2018, no. 15, pp. 1195–1198, Aug. 2018. [Online]. Available: https://doi.org/10.1049/joe.2018.0150

[25] J. Hoyos, M. Dehus, and T. X. Brown, "Exploiting the GOOSE protocol: A practical attack on cyber-infrastructure," in *2012 IEEE*

Globecom Workshops. IEEE, Dec. 2012. [Online]. Available: https://doi.org/10.1109/glocomw.2012.6477809

[26] M. Kabir-Querrec, S. Mocanu, J.-M. Thiriet, and E. Savary, "A test bed dedicated to the study of vulnerabilities in IEC 61850 power utility automation networks," in *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, Verimag. Grenoble, France: IEEE, Sep. 2016. [Online]. Available: https://doi.org/10.1109/etfa.2016.7733644

[27] J. Noce, Y. Lopes, N. C. Fernandes, C. V. N. Albuquerque, and D. C. Muchaluat-Saade, "Identifying vulnerabilities in smart gric communication networks of electrical substations using GEESE 2.0," in *2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)*. IEEE, Jun. 2017. [Online]. Available: https://doi.org/10.1109/isie.2017.8001232

[28] J. G. Wright and S. D. Wolthusen, "Stealthy injection attacks against IEC61850's GOOSE messaging service," in *2018 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*. IEEE, Oct. 2018. [Online]. Available: https://doi.org/10.1109/isgteurope.2018.8571518

[29] M. Chlela, G. Joos, M. Kassouf, and Y. Brissette, "Real-time testing platform for microgrid controllers against false data injection cybersecurity attacks," in *2016 IEEE Power and Energy Society General Meeting (PESGM)*. IEEE, Jul. 2016. [Online]. Available: https://doi.org/10.1109/pesgm.2016.7741747

[30] N. Kush, M. Branagan, E. Foo, and E. Ahmed, "Poisoned GOOSE : Exploiting the GOOSE protocol," vol. 149, Jan. 2014.

[31] F. Zhang, M. Mahler, and Q. Li, "Flooding attacks against secure time-critical communications in the power grid," in *2017 IEEE International Conference on Smart Grid Communications (SmartGridComm)*. IEEE, Oct. 2017. [Online]. Available: https://doi.org/10.1109/smartgridcomm.2017.8340726

[32] Q. Li, C. Ross, J. Yang, J. Di, J. C. Balda, and H. A. Mantooth, "The effects of flooding attacks on time-critical communications in the smart grid," in *2015 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*. IEEE, Feb. 2015. [Online]. Available: https://doi.org/10.1109/isgt.2015.7131802

[33] S. M. S. Hussain, T. S. Ustun, and A. Kalam, "A review of IEC 62351 security mechanisms for IEC 61850 message exchanges," *IEEE Trans.*

Ind. Inf., vol. 16, no. 9, pp. 5643–5654, Sep. 2020. [Online]. Available: https://doi.org/10.1109/tii.2019.2956734

[34] Y. Yang, H. T. Jiang, K. McLaughlin, L. Gao, Y. Yuan, W. Huang, and S. Sezer, "Cybersecurity test-bed for IEC 61850 based smart substations," in *2015 IEEE Power & Energy Society General Meeting*. IEEE, Jul. 2015. [Online]. Available: https://doi.org/10.1109/pesgm.2015.7286357

[35] International Electrotechnical Comission (IEC), "IEC 62351-6, power systems management and associated information exchange - data and communications security - part 6: Security for IEC 61850," 2007.

[36] M. Daboul, J. Orságová, T. Bajanek, and V. Wasserbauer, "Testing protection relays based on IEC 61850 in substation automation systems," pp. 335–340, Jul. 2015.

[37] M. Braendle and F. A. F. Hohlbaum, "Cyber security practical considerations for implementing IEC 62351," https://library.e.abb.com/, 2010.

[38] D. Ishchenko and R. Nuqui, "Secure communication of intelligent electronic devices in digital substations," in *2018 IEEE/PES Transmission and Distribution Conference and Exposition (T&D)*. IEEE, Apr. 2018. [Online]. Available: https://doi.org/10.1109/tdc.2018.8440438

[39] S. M. Farooq, S. M. S. Hussain, and T. S. Ustun, "Performance evaluation and analysis of IEC 62351-6 probabilistic signature scheme for securing GOOSE messages," *#IEEE_O_ACC#*, vol. 7, pp. 32 343–32 351, 2019. [Online]. Available: https://doi.org/10.1109/access.2019.2902571

[40] International Electrotechnical Comission (IEC), "IEC 62351-6, Power systems management and associated information exchange - data and communications security - part 6: Security for IEC 61850," 2020.

[41] ——, "IEC 62351-9, Power systems management and associated information exchange - data and communications security - part 9: Cyber security key management for power system equipment," 2017.

[42] E. B. Kavun, N. Mentens, J. Vliegen, and T. Yalcin, "Efficient utilization of DSPs and BRAMs revisited: New AES-GCM recipes on FPGAs," in *2019 International Conference on ReConFigurable Computing and FPGAs (ReConFig)*. IEEE, Dec. 2019. [Online]. Available: https://doi.org/10.1109/reconfig48160.2019.8994730

[43] B. Buhrow, K. Fritz, B. Gilbert, and E. Daniel, "A highly parallel AES-GCM core for authenticated encryption of 400 gb/s network protocols," in *2015 International Conference on ReConFigurable Computing*

and FPGAs (ReConFig). IEEE, Dec. 2015. [Online]. Available: https://doi.org/10.1109/reconfig.2015.7393321

[44] M. Mozaffari-Kermani and A. Reyhani-Masoleh, "Efficient and high-performance parallel hardware architectures for the AES-GCM," *IEEE Trans. Comput.*, vol. 61, no. 8, pp. 1165–1178, Aug. 2012. [Online]. Available: https://doi.org/10.1109/tc.2011.125

[45] J. Daemen and V. Rijmen, "AES proposal: Rijndael," 1999.

[46] ——, "The block cipher rijndael," in *Smart Card Research and Applications*, J.-J. Quisquater and B. Schneier, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 277–284.

[47] D. McGrew, "Efficient authentication of large, dynamic data sets using Galois/Counter mode (GCM)," in *Third IEEE International Security in Storage Workshop (SISW'05)*. IEEE, 2005. [Online]. Available: https://doi.org/10.1109/sisw.2005.3

[48] P. H. W. Leong, "Recent trends in FPGA architectures and applications," in *4th IEEE International Symposium on Electronic Design, Test and Applications (delta 2008)*. IEEE, Jan. 2008, pp. 137–141. [Online]. Available: https://doi.org/10.1109/delta.2008.14

[49] L. Yizhen, L. Lin, and W. Jun, "The application of pipeline technology: An overview," in *2011 6th International Conference on Computer Science & Education (ICCSE)*. IEEE, Aug. 2011, pp. 47–51. [Online]. Available: https://doi.org/10.1109/iccse.2011.6028582

[50] Q. Liu, Z. Xu, and Y. Yuan, "A 66.1 gbps single-pipeline AES on FPGA," in *2013 International Conference on Field-Programmable Technology (FPT)*. IEEE, Dec. 2013, pp. 378–381. [Online]. Available: https://doi.org/10.1109/fpt.2013.6718392

[51] T. Chen, W. Huo, and Z. Liu, "Design and efficient FPGA implementation of ghash core for AES-GCM," in *2010 International Conference on Computational Intelligence and Software Engineering*. IEEE, Dec. 2010, pp. 1–4. [Online]. Available: https://doi.org/10.1109/cise.2010.5676905

[52] K. M. Abdellatif, R. Chotin-Avot, and H. Mehrez, "Efficient AES-GCM for VPNs using FPGAs," in *2013 IEEE 56th International Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE, Aug. 2013, pp. 1411–1414. [Online]. Available: https://doi.org/10.1109/mwscas.2013.6674921

[53] L. Henzen and W. Fichtner, "FPGA parallel-pipelined AES-GCM core for 100G Ethernet applications," in *2010 Proceedings of ESSCIRC*. IEEE,

Sep. 2010, pp. 202–205. [Online]. Available: https://doi.org/10.1109/esscirc.2010.5619894

[54] S. M. Farooq, S. S. Hussain, and T. S. Ustun, "S-GoSV: Framework for generating secure IEC 61850 GOOSE and sample value messages," *Energies*, vol. 12, no. 13, p. 2536, Jul. 2019. [Online]. Available: https://doi.org/10.3390/en12132536

[55] S. M. S. Hussain, S. M. Farooq, and T. S. Ustun, "Analysis and implementation of message authentication code (MAC) algorithms for GOOSE message security," *#IEEE_O_ACC#*, vol. 7, pp. 80 980–80 984, 2019. [Online]. Available: https://doi.org/10.1109/access.2019.2923728

[56] ——, "A method for achieving confidentiality and integrity in IEC 61850 GOOSE messages," *IEEE Trans. Power Delivery*, vol. 35, no. 5, pp. 2565–2567, Oct. 2020. [Online]. Available: https://doi.org/10.1109/tpwrd.2020.2990760

[57] M. Dworkin, E. Barker, J. Nechvatal, J. Foti, L. Bassham, E. Roback, and J. Dray, "Advanced encryption standard (AES)," Nov. 2001.

[58] S. Chandra, S. Paira, S. S. Alam, and G. Sanyal, "A comparative survey of symmetric and asymmetric key cryptography," in *2014 International Conference on Electronics, Communication and Computational Engineering (ICECCE)*. IEEE, Nov. 2014, pp. 83–93. [Online]. Available: https://doi.org/10.1109/icecce.2014.7086640

[59] S. Gueron and V. Krasnov, "Speeding up counter mode in software and hardware," in *2014 11th International Conference on Information Technology: New Generations*. IEEE, Apr. 2014, pp. 338–340. [Online]. Available: https://doi.org/10.1109/itng.2014.32

[60] M. J. Dworkin, "Recommendation for block cipher modes of operation," National Institute of Standards and Technology, Gaithersburg, MD, USA, Tech. Rep., Dec. 2012. [Online]. Available: https://doi.org/10.6028/nist.sp.800-38f

[61] D. Whiting, R. Housley, and N. Ferguson, "Counter with CBC-MAC (CCM)," *RFC*, vol. 3610, pp. 1–26, 2003.

[62] M. Bellare, P. Rogaway, and D. Wagner, "A conventional authenticated-encryption mode," 2003.

[63] T. Iwata and K. Kurosawa, "OMAC: One-key CBC MAC," in *Fast Software Encryption*, T. Johansson, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 129–153.

[64] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. CRC Press, Dec. 2018. [Online]. Available: https://doi.org/10.1201/9781439821916

[65] ARM Inc. (2010) AMBA® 4 AXI4-Stream Protocol, Version 1.0. [Online]. Available: https://static.docs.arm.com/ihi0051/a/IHI0051A_amba4_axi4_stream_v1_0_protocol_spec.pdf

[66] "IEEE standard for Ethernet," *IEEE Std 802.3-2018 (Revision of IEEE Std 802.3-2015)*, pp. 1–5600, 2018.

[67] N. Nalla Anandakumar, S. K. Sanadhya, and M. S. Hashmi, "FPGA-based true random number generation using programmable delays in oscillator-rings," *IEEE Trans. Circuits Syst. II*, vol. 67, no. 3, pp. 570–574, Mar. 2020. [Online]. Available: https://doi.org/10.1109/tcsii.2019.2919891

[68] S. Choi, Y. Shin, and H. Yoo, "Analysis of ring-oscillator-based true random number generator on FPGAs," in *2021 International Conference on Electronics, Information, and Communication (ICEIC)*. IEEE, Jan. 2021, pp. 1–3. [Online]. Available: https://doi.org/10.1109/iceic51217.2021.9369714

[69] A. Beal, J. Blakely, and N. Corron, "Driven ring oscillators as FPGA entropy sources," in *2019 SoutheastCon*. IEEE, Apr. 2019, pp. 1–6. [Online]. Available: https://doi.org/10.1109/southeastcon42311.2019.9020451

[70] M. Nursalman, A. Sasongko, Y. Kurniawan, and Kuspriyanto, "Improved generalizations of the karatsuba algorithm in GF(2^n)," in *2014 International Conference of Advanced Informatics: Concept, Theory and Application (ICAICTA)*. IEEE, Aug. 2014, pp. 185–190. [Online]. Available: https://doi.org/10.1109/icaicta.2014.7005938

[71] w. El hadj youssef, M. Machhout, M. Zeghid, B. Bouallegue, and R. Tourki, "Efficient hardware architecture of recursive karatsuba-ofman multiplier," in *2008 3rd International Conference on Design and Technology of Integrated Systems in Nanoscale Era*. IEEE, Mar. 2008, pp. 1–6. [Online]. Available: https://doi.org/10.1109/dtis.2008.4540262

[72] Z. Ge, G. Shou, Y. Hu, and Z. Guo, "Design of low complexity GF(2^m) multiplier based on karatsuba algorithm," in *2011 IEEE 13th International Conference on Communication Technology*. IEEE, Sep. 2011, pp. 1018–1022. [Online]. Available: https://doi.org/10.1109/icct.2011.6158033

[73] X. Fang and L. Li, "On karatsuba multiplication algorithm," in *The First International Symposium on Data, Privacy, and E-Commerce*

(ISDPE 2007). IEEE, Nov. 2007, pp. 274–276. [Online]. Available: https://doi.org/10.1109/isdpe.2007.11

[74] G. Zhou, H. Michalik, and L. Hinsenkamp, "Efficient and high-throughput implementations of AES-GCM on FPGAs," in *2007 International Conference on Field-Programmable Technology*, J. Becker, R. Woods, P. Athanas, and F. Morgan, Eds. Berlin, Heidelberg: IEEE, Dec. 2007, pp. 193–203. [Online]. Available: https://doi.org/10.1109/fpt.2007.4439248

[75] Xilinx, "Vivado design suite user guide," Online, Jun. 2021. [Online]. Available: https://www.xilinx.com/support/documentation/sw_manuals/xilinx2018_3/ug973-vivado-release-notes-install-license.pdf

[76] System-on-Chip Engineering, "SMARTzynq brick: Hsr/prp/ptp/gbe out-of-the-box networking solution," Online, Jun. 2021. [Online]. Available: https://soc-e.com/smartzynq-brick-full-working-solution-out-of-the-box/

[77] Xilinx, "Zynq-7000 SoC technical reference manual," Online, Jun. 2021. [Online]. Available: https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf

[78] Wireshark, "Wireshark user's guide," Online, Jun. 2021. [Online]. Available: https://www.wireshark.org/docs/wsug_html/

[79] S. Lemsitzer, J. Wolkerstorfer, N. Felber, and M. Braendli, "Multi-gigabit GCM-AES architecture optimized for FPGAs," in *Cryptographic Hardware and Embedded Systems - CHES 2007*, P. Paillier and I. Verbauwhede, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 227–238.

[80] System-on-Chip Engineering, "PreciseTimeBasic: Ieee 1588-2008 PTPv2 IP core," Online, Jun. 2021. [Online]. Available: https://soc-e.com/products/precisetimebasic-ieee-1588-2008-v2-ptp-ip-core/

[81] *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, IEEE Instrumentation and Measurement Society Std., 2008.

[82] System-on-Chip Engineering, "HPS: Hsr-prp switch IP core," Online, Jun. 2021. [Online]. Available: https://soc-e.com/products/hsr-prp-switch-ip-core-all-hardware-low-latency-switch-for-fpgas/

[83] *IEC 62439-3 ed2.0 Industrial communication networks - High availability automation networks - Part 3: Parallel Redundancy Protocol (PRP) and High-availability Seamless Redundancy (HSR)*, Std., Rev. ed2.0, Jul. 2012. [Online]. Available: http://webstore.iec.ch/Webstore/webstore.nsf/Artnum_PK/46615