Akseli Nousiainen

Applying cybersecurity for IEC 60870-5-104 communication between control station and substation

School of Technology and Innovations
Master's thesis in Electrical Engineering
Energy and Information Technology, M.Sc. (Tech.)

Vaasa 2024

UNIVERSITY OF VAASA
School of Technology and Innovations
Author: Akseli Nousiainen
Title of the Thesis: Applying cybersecurity for IEC 60870-5-104 communication between control station and substation
Degree: Master of Science in Technology
Programme: Electrical Engineering
Supervisor: Kimmo Kauhaniemi
Instructor: Jere Mäki
Evaluator: Mazaher Karimi
Year: 2024 **Number of pages:** 84

ABSTRACT:
IEC 60870-5-104 is a communication protocol used in telecontrol of electric power systems. Despite the critical nature of electric power systems IEC 60870-5-104 itself does not implement security measures such as encryption and authentication making it vulnerable for cybersecurity threats such as eavesdropping and spoofing. The intention of this thesis is to investigate three techniques to provide security for IEC 60870-5-104 communication in different layers of networking and data communication. The investigated techniques are IEC 62351-3 standard providing TLS protection for IEC 60870-5-104, IPsec VPN tunnel, and an application layer authentication mechanism for IEC 60870-5-104 defined by IEC 62351-5 standard.

Investigations were based on literature review and empiric testing. Literature review focused on standards and definitions defining the protection mechanisms and IEC 60870-5-104 with the support of field specific literature. Also researches regarding attack simulations against IEC 60870-5-104 were studied. In the empiric investigations all protection mechanisms were tested and demonstrated using a SCADA server running on a virtual machine and a physical RTU. Wireshark packet analyzer was used for analyzing traffic when IEC 60870-5-104 communication was protected using the investigated techniques. An understanding was formed how IEC 60870-5-104 can be protected on different layers of networking and data communication using the investigated protection techniques. Each technique acts independently from each other enabling protection of IEC 60870-5-104 communication in multiple layers.

Solutions to be used for protecting IEC 60870-5-104 depends on the protection requirements, network architectural restrictions, and support of the used equipment. Important aspects regarding protection of transferred data is to provide end-to-end protection between endpoints and to isolate access to sensitive control system endpoints from untrusted networks directly or indirectly. TLS provides protection for transferred application data but does not provide network level isolation of endpoint hosts. Therefore, the most suitable use case for TLS would be to protect IEC 60870-5-104 communication in a trusted network. The advantage of an IPsec tunnel is the isolation of endpoint networks and data exchanged between them making it most suitable for protecting T104 communication in untrusted networks. IEC 62351-5 is not a networking technique but a set of functionalities added in the application layer of IEC 60870-5-104. As IEC 62351-5 is implemented in the application layer it can provide security measures which cannot be achieved by the protection techniques affecting in the lower layers. The most important security measure that IEC 62351-5 adds is linking and authenticating application layer users between a controlling station and a controlled station which provides security for application processes.

KEYWORDS: telecontrol system, IEC 60870-5-104, man-in-the-middle attack, end-to-end protection, authentication

VAASAN YLIOPISTO
Tekniikan ja innovaatiojohtamisen yksikkö
Tekijä: Akseli Nousiainen
Tutkielman nimi: Tietoturvan soveltaminen valvomon ja ala-aseman väliseen IEC 60870-5-104 -kommunikaatioon
Tutkinto: Diplomi-insinööri
Ohjelma: Sähkötekniikka
Työn valvoja: Kimmo Kauhaniemi
Työn ohjaaja: Jere Mäki
Tarkastaja: Mazaher Karimi
Vuosi: 2024 **Sivumäärä:** 84

TIIVISTELMÄ:
IEC 60870-5-104 on sähkönjakelujärjestelmien kaukokäytössä käytetty kommunikaatioprotokolla, joka ei itsessään toteuta tietoturvamekanismeja kuten salausta tai autentikaatiota. Tämä tekee siitä haavoittuvan erinäisille tietoturvauhkille, esimerkiksi salaamattoman kommunikaation sisältöä voidaan salakuunnella ja viestien alkuperä väärentää. Tämän diplomityön tarkoituksena on tutkia kolmea tekniikkaa IEC 60870-5-104 -kommunikaation tietoturvan parantamiseksi, joista kukin vaikuttaa tiedonsiirron eri kerroksissa. Tarkastellut menetelmät ovat IEC 62351-3 -standardin tarjoama TLS-suojaus IEC 60870-5-104 -protokollalle, IPsec VPN -tunnelointi sekä IEC 62351-5 -standardin määrittelemä autentikaatiomekanismi IEC 60870-5-104 -protokollalle.

Tutkimusmenelminä käytettiin kirjallisuuskatsausta sekä empiiristä tutkimusta. Kirjallisuuskatsauksessa perehdyttiin tietoturvatekniikoita ja IEC 60870-5-104 -protokollaa määritteleviin standardeihin sekä käytettiin tukena alan kirjallisuutta. IEC 60870-5-104 -protokollan tietoturvaan perehdyttiin hyökkäyssimulaatioita käsittelevien tutkimusten kautta. Empiirisissä tutkimuksissa jokaista tietoturvatekniikkaa testattiin ja demonstroitiin virtuaalialustalla pyörivän SCADA-palvelimen ja fyysisen RTU-laitteen muodostamassa testiympäristössä. Wireshark-pakettianalysaattoria käytettiin laitteiden välisen IEC 60870-5-104 -kommunikaation suojauksen analysointiin. Diplomityön tuloksena saavutettiin käsitys siitä, miten tutkituilla suojaustekniikoilla voidaan suojata IEC 60870-5-104 -kommunikaatiota tiedonsiirron eri kerroksissa. Kukin tekniikka on toisistaan riippumaton, mikä mahdollistaa IEC 60870-5-104 -kommunikaation suojauksen usealla eri tiedonsiirron tasolla.

Kommunikaation suojaukseen käytetyn tekniikan valinta riippuu suojauksen vaatimuksista, verkkoarkkitehtuurisista rajoitteista sekä laitteiston tukemista tekniikoista. Tärkeä näkökulma koskien siirrettävän tiedon suojausta on suojauksen toteuttaminen koko matkalta lähettäjältä ja vastaanottajalle. Lisäksi kriittisiin ohjausjärjestelmiin ei tulisi olla suoraa tai epäsuoraa yhteyttä epäluotettavista verkoista. TLS tarjoaa applikaatiodatan suojausta, mutta ei luo suojaavaa verkkokerrosta kommunikoiville laitteille. Siten TLS soveltuu parhaiten IEC 60870-5-104 -kommunikaation suojaamiseen luotetuissa verkoissa. IPsec-tunnelointi puolestaan luo suojaavan verkkokerroksen kahdelle yhdistettävälle sisäverkolle ja niiden väliselle tiedonsiirrolle yhdistävän verkon näkökulmasta soveltuen tiedonsiirron suojaukseen avoimissa verkoissa. IEC 62351-5 -standardin implementaatio ei ole tiedonsiirtotekniikka vaan applikaatiotason tietoturvaa parantava toiminnallisuus, jolla voidaan linkittää ja autentikoida valvomo- ja ala-asemalaitteiden välisiä käyttäjiä, mitä ei voida toteuttaa matalammilla tiedonsiirtokerroksilla.

AVAINSANAT: kaukokäyttöjärjestelmä, IEC 60870-5-104, väliintulohyökkäys, end-to-end-suojaus, autentikaatio

Contents

Figures

Abbreviations

AES	Advanced Encryption Standard
AH	Authentication Header
APCI	Application Protocol Control Information
APDU	Application Protocol Data Unit
ARP	Address Resolution Protocol

ARPANET	Advanced Research Projects Agency Network
ASDU	Application Service Data Unit
CA	certificate authority
ECDHE	Elliptic Curve Diffie-Hellman Ephemeral
ESP	Encapsulating Security Payload
GCM	Galois/Counter Mode
HMAC	hash-based message authentication code
IEC	International Electrotechnical Commission
IEC TC 57	IEC Technical Committee 57
IKE	Internet Key Exchange
IP	Internet Protocol
IPsec	Internet Protocol Security
ISAKMP	Internet Security Association and Key Management Protocol
ISO	International Organization for Standardization
LAN	local area network
MAC	message authentication code
MAC (address)	media access control (address)
OSI	Open Systems Interconnection
RFC	Request for Comments
RSA	Rivest-Shamir-Adleman
RTU	remote terminal unit
SCADA	supervisory control and data acquisition
SHA	Secure Hash Algorithm
SKEME	Secure Key Exchange Mechanism
SKEYID	secret key ID
STARTDT	start data transfer
STOPDT	stop data transfer
TCP	Transmission Control Protocol
TESTFR	test frame
TLS	Transport Layer Security
T101	IEC 60870-5-101
T104	IEC 60870-5-104
UDP	User Datagram Protocol
VPN	Virtual Private Network
WAN	wide area network

1 Introduction

This thesis was done as an assignment of Services department of Grid Automation business unit of Hitachi Energy in Vaasa. Services department is involved with delivery and maintenance of SCADA telecontrol systems built on their own MicroSCADA software products. Typical customers are for example electricity network service providers, energy production service providers, and industrial factories. Typical delivery projects include design, configuration, and commissioning of MicroSCADA telecontrol systems including control station and substation devices. Such projects include deliveries of systems to new infrastructures, extensions of existing systems, and system updates. Maintenance operation focuses on operative MicroSCADA telecontrol systems and includes for example troubleshooting and support in problem situations, fixing faults, and extensions or updates of individual components for running systems.

Delivery projects and maintenance operations provide cybersecurity measures and services as well. Certain cybersecurity measures are implemented by default when new systems are configured, and customers might have their own requirements regarding cybersecurity of an ordered system. Maintenance operation provides also certain cybersecurity services, such as security updates for running systems, and customers might have desires to develop cybersecurity of their systems which need to be met.

This thesis handles cybersecurity of telecontrol systems of electricity distribution networks and focuses on TCP/IP based IEC 60870-5-104 protocol used for operational communication between a control station and a substation. This contains for example telecontrol commands sent from the control station to the substation and telemetry data sent from the substation to the control station. Originally IEC 60870-5-104, or T104 as termed by Clarke & Reynders (2004, p. 174), does not have built-in security measures such as encryption, authentication, and integrity protection. Due to the critical and sensitive nature of telecontrol systems and electric distribution networks this communication should be protected. If an intruder manages to interfere with the network traffic, he might be able to access sensitive information by eavesdropping or even manipulate

messages of T104 communication. Afterwards International Electrotechnical Commission (IEC) has defined a series of standards IEC 62351 to provide security for power system communication protocols defined by IEC Technical Committee 57 (TC 57), including IEC 60870-5 based protocols IEC 60870-5-101 (abbreviated as T101 later in the text) and T104 (IEC, 2007, p. 6, 8).

IEC 62351-3 standard defines deployment of TLS protection for TCP/IP based telecontrol protocols defined by TC 57 providing them authentication, encryption, and integrity protection (IEC, 2014, p. 5). IEC (2007, p. 26) says that initially also another protection technique called IPsec was evaluated to be used instead of TLS but TLS was seen more suitable for the purpose. IPsec provides similar security measures as TLS but acts on a lower layer of networking and data communication. IEC has also developed an application layer security measure in its standard IEC 62351-5 which defines procedures for authenticating application layer messages of IEC 60870-5 based protocols (IEC, 2013a, p. 8, 17−18).

TCP/IP based networking composes of several protocols co-operating hierarchically in distinctive layers of networking and data communication. Concept of layering can be understood via the OSI model which structures networking and data communication in seven layers and gives abstract descriptions of duties and interoperation of the layers. In practice the layered structure of TCP/IP based networking is implemented through the concrete TCP/IP protocol suite which composes of four distinctive layers each containing specific protocols. Protocols on different layers have different duties in transmission of messages from an endpoint to another and they are used in conjunction with each other to implement the whole chain of end-to-end data transmission.

This thesis investigates three cybersecurity techniques provided by MicroSCADA X SYS600 SCADA server software and RTU560 remote terminal unit of Hitachi Energy to protect T104 communication between a controlling station and a controlled station. The techniques are the implementation of IEC 62351-3 standard, IPsec Virtual Private Network (VPN) tunnel, and the implementation of IEC 62351-5. Both SYS600 and RTU560

implement IEC 62351-3 and IEC 62351-5. RTU560 implements IPsec VPN as well. Each protection technique provides security measures on different layers of networking and data communication. The objective is to gain answers for the following questions.

- What kind of cybersecurity deficiencies T104 protocol has?
- What kind of cybersecurity threats can be targeted against T104 communication between a controlling station and a controlled station?
- What kind of protection each investigated technique provides for T104 communication between a controlling station and a controlled station?
- What is the scope of protection of each technique regarding the layered structure of TCP/IP based communication?

The investigations were conducted combining literature review and empiric investigations. Conception of operation of each technique was obtained through their technical specifications and field specific literature. Cybersecurity aspect was examined by reviewing studies found from the literature about attack simulations against T104 communication between a controlling and a controlled station. Inspections of attack simulations were confined to a situation in which an attacker has penetrated into a control system network with the possibility to communicate with the endpoints and interfere with T104 communication between them. Empiric investigations were conducted to demonstrate and support the information acquired as the result of the literature review. Each technique was tested and demonstrated to protect T104 communication in a test setup consisting of a controlling station running MicroSCADA X SYS600 and a controlled station running an RTU560. At the end conclusions are made considering the protection of the investigated techniques against the potential attack scenarios presented in the literature review.

The structure of the thesis is the following. Chapter 2 concentrates on basics of networking and data communication by introducing the OSI model and the TCP/IP protocol suite. Focus regarding the OSI model is on the concept of layering whereas focus on the TCP/IP

protocol suite is to introduce basics of TCP/IP based networking. Chapter 3 presents the principles of protection methods TLS and IPsec. Beside the introduction of TLS also common cryptographic techniques are presented which are also deployed by IPsec and IEC 62351-5. Security measures provided by both TLS and IPsec are introduced comprehensively however the focus is on authentication and integrity protection. Encryption techniques are not investigated as granularly but encryption is assumed to take place and provide protection against eavesdropping attacks.

Chapter 4 focuses on the telecontrol protocol T104 under inspection. A review of cybersecurity deficiencies of T104 is presented in Chapter 5. Possible attack scenarios are also considered focusing on eavesdropping and man-in-the-middle attacks against T104 communication. After this IEC 62351-3 and IEC 62351-5 standards are introduced in Chapter 6. Theory of TLS given in Chapter 3 acts as the basis for IEC 62351-3. After the underlying theory has been covered Chapter 7 presents practical demonstrations of the investigated techniques for protecting T104 communication.

2 Principles of networking and data communication

Forouzan and Fegan (2007, p. 1, 4−5, 27) summarize data communication as a way of exchanging information in a common format through a physical transmission medium between two entities consisting of hardware and software components. Information presented in a common format is called data which is transmitted through the physical medium between the endpoints in physical signals. A protocol is a set of rules that govern data communication between the communicating entities. It defines the common presentation format of the data and common practices for transmitting it. Networking provides facilities and functionalities for data communication between multiple endpoint and intermediary node devices consisting of hardware and software components which are interconnected with each other via physical channels. As data is sent from an endpoint application to another through the physical media it is processed through multiple layers in between that provide different networking services required to get the data to its destination.

This chapter introduces two fundamental concepts for understanding and structuring networking and data communication, the OSI model and the TCP/IP protocol suite. The OSI model is an abstract model specifying principles for open data communication of interconnected computing devices. Zimmermann (1980, p. 425), who was involved with the initial development of the OSI standards, tells that International Organization of Standardization (ISO) developed the OSI model initially in 1977−1979 as the need for standardizing communication in computer networks increased due to the wide range of manufacturer specific solutions. The intention was to build a model to be used as the basis for standardizing the design of communication protocols. Tanenbaum and Wetherall (2011, p. 41−45, 49−53) say that the model was carefully defined based on layered structure starting from the physical layer considering the physical medium and transfer of data in physical form ending up to the topmost application layer serving an application process and a possible end user on the end device. However, the model was complex and impractical to implement and it was never taken widely into use in practice.

Nevertheless, the model has remained as a reference for presenting and understanding networking and data communication.

Tanenbaum and Wetherall (2011, p. 41, 45−46, 53) say that conversely TCP/IP model and its implementation, the TCP/IP protocol suite, became widely popular in practice and is today the foundation of computer networking and the Internet. The trajectory of TCP/IP protocol stack is opposite to the one of OSI model. TCP/IP protocol stack was initially developed in the early 1970s for communication used in a research network ARPANET developed by the United States Department of Defense. The need was to interconnect several networks distributed in geographically large areas and achieve redundancy of communication by being able to reach endpoints via multiple routes. A stack of specific protocols was developed to serve the purpose. However, TCP/IP model is not a fundamental description of how protocols should be implemented in a standardized manner.

2.1 Introducing the OSI model

Zimmermann (1980) provides an overview of the Open Systems Interconnection (OSI) reference model. The OSI model is based on a layered structure consisting of seven layers each having their independent responsibilities in networking and data communication. The structure and layers of the OSI model are presented in Figure 1. Considering a networking system, the principle is that layers co-operate vertically so that a lower layer always serves its upper layer and the upper layer “adds value” (Zimmermann, 1980, p. 426) for the service provided by the lower layer. Finally, the topmost layer, application layer, serves an application process. Considering networking peer systems, implementations of a specific layer communicate horizontally together acting as peers. Another principle of the OSI model is its concern of the external behavior of a communicating entity, i.e., how it is seen by other communicating entities. OSI model is not concerned of internal implementations of a layer but specifies formally what kind of services a layer must provide to serve its upper layer.

The basis of the OSI reference model is defined in ISO/IEC standard 7498-1. ISO/IEC (1994, p. 30, 32−37) defines the outline of each layer describing its purpose, services, and functionalities to be implemented to serve the layer above, and how the layer utilizes services provided by the layer below. The three highest layers are related to handling end-to-end communication of application processes. The highest layer is the application layer which provides for an application process and a possible end user access to networking facilities provided by the lower layers and finally to communicate with other peer application processes and end users using an application layer protocol. Presentation layer beneath the application layer is responsible for converting the application specific syntaxes of application layer data into general formats that can be used in data transfer between application processes. However, original application syntaxes must be preserved. This leaves application layer entities independent from the general transfer syntax and enables variance in application layer syntaxes. Encryption of data can also be performed in the presentation layer (Cloudflare, n.d.-a; Miller, 2021, p. 4). Session layer establishes and maintains communication of presentation layer entities which enables dialogues for application layer processes, meaning for example synchronization and ordering of data exchange (ISO/IEC, 1994, p. 35).

ISO/IEC (1994, p. 37−52) defines that four lowest layers are responsible for the transmission of data between endpoints. Transport layer is responsible for dividing the data exchanged by upper layer sessions into transferrable units and managing a reliable transmission between endpoints which appears transparent for the session layer entities. Beneath transport layer operates the network layer which is responsible for routing transport layer packets between different networks. Therefore, devices performing network layer tasks do not have to be endpoints of communication but can also be intermediate nodes that forward traffic towards the correct destination. Such nodes do not have to implement services of higher layers as presented in Figure 1. Data link layer is below the network layer and establishes links of physical connections which the network layer can use to define routes between devices. Data link layer provides error detection and error recovery for the physical layer which underlies the whole infrastructure of

networking and data communication. Physical layer is responsible for physical transfer of data. It determines requirements and definitions of the transfer medium and the presentation of data in physical signals.

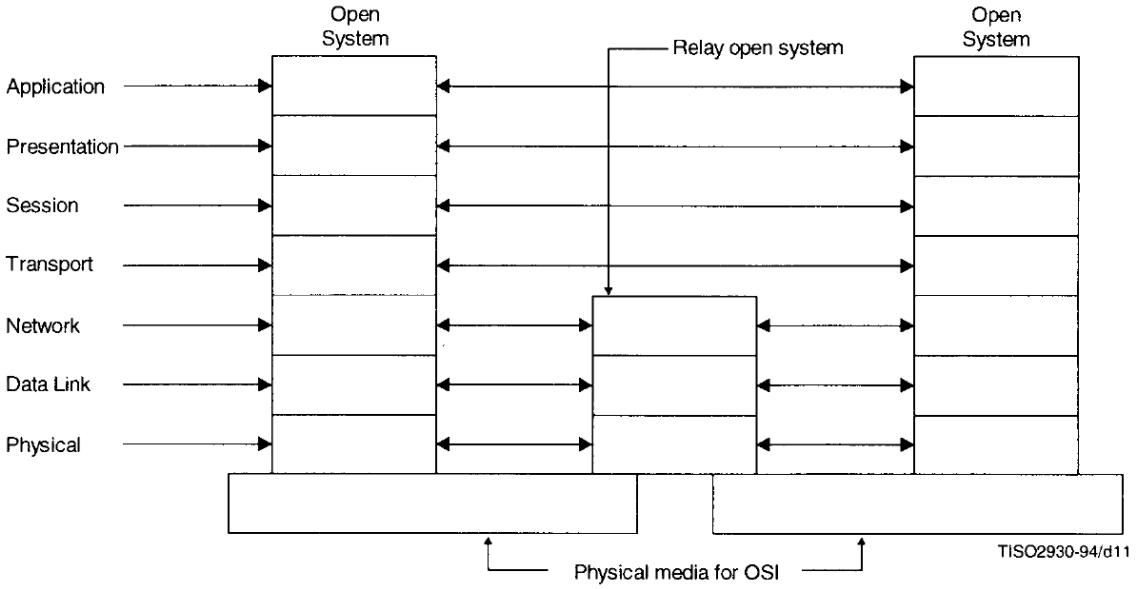


Figure 1. Structure of the OSI model including a relaying open system. (ISO/IEC, 1994, p. 29)

2.2 Relevant concepts of the TCP/IP protocol suite

A concrete picture of networking and data communication according to the TCP/IP protocol suite is given by Forouzan (2010, p. 29−32, 35−37). TCP/IP protocol suite has also a layered structure, originally consisting of four layers: data link layer, network layer, transport layer, and application layer. However, nowadays also physical layer is conceived as a part of the TCP/IP suite. TCP/IP suite is actually a collection of several protocols operating on different layers. Beneath the communication underlies the physical layer which comprises the physical medium used for connecting devices and transferring data between them in bits presented as physical signals. Data link layer turns the physical matter into links which enable logical data communication. Data link layer interprets bit sequences as frames that can be transferred between devices through the links. Different devices, also called nodes, connected via the links are identified by physical addresses meaning MAC (media access control) addresses in practice. TCP/IP protocol suite

does not define protocols for physical and data link layers but supports standard technologies.

Forouzan (2010, p. 32−33, 37−38) explains that above the data link layer is the network layer which provides a layer of abstraction for defining individual endpoints and routes between them. Network layer provides interconnection of multiple networks each consisting of endpoint devices and communication links. Physical addressing identifies nodes of data links but network layer identifies endpoint devices individually among interconnected networks by logical addressing meaning IP addressing in practice. Each network has its own individual IP address space and data is sent to the destination according to its unique IP address. If the destination resides in another network data is sent to a router which acts as an interconnecting node between separate networks and directs the traffic to the correct network or to another router towards the correct destination. Therefore, all nodes need to have an IP address. At network layer data is handled and transferred in datagrams using the Internet Protocol (IP).

According to Forouzan (2010, p. 32−35) data at transport layer is exchanged between end application processes in packets. Packets are sliced into datagrams to be transferred at network layer in a random order and through different routes. Transport layer gathers and organizes datagrams into packets and delivers packets for the end application process. Typical protocols used at transport layer are Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). Application processes listen behind TCP or UDP ports and port numbering is used for directing a packet to a specific port for a specific application process. Furthermore, the application process behind the port has its own application layer protocol with its own addressing system. Application layer data is sent in messages.

Forouzan (2010, p. 36−40) explains that protocols operate in a layered manner. Each layer encapsulates a data unit of the upper layer inside a header and possibly a trailer which contain layer specific information regarding the data unit to be transferred. For

example, a TCP packet with transport layer source and destination ports of the end application processes is encapsulated in a network layer header containing the IP addresses of the endpoints. When the packet goes to data link layer new header and trailer are added which contain the physical addresses of the source and destination nodes so that the data can be transferred between them. Figure 2 presents delivery of an application layer message. Each layer encapsulates the data unit of the upper layer with layer specific information to be handleable within the layer and finally data is sent in physical signals to the receiver. At the receiver's end packets are decapsulated by each layer and passed to the upper layer finally ending up to the application layer.

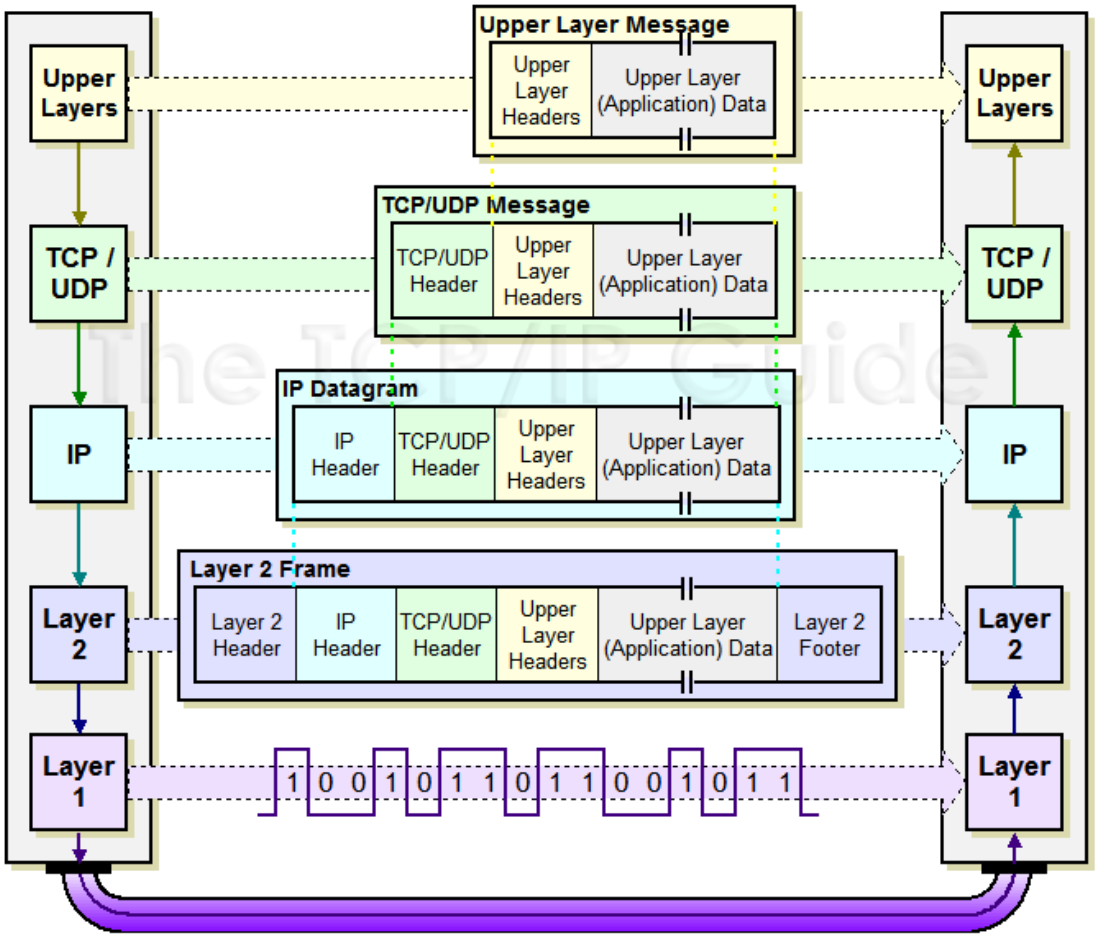


Figure 2. Diagram illustrating encapsulation in different layers of the TCP/IP suite. (Kozierok, 2005)

2.3 Conclusions regarding this thesis

It should be noted that the OSI model and the TCP/IP protocol suite represent two different concepts: the OSI model is an abstract model whereas the TCP/IP protocol suite is an implementation of protocols operating in a layered manner. In the context of this thesis what should be considered regarding the OSI model is the concept of layering. Also, some standards and literature addressed in this thesis refer to the OSI model and it is therefore relevant. However, networking and data communication presented in this thesis operate on the TCP/IP suite and therefore the practical substance it provides is essential for the objectives of the thesis.

Similarities exist between the structure and functionalities of the models. Both structures define application, transport, and network layers, and regarding physical and data link layer the TCP/IP protocol suite identifies them whereas does not specify them. The three application layers of the OSI model can be seen combined in implementations of the application layer in the TCP/IP suite but are not specified distinctively (Forouzan, 2010, p. 29−30). Nevertheless, most relevant layers regarding the protection techniques and T104 protocol presented in this thesis are the network, transport, and application layers which are similar in both models. Also, presentation layer from the OSI model should be noted as it is seen relevant regarding TLS.

3 General methods for protecting TCP/IP based traffic

Target of this chapter is to introduce two common techniques, TLS and IPsec, for protecting security objectives of networking and data communication. Three such objectives are confidentiality, integrity, and authenticity of data transferred between communicating parties. Confidentiality means that unauthorized parties are not able to read transferred data (Frankel et al., 2008, p. 2-3). Also, term privacy is used in some sources referring to the same objective (Rescorla & Dierks, 2008, p. 4). Frankel et al. (2008, p. 2-3, 3-3) say that authentication requires that the identities of the communicating parties are verified. Integrity protection inspects whether data has altered during transmission.

Confidentiality or privacy is protected with encryption. In encryption plain text is processed to an unreadable form by an algorithm which uses an encryption key to produce a ciphertext which cannot be decrypted without an appropriate decryption key (Cloudflare, n.d.-b). Authentication is commonly implemented through public-key cryptography (Rescorla & Dierks, 2008, p. 4) and certificates which aim to ensure the authenticity of public keys (Fortinet, n.d.-a). Pre-shared secret symmetric keys can also be used for authentication (Forouzan, 2008, p. 569). Bellare et al. (1996, p. 3) explain that integrity is commonly verified using a technique called message authentication code (MAC) which calculates a digest of transferred data. By calculating a MAC in both ends of communication and comparing the results it can be checked if the original message has remained unaltered during the transmission.

3.1 Protecting application layer data with TLS

Transport Layer Security (TLS) is a protocol which provides security features to protect privacy, integrity, and authenticity of transferred data (Rescorla & Dierks, 2008, p. 4). TLS encapsulates data provided by the application layer (Forouzan, 2008, p. 508) and TLS packets act as a payload for a reliable transport layer protocol such as TCP (Forouzan, 2008, p. 509; Rescorla & Dierks, 2008, p. 4). TLS is independent of the higher-level application protocol it encapsulates and can be implemented by any application layer

protocol (Forouzan, 2008, p. 508; Rescorla & Dierks, 2008, p. 5). Ristić (2014, p. 3) locates TLS into the presentation layer of the OSI model. This section gives an overview of the operation of TLS 1.2 defined in RFC 5246.

3.1.1 Encryption and key exchange

Encryption is used for providing privacy of transferred data. Ristić (2014, p. 5, 12–13) explains that TLS uses asymmetric cryptography to negotiate a symmetric key used for performing the encryption of data. Asymmetric cryptography is also known as public-key cryptography. It is based on two separate keys: one of the encryption keys is called private key and the other is called public key. The keys are related to each other mathematically so that data encrypted with the private key can only be decrypted with the public key, and vice versa. The private key must be kept hidden from external parties but the public key can be shared publicly. Conversely, in symmetric cryptography the same key is used in both ends and it must be kept safe from external parties.

Asymmetric encryption enables exchange of secrets in public infrastructures such as the Internet. However, symmetric cryptography is faster and not as laborious as asymmetric cryptography (Ristić, 2014, p. 13). Therefore, asymmetric cryptography is used in TLS to share parameters for calculating a master secret between communicating parties which is used for creation of symmetric encryption keys (Rescorla & Dierks, 2008, p. 26; Ristić, 2014, p. 35, 48–49). Master secret is derived from a random value provided by the client, a random value provided by the server, and a premaster secret which are exchanged in the TLS handshake (Rescorla & Dierks, 2008, p. 64; Ristić, 2014, p. 48–49). The master secret is further used for deriving encryption keys (Rescorla & Dierks, 2008, p. 25–26).

The derivation of the premaster secret depends on the key exchange algorithm. Two common algorithms to exchange the premaster secret are the RSA and Diffie-Hellman algorithms which are shortly introduced here. Ristić (2014, p. 38) explains that in RSA key exchange the client generates the premaster secret and encrypts it with the public RSA key of the server. The encrypted premaster secret is then shared with the server

which decrypts it using its private RSA key. After exchange of the random values and the premaster secret the master secret can be calculated in both ends. The mathematics behind the RSA algorithm make decryption of the ciphertext infeasibly complex without knowing the private key (Forouzan, 2008, p. 301). However, the weakness of RSA key exchange is that the RSA keys are static and if someone gets access to the private key of the server he can reveal the premaster secret and calculate the master secret (Ristić, 2014, p. 38).

Forouzan (2008, p. 447–449) explains that Diffie-Hellman algorithm allows two parties to independently calculate a shared secret using public parameters with certain mathematical characteristics and a random private parameter kept secret from others. Using the commonly agreed public parameters and the individual private parameter, each party calculates independently a so-called half-key. Due to the mathematical characteristics of the public parameters the half-keys have a mathematical connection. Parties exchange the half-keys with each other and each party can calculate independently the same result using the half-key of the other party and its own private parameter. Diffie-Hellman algorithm has the characteristic that reversing the calculation, i.e., solving the private parameters from the half-keys, is troublesome (Forouzan, 2008, p. 450; Ristić, 2014, p. 39).

Using earlier exchanged client and server random values and the common premaster secret derived through Diffie-Hellman algorithm the master secret can be calculated in both ends (Cloudflare, n.d.-c). Ristić (2014, p. 39, 257) tells that Diffie-Hellman parameters can be fixed but it is recommended to use Diffie-Hellman key exchange algorithms which provide forward secrecy. This means that each session uses unique keys and if keys of one session are compromised the keys of the earlier sessions are not.

3.1.2 Authentication

Authentication is incorporated with the key exchange and is based on public key cryptography (Ristić, 2014, p. 41). Stallings (2017, p. 457–458) explains how certificates are

used for authenticating public keys. An authenticating party shares its public key to the other via a certificate. A certificate is used for binding a public key to an identity and verifying the authenticity of its owner. It is issued by a party called the certificate authority (CA) which must be trusted by both parties forming a secured connection. The certificate is signed with the private key of the CA to verify that it is really issued by the CA. Public key of the CA in turn can be used for decrypting the signature and verifying the authenticity of the certificate. Therefore, the receiver must possess the public key of the CA.

The public key shared with a certificate can be used for authentication in different ways depending on the key exchange method. Rescorla and Dierks (2008, p. 92–93) explain that in RSA key exchange the premaster secret is encrypted with the public key of the server as presented in the previous section and by proving the possession of the corresponding private key the server authenticates itself. In Diffie-Hellman key exchange the exchanged parameters can be signed by the authenticating party with its private key and the other party can verify the signature with the public key shared via the certificate. Further in the TLS handshake process hashes are exchanged which aim to verify the possession of genuine keys. The handshake process is presented more detailed in Section 3.1.4.

3.1.3 Integrity protection

TLS 1.2 deploys hash-based message authentication code (HMAC) algorithms to verify integrity of transferred data (Rescorla & Dierks, 2008, p. 14). Ristić (2014, p. 9–10) explains that HMAC calculates a hash of the transferred message and encrypts it with a secret key known only by the sender and the receiver. This hash is a fixed-length digest of the message calculated with an algorithm called cryptographic hash function. Characteristics of such function is that its input is computationally infeasible to recover and an equal output of such function is computationally infeasible to achieve with two different inputs. According to Bellare et al. (1996, p. 3) the encrypted hash acting as an authentication tag is attached to the transferred message and delivered to the receiver. Using the

same HMAC algorithm and secret key the receiver can calculate the authentication tag of the message and compare it to the one received. If they match the receiver can be sure that the received content of the message corresponds with the originally sent content.

3.1.4 Structure and functions of TLS

In the specification of TLS 1.2 Rescorla and Dierks (2008, p. 15) define that the structure of TLS is layered and functionalities of TLS are performed by several subprotocols. Record protocol works on the lowest level of TLS on top of a transport protocol such as TCP. Record protocol encapsulates higher level protocols including the handshake protocol, change cipher spec protocol, application data protocol, and alert protocol.

Record protocol is in charge of structuring and processing data of the higher-level protocols to be transferred by the underlying transport protocol. Record protocol fragments data into definite-sized transferable records, performs the encryption according to the negotiated parameters, and calculates the MAC of the data to verify its integrity. In the other end record protocol decrypts the data, verifies its integrity applying MAC calculation, and reconstructs the fragmented data.

Encryption and MAC calculation are accompanied with each other (Ristić, 2014, p. 42–43). Like encryption keys, MAC keys are derived from the master secret (Rescorla & Dierks, 2008, p. 25–26). HMAC of a TLS fragment is calculated before encrypting the fragment and the HMAC is encrypted along with the data fragment (Rescorla & Dierks, 2008, p. 22–23, 84). Encryption of the hash provides authentication because only the sender and receiver know the secret key (Bellare et al., 1996, p. 3). HMAC also includes sequence numbering of TLS records which provides protection against loss or repetition of records (Rescorla & Dierks, 2008, p. 21).

Handshake protocol is in charge of establishing a protected TLS connection through a handshake procedure. A TLS handshake consists of negotiation of the cipher suite used,

authentication of communicating parties, and exchange of parameters used for calculating the cryptographic keys for protecting the connection. Figure 3 presents the trajectory of a complete TLS handshake with mutual authentication. It should be noted that different cipher suites use different mechanisms and therefore handshake procedures and usage of messages vary between cipher suites. Messages which apply only to certain cipher suites and therefore do not always occur are marked with an asterisk.

Ristić (2014, p. 26–30) explains that a TLS handshake is initiated by the client with a ClientHello message which for example introduces the cipher suites supported by the client, meaning the set cryptographic algorithms used for key exchange, authentication, encryption, and integrity validation. According to the ClientHello message the server selects the most suitable cipher suite and announces it for the client with a ServerHello message. Both hello messages also contain a piece of random data which is used later in the generation of the master secret and gives uniqueness for a particular handshake preventing replay attacks.

After hello messages have been exchanged the server sends its certificate for the client, typically in X.509 form (Ristić, 2014, p. 30). Stallings (2017, p. 462) explains how certificates are validated. Certificate holds a digital signature of the CA signed with its private key and information of the algorithm used for signing. The client must possess the public key of the root certificate so that it can verify the issuer of the certificate and identity of the public key in the certificate. The certificate holds the public key of the server and information of the public key algorithm used for authentication against the corresponding private key (Ristić, 2014, p. 30, 41). Depending on the used cipher suite certificates might also contain other relevant information, for example static Diffie-Hellman parameters (Rescorla & Dierks, 2008, p. 92).

Some key exchange methods require additional exchange of parameters which need to be communicated with a separate ServerKeyExchange message (Rescorla & Dierks, 2008, p. 50–51). For example, Diffie-Hellman Ephemeral exchanges signed temporary public

parameters in this message (Rescorla & Dierks, 2008, p. 50–52, 92–93). If mutual authentication is requested the server sends a CertificateRequest message for the client requesting its certificate (Ristić, 2014, p. 33). After this the server sends a ServerHelloDone message to indicate that its ServerHello process has been fulfilled and waits for the client to continue the handshake (Rescorla & Dierks, 2008, p. 55).

The client continues the handshake by sending its certificate (Ristić, 2014, p. 32–33). Rescorla & Dierks (2008, p. 57–58, 61) define that whether or not the certificate was requested the client must next send a mandatory ClientKeyExchange message which contains key exchange parameters according to the key exchange algorithm. If RSA key exchange is used the message contains the RSA encrypted premaster secret, if Diffie-Hellman key exchange is used the Diffie-Hellman parameters are sent if not included in the certificate. To prove the authenticity of the public key delivered for the server the client sends a CertificateVerify message which contains a hash of all handshake messages performed so far and signs it with its private key (Rescorla & Dierks, 2008, p. 62). Lake (2021) explains that in the other end the server verifies the signature by calculating the same hash, decrypting the hash sent by the client, and comparing the two hashes. If hashes match the server can verify that the certificate is from the same entity engaged with the handshake. Hash also contains the random values exchanged in the Hello messages which bring uniqueness for the specific key exchange and therefore counter replay attacks (Rescorla & Dierks, 2008, p. 62, 92; Ristić, 2014, p. 29).

Ristić (2014, p. 31) says that at this point all the messages required for negotiating session keys and algorithms have been exchanged. The communicating parties can calculate the cryptographic keys used for protecting the connection and turn into encryption. Client indicates this for the server with ChangeCipherSpec message. Rescorla & Dierks (2008, p. 14–15, 63–64) explain that finally the client deploys a pseudorandom function to generate a set of data called verify_data based on HMAC calculations applying the negotiated master secret as a key and a hash of all the handshake messages as the seed. The client sends the data to the server in a Finished message and the server must

validate it. The server replies with a ChangeCipherSpec message and forms a similar Finished message which the client must validate. After a successful handshake data carried by the application data protocol of TLS can be protected by the record protocol.

Validation of Finished messages verifies integrity of the entire handshake. If an attacker has tampered handshake messages in transit the client and the server come up with different hashes of handshake messages and validation of a Finished message fails (Rescorla & Dierks, 2008, p. 94). Validation also verifies authenticity of a party as it proves that it possesses the negotiated master secret involved with the calculation of verify_data (Rescorla & Dierks, 2008, p. 92). Without the correct master secret valid Finished messages cannot be formed (Ristić, 2014, p. 32). In RSA key exchange ability of the server to form correct Finished messages authenticates the server because the premaster secret sent by the client is encrypted with the public key of the server (Rescorla & Dierks, 2008, p. 92). Same objective can be seen to happen in Diffie-Hellman key exchange when parameters are signed. Formation of the genuine master secret includes validating authenticity of the parameters used for calculating the premaster secret (Rescorla & Dierks, 2008, p. 92).

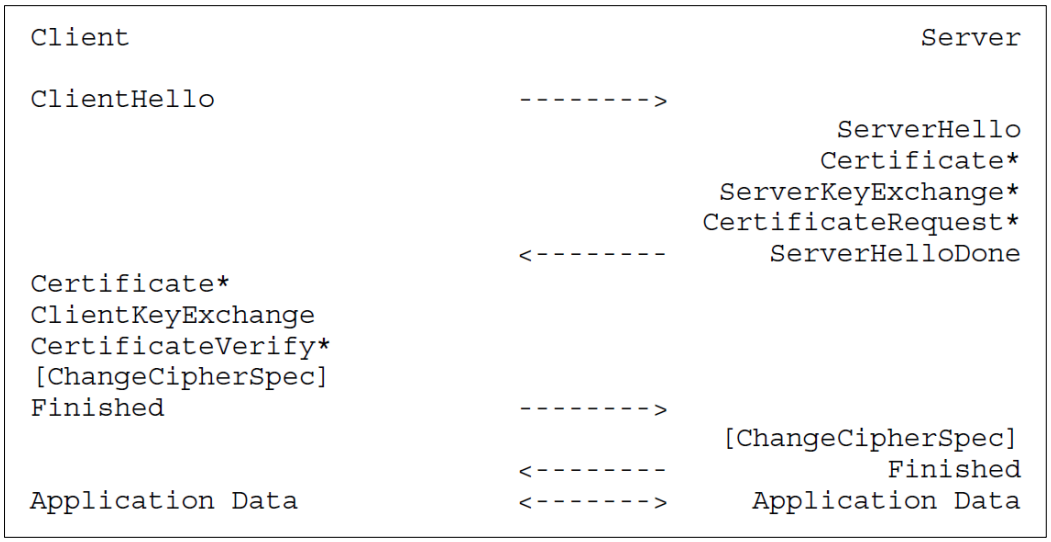


Figure 3. Trajectory of a complete TLS handshake with mutual authentication. (Rescorla & Dierks, 2008, p. 36)

3.2 Network layer protection with IPSec

According to Kent and Atkinson (1998a, p. 3, 5–7) IPsec (Internet Protocol Security) is a collection of protocols and associated algorithms used for providing security services for IP based traffic at network layer. IPsec can provide various security measures including encryption to provide confidentiality of transferred data, authentication to verify origin of data, integrity checking of transferred packets, and a type of access control through security associations addressed to defined networking entities. IPsec can be deployed to secure transport layer traffic between individual hosts or network layer traffic involved with gateways, such as routers and firewalls, connecting separate networks. IPsec features several functionality options which are discussed here only in general level but the focus is on the net-to-net tunneling using ESP protocol.

3.2.1 Structure of IPSec

Kent and Atkinson (1998a, p. 7, 9) define that IPsec can work in two different modes, transport and tunnel mode. In transport mode the transport layer payload of an IP datagram is protected and encapsulated inside an IPsec header. In tunnel mode the whole IP datagram is encapsulated at network layer into an IPsec header and a new IP layer header is added to the datagram. The protective encapsulation hides the original IP datagram and the datagram is processed according to the new IP header defining its properties, for example source and destination IP addresses. This isolates the original network layer datagram into a so-called tunnel for transfer and after unwrapping IPsec protection in the endpoint of the tunnel the original IP datagram is unveiled again.

Kent and Atkinson (1998a, p. 10) define that a host implementation must support both transport and tunnel mode whereas a gateway must support only tunnel mode. Typically transport mode is used for end-to-end protection of a connection between two hosts (Forouzan, 2008, p. 551). Tunnel mode is used for forming a secure connection when at least other endpoint is a gateway, such as a router or a firewall, and connecting networks located separate from each other (Stallings, 2017, p. 667). Typically, a tunnel is formed

between two gateways to form a protected connection between two private local area networks (LANs) through a public wide area network (WAN), such as the Internet (Barker et al., 2020, p. 11–13). Figure 4 presents the principle of tunnel mode.

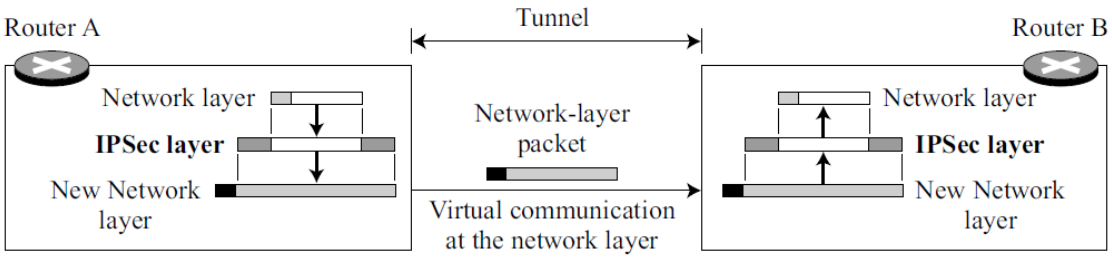


Figure 4. Operating principle of tunnel mode. (Forouzan, 2008, p. 552)

Kent and Atkinson (1998a, p. 6–7, 9) define that IPsec can operate using two different protocols, Authentication Header (AH) protocol and Encapsulating Security Payload (ESP) protocol. AH protocol can verify integrity and authenticity of an IP datagram. ESP protocol provides encryption, authentication, and integrity checking for an IP datagram through a protective encapsulation. It adds an ESP trailer in the end of the original IP datagram acting as the payload for an ESP datagram (Forouzan, 2008, p. 554–555) and encrypts the payload and ESP trailer with the key and algorithm negotiated for the security association in question (Kent & Atkinson, 1998b, p. 11). After this an ESP header containing identifying information regarding the ESP datagram is added in front of the encrypted payload and a piece of data for authenticating the ESP datagram is added in the end of the ESP trailer (Forouzan, 2008, p. 554-555). Finally, a new IP header with the IP addresses of the tunnel endpoints is added in front of the ESP datagram (Barker et al., 2020, p. 37; Forouzan, 2008, p. 554–555). Figure 5 presents the structure of a complete ESP datagram with a new IP header and authentication data.

Authentication data added in the end of the ESP datagram is formed by the sender by applying a hash function for the whole ESP datagram including the ESP header, the encrypted payload, and the ESP trailer (Forouzan, 2008, p. 554–555). This is done by performing a keyed MAC calculation for all parts of the ESP datagram (Kent & Atkinson,

1998b, p. 10). Authenticity of the hash can be verified by the receiver by performing the same calculation and comparing the results (Kent & Atkinson, p. 1998b, p. 15).

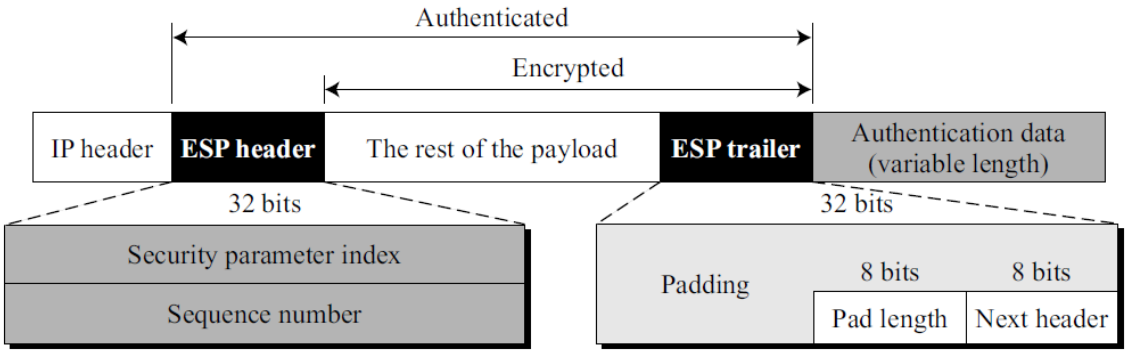


Figure 5. Structure of a complete ESP datagram with a new IP header and authentication data added in the end. (Forouzan, 2008, p. 554)

3.2.2 Security associations and Internet Key Exchange

IPsec connections between hosts are defined by security associations which specify the endpoints involved, IPsec protocol used (AH or ESP), security measures applied, and parameters for protection, such as cryptographic algorithms and keys (Forouzan, 2008, p. 557–559). Each IPsec connection consists of a pair of security associations, one for inbound and one for outbound traffic (Barker et al., 2020, p. 48). Kent and Atkinson (1998a, p. 21–24) explain that attributes of security associations are recorded in a security association database in which security associations are identified by a triple including the distinguishing security parameter index of the security association, the destination address of the tunnel endpoint which the security association applies to, and the IPsec protocol used by the security association (AH or ESP).

Security associations are negotiated using a protocol called Internet Key Exchange (IKE) (Forouzan, 2008, p. 563). Barker et al. (2020, p. 33) explain that two versions of IKE exist, the older IKEv1 and the revised IKEv2. IKEv2 has enhancements compared to the older protocol and it is considered more secure. Therefore, its usage is recommended. However, in this thesis the negotiation of security associations was investigated according to IKEv1. Forouzan (2008, p. 563) explains that IKEv1 is based on three other protocols. It

uses the framework of ISAKMP protocol to form and exchange messages for negotiating security associations. For negotiating shared secrets IKEv1 uses Diffie-Hellman key exchange accompanied with additional security features derived from two other key exchange protocols, Oakley and SKEME.

IKEv1 performs the negotiation of security associations for IPsec in two phases (Forouzan, 2008, p. 566). In phase 1 so-called ISAKMP security associations are negotiated which are used for forming a secure connection for performing phase 2 (Carrel & Harkins, 1998, p. 5, 16; Forouzan, 2008, p. 566). Phase 1 also generates and provides authenticated key material which is used and refined in phase 2 (Carrel & Harkins, 1998, p. 9–10, 16–17). Phase 2 in turn negotiates security associations for establishing the actual IPsec connection (Forouzan, 2008, p. 566). In IKEv1 negotiation of phase 1 can be done in two different modes, either in main mode consisting of six messages or in aggressive mode compressed to three messages (Forouzan, 2008, p. 567). Main mode provides more security compared to aggressive mode and should be used (Barker, 2020, p. 76). Main mode provides identity protection but aggressive mode does not unless public key encryption is used (Carrel & Harkins, 1998, p. 6, 13). Only one mode is defined for phase 2 which is called quick mode (Forouzan, 2008, p. 567).

Forouzan (2008, p. 563–566) explains that in phase 1 the endpoints negotiate a shared secret based on Diffie-Hellman key exchange. IKEv1 implements certain measures to protect the key exchange. Before exchanging Diffie-Hellman parameters pieces of identifying data called cookies are exchanged by the endpoints. Cookies are included in the key exchange messages to identify the particular key exchange and the other party involved. Cookies prevent clogging attacks in which an attacker sends numerous Diffie-Hellman half-keys causing the receiver to initiate numerous laborious Diffie-Hellman calculation processes which could cause a denial-of-service as the receiver is too loaded to serve other clients. If the exchanged cookies are not included in the key exchange messages they are ignored and the key exchange is aborted. To prevent reusage of Diffie-Hellman parameters IKE includes pieces of random data called nonces to the key exchange

messages to provide uniqueness for the key exchange. To prevent man-in-the-middle attacks against the key exchange endpoints must authenticate themselves using a secret pre-shared key, a private-public key pair, or a digital signature. Nonces, cookies, and authentication methods are also utilized for deriving authenticated key material for authentication and encryption.

Main mode negotiation of phase 1 consists of six messages. Carrel and Harkins (1998, p. 8) state that specific contents of the messages vary depending on the selected authentication method but intentions of the messages are the same. Forouzan (2008, p. 567–568) explains that the first two messages negotiate security association attributes for forming a protected connection and exchange cookies to bind the participants for the specific key exchange. In the third and fourth messages Diffie-Hellman half-keys are exchanged for calculating a shared secret. At the same time nonces are exchanged. Finally, in messages five and six the parties exchange authenticating hashes based on parameters exchanged in the key exchange. The hashes use a parameter called SKEYID (secret key ID) as the key which is derived through the authentication method in use. Figure 6 presents messages of phase 1 when pre-shared key is used for authentication. The asterisks remark that payloads of messages five and six are encrypted.

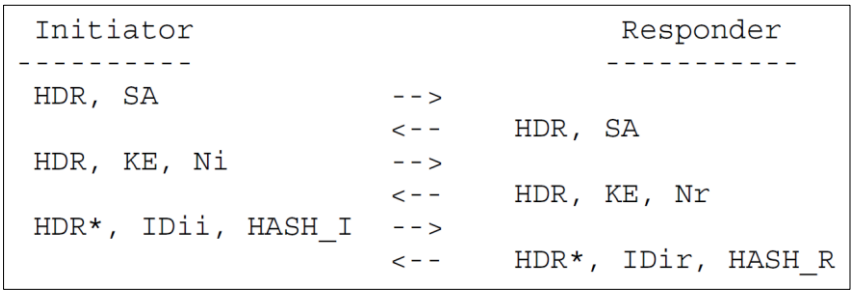


Figure 6. Messages of IKEv1 phase one when pre-shared key is used for authentication. (Carrel & Harkins, 1998, p. 16)

Carrel & Harkins (1998, p. 9–10, 12) define that SKEYID is derived differently depending on the authentication method through a keyed hash function. For example, if pre-shared key is used for authentication, it is used as the key for the hash function using the nonces

as input to generate SKEYID. If public keys are used, nonces are exchanged encrypted with the public key of the counterpart and used as the key for generating SKEYID. SKEYID is further used as the key to derive authenticated key material for authentication and encryption through a hash function using the shared secret derived through Diffie-Hellman key exchange and the cookies as inputs. As mentioned, SKEYID is also used as the key for hashes exchanged in messages five and six.

In phase 2 the parties negotiate security association attributes for creating the IPsec connection and derive key material for its protection (Tiller, 2000, p. 200). Quick mode used in phase 2 consists of three messages presented in Figure 7 which are protected by the security associations created in phase 1 (Carrel & Harkins, 1998, p. 16, 18), i.e., the messages are authenticated and encrypted with the algorithms and keys negotiated in phase 1 (Tiller, 2000, p. 199–200). In the first two messages the parties negotiate the security association attributes (SA) for IPsec (Tiller, 2000, p. 199–201) and exchange new nonces (Ni and Nr) which are used for proof of liveliness and refining key material derived from phase 1 to be used for IPsec keys (Carrel & Harkins, 1998, p. 16–18). Hashes of the messages are exchanged for authentication and are calculated using the key material derived from phase 1 (Forouzan, 2008, p. 576). Finally in the third message the initiator sends a hash based on both nonces to prove its liveliness before forming the IPsec connection (Tiller, 2000, p. 201).

New key material for IPsec security association is derived using key material derived from phase 1, security parameter index of the negotiated security association, and nonces exchanged in phase 2 (Carrel & Harkins, 1998, p. 18–19). Because security parameter indexes are distinctive for inbound and outbound security associations the keys used in different directions are separate (Forouzan, 2008, p. 576–577). Tiller (2000, p. 199, 203) explains that it should be noted that exposure of key material derived in phase 1 also exposes key material derived in phase 2. An option for a new Diffie-Hellman key exchange in phase 2 exists providing perfect forward secrecy for IPsec keys and protection against compromise of phase 1.

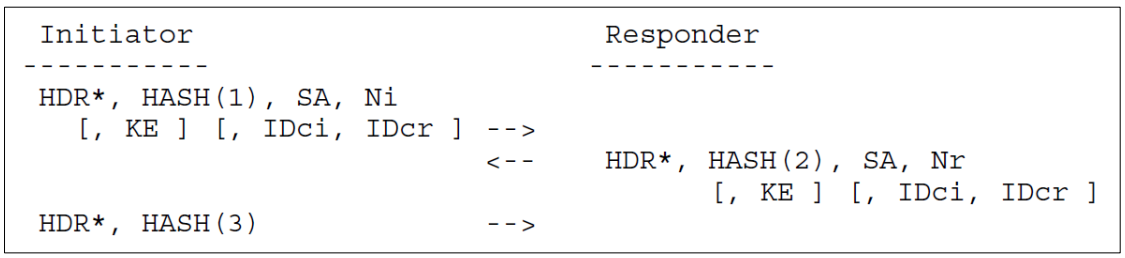


Figure 7. Messages of quick mode in phase 2. (Carrel & Harkins, 1998, p. 18)

4 IEC 60870-5-104 protocol

SCADA (supervisory control and data acquisition) systems are used for monitoring and controlling geographically widespread automation infrastructures, such as the electric power system. In the core of a telecontrol system is the SCADA server software which is used for modelling, controlling, and monitoring processes of the electric distribution network. A SCADA server can be located in a separate control station and have multiple remote substations under its supervision or it is possible that a substation has its own SCADA server at the local site. An RTU (remote terminal unit) is a device which is used for linking a substation to the SCADA server. A substation can have one or multiple RTUs and an RTU can act as a singular substation device or as a connecting node for multiple substation devices under it.

RTUs perform control actions sent from the SCADA server to the field, for example control switching devices, or forward control data to other control devices in the substation, such as protection relays. Likewise, RTUs gather state and measurement data concentratedly from the substation and pass it up to the SCADA server. Depending on the environment RTUs are connected to the SCADA server by different communication links, either private or public, or by a LAN. Protocols defined by IEC in its 60870-5 series of standards are commonly used for telecontrol in European power systems to perform communication between a SCADA server and an RTU.

4.1 About IEC 60870 and IEC 60870-5 series of standards

IEC (1988, p. 9, 11) explains that IEC 60870 is a series of standards which defines technical requirements for equipment and techniques used in telecontrol systems. IEC 60870 consists of six parts which consider different technical aspects required for data transmission. For example, part 2 defines operating conditions, referring for example to electrical and electromagnetic requirements of telecontrol systems, and part 4 specifies performance requirements of telecontrol systems. Part 5 specifies requirements for communication protocols used in telecontrol systems.

Clarke and Reynders (2004, p. 177–181, 183–184) explain that the first five parts of IEC 60870-5 series define requirements for a transmission protocol regarding different layers of networking and data communication. Ultimately, IEC 60870-5 defines four companion standards which define actual communication protocols for transmission of telecontrol data. It should be noted that principally IEC 60870 series defines data transmission using low-bandwidth serial data transmission (IEC, 1988, p. 9). Under interest here are the companion standard IEC 60870-5-101 which defines a communication protocol for telecontrol and telemetry in SCADA systems and the companion standard IEC 60870-5-104 which harnesses the application layer of T101 to be transmitted using TCP/IP protocol suite (Clarke & Reynders, 2004, p. 170–171).

4.2 Application layer of IEC 60870-5-101 protocol

IEC 60870-5-101 standard defines a communication protocol used for telemetry control and monitoring functions between a controlling station and a controlled station in SCADA systems. Clarke and Reynders (2004, p. 170–171) tell that T101 was the first protocol implemented in accordance with IEC 60870-5. T101 is applicable for all SCADA applications in general but is mainly used in electrical industries and particularly in Europe. It is based on the enhanced performance architecture which is composed of the physical layer, the data link layer and, the application layer of the OSI model (IEC, 1992, p. 13; IEC, 2003, p. 10–11). T101 also defines an additional user process layer on top of application layer (IEC, 2003, p. 11–12). At lower layers T101 uses serial communication (IEC, 2003, p. 11) and therefore T101 protocol is not relevant for this thesis considering the lower layers. However, definitions in application and user process layers apply also for TCP/IP based T104 for large extent (Clarke & Reynders, 2004, p. 300; IEC, 2006, p. 21).

As a basis for structuring and presenting application data T101 implements an Application Service Data Unit (ASDU) which is fundamentally defined in IEC 60870-5-3 (IEC, 2003, p. 27). The structure of a T101 ASDU is shown in Figure 8. An ASDU consists of a data unit identifier and one or more information objects (IEC, 2003, p. 27). The data unit

identifier defines identifying and defining information regarding an ASDU whereas information objects contain the actual operational information transmitted by the ASDU. Information fields constructing an ASDU and defining parameters of an ASDU are implemented according to the data structures defined in IEC 60870-5-4 (IEC, 2003, p. 29).

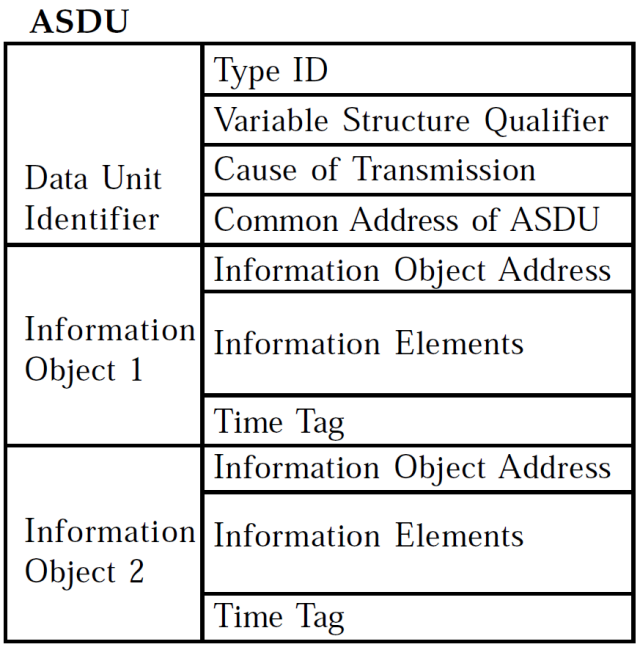


Figure 8. Structure of a T101 ASDU. (Clarke & Reynders, 2004, p. 204)

IEC (2003, p. 29–33) defines that type identification field of the data unit identifier specifies the type of data transmitted by the following information objects. T101 has a definite set of types defined representing different functionalities used for operation of a control system in monitor and control directions. The value of type identification field is presented with 8 bits and distinctive values have designated meanings. For example, values 1−44 represent different types of process information in monitor direction. ASDU with type identification field with decimal value 1 carries single-point information (M_SP_NA_1). Correspondingly values 45−69 represent process information in control direction. For example, decimal value 45 is dedicated for a single command (C_SC_NA_1). IEC (2003, p. 36−37) defines that the cause of transmission field of the data unit identifier defines the purpose of the ASDU. Clarke and Reynders (2004, p. 211–212) explain that each type of ASDU has only certain cause of transmission codes related to it. Cause of

transmission field also carries a confirmation bit called P/N-bit which applies to replies for control commands in monitor direction. The P/N bit is used by the controlled station to indicate if a control command was successful or not. Cause of transmission field can also define the station address of the originator which is used for routing response ASDUs from the controlled station back to the controlling station.

Clarke and Reynders (2004, p. 213–214) explain that addressing of an ASDU is based on common address and information object address attributes. Common address identifies a controlled station and is part of the data unit identifier field. An information object address defines an individual data point under a station. The combination of a common address and an information object address identifies an individual and unique data point under a specific station.

Information objects transmitted by an ASDU contain the actual data used in the operational procedures. An information object consists of the information object address, information element containing the actual operational data, and optionally a time tag (Clarke & Reynders, 2004, p. 217–219; IEC, 2003, p. 27). T101 defines different structures for information elements according to their purpose (IEC, 2003, p. 44–58) which are used by ASDUs defined in the standard (IEC, 2003, p. 58–119). For example, particular information elements are defined for carrying monitoring data such as measurement values and particular information elements are defined for carrying control data such as control commands (Clarke & Reynders, 2004, p. 218).

Regarding operation and interaction of application processes controlling and monitoring a SCADA system, user process layer consists of application functions which deploy the lower communication layers to exchange information with ASDUs between different user process entities (IEC, 1995, p. 15). Basic application functions specified to be used in T101 communication for controlling and monitoring in a SCADA system are based on functions defined in IEC 60870-5-5 (IEC, 2003, p. 120). Couple of examples of user process functions are presented in the following chapters.

4.3 TCP/IP based telecontrol protocol IEC 60870-5-104

Companion standard IEC 60870-5-104 harnesses application and user process layers of T101 to be carried by TCP/IP protocol suite enabling communication in Ethernet LANs and WANs (IEC, 2006, p. 17, 19, 22). T104 implements mostly the user process and application layer of T101 but on the lower layers it replaces the serial communication with the protocols of TCP/IP suite. According to Clarke and Reynders (2004, p. 305–306) T104 uses mainly the same ASDUs and application functions as T101 with some deviations due to different timing characteristics of the underlying data transmission techniques. Figure 9 presents the structure of T104 with respect to the layers of the OSI model.

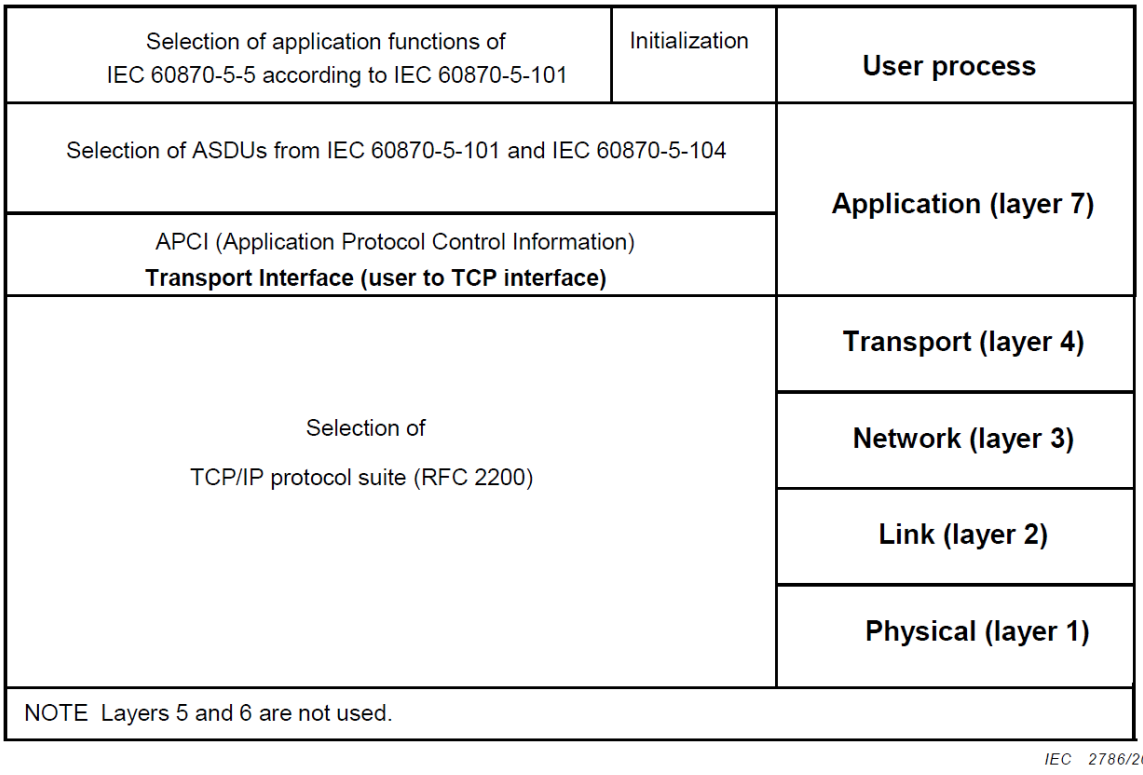


Figure 9. Structure of T104 with respect to the OSI model. (IEC, 2006, p. 21)

IEC (2006, p. 23, 25, 27) defines a structure called Application Protocol Control Information (APCI) for T104 to be used in conjunction with an ASDU for controlling information flow of ASDUs transmitted in TCP streams. APCI distinguishes an ASDU from a

TCP stream and together an APCI and an ASDU form an Application Protocol Data Unit (APDU) presented in Figure 10. APCI indicates the beginning of an APDU and indicates its length. Start of an APDU is indicated in the beginning of the APCI with a hexadecimal value 68 followed by the length of the APDU. An APCI also contains a control field consisting of four octets which is used for controlling and supervising T104 communication. There are three types of control fields formats for different purposes. Relevant definitions regarding this thesis are introduced.

IEC (2006, p. 27, 33, 37) defines that a controlled station can have several T104 connections open simultaneously but only one can have active communication at once. Control field of U format (unnumbered control functions) is used for activating and terminating data transfer from a controlled station to a controlling station by using STARTDT (start data transfer) and STOPDT (stop data transfer) procedures in which the controlling station sends a STARTDT or STOPDT activation message for the controlled station and a confirmation message is required as response from the controlled station to change the state of communication. Control fields of U format can also carry test frames (TESTFR) for testing a T104 connection and they can be initiated by either station. IEC (2006, p. 25, 27, 29) defines that APDUs with control field of I format (information transfer format) carry process information and have sequence numbers for the ASDUs exchanged in the active T104 communication. I format ASDUs transmitted and received are calculated separately and are incremented one by one. Sequence numbering is used for detection of duplication and loss of ASDUs.

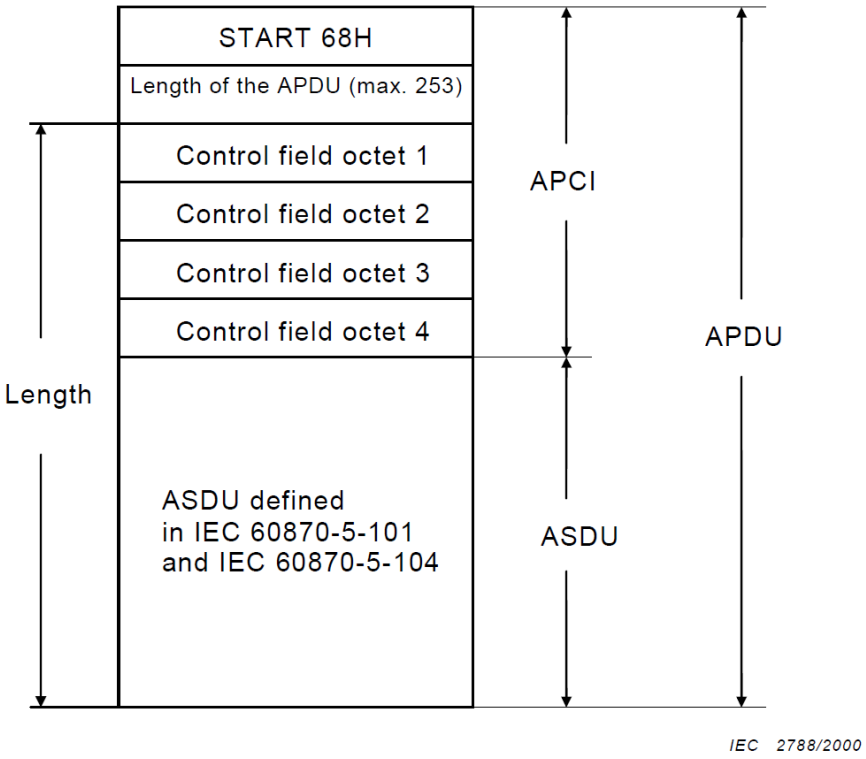


Figure 10. Structure of a T104 APDU. (IEC, 2006, p. 25)

5 Cybersecurity of IEC 60870-5-104

This chapter considers security deficiencies of T104 and possible attacks conducted against T104 communication. Possible threats include different types of man-in-the-middle attacks for which T104 communication is susceptible due to inherent lack of security measures regarding confidentiality, authentication, and integrity protection. Exploitation of these deficiencies enables an attacker to gain access to T104 communication and ultimately leads into manipulation of communication and sensitive process objects.

5.1 Relevant cybersecurity threats in general

Man-in-the-middle attack is a cyberattack in which an external party manages secretly intercept communication of two legitimate parties. This enables the external party to eavesdrop the communication or manipulate the communication imperceptibly. Eavesdropping is an attack in which an external party secretly listens communication of legitimate parties (Fortinet, n.d.-b). In a replay attack the attacker manages to capture a packet and delays its delivery or resends it to the receiver (Kaspersky, n.d.). Resending a legitimate message might cause harmful consequences or give advantage for the attacker, delaying the communication might lead to a denial-of-service compromising availability. In a spoofing attack the attacker disguises his identity as a legitimate party (Rapid7, n.d.). If an attacker manages to get in the middle of the communication, i.e., legitimate traffic is intercepted and routed through the attacker the attacker might be able to tamper and forge the contents of original messages (OWASP Foundation, n.d.).

5.2 Vulnerabilities of IEC 60870-5-104

T104 lacks essential security measures requested in modern networking and data communication exposing T104 communication for different kinds of man-in-the-middle attacks. T104 does not have inherent encryption of packets leading the application layer payload to be sent in plain text and vulnerable for eavesdropping attacks (Erdődi et al., 2022, p. 9). This compromises the confidentiality of critical information of ASDUs. T104

does not have inherent authentication mechanisms which verify the authenticity of the origin of T104 messages (Maynard et al., 2014, p. 37). T104 also lacks integrity checking to verify modification of transferred messages (György & Holczer, 2020, p. 140). At the application layer T104 does not have measures to check integrity but is dependent on measures provided by lower layers (IEC, 2013a, p. 13; Pidikiti et al., 2013, p. 136). It is also noted that T104 frame lengths are limited to 255 octets which restricts capabilities for implementing security measures in the application protocol (IEC, 2013a, p. 13; Pidikiti et al., 2013, p. 137). IEC (2013a, p. 12) states that T104 uses poor sequence numbering which makes it susceptible for replay attacks. These several security measure deficiencies expose legitimate T104 communication for interception, modification, and spoofing by an unintended party (Matoušek, 2017, p. 26).

5.3 Examples of attacks against IEC 60870-5-104 communication

To get a picture of how vulnerabilities of T104 protocol can be exploited some attack simulation studies found in the literature were investigated more thoroughly. They demonstrate how the lack of encryption, authentication, and integrity protection of the protocol can be exploited between communication of a controlling and a controlled station. Common for the attacks is that they capture or intercept legitimate unencrypted T104 traffic and use contents of legitimate messages to form forged T104 messages. These messages can either be used for manipulating the controlled station or feed false data for the controlling station. The assumption in the presented scenarios is that the attacker has already penetrated in the target environment and has access to a device which is in the same subnet as the controlling and the controlled station. Generally, passive and active reconnaissance techniques were first used for gathering information of the target environment and accessing legitimate T104 traffic. After this packet injection or man-in-the-middle attacks were performed against the target.

After gaining access to the target network the attacker must first acquire information of the target environment through passive or active reconnaissance techniques. Passive reconnaissance techniques do not interact with target devices but eavesdrop passively

traffic in the network. This can be done with network packet analyzers such as Wireshark or tcpdump listening network traffic in promiscuous mode (Saif Qassim et al., 2018, p. 155) which records all the traffic entering a network adapter (Awati, 2021). Wireshark has features to filter T104 packets and interpret contents of application layer APCIs and ASDUs (Wireshark, n.d.-a, n.d.-b). Since T104 communication is not encrypted the attacker can read contents of T104 packets which happen to cross the attackers network interface (Erdődi et al., 2022, p. 5). One possibility to monitor network traffic is to connect to a span port of a network switch in which copy of all traffic is mirrored typically for debugging purposes (FS, n.d.; Maynard et al., 2014, p. 33).

From this traffic the attacker can gather information and make conclusions of the engaged network assets. By default controlled stations acting as T104 servers listen TCP port 2404 so traffic directed to that port can be assumed to be sent by the controlling station to the controlled station (IEC, 2006, p. 45; Saif Qassim et al., 2018, p. 156). From these TCP packets the attacker can obtain IP and MAC addresses of controlling and controlled stations. MAC address of a network interface can give device specific information such as the manufacturer (Erdődi et al., 2022, p. 5). From ASDUs the attacker can discover T104 specific information, for example station numbers and information object addresses of monitor and control points in use (Erdődi et al., 2022, p. 5; Saif Qassim et al., 2018, p. 156).

Erdődi et al. (2022, p. 5–6) explain that instead of listening random packets entering the network adapter the attacker can also use more systematic active reconnaissance techniques which probe information from the network by trying to interact with devices in the network. Even though these techniques might be more efficient than passive reconnaissance it should be noted they cause noise in the network making an attacker to be detected more easily. To recognize potential active T104 hosts the attacker might try to perform port scans to TCP 2404 port in the target network to identify T104 servers. Lack of authentication enables an attacker to connect to the T104 server of a controlled station (György & Holczer, 2020, p. 146). After identifying potential T104 devices the

attacker can try to make connections to them. However, only one T104 session can be active at once and therefore the attacker must interfere with the existing legitimate T104 session (Erdodi et al., 2022, p. 6; IEC, 2006, p. 37). Three different scenarios for an attacker to hijack communication with a controlled station were presented in the investigated studies.

Erdődi et al. (2022) explain how to inject singular packets to an existing T104 session. Saif Qassim et al. (2018) terminated the TCP connection and the active T104 session between a controlling station and a controlled station and formed a new spoofed TCP connection and T104 session with the controlled station on behalf of the legitimate controlling station. Maynard et al. (2014) gained a man-in-the-middle position in which they got access to all traffic between a controlling station and a controlled station and could manipulate legitimate T104 ASDUs. György and Holczer (2020) explain how TCP and APCI sequence numbers can be dealt when injecting arbitrary packets in a man-in-the-middle position.

Forged T104 packets must be formed properly so that they can be injected to an existing T104 communication stream. The attacker must capture the last packet sent by the control center and the last packet sent by the controlled station to get information of current communication to form a proper packet to be injected in the TCP stream. The packets must use a valid spoofed source IP address and TCP port (Erdődi et al., 2022, p. 7). TCP packets must have valid TCP sequence numbers and T104 messages valid APCI sequence numbers to be accepted by the controlled station, otherwise they get dropped (Erdődi et al., 2022, p. 7; György & Holczer, 2020, p. 148).

Erdődi et al. (2022, p. 6) explain that specific T104 messages can be used for enumerating T104 specific information from the environment. They suggest that T104 STARTDT messages with different source IP addresses can be used for enumerating valid controlling station IP addresses because configuration of an RTU might restrict acceptable IP addresses of controlling stations and answer only for legitimate IPs. Conversely, an attacker

can send TESTFR messages with spoofed RTU IP addresses to enumerate RTUs registered for a controlling station. However, they state that most useful is to send general interrogation requests to the network with a spoofed controlling station IP address to expose T104 devices. General interrogation requests a controlled station to send values of all its process variables (IEC, 1995, p. 61).

Erdődi et al. (2022, p. 6–7) demonstrated how to inject spoofed packets into T104 communication by sending a general interrogation request for active reconnaissance and perform an attack which injects a 0x45 single command to control a breaker. Both of the packet injections were successful and the requested actions were performed by the controlled center. However, because the legitimate controlling station continued to act independently beside the spoofed T104 packets and the following legitimate T104 packet reused sequence numbers of the forged packets the controlled station terminated the existing legitimate connection in both demonstrations. Saif Qassim et al. (2018, p. 156–157) made a similar kind of control command injection attack but they terminated the original legitimate TCP connection between a controlling station and a controlled station and formed a new TCP connection with the controlled station on behalf of the legitimate controlling station. This gave them a possibility to start a new T104 communication with the controlled station using the STARTDT procedure and send arbitrary commands to the controlled station.

A steadier position can be achieved by the attacker if he manages to place himself in the middle of the communication between a controlling and a controlled station. This is called the man-in-the-middle position in which the traffic is routed via the attacker giving the attacker full control to manipulate it (Maynard et al., 2014, p. 33). The attacker can forward T104 messages between the control and the controlled station, modify legitimate messages, inject arbitrary messages in the TCP stream, or block communication between the entities (Erdődi et al., 2022, p. 7; Maynard et al., 2014, p. 33).

Maynard et al. (2014, p. 33) explain that to achieve a man-in-the-middle position the attacker can for example use a technique called ARP spoofing. In ARP spoofing the attacker sends false ARP (Address Resolution Protocol) messages to the controlling station and controlled station devices to forge their ARP tables which link MAC addresses and IP addresses of devices in a subnet. By sending false ARP messages to the controlling station device the machine acting as the attacker can link the IP address of the controlled station device to its own MAC address so that packets intended for the controlled station device are sent to the attacker. Correspondingly the ARP table of the controlled station device can be forged to link the IP address of the controlling station device with the MAC address of the attacker. By sending forged ARP messages continuously the attacker can maintain the man-in-the-middle position (Erdődi et al., 2022, p. 7).

Erdődi et al. (2022, p. 7) say that with ARP spoofing a TCP connection reset can be avoided. György and Holczer (2020, p. 148–149) explain how sequence numbers towards the controlling station and the controlled station should be handled in a man-in-the-middle position to avoid the termination of the connection when arbitrary packets are added to the communication and sent for the controlled station. This requires continuous update of TCP and APCI sequence numbers towards the controlling and controlled stations separately. Injecting packets without a reset requires keeping track and handling of both the TCP sequence numbers of the existing TCP session and the APCI sequence numbers of the existing T104 communication precisely to avoid termination of the active communication (Erdődi et al., 2022, p. 7; György & Holczer, 2020, p. 148–149).

Maynard et al. (2014) demonstrated how the man-in-the-middle position can be used for feeding forged process state values of a controlled station to the controlling station. They used a Kali Linux (a Linux distribution designed particularly for penetration testing and security auditing (Kali, n.d.)) to simulate a machine compromised by the attacker, and deployed a tool called Ettercap to perform ARP spoofing and to form the man-in-the-middle position. They also used a custom Ettercap plugin for tampering T104 packets.

The goal was to falsify an earth fault indication of the controlled station so that if an earth fault occurs the controlling station does not get aware of it.

In the man-in-the-middle position Maynard et al. (2014) could monitor all data sent between an RTU and a SCADA server and Ettercap could be used for detecting T104 packets containing the earth fault indication data to be modified. The earth fault indication data was conveyed in a M_SP_TB_1 single point information ASDU (IEC, 2003, p. 44) with value 1 of the SPI field indicating an earth fault. By detecting these records, changing their value to 0, and forwarding them to the SCADA server the earth fault could be hidden from the SCADA server and the operator. In the end they state that in real life instead of Kali Linux used in their test setup an attacker would probably plant a malware into the target environment to forge values of T104 ASDUs.

Erdődi et al. (2022, p. 7–9) also performed denial-of-service attacks against T104 communication deploying packet injection and ARP spoofing. Such attacks make the controlled station unavailable for the controlling station which is naturally a serious issue in the case of control systems. In the packet injection approach they reset a controlled station by injecting a C_RP_NA_1 command (IEC, 2003, p. 107–108) in T104 communication. After the controlled station had recovered and the controlling station had started a new session with a STARTDT message they injected another reset command for the controlled station. By repeating the process continuously, the controlled station could be made unavailable for the controlling station. In the man-in-the-middle approach they gained the man-in-the-middle position via ARP spoofing in which they were able to block the communication between the legitimate entities by not forwarding the captured messages to their endpoints. They state that denial-of-service attack using packet injection is less noisy than using ARP spoofing which requires continuous sending of ARP messages to maintain the man-in-the-middle position.

6 Security measures defined by IEC 62351

In IEC 62351 series of standards IEC (2007, p. 6, 23, 24–29) specifies security measures for power system communication protocols defined by IEC TC 57 and has specifications also regarding T104. IEC 62351 standards are concerned of protection of T104 in several layers, i.e., it addresses security measures for protecting transfer of data as well as protection regarding the application layer which cannot be achieved by protection in the lower layers. Protection of data transfer is covered by part 62351-3 which defines usage of TLS protection for TCP/IP based protocols defined by TC 57. However, IEC does not see transport layer protection solely sufficient for protecting T104 communication but emphasizes the importance of user authentication and authorization on application layer. To counter spoofing and tampering of application layer messages of 60870-5 based protocols IEC has defined application layer protection mechanisms in part 62351-5 which provides authentication, integrity protection, and authorization of control system operations.

6.1 TLS protection according to IEC 62351-3

IEC 62351-3 specifies deployment of TLS version 1.2 defined in RFC 5246 for TCP/IP based communication protocols defined by IEC TC 57, including T104 (IEC, 2007, p. 26; IEC, 2014, p. 5). IEC 62351-3 gives general requirements for deployment of TLS considering telecontrol systems and IEC 60870-5-7 determines specifications for T104 according to those requirements (IEC, 2013b, p. 34). TLS provides end-to-end protection of TCP/IP based data transfer between software instances countering eavesdropping via encryption (providing confidentiality), modification of transferred data via MAC calculation (providing integrity), and spoofing via endpoint authentication by deploying X.509 certificates (providing authenticity) (IEC, 2007, p. 26–27; 2014, p. 5, 7). IEC (2007, p. 26) says that deployment of IPsec and TLS were compared but because TLS itself provides end-to-end protection between software instance endpoints TLS was seen more suitable. However, these techniques can be used in tandem.

6.2 Application layer authentication provided by IEC 62351-5

IEC 62351-5 provides application layer security for IEC 60870-5 based protocols and their derivatives through authentication of individual ASDUs (IEC, 2013a, p. 8, 17–18). IEC 62351-5 defines the formal procedures and message formats regarding authentication (IEC, 2013a, p. 8) and IEC 60870-5-7 specifies ASDUs used for authentication regarding T101 and T104 protocols (IEC, 2013b, p. 7). Basically IEC 62351-5 and IEC 60870-5-7 extend functions in user process layers of T101 and T104 whereas the term is not directly used in the standards. It should be noted that the 2013 version of the standard is investigated in this thesis. IEC released the latest revision of 62351-5 standard in 2023 including some major changes in functionalities compared to the old version (IEC, 2023, p. 6–7). However, the 2013 version was relevant because the versions of SYS600 and RTU560 used in this thesis had implementations of the standard based on it.

In IEC 62351-5 standard IEC (2013a, p. 17–18, 20, 58) requires authentication of application layer ASDUs classified as critical. The standard defines that all output operations, such as controls and setpoint adjustments, and changes in security parameters should be considered as critical and therefore challenged. Optionally additional operations can be classified as critical. Authentication is based on challenge-response mechanism between the sender and the receiver which is based on keyed MAC calculation. In essence the authentication mechanism deploys two types of keys, session keys and update keys. Session keys are used for authenticating critical messages using the MAC calculation. Separate session keys are used for authentication in control and monitor directions so compromise of one key does not compromise trust in both directions. Session keys are renewed regularly by the controlling station during a session. Controlling station encrypts the new session keys with the update key and delivers them for the controlled station.

IEC (2013a, p. 17–18) explains that either the controlled station or the controlling station can initiate a challenge. The receiver of a critical message creates a challenge message containing a piece of random challenge data and the MAC algorithm to be used for

calculating the authentication tag of the response. Random challenge data is to be included in the MAC calculation and protects against replay attacks. The receiver sends the challenge message to which the sender of the critical ASDU responds with a MAC based on the contents of the critical ASDU and the challenge data deploying the session key. The receiver performs the same calculation and compares the results of the two MACs. If they match the critical ASDU is processed but if the results do not match the ASDU is not processed.

In IEC 62351-5 standard IEC (2013a, p. 16, 21–22, 45, 76) provides a capability to link individual users between a controlled station and a controlling station and manage these users and their privileges. Individual users of a controlled station are identified by a user number and they have individual update keys. Creation and deletion of users and controlling their privileges according to the role-based access control defined in IEC 62351-8 (for example Operator and Viewer privileges) is intended to be done by an external authority. The authority should deliver the user information for the controlling station which in turn provides the user information for the controlled station with application protocol specific message structures defined by IEC 62351-5. Authentication mechanism also enables to identify which user has sent a critical ASDU, for example performed a control action, and enables collection of security statistics which can be used for identifying possible cyberattacks.

As a conclusion, MAC calculation provides user authentication and integrity protection of critical application layer messages. Session and update keys bound to an individual user authenticate the sender of a critical ASDU. MAC calculations in both ends based on the contents of the ASDU and the random challenge data verify the integrity and uniqueness of the ASDU. If the MAC has been calculated using the same algorithm and the same key but results differ between the sender and the receiver the content of the ASDU has changed after sending. Without the valid update key a user is not able to negotiate valid session keys and is not authenticated. Random challenge data prevents reusing authentication responses. Therefore IEC 62351-5 provides protection against tampering,

spoofing, and replaying of critical ASDUs. IEC 62351-5 does not provide encryption and therefore IEC (2013a, p. 99) recommends using an implementation of the standard accompanied with TLS protection according to IEC 62531-3. Deployment of IPsec is not defined in the standard.

7 Demonstrations

The empiric part of the thesis consists of demonstrations in which T104 communication and its protection is analyzed. The test setup consists of a Hyper-V virtual machine running MicroSCADA X SYS600 SCADA server software of Hitachi Energy on a modern Windows Server operating system and a physical RTU560 of Hitachi Energy. SYS600 acts as the controlling station, RTU560 acts as the controlled station, and T104 is used for communication between them. Packet analyzing tool Wireshark was used for capturing and reading the traffic. First plain T104 traffic was captured and analyzed to demonstrate data structures and functionalities of T104 presented in the theoretical part. After this the investigated protection techniques, TLS protection according to IEC 62351-3, IPsec ESP tunnel, and implementations of IEC 62351-5, were tested and their effects were analyzed for protecting T104 communication between the SCADA server and the RTU.

7.1 Example of IEC 60870-5-104 communication

In the first demonstration an example of unprotected T104 communication is presented in which a control command is given from SYS600 to RTU560 to open a disconnector. The intention of this demonstration is to present operation of T104 but also to realize what information can be discovered from unprotected plain T104 communication using a packet analyzer. The communication was recorded with Wireshark which can extract and interpret APCI and ASDU fields from T104 messages (Wireshark, n.d.-a, n.d.-b). First configuration of the test setup is introduced and then most relevant fields of T104 messages are inspected regarding controlling the disconnector.

Figure 11 presents a visualization of the test setup. T104 communication must be configured both to SYS600 and RTU560. Figure 12 presents configuration of the T104 controlled station under a T104 master line in PC-NET program of SYS600 which is responsible for the communication of SYS600. Station Address attribute represents the common address of the RTU (10) and Internet Address attribute carries the IP address of RTU560 (10.10.50.10) which SYS600 uses for T104 communication. Correspondingly a T104 line

must be configured for RTU560 for which the SCADA server is configured as the controlling station of T104 communication.

The process objects supervised in SYS600 must be configured in its database. Figure 13 presents the process objects associated with the disconnector. The common address, the information object addresses, and the information types of the process objects received from RTU560 must be specified. Information object address 1003 was configured for the process object controlling the disconnector. The supervised process objects must be configured as signals for the actual input and output hardware of RTU560 and link them to the SCADA line with the common address and the information object addresses configured in SYS600. Figure 14 presents an extract from RTUtil tool used for configuration of RTU560 in which the process object for controlling the disconnector is configured to a binary output board of RTU560.

The process to be supervised is modelled in SYS600 as a display which visualizes the process objects in the database. The display is used for monitoring and controlling the state of the associated process objects which in turn are linked to RTU560 via T104 communication. Figure 15 presents the display of the setup used in this demonstration. Real switching devices were not used but the process was simulated using logical signals connected to the binary input card of RTU560. In Figure 15 the disconnector is in a closed state.

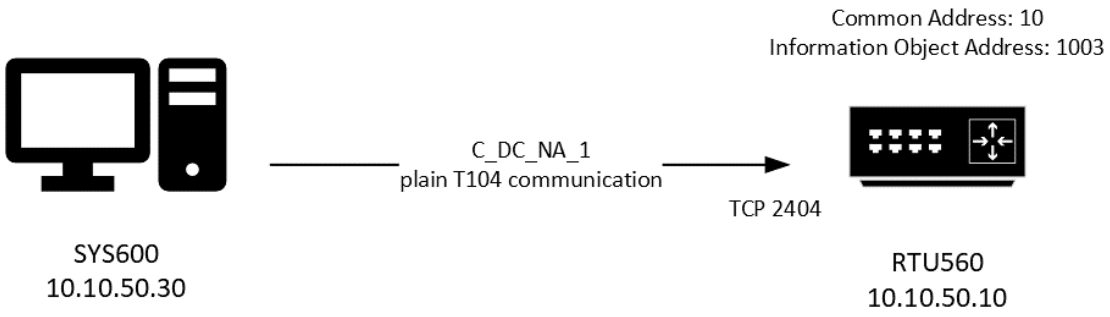


Figure 11. Test setup of the first demonstration in which SYS600 and RTU560 communicate using unprotected T104.

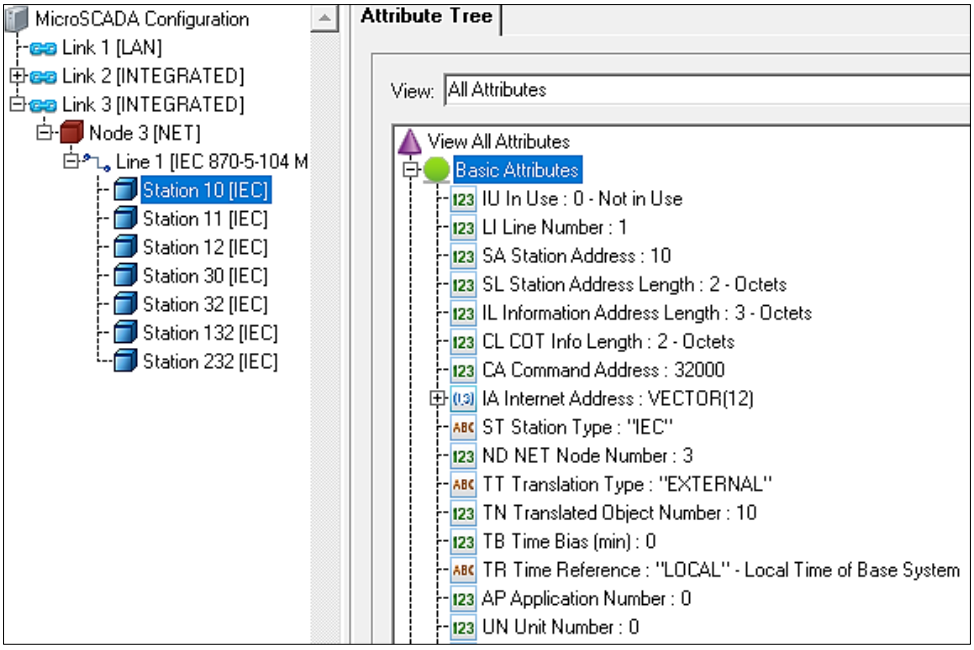


Figure 12. Configuration of T104 communication between SYS600 and RTU560 in PC-NET program of SYS600.

LN	IX	[UN]	[OA]/IN	[OB]/E	OI			
STA10_J01_Q1	10	10	1002		STA10	J01	Q1	Disconn. position indication
STA10_J01_Q1	13	10	1003		STA10	J01	Q1	Disconn. command
STA10_J01_Q1	15				STA10	J01	Q1	Disconn. device control block
STA10_J01_Q1	16				STA10	J01	Q1	Disconn. open interlocked
STA10_J01_Q1	17				STA10	J01	Q1	Disconn. close interlocked
STA10_J01_Q1	18				STA10	J01	Q1	Cause of interlocking
STA10_J01_Q1	19				STA10	J01	Q1	Disconn. selection on monitor
STA10_J01_Q1	20				STA10	J01	Q1	Disconn. command event
STA10_J01_Q1	113				STA10	J01	Q1	Disconn. command
STA10_J01_Q1	50010				STA10	J01	Q1	Topological state of Disconn. position indication

Figure 13. Process objects associated with the disconnector in the database of SYS600.

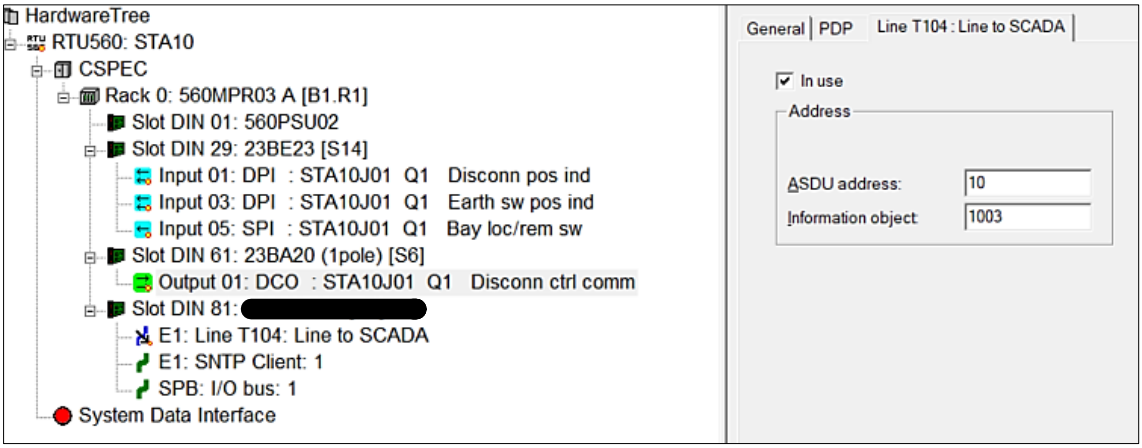


Figure 14. Configuration of the process object for controlling the disconnector for RTU560 in RTUtil configuration tool.

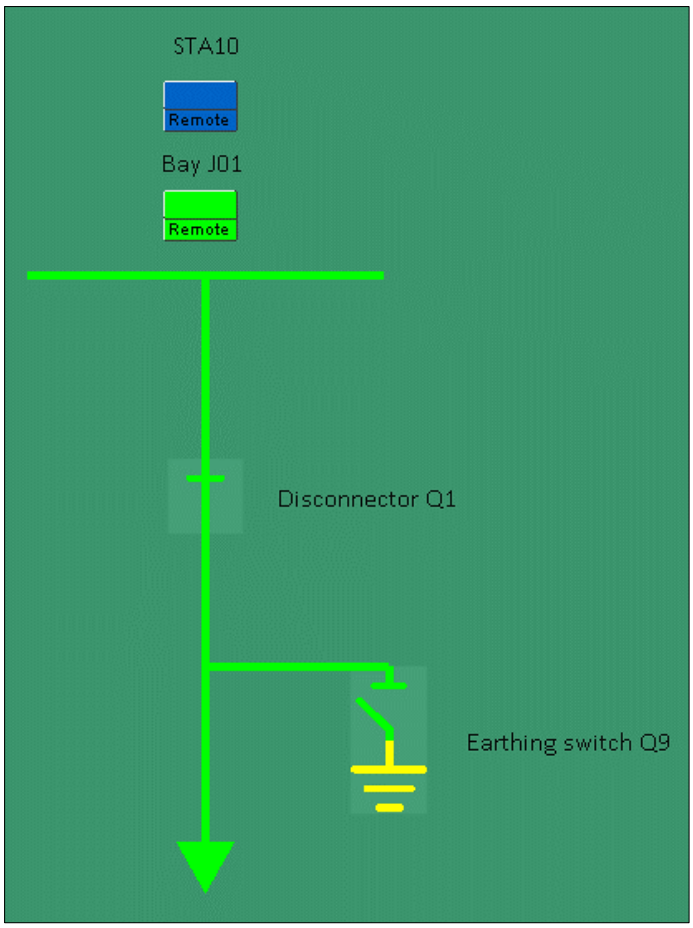


Figure 15. Display of the test setup in SYS600. In this situation the disconnector is closed.

Figure 16 presents the control command sent from SYS600 commanding RTU560 to open the disconnector. Figure 17 presents the same command in raw hexadecimal format. Regarding passive reconnaissance techniques discussed in Section 5.3 it can be seen that T104 communication is targeted to TCP port 2404 which can be used for identifying the controlled station (Saif Qassim et al., 2018, p. 156). Also, MAC addresses can be seen which can be used for making conclusions regarding the associated network assets (Erdődi et al., 2022, p. 5). All application layer data can be read and Wireshark can analyze the APCI and ASDU structures of the T104 message. TCP sequence numbers can be seen (Seq and Ack).

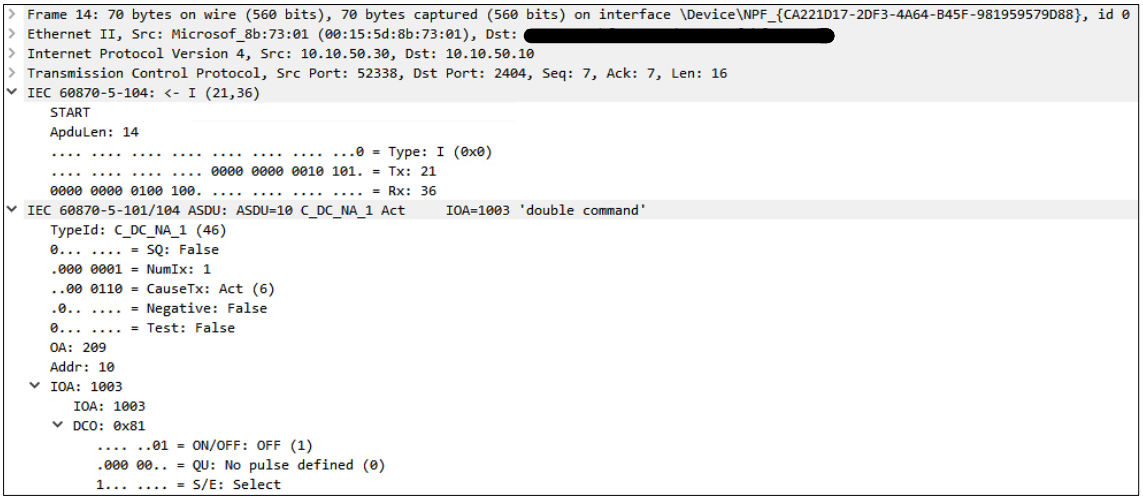


Figure 16. Select request of the two-phased double command ASDU for operating the disconnector.

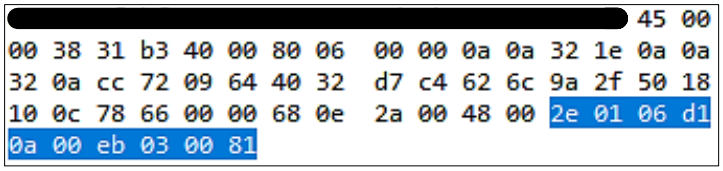


Figure 17. Raw hexadecimal data of the select request of the two-phased double command ASDU operating the disconnector presented in the highlighted part.

From the data unit identifier in the beginning of the ASDU it can be seen that this message activates a double command by carrying decimal value 46 (C_DC_NA_1) in the type identification field and indicating activation in the cause of transmission field (IEC, 2003, p. 98–99). Common address 10 of RTU560 can be seen as well as originator address 209 which refers to the station address of SYS600. After the data unit identifier is the information object with information object address 1003 carrying the actual control information in the information element. Information element has the structure of a double command (DCO) (IEC, 2003, p. 51) which is used by the double command ASDU (IEC, 2003, p. 98–99). Value 1 of the two least significant bits (binary value 01) indicates OFF (IEC, 2003, p. 51) and is used for opening the disconnector. Correspondingly, decimal value 2 (10 in binary) indicates ON (IEC, 2003, p. 51) and would be used for closing the disconnector.

In SYS600 control of the disconnector is defined to be a two-phased select and execute command specified in IEC 60870-5-5. Two-phased operation is also required in the configuration of RTU560. Idea of the two-phased operation is to provide security against accidental operation due to human or machine error (Clarke & Reynders, 2004, p. 114, 296). IEC (1995, p. 69, 71, 73) specifies that first in such command the controlling station must verify the readiness of the controlled station for performing a control operation by sending a preparing select request. If the controlled station can perform the aimed control operation it confirms it with an activation confirmation message. After this the controlling station can send a separate execute message requesting the controlled station to perform the action. If the controlled station consents the request it replies with an activation confirmation message with P/N-bit of the cause of transmission set to 0 indicating a positive confirmation (IEC, 2003, p. 37) and performs the requested command (IEC, 1995, p. 71). Finally, the controlled station sends an activation termination message to indicate that execution of the control process has finished (IEC, 1995, p. 71).

Figure 18 presents communication of the whole control operation. The first message presented in Figures 16 and 18 activates the select request. Cause of transmission field has the value 6 (activation) and S/E bit of the information element is set to 1 (select) (IEC, 2003, p. 54, 98–99). Controlled station permits the operation with an activation confirmation message. In the third message the controlling station sends the execute request for which the controlled station sends a confirmation and performs the command. Finally, the controlled station sends the termination message. As a side note, Figure 18 presents also the APCI sequence numbers of the T104 ASDUs increasing by one after each ASDU.

Conversely, if the controlled station refuses to execute a command it replies with a negative confirmation by setting the P/N-bit to 1 (IEC, 2003, p. 37). A demonstration was conducted in which a direct command was attempted instead of the required two-phased operation. Figure 19 presents a negative confirmation by the controlled station which requires two-phased operation for controlling the disconnector. As a direct execute is requested it refuses to operate and sends a negative confirmation in response.

Source	Destination	Protocol	Info
10.10.50.30	10.10.50.10	IEC 60870-5 ASDU	<- I (21,36) ASDU=10 C_DC_NA_1 Act IOA=1003
10.10.50.10	10.10.50.30	IEC 60870-5 ASDU	-> I (36,22) ASDU=10 C_DC_NA_1 ActCon IOA=1003
10.10.50.30	10.10.50.10	IEC 60870-5 ASDU	<- I (22,37) ASDU=10 C_DC_NA_1 Act IOA=1003
10.10.50.10	10.10.50.30	IEC 60870-5 ASDU	-> I (37,23) ASDU=10 C_DC_NA_1 ActCon IOA=1003
10.10.50.10	10.10.50.30	IEC 60870-5 ASDU	-> I (38,23) ASDU=10 C_DC_NA_1 ActTerm IOA=1003

Figure 18. T104 communication of the two-phased control operation of the disconnector.

```
> Frame 103: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface \Device\
> Ethernet II, Src: [redacted] Dst: Microsof_8b:73:01 (00:15:5d:
> Internet Protocol Version 4, Src: 10.10.50.10, Dst: 10.10.50.30
> Transmission Control Protocol, Src Port: 2404, Dst Port: 52338, Seq: 67, Ack: 73, Len: 16
> IEC 60870-5-104: -> I (39,24)
v IEC 60870-5-101/104 ASDU: ASDU=10 C_DC_NA_1 ActCon_NEGA IOA=1003 'double command'
    TypeId: C_DC_NA_1 (46)
    0... .... = SQ: False
    .000 0001 = NumIx: 1
    ..00 0111 = CauseTx: ActCon (7)
    .1.. .... = Negative: True
    0... .... = Test: False
    OA: 209
    Addr: 10
  v IOA: 1003
      IOA: 1003
    v DCO: 0x01
        .... ..01 = ON/OFF: OFF (1)
        .000 00.. = QU: No pulse defined (0)
        0... .... = S/E: Execute
```

Figure 19. Demonstration of a negative confirmation for a control command due to requesting for a direct execution instead of the required two-phased operation.

In the display of SYS600 the select and execute command procedure is seen as a two-stepped process. The select message is sent when an operator chooses to open the disconnector. After confirmation by the controlled station another dialogue is prompted allowing the operator to send the execute request. This is presented in Figure 20. Figure 21 presents the display of SYS600 after a successful opening of the disconnector.

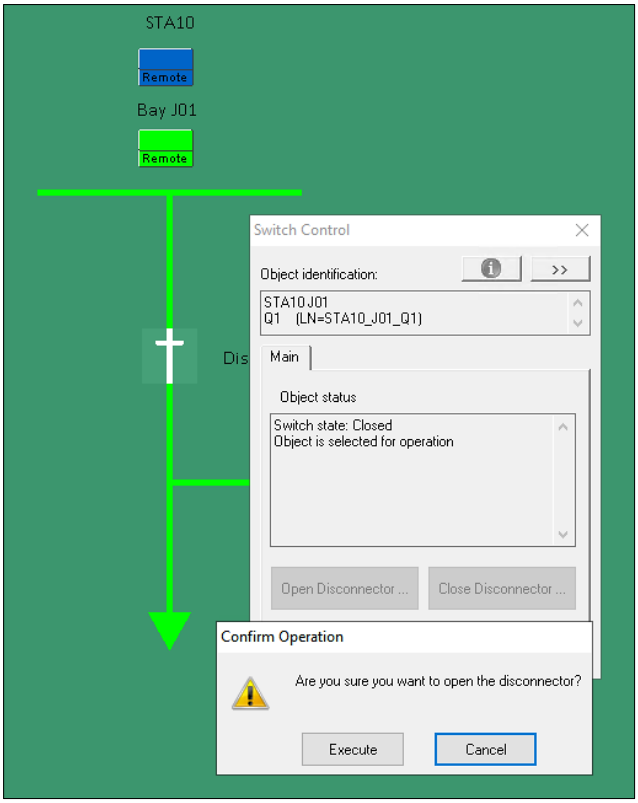


Figure 20. Two-phased control command for operating the disconnector in the display of SYS600.

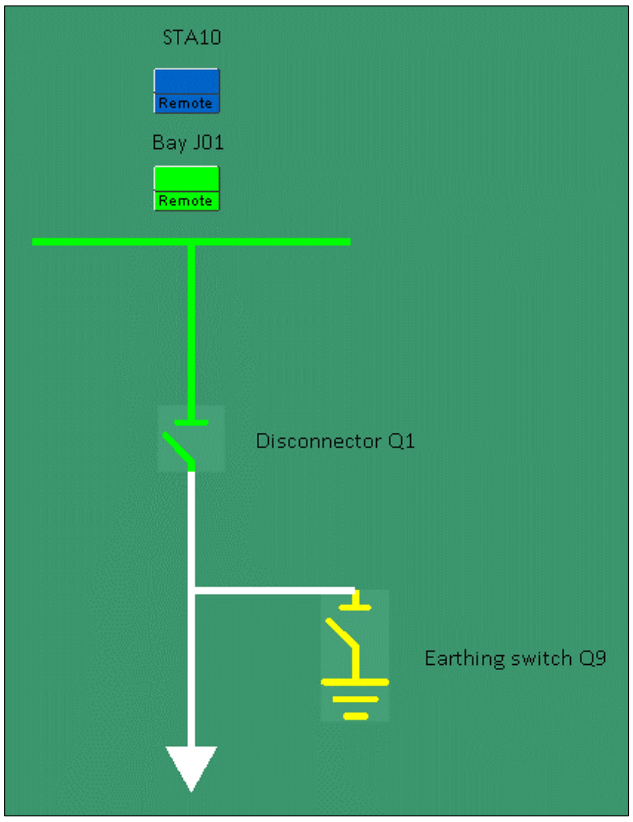


Figure 21. Display of SYS600 after the disconnector has been successfully opened.

7.2 TLS protection by IEC 62351-3

In the second demonstration TLS protection according to IEC 62351-3 was used for protecting T104 communication. Both SYS600 and RTU560 had implementations of IEC 62351-3 and were configured to use TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 cipher suite. The name of the cipher suite is a combination of the cryptographic algorithms used. According to Ciphersuite.info (n.d.) it is considered as a secure state-of-art cipher suite. Abbreviation ECDHE comes from Elliptic Curve Diffie-Hellman Ephemeral and it is the key exchange algorithm used by the cipher suite. According to Blake-Wilson et al. (2006, p. 3–7) it is a type of Diffie-Hellman key exchange algorithm based on elliptic curve cryptography and provides perfect forward secrecy. RSA cryptography is used for signing and authenticating the Diffie-Hellman parameters exchanged between parties. AES 256 GCM is the algorithm used for encrypting data and SHA384 is the hash algorithm used for integrity checking in conjunction with encryption (Ciphersuite.info, n.d.).

RSA authentication required public keys to be exchanged by the endpoints. XCA software based on OpenSSL was used for generating a self-signed root certificate acting as the CA. An RSA private-public key pair was generated for the CA. Individual RSA key pairs and leaf certificates signed with the private key of the CA were generated for SYS600 and RTU560. The individual certificate was imported to its corresponding device along with the root certificate containing the public key of the CA. In configurations of SYS600 and RTU560 the certificates were configured to be used for TLS protection of the T104 connection between them.

Demonstration of TLS protection of T104 communication was conducted. Unnecessary repetition regarding TLS protection introduced in Section 3.1 is avoided here but some notifications and points are remarked. Before T104 communication can be started the protected TLS connection is formed. SYS600 (10.10.50.30) acts as the client and starts to form the protected connection with RTU560 (10.10.50.32) by initiating a TLS handshake presented in Figure 22. Figure 23 presents the ServerHello message which includes the

cipher suite selected by RTU560 according to the proposals of the ClientHello message, TLS_ECDHE_RSA_WITH_AES_GCM_SHA384. Also, the server random used for generation of the master secret is presented. Mutual authentication takes place here and RTU560 requests the client certificate from SYS600 with a CertificateRequest message. After sending the ChangeCipherSpec message both participants turn into encryption and therefore the Finished message completing the handshake is also encrypted.

Source	Destination	Protocol	Info
10.10.50.30	10.10.50.32	TLSv1.2	Client Hello
10.10.50.32	10.10.50.30	TLSv1.2	Server Hello
10.10.50.32	10.10.50.30	TLSv1.2	Certificate, Server Key Exchange, Certificate Request, Server Hello Done
10.10.50.30	10.10.50.32	TLSv1.2	Certificate, Client Key Exchange, Certificate Verify, Change Cipher Spec, Encrypted Handshake Message
10.10.50.32	10.10.50.30	TLSv1.2	Change Cipher Spec, Encrypted Handshake Message

Figure 22. TLS handshake for forming the TLS protection of T104 communication. SYS600 acts as the client and RTU560 acts as the server.

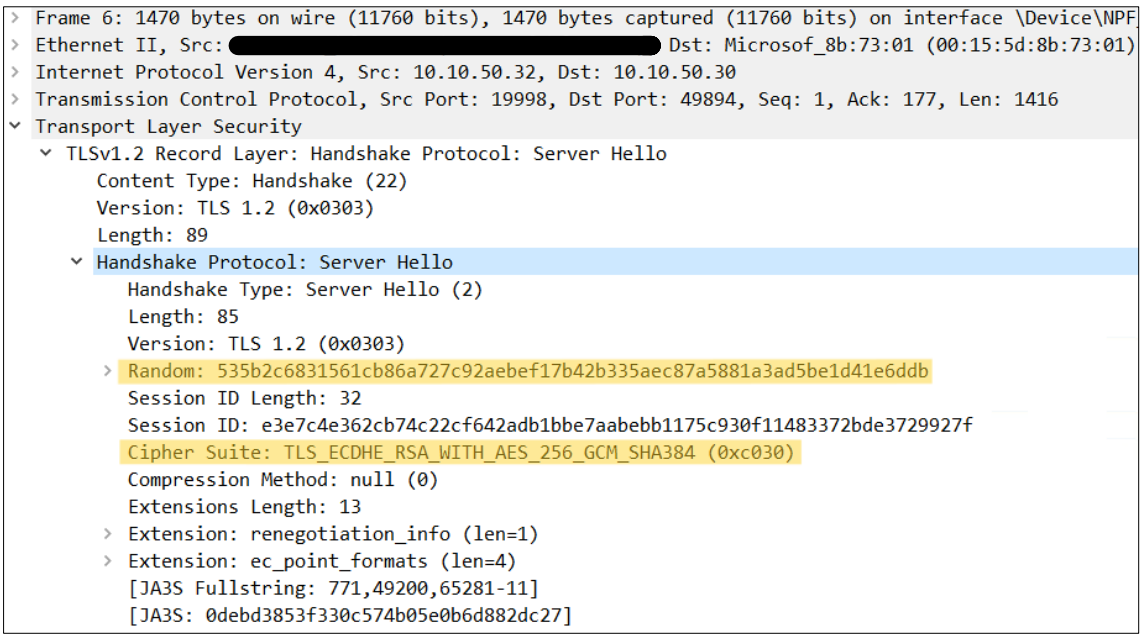


Figure 23. ServerHello message containing the cipher suite selected by RTU560 and the server random.

After a successful handshake TLS protected application data is exchanged. As seen from Figure 24 TLS record layer protocol is carried on top of the transport layer and TCP ports used by the endpoints can be seen. The default TCP port used for secured T104 is 19998 (IEC, 2013b, p. 37). TLS protection does not affect on the network layer and therefore original IP addresses are used. Devices in the same network can communicate with each

other using TLS protection as presented in Figure 25. It can be seen from Figure 24 that application layer data of T104 is carried encrypted by the record protocol of TLS.

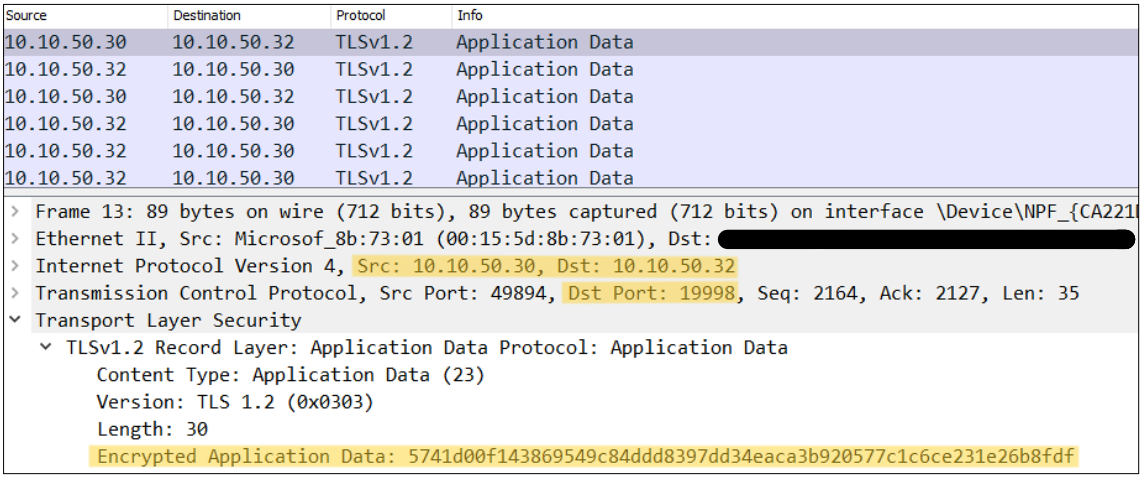


Figure 24. TLS protected T104 application data exchanged between SYS600 and RTU560.

Figure 25. Visualization of the TLS demonstration.

7.3 Protection with IPsec ESP tunnel

RTU560 has a built-in IPsec VPN functionality which was tested in the third demonstration. Two scenarios were implemented considering different network architectures. Figure 26 presents the network architecture of the first setup. The idea was to demonstrate a setup in which a control station network or a so-called SCADA network considered as trusted is connected to RTU560 at a remote site with an IPsec ESP tunnel through an untrusted network. IPFire firewall represents a firewall and a border router of the SCADA network in between the trusted and untrusted networks. IPFire also supports IPsec VPN

and an IPsec ESP tunnel was configured between IPFire and RTU560 to make the RTU accessible for the SCADA server.

Figure 26 presents the trusted SCADA network on the left with IP address area 192.168.100.0/24 and the untrusted network on the right with IP address area 10.10.50.0/24 in which the IPsec tunnel was formed. Behind the IPsec tunnel RTU560 had a virtual IP address 192.168.1.132 seen only through the IPsec tunnel. T104 communication from SYS600 to RTU560 was first routed from the SCADA server to the firewall server which in turn routed it in the VPN tunnel. The traffic in the SCADA network between the SCADA server and the firewall server was unprotected but traffic forwarded from the firewall server to RTU560 was protected with ESP protocol. It should be noted here that the type of untrusted network and network connection is not specified but the idea is to present how IPsec is typically deployed. Auxiliary accessories and security mechanisms should be implemented depending on the use case and are not considered here.

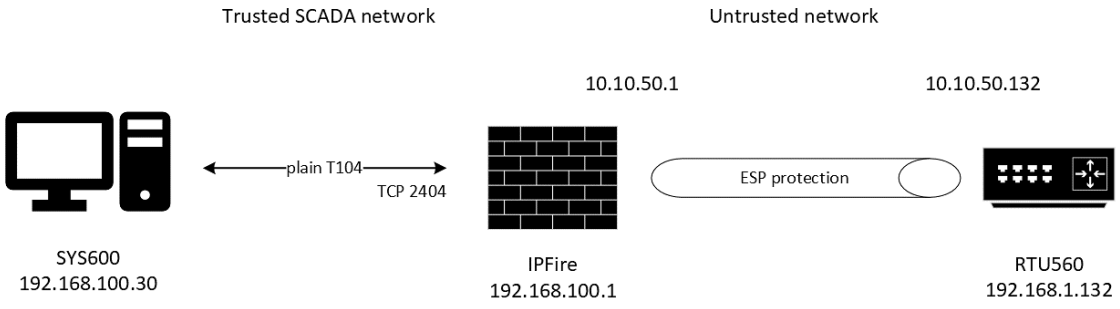


Figure 26. Setup of the first IPsec demonstration.

Parameters for creating security associations needed to be configured for both endpoint devices of the IPsec tunnel. Selections of the cryptographic algorithms were made randomly for demonstration purposes, however, in practice cryptographically secure algorithms and options should be selected. For example, National Institute of Standards and Technology has defined approved algorithms and options for IKE and IPsec security associations (Barker et al., 2020, vii–viii). The older key exchange protocol IKEv1 was used

as it was presented in the theory part as reference for the introduction of IPsec. For the demonstration purpose pre-shared key was used for authentication due to its simplicity, however, usage of public keys or digital signatures is recommended (Barker et al., 2020, p. 66–69).

The public IP addresses used for forming the IPsec tunnel needed to be configured for both ends (10.10.50.1 and 10.10.50.132). Also, private networks connected through the IPsec tunnel needed to be defined (192.168.100.0/24 and 192.168.1.0/24). Common algorithms for different protection measures needed to be defined for ISAKMP security associations of phase 1 and IPsec security associations of phase 2 including the encryption algorithm, the integrity protection algorithm, and the key exchange algorithm. Figure 27 presents an example of configuring security association attributes of phase 1 and phase 2 in the graphical user interface of IPFire firewall server.

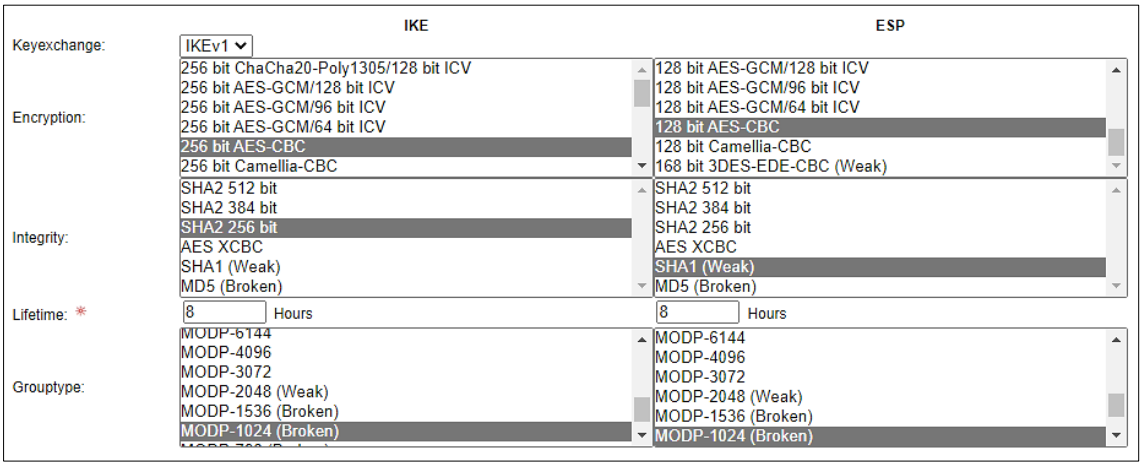


Figure 27. Configuration of phase 1 and phase 2 security association attributes in the graphical user interface of IPFire.

In Figure 28 the formation of the IPsec tunnel is presented containing all six negotiation messages of main mode and three negotiation messages of quick mode. The negotiation is done through the default port of IKE in both ends, UDP 500, by ISAKMP protocol. ISAKMP (Internet Security Association and Key Management Protocol) is a general framework for negotiating security associations for different network security services

at various layers of networking and data communication (Turner et al., 1998, p. 4–5), in this case IPsec. RTU560 acts as the initiator of the connection and the firewall server as the responder.

Source	Destination	Protocol	Info
10.10.50.132	10.10.50.1	ISAKMP	Identity Protection (Main Mode)
10.10.50.1	10.10.50.132	ISAKMP	Identity Protection (Main Mode)
10.10.50.132	10.10.50.1	ISAKMP	Identity Protection (Main Mode)
10.10.50.1	10.10.50.132	ISAKMP	Identity Protection (Main Mode)
10.10.50.132	10.10.50.1	ISAKMP	Identity Protection (Main Mode)
10.10.50.1	10.10.50.132	ISAKMP	Identity Protection (Main Mode)
10.10.50.132	10.10.50.1	ISAKMP	Informational
10.10.50.132	10.10.50.1	ISAKMP	Quick Mode
10.10.50.1	10.10.50.132	ISAKMP	Quick Mode
10.10.50.132	10.10.50.1	ISAKMP	Quick Mode

Figure 28. Messages of IKE phases 1 and 2 for negotiating the IPsec connection.

Figure 29 presents the first main mode message sent by RTU560 which is carried and structured by the ISAKMP protocol. The ISAKMP message is constructed of a ISAKMP header and a set of payloads carrying the data used by IKE as explained by Tiller (2000, p. 164, 166–167). An individual payload in turn constructs of a general payload header and the data specified by the type of the payload. Multiple payloads are chained to each other with the generic payload headers and certain payloads are nested into another to carry information regarding the upper-level payload.

In the beginning of the ISAKMP message the ISAKMP header can be seen followed by the payloads. In the first message the RTU sends its cookie in the Initiator SPI field and its proposals for the security association attributes in the security association payload (Tiller, 2000, p. 164). Further, the security association payload contains a proposal payload which in turn carries a transform payload with the actual proposals for IKE attributes for phase 1 (Tiller, 2000, p. 166–170). Figure 30 presents the transform payload containing the proposed ISAKMP security association attributes which were configured in Figure 27. The responder sends its cookie and selection of IKE attributes in response.

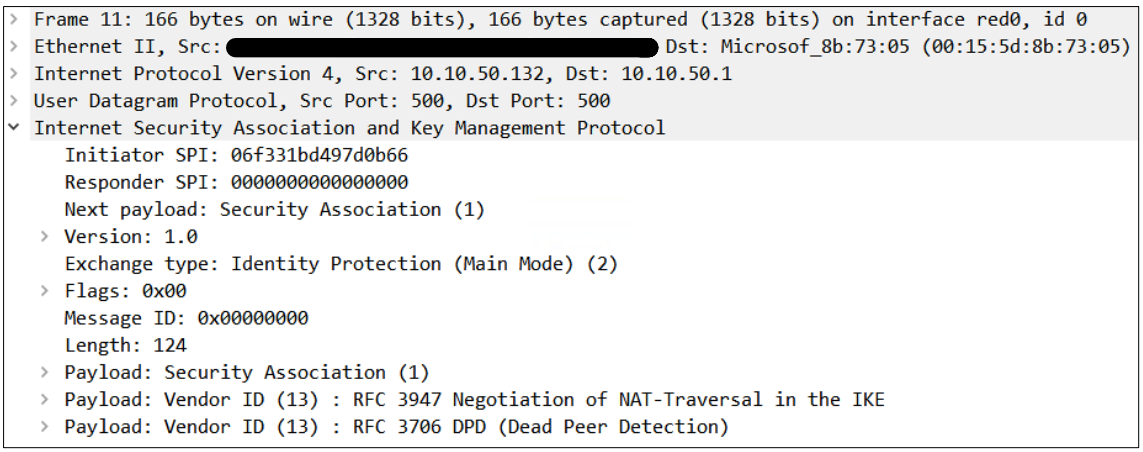


Figure 29. First main mode message sent by RTU560.

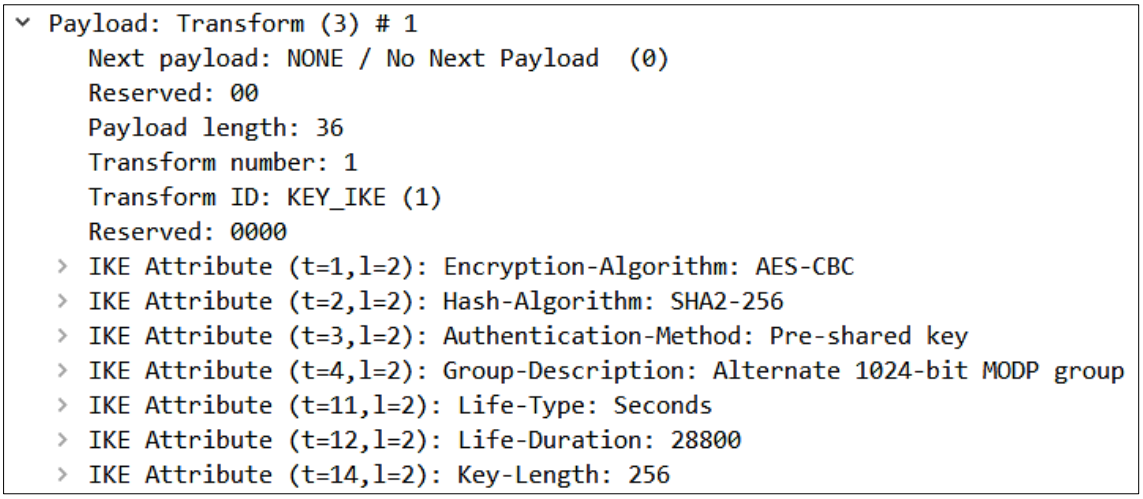


Figure 30. Proposals for the phase 1 security association attributes by RTU560.

In the third message presented in Figure 31 RTU560 sends its nonce in a nonce payload and the public Diffie-Hellman data required for calculating the shared secret in a key exchange payload (Forouzan, 2008, p. 582, 585; Turner et al., 1998, p. 31, 38). The responder responds with the corresponding attributes. After this the participants can calculate the shared secrets and turn to encrypt ISAKMP payloads in the fifth and sixth messages. Figure 32 presents the fifth message. Also, quick mode messages are encrypted as presented in Figure 33. It can be noted that the cookies exchanged in the first two messages are carried along in all messages to identify the negotiation.

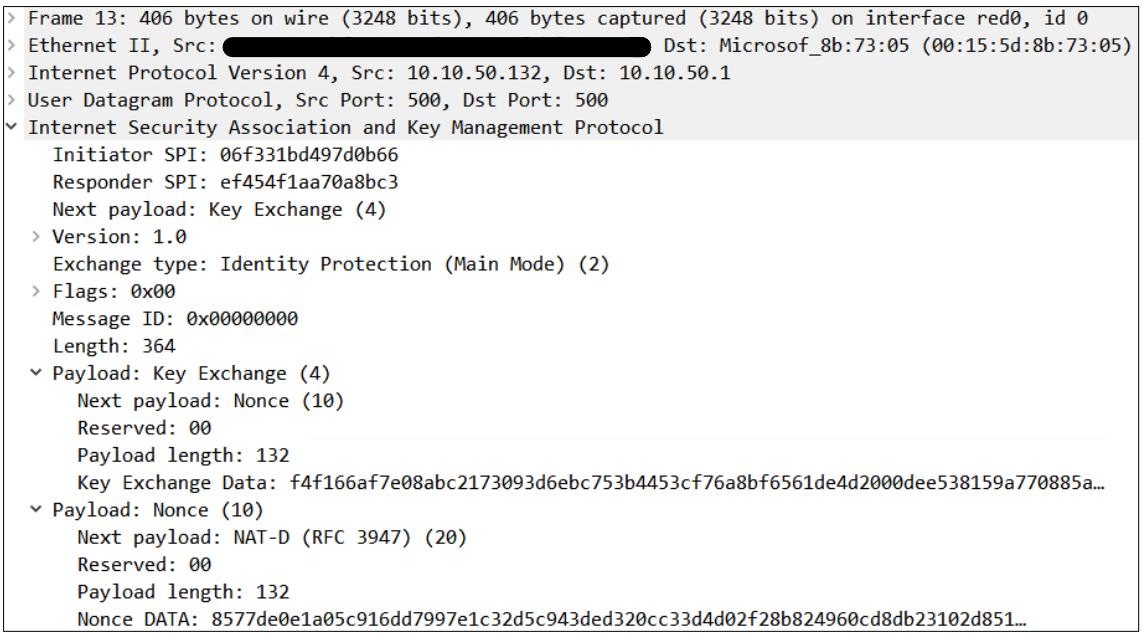


Figure 31. Third main mode message sent by RTU560 containing its nonce and Diffie-Hellman public data.

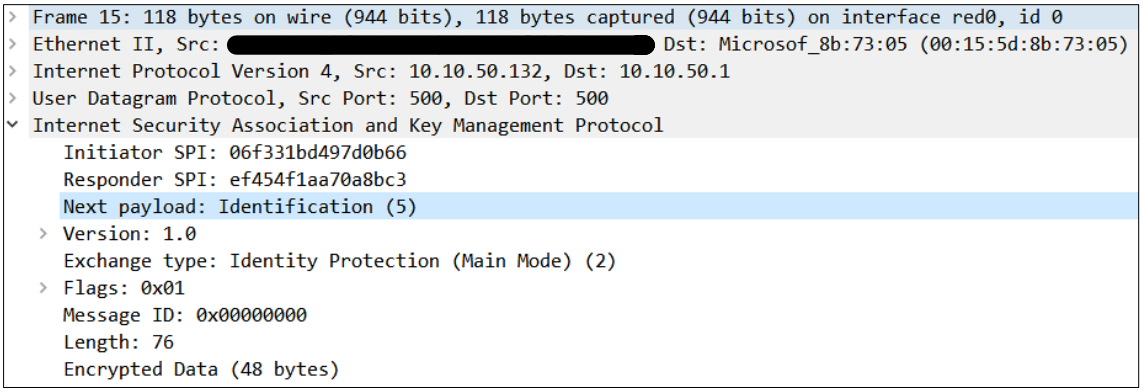


Figure 32. Fifth main mode message sent by RTU560 with encrypted payload.

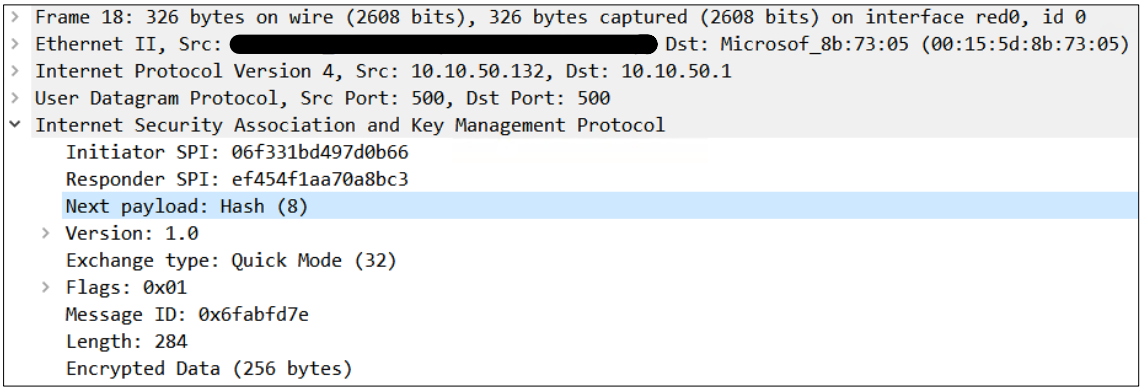


Figure 33. First encrypted quick mode message sent by RTU560 carrying a hash.

After the security associations for IPsec have been created traffic is encapsulated by ESP protocol. Figure 34 presents ESP protected traffic and Figure 35 presents an ESP datagram transferred on top of the IP layer. Wireshark can interpret the ESP header but not other contents. IP datagrams of these packets use the IP addresses of the tunnel endpoints. ESP does not use a transport layer port but is bound to a security parameter index representing the security association negotiated by IKE. As discussed in Section 3.2.2, inbound and outbound datagrams have different security parameter indexes.

Source	Destination	Protocol	Info
10.10.50.1	10.10.50.132	ESP	ESP (SPI=0xcf8875d2)
10.10.50.132	10.10.50.1	ESP	ESP (SPI=0xcff7848b)
10.10.50.1	10.10.50.132	ESP	ESP (SPI=0xcf8875d2)
10.10.50.1	10.10.50.132	ESP	ESP (SPI=0xcf8875d2)
10.10.50.132	10.10.50.1	ESP	ESP (SPI=0xcff7848b)
10.10.50.132	10.10.50.1	ESP	ESP (SPI=0xcff7848b)
10.10.50.1	10.10.50.132	ESP	ESP (SPI=0xcf8875d2)
10.10.50.132	10.10.50.1	ESP	ESP (SPI=0xcff7848b)

Figure 34. ESP protected traffic.

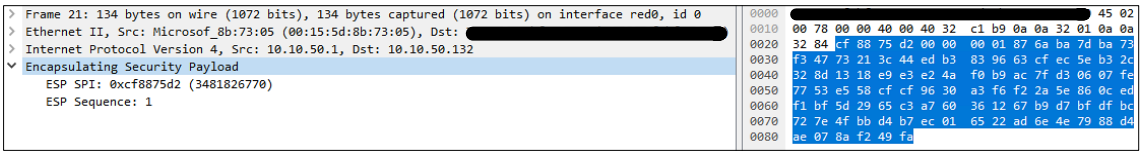
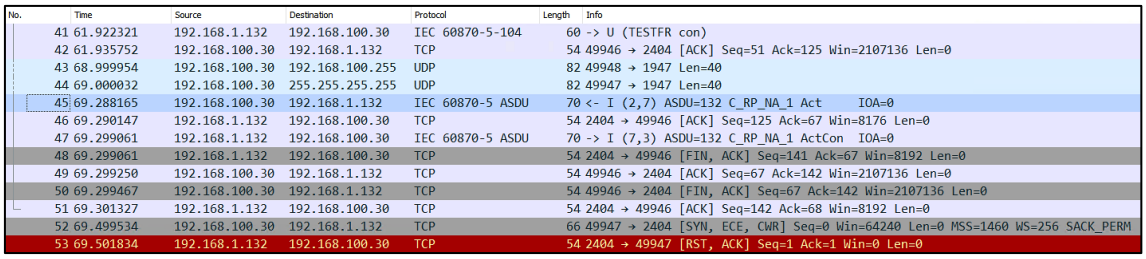


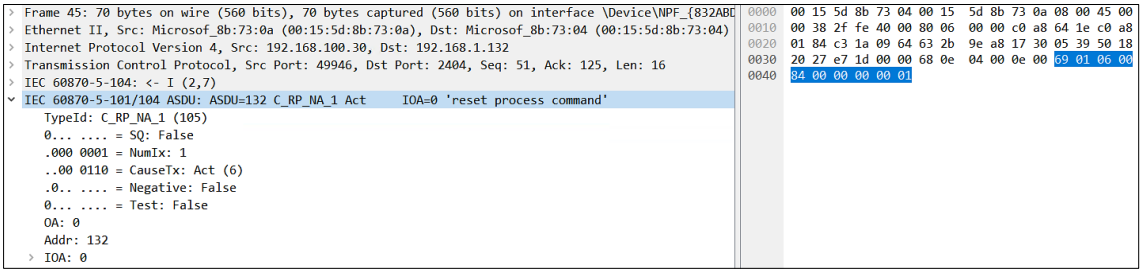
Figure 35. An IP datagram carrying ESP protected data.

Furthermore, Figure 36 presents unprotected T104 traffic between the SCADA server and IPFire. In the demonstration a reset command C_RP_NA_1 was sent from SYS600 to RTU560 which can be seen in Figure 36 in packet number 45 and its contents are presented in Figure 37. Figure 38 presents unfiltered traffic captured simultaneously from the public interface of IPFire. The reset command is most probably carried by the ESP protected packet number 80 which is also presented in Figure 39.



No.	Time	Source	Destination	Protocol	Length	Info
41	61.922321	192.168.1.132	192.168.100.30	IEC 60870-5-104	60	-> U (TESTFR con)
42	61.935752	192.168.100.30	192.168.1.132	TCP	54	49946 → 2404 [ACK] Seq=51 Ack=125 Win=2107136 Len=0
43	68.999954	192.168.100.30	192.168.100.255	UDP	82	49948 → 1947 Len=40
44	69.000032	192.168.100.30	255.255.255.255	UDP	82	49947 → 1947 Len=40
45	69.288165	192.168.100.30	192.168.1.132	IEC 60870-5 ASDU	70	<- I (2,7) ASDU=132 C_RP_NA_1 Act IOA=0
46	69.290147	192.168.1.132	192.168.100.30	TCP	54	2404 → 49946 [ACK] Seq=125 Ack=67 Win=8176 Len=0
47	69.299061	192.168.1.132	192.168.100.30	IEC 60870-5 ASDU	70	-> I (7,3) ASDU=132 C_RP_NA_1 ActCon IOA=0
48	69.299061	192.168.1.132	192.168.100.30	TCP	54	2404 → 49946 [FIN, ACK] Seq=141 Ack=67 Win=8192 Len=0
49	69.299250	192.168.100.30	192.168.1.132	TCP	54	49946 → 2404 [ACK] Seq=67 Ack=142 Win=2107136 Len=0
50	69.299467	192.168.100.30	192.168.1.132	TCP	54	49946 → 2404 [FIN, ACK] Seq=67 Ack=142 Win=2107136 Len=0
51	69.301327	192.168.1.132	192.168.100.30	TCP	54	2404 → 49946 [ACK] Seq=142 Ack=68 Win=8192 Len=0
52	69.499534	192.168.100.30	192.168.1.132	TCP	66	49947 → 2404 [SYN, ECE, CWR] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
53	69.501834	192.168.1.132	192.168.100.30	TCP	54	2404 → 49947 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Figure 36. Unprotected traffic between the SCADA server and IPFire. Packet number 45 activates the reset command.



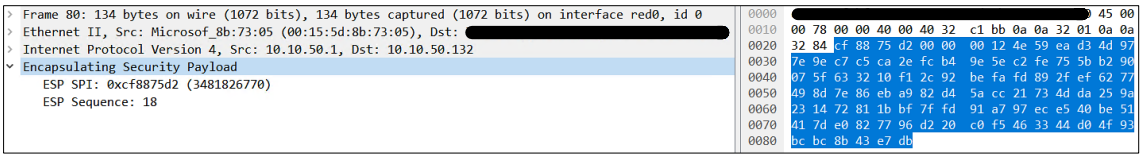
```
> Frame 45: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface \Device\NPF_{832ABD
> Ethernet II, Src: Microsof_8b:73:0a (00:15:5d:8b:73:0a), Dst: Microsof_8b:73:04 (00:15:5d:8b:73:04)
> Internet Protocol Version 4, Src: 192.168.100.30, Dst: 192.168.1.132
> Transmission Control Protocol, Src Port: 49946, Dst Port: 2404, Seq: 51, Ack: 125, Len: 16
> IEC 60870-5-104: <- I (2,7)
v IEC 60870-5-101/104 ASDU: ASDU=132 C_RP_NA_1 Act     IOA=0 'reset process command'
    TypeId: C_RP_NA_1 (105)
    0... .... = SQ: False
    .000 0001 = NumIx: 1
    ..00 0110 = CauseTx: Act (6)
    .0.. .... = Negative: False
    0... .... = Test: False
    OA: 0
    Addr: 132
  > IOA: 0
```

```
0000  00 15 5d 8b 73 04 00 15  5d 8b 73 0a 08 00 45 00
0010  00 38 2f fe 40 00 80 06  00 00 c0 a8 64 1e c0 a8
0020  01 84 c3 1a 09 64 63 2b  9e a8 17 30 05 39 50 18
0030  20 27 e7 1d 00 00 68 0e  04 00 0e 00 69 01 06 00
0040  84 00 00 00 00 01
```

Figure 37. T104 reset command sent from SYS600 which is unprotected in the SCADA network.

No.	Time	Source	Destination	Protocol	Length	Info
78	51.181094700	192.168.1.132	192.168.100.30	IEC 60870-5-104	60	-> U (TESTFR con)
79	51.195334300	10.10.50.1	10.10.50.132	ESP	118	ESP (SPI=0xcf8875d2)
80	58.547730000	10.10.50.1	10.10.50.132	ESP	134	ESP (SPI=0xcf8875d2)
81	58.548847200	10.10.50.132	10.10.50.1	ESP	118	ESP (SPI=0xcff7848b)
82	58.548847200	192.168.1.132	192.168.100.30	TCP	54	2404 → 49946 [ACK] Seq=125 Ack=67 Win=8176 Len=0
83	58.557325200	10.10.50.132	10.10.50.1	ESP	134	ESP (SPI=0xcff7848b)
84	58.557325300	10.10.50.132	10.10.50.1	ESP	118	ESP (SPI=0xcff7848b)
85	58.557325200	192.168.1.132	192.168.100.30	IEC 60870-5 ASDU	70	-> I (7,3) ASDU=132 C_RP_NA_1 ActCon IOA=0
86	58.557325300	192.168.1.132	192.168.100.30	TCP	54	2404 → 49946 [FIN, ACK] Seq=141 Ack=67 Win=8192 Len=0
87	58.558775800	10.10.50.1	10.10.50.132	ESP	118	ESP (SPI=0xcf8875d2)
88	58.558843800	10.10.50.1	10.10.50.132	ESP	118	ESP (SPI=0xcf8875d2)
89	58.560133900	10.10.50.132	10.10.50.1	ESP	118	ESP (SPI=0xcff7848b)
90	58.560133900	192.168.1.132	192.168.100.30	TCP	54	2404 → 49946 [ACK] Seq=142 Ack=68 Win=8192 Len=0
91	58.759516000	10.10.50.1	10.10.50.132	ESP	134	ESP (SPI=0xcf8875d2)
92	58.760506900	10.10.50.132	10.10.50.1	ESP	118	ESP (SPI=0xcff7848b)
93	58.760506900	192.168.1.132	192.168.100.30	TCP	54	2404 → 49947 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Figure 38. Traffic from the public interface of IPFire containing ESP protected datagrams. Packet number 80 is most probably the reset command.



```
> Frame 80: 134 bytes on wire (1072 bits), 134 bytes captured (1072 bits) on interface red0, id 0
> Ethernet II, Src: Microsof_8b:73:05 (00:15:5d:8b:73:05), Dst: [redacted]
> Internet Protocol Version 4, Src: 10.10.50.1, Dst: 10.10.50.132
v Encapsulating Security Payload
    ESP SPI: 0xcf8875d2 (3481826770)
    ESP Sequence: 18
```

```
0000  [redacted] 45 00
0010  00 78 00 00 40 00 40 32  c1 bb 0a 0a 32 01 0a 0a
0020  32 84 cf 88 75 d2 00 00  00 12 4e 59 ea d3 4d 97
0030  7e 9e c7 c5 ca 2e fc b4  9e 5e c2 fe 75 5b b2 90
0040  07 5f 63 32 10 f1 2c 92  be fa fd 89 2f ef 62 77
0050  49 8d 7e 86 eb a9 82 d4  5a cc 21 73 4d da 25 9a
0060  23 14 72 81 1b bf 7f fd  91 a7 97 ec e5 40 be 51
0070  41 7d e0 82 77 96 d2 20  c0 f5 46 33 44 d0 4f 93
0080  bc bc 8b 43 e7 db
```

Figure 39. T104 reset command originated from SYS600 which is ESP protected in the IPsec tunnel.

An experimental test case presented in Figure 40 was to form an IPsec tunnel between the Windows firewall of the SCADA server and RTU560. This was successfully done using the Connection Security Rules of the Windows Defender Firewall which enable to form

different types of IPsec connections with selected endpoints. The IPsec connection was a similar net-to-net connection as in the previous demonstration. Two different IP addresses were given for the network interface of the Windows server, one for forming the IPsec tunnel and the other IP address acting as the internal SCADA network IP address seen by RTU560 through the tunnel. The virtual IP address of RTU560 was again seen by the SCADA server through the tunnel. A static route was defined to route traffic to the virtual IP address of RTU560 through the IPsec tunnel. Figure 41 presents the security associations of phase 1 and phase 2 created to the Windows firewall of the SCADA server after the formation of the IPsec tunnel. Similar settings for security association attributes were used as in the first IPsec demonstration.

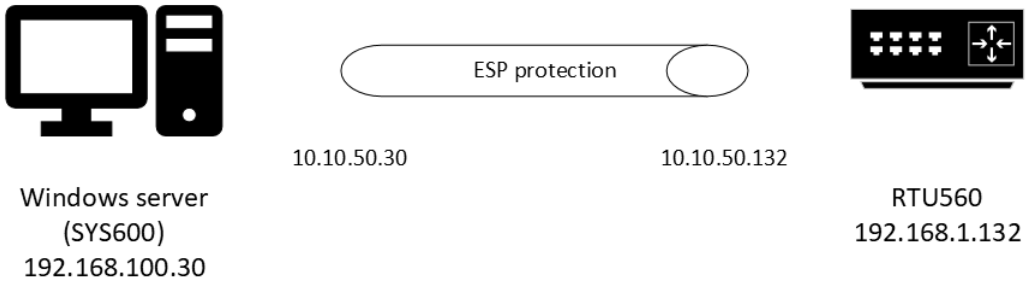


Figure 40. Setup of the second IPsec demonstration. ESP tunnel is formed from the Windows firewall of the SCADA server to RTU560 providing end-to-end protection between the endpoint hosts.

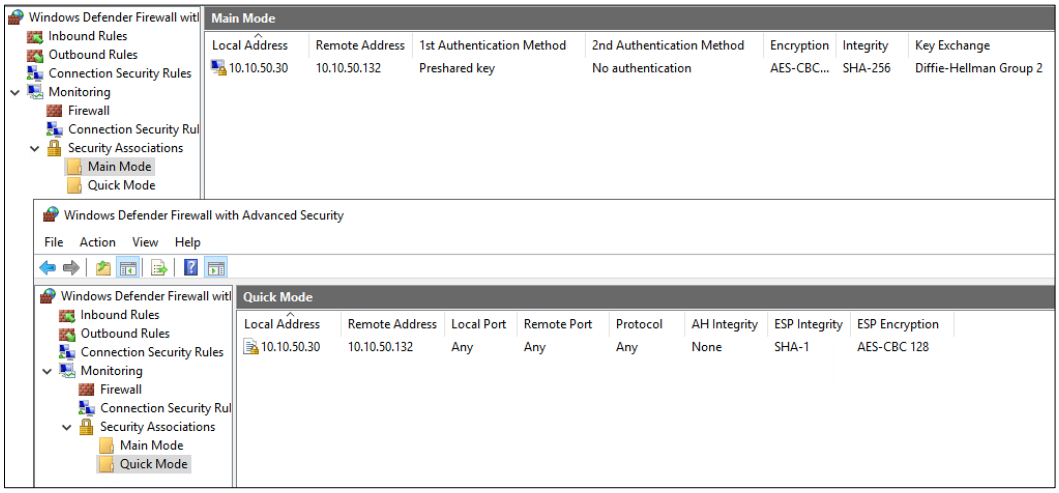


Figure 41. Security associations of phase 1 and phase 2 created to the Windows firewall of the SCADA server.

Couple of notes can be done from the second IPsec setup. First, as the protection is performed on the firewall of the operating system it is independent of SYS600. Second, compared to the first IPsec demonstration this setup provides an end-to-end protection for network traffic at host level. As the virtual IP address of the RTU is only visible through the IP tunnel it is only visible for the Windows server as it hosts the IPsec tunnel. This provides a sort of network segmentation as only the Windows server can access the RTU from the SCADA network. However, as the IPsec tunnel is formed straight to the SCADA server the presented setup as such should be only used in trusted networks to avoid the SCADA server directly facing an untrusted network.

7.4 Application layer authentication according to IEC 62351-5

In the last demonstration authentication extensions for T104 according to IEC 62351-5 were tested. Similar demonstration was performed for controlling a disconnector as in Section 7.1 with the addition of the challenge-response mechanism. Pre-shared update keys were deployed for delivering session keys used for authenticating critical ASDUs. Configuration of the demonstration required defining a user for RTU560 and adding it also to a user management tool of SYS600 acting as the authority. A common update key needed to be defined between RTU560 and the user management tool and a common MAC algorithm for calculating authentication responses needed to be specified. Also, a so-called key wrap algorithm needed to be specified which is used by the controlling station to encrypt session keys with the update key for the delivery to the controlled station (IEC, 2013a, p. 20, 91).

The demonstration presents how the access rights of a preconfigured RTU user can be linked with a user in SYS600. The protocol based online user management functionality described in Section 6.2 was not tested so therefore this is not an all-encompassing demonstration of IEC 62351-5. However, the challenge-response mechanism is demonstrated as well as an idea of access control provided by IEC 62351-5 is given.

Figure 42 presents a similar two-phased command procedure of a disconnector sent from SYS600 to RTU560 as presented in Section 7.1 with the addition of the challenge-response mechanism. As a critical double command ASDU is sent to RTU560 it challenges the sender of the message which in this case is the user linked between SYS600 and RTU560 who originated the command. SYS600 sends a reply for the challenge and as the authentication succeeds RTU560 confirms the select message. Same procedure is repeated for the execute phase.

Figure 43 presents the first challenge message sent by RTU560 and Figure 44 presents the reply from SYS600. Structures of the ASDUs, as well as structures of other ASDUs of T104 regarding IEC 62351-5, are specified in the standard IEC 60870-5-7 (IEC, 2013b, p. 7). In the challenge message in Figure 43 the blue highlighted hexadecimal value indicates the algorithm to be used for the MAC computation of the response value 4 representing HMAC_SHA256 (IEC, 2013a, p. 34; IEC, 2013b, p. 16). The yellow value in the end of the message is the challenge data of RTU560 to be compounded with the MAC calculation (IEC, 2013b, 16). In Figure 44 presenting the response the red value indicates the number of the user (8) originating the critical ASDU and the yellow value is the result of the MAC computation used for authentication (IEC, 2013b, p. 17).

```
10.10.50.30     10.10.50.232    IEC 60870-5 ASDU    <- I (58,55) ASDU=232 C_DC_NA_1 Act     IOA=1203
10.10.50.232    10.10.50.30     IEC 60870-5 ASDU    -> I (55,59) ASDU=232 S_CH_NA_1 Auth    IOA=3521
10.10.50.30     10.10.50.232    IEC 60870-5 ASDU    <- I (59,56) ASDU=232 S_RP_NA_1 Auth    IOA=3563
10.10.50.232    10.10.50.30     IEC 60870-5 ASDU    -> I (56,60) ASDU=232 C_DC_NA_1 ActCon  IOA=1203
10.10.50.30     10.10.50.232    IEC 60870-5 ASDU    <- I (60,57) ASDU=232 C_DC_NA_1 Act     IOA=1203
10.10.50.232    10.10.50.30     IEC 60870-5 ASDU    -> I (57,61) ASDU=232 S_CH_NA_1 Auth    IOA=3777
10.10.50.30     10.10.50.232    IEC 60870-5 ASDU    <- I (61,58) ASDU=232 S_RP_NA_1 Auth    IOA=3820
10.10.50.232    10.10.50.30     IEC 60870-5 ASDU    -> I (58,62) ASDU=232 C_DC_NA_1 ActCon  IOA=1203
10.10.50.232    10.10.50.30     IEC 60870-5 ASDU    -> I (59,62) ASDU=232 C_DC_NA_1 ActTerm IOA=1203
```

Figure 42. The critical two-phased double command procedure originated from SYS600 to RTU560 with the challenge-response mechanism.

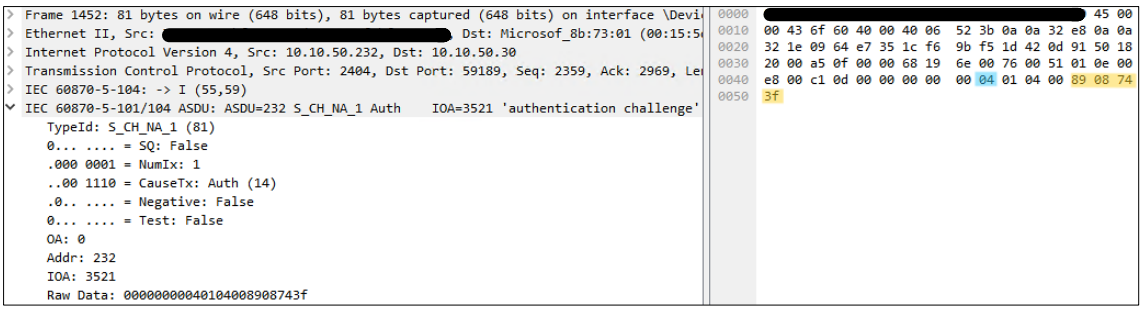


Figure 43. RTU560 challenges the critical double command.

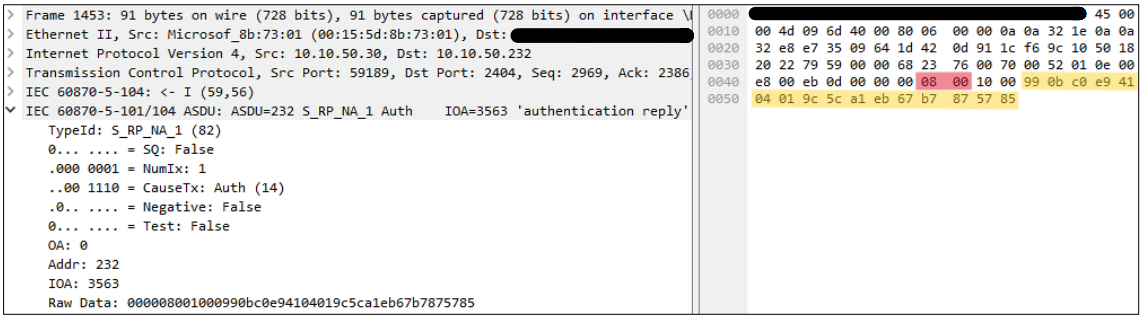


Figure 44. SYS600 replies to the challenge.

Figures 45 and 46 represent a situation in which the user linked between SYS600 and RTU560 does not have sufficient privileges for controlling the disconnector. The role of the user number 8 was changed from an operator to a viewer in the configuration of RTU560 making it insufficient for operating the disconnector. Figure 45 presents that failure of authentication aborts the control operation. Figure 46 presents the terminating authentication error ASDU specifying with error code 7 (highlighted with blue) that the authentication of user number 8 (highlighted with red) is successful but it is not permitted to perform the requested operation (IEC, 2013a, p. 44; IEC, 2013b, p. 22).

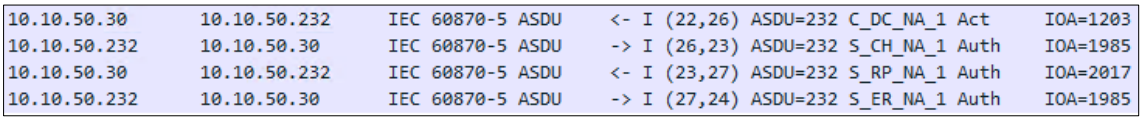


Figure 45. Termination of the critical double command due to the authentication failure.

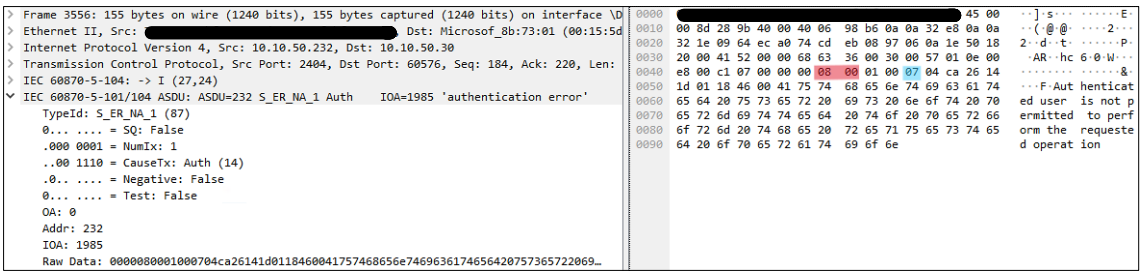


Figure 46. Authentication error due to the insufficient user privileges.

Another authentication failure terminating the control operation is presented in Figures 47 and 48. In this case the MAC algorithm of the authentication response was intentionally configured wrong leading to a wrong authentication response. The idea was to simulate a situation in which the contents of the message would not match the authentication tag which would happen if the contents of the message are tampered in transmission. Figure 47 presents that the wrong authentication tag leads into the termination of the control procedure. Error code 1 (highlighted with blue) in Figure 48 specifies that this happens due to the incorrect result of MAC calculation used for authentication (IEC, 2013a, p. 44; IEC, 2013b, p. 22).

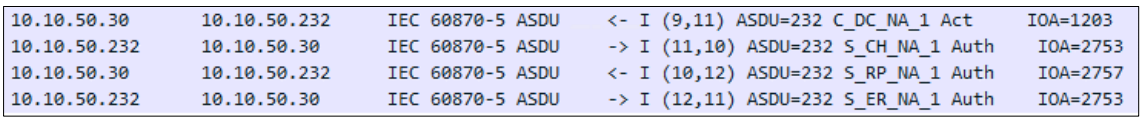


Figure 47. Termination of the critical double command due to the authentication failure.

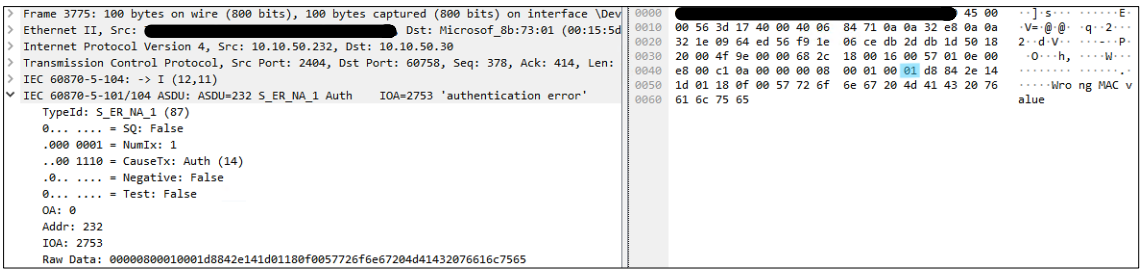


Figure 48. Authentication error due to the wrong result of the MAC calculation.

8 Conclusions

Three different techniques were investigated to protect IEC 60870-5-104 protocol which inherently lacks encryption, authentication, and integrity checking of transmitted data. It was clarified that each technique operates on a different layer of networking and data communication. According to the abstract OSI model and the concrete TCP/IP protocol suite IPsec operates on the network layer, TLS on the presentation layer on top of the transport layer securing application layer data, and IEC 62351-5 authentication mechanism is built into the application layer protocol and extends functions of the user process layer of T104. Each technique operates independently from each other. Basically, all techniques could be used simultaneously providing security on multiple layers, however, this was not tested in practice. One should be able add authentication extensions to T104, protect the application layer data with TLS, and encapsulate the IP datagram carrying the TCP payload with IPsec.

TLS handshake authenticates software process endpoints: T104 client of the controlling station and T104 server of the controlled station authenticate each other with certificates and TLS itself provides end-to-end protection for T104 communication by encrypting and checking integrity of T104 application data. From network layer perspective TLS protection does not make changes to T104 communication: protection is implemented on top of the transport layer and the TCP connection so endpoints in the same subnet can communicate directly to each other similarly as using unprotected T104. As TLS does not provide network isolation for the endpoints but exposes them for a direct or indirect connection from the intermediary network it is not suitable to be used in the case of untrusted networks.

Perhaps the most suitable use case for TLS protection would be to protect T104 communication when endpoints are connected via trusted networks such as private data links of a service provider. This provides protection against an attacker who manages to penetrate into the control system network such as in cases presented in Section 5.3. An attacker trying to sniff network traffic can see the TLS protected traffic sent from a SCADA

server to TCP port 19998 of an RTU but cannot see contents of T104 messages. Without authentication and negotiated secrets the attacker cannot get involved with the TLS connection. The attacker could possibly alter contents of the encrypted application data but MAC integrity checks should expose the changes. Sequence number of a TLS record is included in the MAC protecting against replay attacks (Rescorla & Dierks, 2008, p. 94–95; Ristić, 2014, p. 42). Denial-of-service attacks could be conducted against TLS. For example, the attacker could perform ARP spoofing and block protected T104 traffic. ARP spoofing should work because it affects on lower layers of the TCP/IP protocol suite.

IPsec differs from the other two techniques in the way that it is not built in the T104 client or server software but requires a network device which supports it. ESP tunnel investigated in this thesis connects two separate networks together with a protective tunnel through an intermediary network so that the data transferred between the networks is protected in the intermediary network. Another remarkable characteristic of IPsec tunneling is that it isolates the networks behind its endpoints so that they can only be reached through the tunnel providing network segmentation. Due to the isolation networks behind the tunnel endpoints do not face the intermediary network directly or indirectly which is desired if the intermediary network is untrusted as was the case in the first IPsec demonstration of Section 7.3.

Considering attack scenarios presented in Section 5.3 the scope of IPsec protection has a notable significance. T104 communication cannot be compromised in the tunnel but is exploitable from its endpoints. The attacker cannot access ESP encapsulated T104 traffic from the side because he is not part of the IPsec tunnel. He cannot eavesdrop T104 communication, inject packets, or gain a man-in-the-middle position when it is protected by the tunnel. However, the attacker can exploit T104 communication from its endpoints. In the first demonstration of Section 7.3 T104 communication could be exploited if the attacker gets access to the trusted SCADA network.

In the case of the first IPsec demonstration in Section 7.3 additional TLS protection would provide protection for T104 communication in the SCADA network while IPsec provides network isolation considering the untrusted network. Alternatively, possibly the results of the second IPsec demonstration could be applied for providing end-to-end protection between the endpoint hosts by using IPsec tunnels. A separate IPsec tunnel could be formed between the Windows firewall of the SCADA server and IPFire to protect traffic in the trusted SCADA network. Traffic to the virtual IP of the RTU could be first routed through that tunnel and then from IPFire to the RTU via the tunnel of the first demonstration through the untrusted network. This would also provide network segmentation as the virtual IP address of the RTU would be seen only by the SCADA server and not by other hosts in the SCADA network. Compromise of T104 communication would require compromise of one of the tunnel endpoints, either the SCADA server, IPFire, or the RTU. Nevertheless, regarding a denial-of-service attack the attacker could prevent the formation of the IPsec tunnel or somehow block ESP traffic and that way disable T104 communication.

IPsec is purely a networking technique which can be an advantage in some situations. As IPsec protection is carried out on operating system level in contrast to TLS protection, which is done in the control system software, it is independent of the support of the control system software run by a SCADA server or an RTU. Furthermore, as protection is provided in the network layer it is also independent of the application layer protocol. If a control system software or an application layer protocol lacks the support of TLS protection IPsec protection can provide a generic protection between hosts. Such situation can be actual for systems running legacy software or systems using unprotected protocols. In such cases the host level end-to-end protection approach provided by the second IPsec demonstration of Section 7.3 can be relevant.

IEC 62351-5 provides purely application and user process layer mechanisms to protect authenticity and integrity of critical T104 messages. Regarding attack scenarios of Section 5.3 IEC 62351-5 provides application layer authentication and integrity protection

of T104 messages classified as critical. Therefore, even though an attacker can access T104 communication it cannot perform critical operations, such as control commands through packet injection. As the implementation of IEC 62351-5 is built in the application layer it provides protection which cannot be provided by the techniques of the lower layers. The most remarkable functionalities are the linkage and authentication of application users between a controlling station and a controlled station and management of the privileges of the linked users. This provides protection for example if the SCADA application of the SCADA server is compromised and the lack of authentication or insufficient user privileges deny the attacker to operate a controlled station. However, IEC 62351-5 does not provide confidentiality of transferred data and therefore TLS or IPsec should be used in conjunction.

The solution for protecting T104 communication depends on several circumstances. The basis is the kind of protection required. This can become as a requirement from a customer or internally from the service provider. Security measures provided by each investigated technique have been covered in this thesis whereas not completely. Network architecture of a telecontrol system defines what kind of protection is applicable or suitable for a use case. Characteristics and possible restrictions regarding network architecture regarding each technique have been discussed in this thesis. Technical limitations regarding control station or substation equipment might also drive into using certain techniques. For instance, if legacy software is used it might restrict usage of newer protection techniques provided by IEC 62351 series of standards but enable usage of IPsec. Also providing protection in a layered manner can be relevant to be considered which leads into deploying multiple protection techniques simultaneously. One should also be aware that the discussed protection frameworks have possibly their own vulnerabilities which could be exploited and for example misconfigurations of the techniques could expose them for security breaches.

References

Awati, R. (2021). *What is promiscuous mode?* TechTarget. Retrieved 2.1.2024 from https://www.techtarget.com/searchsecurity/definition/promiscuous-mode

Barker, E., Dang, Q., Frankel, S., Scarfone, K., & Wouters, P. (2020). *Guide to IPsec VPNs* (NIST SP 800-77r1). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-77r1

Bellare, M., Canetti, R., & Krawczyk, H. (1996). Keying Hash Functions for Message Authentication. In N. Koblitz (Ed.), *Advances in Cryptology—CRYPTO '96* (Vol. 1109, pp. 1–15). Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-68697-5_1

Blake-Wilson, S., Bolyard, N., Gupta, V., Hawk, C., & Moeller, B. (2006). *Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)* (RFC4492). Internet Engineering Task Force. https://doi.org/10.17487/rfc4492

Carrel, D., & Harkins, D. (1998). *The Internet Key Exchange (IKE)* (RFC 2409). Internet Engineering Task Force. https://doi.org/10.17487/RFC2409

Ciphersuite.info. (n.d.). *TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384*. Retrieved 3.2.2024 from https://ciphersuite.info/cs/TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384/

Clarke, G., & Reynders, D. (2004). *Practical Modern SCADA Protocols: DNP3, 60870. 5 and Related Systems* (first edition). Elsevier Science & Technology. ISBN: 9780080480244.

Cloudflare. (n.d.-a). *What is the OSI Model?* Cloudflare. Retrieved 23.12.2023 from https://www.cloudflare.com/learning/ddos/glossary/open-systems-interconnection-model-osi/

Cloudflare. (n.d.-b). *What is encryption?* Cloudflare. Retrieved 23.12.2023 from https://www.cloudflare.com/learning/ssl/what-is-encryption/

Cloudflare. (n.d.-c). *What happens in a TLS handshake? | SSL handshake*. Cloudflare. Retrieved 9.1.2024 from https://www.cloudflare.com/learning/ssl/what-happens-in-a-tls-handshake/

Erdődi, L., Kaliyar, P., Houmb, S. H., Akbarzadeh, A., & Waltoft-Olsen, A. J. (2022). Attacking Power Grid Substations: An Experiment Demonstrating How to Attack the

SCADA Protocol IEC 60870-5-104. *Proceedings of the 17th International Conference on Availability, Reliability and Security*, 1–10. https://doi.org/10.1145/3538969.3544475

Forouzan, B. A. (2008). *Introduction to cryptography and network security* (first edition). McGraw-Hill Higher Education. ISBN: 978-0-07-287022-0.

Forouzan, B. A. (2010). *TCP/IP protocol suite* (fourth edition). McGraw-Hill Higher Education. ISBN: 978-0-07-337604-2.

Forouzan, B. A., & Fegan, S. C. (2007). *Data communications and networking* (fourth edition). McGraw-Hill Higher Education. ISBN: 978-0-07-296775-3.

Fortinet. (n.d.-a). *What is Public Key Infrastructure (PKI)?* Fortinet. Retrieved 2.1.2024 from https://www.fortinet.com/resources/cyberglossary/public-key-infrastructure

Fortinet. (n.d.-b). *What Are Eavesdropping Attacks?* Fortinet. Retrieved 5.2.2024 from https://www.fortinet.com/resources/cyberglossary/eavesdropping

Frankel, S. E., Hoffman, P., Orebaugh, A. D., & Park, R. (2008). *Guide to SSL VPNs* (NIST SP 800-113). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-113

FS. (n.d.). *Port Mirroring Explained: Basis, Configuration & FAQs*. FS. Retrieved 2.2.2024 from https://community.fs.com/article/port-mirroring-explained-basis-configuration-and-fa-qs.html

György, P., & Holczer, T. (2020). Attacking IEC 60870-5-104 Protocol. In I. Fazekas, A. Hajdu, & T. Tómács (Eds.), *Proceedings of the 1st Conference on Information Technology and Data Science* (Vol. 2874, pp. 140–150). CEUR. https://ceur-ws.org/Vol-2874/#paper13

IEC. (1988). *Telecontrol equipment and systems – Part 1: General considerations—Section One: General principles* (870-1-1).

IEC. (1992). *Telecontrol equipment and systems – Part 5: Transmission protocols – Section 3: General structure of application data* (870-5-3).

IEC. (1995). *Telecontrol equipment and systems – Part 5: Transmission protocols – Section 5: Basic application functions* (870-5-5).

IEC. (2003). *Telecontrol equipment and systems – Part 5-101: Transmission protocols – Companion standard for basic telecontrol tasks* (60870-5-101).

IEC. (2006). *Telecontrol equipment and systems – Part 5-104: Transmission protocols – Network access for IEC 60870-5-101 using standard transport profiles* (60870-5-104).

IEC. (2007). *Power systems management and associated information exchange – Data and communications security – Part 1: Communication network and system security – Introduction to security issues* (62351-1).

IEC. (2013a). *Power systems management and associated information exchange – Data and communications security – Part 5: Security for IEC 60870-5 and derivatives* (62351-5).

IEC. (2013b). *Telecontrol equipment and systems – Part 5-7: Transmission protocols – Security extensions to IEC 60870-5-101 and IEC 60870-5-104 protocols (applying IEC 62351)* (60870-5-7).

IEC. (2014). *Power systems management and associated information exchange – Data and communications security – Part 3: Communication network and system security – Profiles including TCP/IP* (62351-3).

IEC. (2023). *Power systems management and associated information exchange – Data and communications security – Part 5: Security for IEC 60870-5 and derivatives* (62351-5).

ISO/IEC. (1994). *Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model* (7498-1).

Kali. (n.d.). *What is Kali Linux?*. Kali Linux. Retrieved 2.2.2024 from https://www.kali.org/docs/introduction/what-is-kali-linux/

Kaspersky. (n.d.). *What Is a Replay Attack?* Kaspersky. Retrieved 5.2.2024 from https://www.kaspersky.com/resource-center/definitions/replay-attack

Kent, S., & Atkinson, R. (1998a). *Security Architecture for the Internet Protocol* (RFC 2401). Internet Engineering Task Force. https://doi.org/10.17487/RFC2401

Kent, S., & Atkinson, R. (1998b). *IP Encapsulating Security Payload (ESP)* (RFC 2406). Internet Engineering Task Force. https://doi.org/10.17487/RFC2406

Kozierok, C. M. (2005). *IP Datagram Encapsulation*. The TCP/IP Guide. Retrieved 4.1.2024 from http://www.tcpipguide.com/free/t_IPDatagramEncapsulation.htm

Lake, J. (2021). *TLS (SSL) Handshakes Explained*. Comparitech. Retrieved 26.12.2023 from https://www.comparitech.com/blog/information-security/tls-ssl-handshakes-explained/

Matoušek, P. (2017). *Description and analysis of IEC 104 Protocol* (Technical report no. FIT-TR-2017-12). Faculty of Information Technology, Brno University of Technology. Retrieved 2.5.2023 from https://www.fit.vut.cz/research/publication-file/11570/TR-IEC104.pdf

Maynard, P., McLaughlin, K., & Haberler, B. (2014). Towards Understanding Man-In-The-Middle Attacks on IEC 60870-5-104 SCADA Networks. *2nd International Symposium for ICS & SCADA Cyber Security Research 2014*. https://doi.org/10.14236/ewic/ics-csr2014.5

Miller, R. (2001). *The OSI Model: An Overview*. SANS. Retrieved 6.11.2023 from https://www.sans.org/white-papers/543/

OWASP Foundation. (n.d.). *Manipulator-in-the-middle attack*. OWASP Foundation. Retrieved 5.2.2024 from https://owasp.org/www-community/attacks/Manipulator-in-the-middle_attack

Pidikiti, D. S., Kalluri, R., Kumar, R. K. S., & Bindhumadhava, B. S. (2013). SCADA communication protocols: Vulnerabilities, attacks and possible mitigations. *CSI Transactions on ICT*, 1(2), 135–141. https://doi.org/10.1007/s40012-013-0013-5

Rescorla, E., & Dierks, T. (2008). *The Transport Layer Security (TLS) Protocol Version 1.2* (RFC 5246). Internet Engineering Task Force. https://doi.org/10.17487/RFC5246

Rapid7. (n.d.). *What is a Spoofing Attack? Detection & Prevention*. Rapid7. Retrieved 5.2.2024 from https://www.rapid7.com/fundamentals/spoofing-attacks/

Ristić, I. (2014). *Bulletproof SSL and TLS*. Feisty Duck. ISBN: 978-1-907117-04-6.

Saif Qassim, Q., Jamil, N., Daud, M., Ja'affar, N., Yussof, S., Ismail, R., & Azlan Wan Kamarulzaman, W. (2018). Simulating command injection attacks on IEC 60870-5-104 protocol in SCADA system. *International Journal of Engineering & Technology*, 7(2.14), 153. https://doi.org/10.14419/ijet.v7i2.14.12816

Stallings, W. (2017). *Cryptography and network security: Principles and practice* (seventh edition). Pearson. ISBN: 978-1-292-15858-7.

Tanenbaum, A. S., & Wetherall, D. (2011). *Computer networks* (fifth edition). Pearson Prentice Hall. ISBN: 978-0-13-212695-3.

Tiller, J. S. (2000). *A Technical Guide to IPSec Virtual Private Networks* (first edition). Auerbach Publishers, Incorporated. ISBN: 978-0-203-99749-9.

Turner, J., Schertler, M. J., Schneider, M. S., & Maughan, D. (1998). *Internet Security Association and Key Management Protocol (ISAKMP)* (RFC 2408). Internet Engineering Task Force. https://doi.org/10.17487/RFC2408

Wireshark. (n.d.-a). *Display Filter Reference: IEC 60870-5-104-Apci*. Wireshark. Retrieved 12.2.2024 from https://www.wireshark.org/docs/dfref/1/104apci.html

Wireshark. (n.d.-b). *Display Filter Reference: IEC 60870-5-104-Asdu*. Wireshark. Retrieved 12.2.2024 from https://www.wireshark.org/docs/dfref/1/104asdu.html

Zimmermann, H. (1980). OSI Reference Model—The ISO Model of Architecture for Open Systems Interconnection. *IEEE Transactions on Communications*, 28(4), 425–432. https://doi.org/10.1109/TCOM.1980.1094702