Mälardalen University
School of Innovation, Design and Engineering
Västerås, Sweden

Thesis for the Degree of Master of Science in Comptuer Science in Intelligent Embedded Systems 30.0 credits

DEVELOPMENT OF GENERIC COMMUNICATION MIDDLEWARE FOR EMBEDDED SENSOR SYSTEMS TRANSMITTING HEALTH PARAMETERS

Subaharan Kailayanathan
skn09011@student.mdh.se

Saji Kamdod
skd18002@student.mdh.se

Examiner: Saad Mubeen
Mälardalen University, Västerås, Sweden

Supervisors: Annica Kristofferson
Mälardalen University, Västerås, Sweden

Maryam Vahabi
Mälardalen University, Västerås, Sweden

June 22, 2020

Abstract

Health technology or e-Health is one of the most rapidly growing areas in healthcare today and it has been an important requirement as a new concept of healthcare industry. Since global society has been changed to aging society and the healthcare cost has been increasing in the 21st century since 2007. As the total number of people aged 65 or older is expected to increase from 12% to 22% in 2050 which is double the rate, and at the same time there is a decrease in in-fertility rates and increase in life expectancy due to the increase in life quality, there is a need to investigate the needs and requirements of an intelligent embedded sensor systems in health applications, and to develop a new communication protocol or set of protocols that can be used to send data collected from a hub within a house, home-care or a complex and send it securely and reliably to a central database where the gathered data can be monitored by a medical professional to make decisions for further interventions. The employed communication protocol should also be able to securely transmit confidential parameters from the hospital network to a central server outside of the hospital network. The final protocol must be inline with the regulations of the EU. This thesis is done in collaboration with Tjeders AB, Stille AB, and Embedded Sensor Systems for Health Plus (ESS-H+) research profile at Mälardalen University. In this thesis, different communication protocols such as IPSec and TLS and algorithms such as AES and RSA are examined and based on the requirements provided by the companies certain of these protocols and algorithms will be used in the final implementation. Different performance metrics such as overhead, round trip delay and throughput will be measured for the chosen communication protocols and recommendations will be given on which of the protocols and algorithms needs to be used to obtain an optimized, secure and reliable network.

Acknowledgement

We would like to express our sincere and humblest gratitude to our academic supervisors, Annica Kristoffersson and Maryam Vahabi for their continuous support and help throughout this thesis work. The feedback provided by them has been incredibly invaluable for us taking the thesis work forward. We also would like to extend our gratitude towards Professor Maria Lindén, and Professor Mats Björkman for their time and feedback. Also providing us with the equipment and all the necessary help possible.

We take this opportunity to also thank Mikael Lauren from Tjeders AB and John from Stille AB for their continued support and feedback throughout the duration of the thesis.
Finally, we like to thank our families and friends for their unconditional love and support during this thesis.

Sincerly,
Subaharan Kailayanathan & Saji Kamdod

Contents

List of Figures

List of Tables

1 Introduction

Health technology or e-Health is the use of Information and Communication Technologies (ICT) in the health sector and it is one of the most rapidly growing areas in healthcare today [1]. Also, e-Health has been an important requirement as a new concept of healthcare industry since global society has seen a change in the rise of old age population which leads to an increase in healthcare cost which has been rapidly growing in the 21st century [2]. The use of intelligent embedded sensor systems is a promising approach to address this challenge and make sure of the functions of these systems to bring out a change in reducing the cost of healthcare per person and making the health care system more efficient. However, deploying a e-health system in reality is a complex task because of many obstacles such as strict and diverse regulations, technologies and lack of understanding of e-Health. Due to these reasons, many healthcare device manufacturers have had difficulties in developing and commercializing their e-Health devices and services. The scope of the e-Health industry includes three major domains namely mobile healthcare, remote healthcare, and e-hospital [3]. All these domains rely heavily on intelligent embedded sensor systems to achieve their objectives. Recent advances in wireless sensor networking have opened up new opportunities in healthcare systems such as a home-care network where one must provide a middleware inter-operability between devices and support the complex and unique relationships among devices such as implants, health monitoring devices, and their outside control unit [4]. The future is the integration of the existing specialized medical technology with wireless embedded sensor systems and networks. This will co-exist with the installed infrastructure by augmenting data collection and real-time response. One such example of an area in which future medical systems can benefit the most from wireless sensor networks is in-home assistance. In-home networks may assist residents by with emergency services, health data lookup, vitals monitoring and assistance. Wireless communication tends to be one of the major trends in medical applications to increase usability and comfort in the long time patient monitoring [5]. However systems currently available on the market concentrate mostly on home-care assistance and not on services and systems where the data is streamed continuously making monitoring easy. There exists a couple of short-range wireless transmission standards that allow appropriate data rates suitable to continuously transmit a sufficient set of patient vital status information. For this reason, a reliable and robust communication protocol needs to be designed so that the confidential and sensitive data and parameters can be transmitted from the home-care to the front-end securely. Also, checks and balances needs to be in place to take care of exceptions that might occur such as loss of network or failure to transmit data. This thesis aims at creating this kind of communication protocol that is safe, reliable, robust and can be translated to other systems.

1.1 Motivation and Problem formulation

According to the recent statistics, the total number of people aged 65 or older is expected to increase from 12% to 22% in 2050 which is double the rate, and at the same time there is a decrease in in-fertility rates and increase in life expectancy due to the increase in life quality [6]. The motivation of this thesis is to investigate the needs and requirements of an intelligent embedded sensor systems in health applications, and to develop a new communication protocol or set of protocols that can be used to send data collected from a hub within a house, home-care or a complex and send it securely and reliably to a central database where the gathered data can be monitored by a medical professional to make decisions for further interventions or studies. This communication protocol should also be able to securely transmit parameters from an operating table in hospitals when it is not in use to a receiver within the hospital network and then transmit it to a central server outside of the hospital network. The final protocol must be inline with the regulations of the EU. This thesis is done in collaboration with Tjeders AB [1], Stille AB [2] and and a research profile for Embedded Sensor Systems for Health Plus (ESS-H+) [3], based at Mälardalen University.

[1] https://tjeders.se/om-tjeders/
[2] https://www.stille.se/about-us/
[3] https://www.mdh.se/en/malardalen-university/research/embedded-systems/embedded-sensor-systems-for-health-plus

The main aim of this thesis is to develop a generic protocol that can transfer data securely and reliably to a central location and also can be easily ported from one system to another. Next, is the investigation of pre-existing protocols and making a decision on whether these protocols can facilitate in achieving the end goal. Validating these current protocols becomes an important part if they do. If they do not, then investigation needs to be done to invent a new protocol or a combination of limited subset of the existing protocols with discussion with the stakeholders.

The research questions within this thesis are:

1. What communication requirements must be fulfilled to achieve regulations for communication of medical devices transmitting health parameters?
2. Which protocols can transfer the data securely and reliably using a generic middleware for embedded sensor systems?
3. What changes should be made so that the requirements can be achieved with a combination of limited subset of the existing protocols?

1.2 Initial Company Requirements

The companies involved in this thesis are Tjeders AB and Stille AB. During the first meeting, the discussion was about developing a secure communication protocol for securely transmitting health parameters data from home-care and operating tables for Tjeders AB and Stille AB respectively. The goal was to develop a standard protocol that was in-line with the Medical Device Regulation (MDR) and this data had to transmitted over Bluetooth 5.0 within the internal communication and for external communication the suggestion from Stille AB was to use IPSec communication. Later, the suggestion from Tjeders was to use MQTT with TLS 1.3 for transmitting the data from home-care to their centralized server. The end goal is to measure the cost and security of this communication profile while comparing them to other approaches.

1.3 Overview

The rest of the thesis is structured as follows. Section 2 describes relevant technical concepts needed to understand this thesis. Section 3 will look into looks into most relevant related works in different topics. Section 4 will explain the research methodology used throughout the thesis. Section 5 explains how the architecture and topology are designed for different use cases and describes the proposed setup that is used in the experiments. Section 6 describes the information about specific software/hardware that have been used in this thesis by comparing with their counterparts. Section 7 describes different scenarios in which the experiments are setup for different network segments. Section 8 describes the experimental results and discussion of results for the different scenarios. Sections 9, and 10 discusses the results, conclusions and the possible future work respectively.

2 Background

This section of the report covers the technical and theoretical aspects that are relevant to the thesis. These include Standards and Regulations, Embedded Sensor Systems, Middleware, Communications protocols such as Zigbee, Z-wave and LoRaWAN. All topics described in this underlying section has an important role in understanding this thesis.

2.1 Standards and regulations

This section will explain the different standards and regulations that are currently in use along with the different requirements that need to be meant to ensure the safety of the medical devices, device software and communication profile.

2.1.1 Medical Device Regulations (MDR)

The European Medical Device Regulation (EU MDR) provides required parameters for the quality and safety of the medical devices that are being manufactured or supplied within Europe. MDR aims to establish a robust and transparent regulatory structure for medical devices, to guarantee a high level of health and security and also to support innovation. The transition chronology is shown in Figure 1. The new MDR imposes strict demands on medical device manufacturers and the notified bodies who must be involved in the approval process of medical devices other than self-declaration class I devices. The MDR differs in several important ways from the EU's current directives for Medical Devices Directives (MDD) and active implantable medical devices [7]. MDR requests increased post-market surveillance authority by the Notified Body with unannounced audits, along with product sample checks and product testing that will strengthen the EU's enforcement policy and help reduce the risks from unsafe devices. Annual safety and performance reporting by device manufacturers will also be required in some cases. The MDR plans to allow the EU commission or expert panels to be defined to publish Common Specifications which shall then be taken into account by manufacturers as well as notified bodies; these common specifications shall exist in parallel to the harmonized standards and state of the art. Under the MDR, all currently approved devices must be re-certified in accordance with the new requirements.

Figure 1: MDR Transition Period. [4]

2.1.2 ISO/IEEE 11073-20702 - Medical Devices Communication Profile

The ISO/IEEE 11073 family of standards focuses on the interoperability between medical devices and external computer systems. Currently, no part of the 11073 standard series allows the ability to achieve plug-and-play-enabled communication of medical devices in an Internet Protocol (IP)-based distributed Point of Care (PoC) medical device communication system. Thus, this standard defines an event propagation method for a distributed PoC medical device communication system based on Web Services. The Service-oriented device communication is defined by another standard known as the Medical Devices Profile for Web Services (MDPWS). This new system will be able to make the complexity of a complete Operating Research (OR) integration manageable, thereby improving the patient's safety. The primary function of MDPWS is that it provides a safe platform

[4] https://www.freyrsolutions.com/medical-devices-regulation-mdr

for the device communication required for medical and safety issues [8]. The loose-coupled, non-centralized service-oriented device communication between several medical devices is shown in the left part of Figure 2.

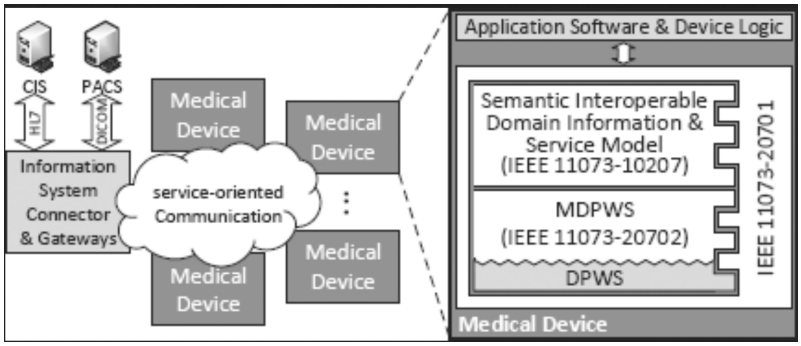

Figure 2: Device communication with IEEE 11073 standards. [9]

2.1.3 IEC 62304 - International Standard for Medical Device Software

IEC 62304 is an international standard that describes the life cycle requirements for medical software and devices that are in development. This standard is also applicable to the development and maintenance the medical device software that is within a medical device or within an embedded system [5]. It is currently implemented and put to use by the European Union (EU) and the United States (US). Therefore it can be used as a benchmark to comply with regulatory requirements. The IEC 62304 standard calls out specific cautions on using software, particularly Software Of Unknown Pedigree or Provenance (SOUP) [10]. An overview of the standard and the software life cycle is shown in Figure 3. The standard provides a risk-based decision model on when the use of SOUP is acceptable and defines testing requirements for SOUP to support reasoning on why such Software should be used.

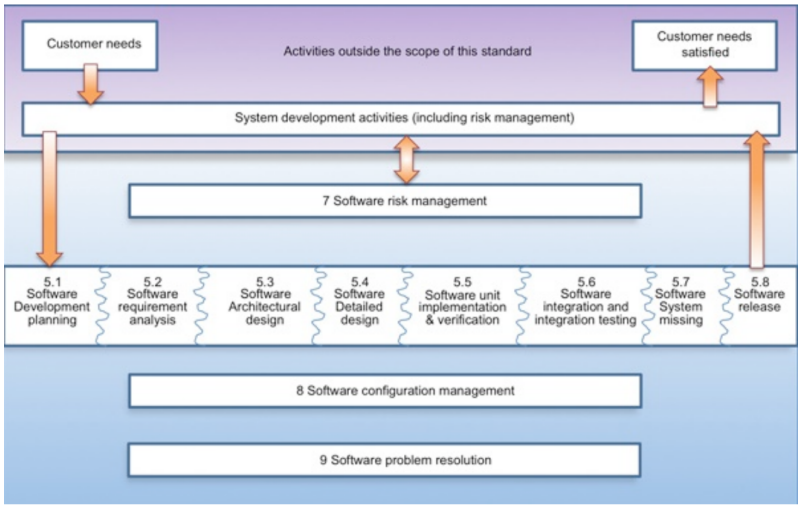

Figure 3: Overview of software development processes and activities. [6]

The IEC 62304 standard was released in 2006, and it provides a framework for software development cycle for processes within an activity or task that is needed to obtain the safe design and maintenance of a medical device software [11]. These tasks, activities, and processes that are within

[5] https://web.archive.org/web/20121104013821/ec.europa.eu/enterprise/policies/european-standards/harmonized-standards/medical-devices/

[6] http://www.medicalsysconsult.com/portfolio/legacy-software-iec-62304-dhf-audit.html

the clause 5 of the standard also establish a common framework for the device software life-cycle process which can be easily understood and implemented within a team working on a project or between teams [12]. This standard becomes applicable within the development and maintenance of a device software when three of the following scenario occurs:

- Software itself is a medical device.
- Software is used in part or in whole within the medical device software.
- Software is utilized when manufacturing a medical device.

2.2 Communication protocols

This section will explain the different communication protocols that can be used to obtain the end result. The architecture of these protocols, along with their uses, advantages and disadvantages, are mentioned. Based on these criteria, the decision will be made to include the protocols in the final design.

2.2.1 ZigBee

ZigBee is a wireless technology standard that defines a set of communication protocols for short-range communications based on the IEEE 802.15.4 standard. Wi-Fi and Bluetooth are the two examples that are short-range communication standards. ZigBee standard is built explicitly for control and sensor networks. ZigBee is a standard that addresses the need for very low-cost implementation of low power devices with low data rate for short-range wireless communications. It is one of the most commonly used standards for IoT and is an open-source standard developed by ZigBee Alliance. The most prominent uses of ZigBee is in home automation, medical data collection and industrial control systems wherein the purpose is to collect information or perform control tasks inside a building [13]. A standard ZigBee architecture is shown in Figure 4.

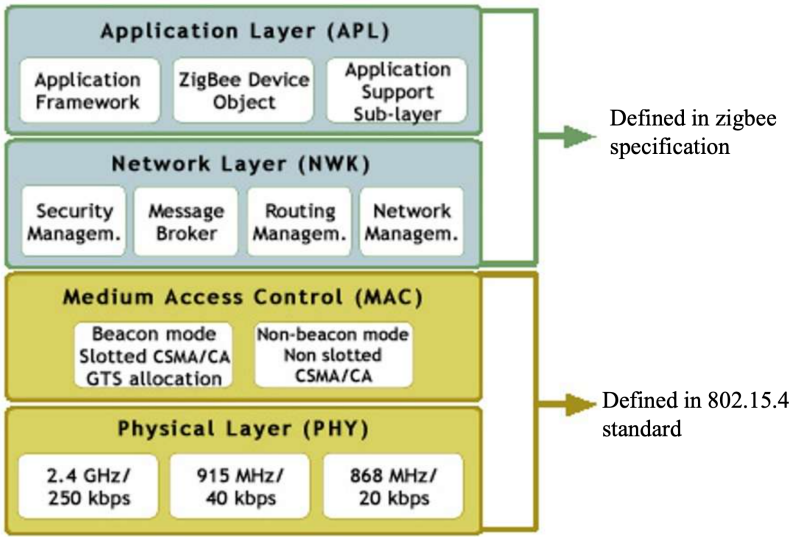

Figure 4: ZigBee Architecture. [14]

Packets of data can be sent between nodes and may be routed by intermediary devices to more distant nodes that would otherwise be out of range. Each device has both a MAC address and a ZigBee network address, while the entire network has its own Personal Area Network (PAN) ID shared by all devices. There are three different ZigBee device types that operate on the layers in any self-organizing application network; ZigBee coordinator node, Full Function Device (FFD) and Reduced Function Device(RFD) [15]. ZigBee coordinator node is the root of a network tree and a bridge to other networks, and it is able to store information about the network. It acts as a repository for other security keys. FFD is an intermediary router transmitting data from other

devices, and it needs less memory than ZigBee coordinator node, can operate on all topologies and can also act as a coordinator. Some of the characteristics of the ZigBee include low power consumption with battery life ranging from months to years, high density of nodes per network, simpler implementation, low data rate and small packet device (128 Bytes).

2.2.2 Bluetooth

Bluetooth is a wireless technology standard used for exchanging data between devices over short distances using short-wavelength UHF radio waves between the 2.402 GHz to 2.480 GHz frequencies. Bluetooth is managed by the Bluetooth Special Interest Group (SIG), and the IEEE standardized Bluetooth is known as IEEE 802.15.1 [16]. Bluetooth SIG oversees the development of a specification, manages and protects the program and trademarks. Bluetooth is a packet-based protocol with a master/slave architecture wherein one master may communicate with up to seven slaves in network devices in a network using the Bluetooth specification. Piconets use the master clock to begin the transmission of packets. The master clock ticks at a period of 312.5 μs, two clock ticks then make up a slot of 625 μs, and two slots make up a slot pair of 1250 μs. In most of the cases, the master devices transmit in even slots and receive data in odd slots, and the slave does it in the reverse order [17]. One packet maybe 1,3 or 5 slots long, but in all cases the master transmits in even slots and slave in odd slots. Bluetooth has been updated from versions 1 to 5 with various improvements and additions to the standards, the main component of Bluetooth version 4.0 added is Bluetooth Low Energy (BLE) specification. All the improvements made to the newer version of Bluetooth is towards BLE specification, which was first introduced in version 4.0. The standard Bluetooth protocol stack is shown in Figure 5. The main aim of BLE is to provide low energy consumption for the services while maintaining the same performance standards. In terms of improving the battery life of devices, it represents a significant progression. Bluetooth 5.0 is the latest version of this standard and is most commonly used for communication between various smart home and Internet of Things (IoT) devices.

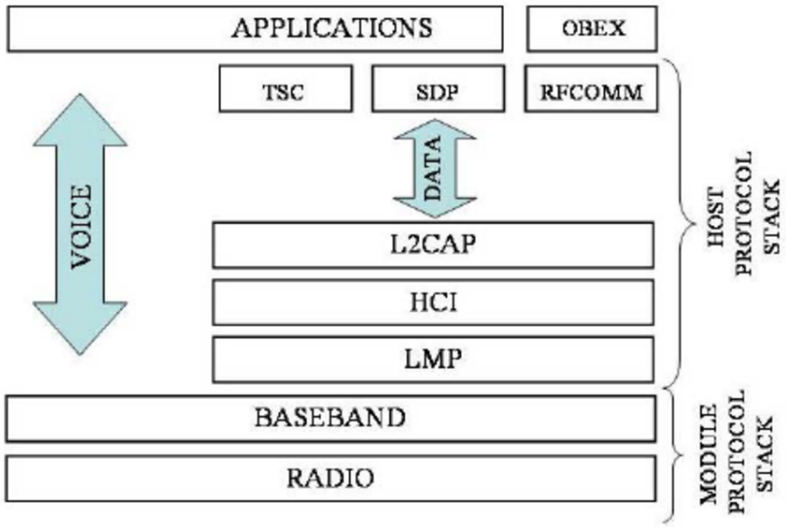

Figure 5: Bluetooth protocol stack. [18]

BLE is designed such that the peripherals use low energy hence reducing the energy consumption [19]. The primary benefits with Bluetooth 5.0 over the previous versions are improved speed and range. With version 5.0, it can transfer data up to 2 Mbps which is almost double the speed of Bluetooth 4.2 and communicate over larger distances of up to 243 m.

2.2.3 Z-wave

Z-Wave is a wireless protocol developed by Zensys that is used for automation devices for homes and commercial setting. This protocol can be used on all major electrical devices in the house, such as on/off light switches, HVAC, televisions and home security. Z-Wave distinguishes from other wireless protocols by enabling secure and reliable transmission of short messages which can be maximum of 64 bytes from the control unit to the devices in the network with minimum noise. In addition to routing, one of the most distinguishable properties of Z-wave is Beaming, i.e., the nodes have a battery-saving characteristic. The Beam is a carrier signal that is transmitted for a set period of time. The Z-wave protocol architecture contains four layers: Application layer, a Routing layer, MAC layer, and transfer layer. How the data is stored in these various layers is shown in Figure 6.

Each device represents a node and includes the network that uses the Z-Wave protocol, and the devices are distinguished by a network ID to separate these nodes from neighbour Z-Wave networks nodes. Radio Frequency (RF) wave is used to communicate the devices with one another, the RF lets the insert or departure nodes travel to and from the home network, and this enables resulting in reduced installation costs [7]. It also uses a Mesh network configuration that enables each appliance in the network to send and receive action commands through walls or ceilings. This enables the z-wave to propagate through the entire area of the house or network [20].

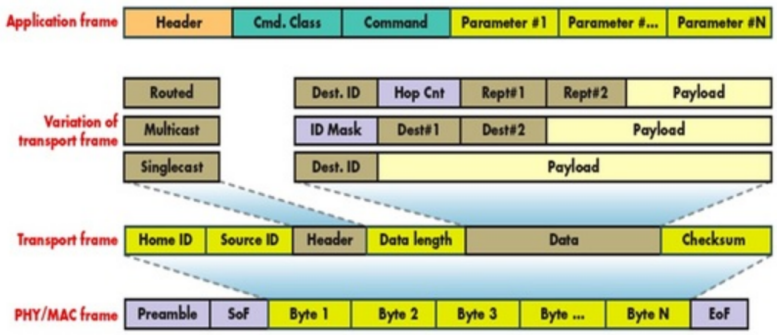

Figure 6: Z-wave data in the different layers. [8]

2.2.4 LoRaWAN

LoRa is an acronym for Long Range, and it is a wireless technology where a low powered sender transmits small data packages between 0.3 to 5.5 kbps to a receiver over a long distance. This gateway can handle hundreds of devices at the same time. A LoRa end node consists of 2 parts: a radio module with antenna and a microprocessor to process the sensor data. The end nodes are often battery-powered and a LoRa device has a wireless transceiver, if this device also has sensors, then it acts as a remote sensor. The LoRa gateway consists of the same parts as the end node. This gateway can receive data from the same end node and can listen to multiple frequencies simultaneously, in every spreading factor at each frequency.

The LoRaWAN network architecture is shown in Figure 7, the architecture is deployed in a star topology and the communication between the end node and gateway is bidirectional which means that the end node can send data to the gateway but it can also receive data from the gateway. The LoRaWAN protocol does not support direct communication between end nodes. If one wants to achieve this without the use of gateways, then the RadioHead Packet library for embedded microprocessors enables it by providing a complete object oriented library for sending and receiving packets.

[7] "An Introductory Guide to Z-Wave Technology", Feb. 2013.
[8] https://www.rfwireless-world.com/Tutorials/z-wave-protocol-stack.html
[9] https://www.rs-online.com/designspark/lorawan-network-server-integrated-or-cloud-hosted-1

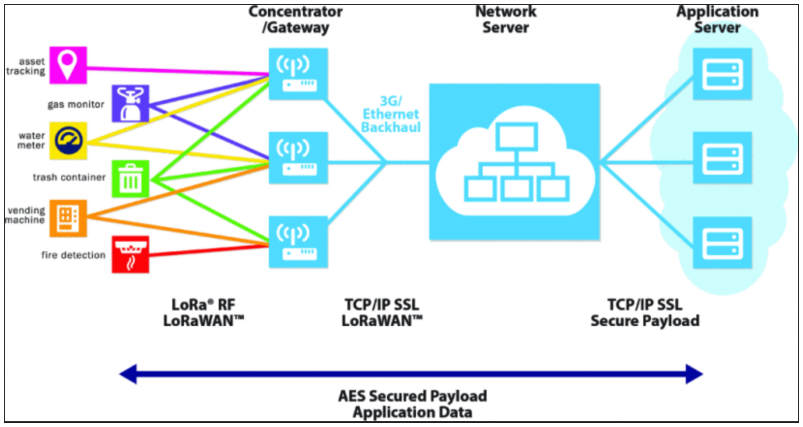

Figure 7: LoRa Network Architecture.[9]

2.3 Cybersecurity for Medical Devices

As Software is becoming more and more integral to Medical Devices, new opportunities arise from their networking and data exchange. But this is also exposing them to the same risks as common objects such as laptops and smartphones, i.e. unauthorized access to the device. A basic definition related to each cybersecurity requirements within the Medical Devices Regulations is that "risk" [10] means the following;

'risk' means the combination of the probability of occurrence of harm and the severity of that harm
This definition is intentionally broad to fulfill the primary goal by the MDR. It is acknowledged that in the domain of medical devices, security risks have to be decreased to an acceptable level.

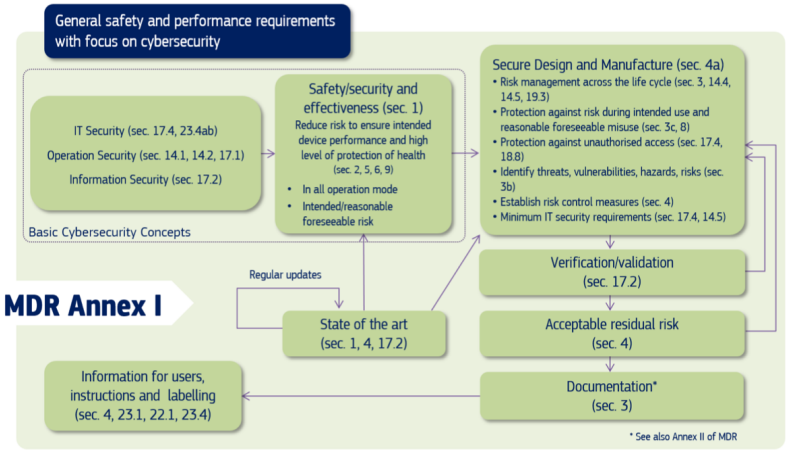

Figure 8: Cybersecurity requirments in MDR Annex I. [2].

Core concepts involved in IT security specifically for medical devices are the following [21]:

[10] Article 2 (23) of Regulation (EU) 2017/745 – MDR and Article 2(16) of Regulation (EU) 2017/746

Confidentiality	Ensures that no unauthorized part except the authorized to access information regarding users.
Integrity	Ensures that the content of the information remains the same, and intentionally or unintentionally not manipulated.
Availability	Ensures that the data is up-to-date and available whenever needed

2.4 Middleware

Middleware is a software layer that lies between an operating system and the service [22]. Functioning as a hidden layer, middleware allows communication and data administration for distributed systems. It commonly is known as "software glue", as it connects two applications. Since distributed systems are made up of heterogeneous devices, that need to interact and collaborate together, a middleware application enables support for heterogeneity. It enables support for multiple languages and OS where developer gets freedom to write the application using any of the supported language under any platform and still can connect with each other. This can be visualized in Figure 9.

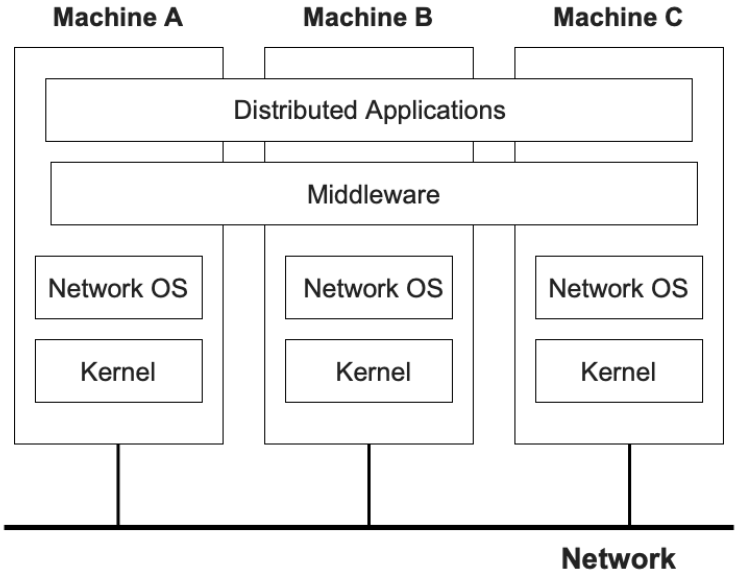

Figure 9: Middleware in Software architecture.

Functioning to eliminate the challenges of integration, middleware applications act as messaging services and enable data management and communication in distributed applications. Middleware covers services such as content management systems, application servers and web servers that support the development and delivery of services. Middleware serves a variety of functions. First, it manages connectivity to various back-end resources. A middleware component might create a connection pool to provide fast and efficient access to a popular back-end database. It can also create connections to message queues and topics. Furthermore, a piece of middleware software may manage connections to cloud-based resources, like the Amazon Simple Storage Service.

Second, middleware software has the capacity to implement logic based on the request made by the client. For example, a middleware component might recognize that a client browser making a given request has the language header set to English, and, as a result, the queries it makes to the back-end might be tweaked to return nothing but English-based results. Or, perhaps a server could identify the geographical location of the client making the request based on its IP address and return data to the client that prioritizes results that are located close by. The ability to take a

request from the user, perform logic and then customize the results is an important job performed by middleware software.

Finally, middleware plays an important role in securing access to back-end resources. Middleware software has the ability to challenge clients; it requires both a secure connection – using a technology like SocketSL – and authentication – using either a username and password combination or a digital certificate. This security information is then used to check if the client making the request has the authority to access the data in question. If the rights are affirmed, the data is sent from the middleware server to the client using a secure and encrypted connection.

2.5 Security Protocols and Algorithms

This section will go through the different security protocols and algorithms and also explain their working and different use cases while also mentioning their pros and cons. Based on these criteria, decision will be made to include some of the protocols in final design.

2.5.1 Message Queuing Telemetry Transport (MQTT)

MQTT is an open-source publish-subscribe messaging protocol designed for constrained devices and low-bandwidth, high latency or unreliable networks [23]. It is a lightweight and simple protocol and designed to be easy to implement [24]. For data communication over the protocol such as TCP/IP, the controller that is used to distribute the message is called MQTT broker. The broker forwards, filters and prioritizes publish requests from the publishers to the subscribers. To communicate via MQTT protocol, the publisher must define two elements, including message and topic for the MQTT broker. The message element is the string data that the publisher wants to share to the subscribers via MQTT broker. The topic is a string that is used by the broker to filter and decide which subscribers should receive which message.

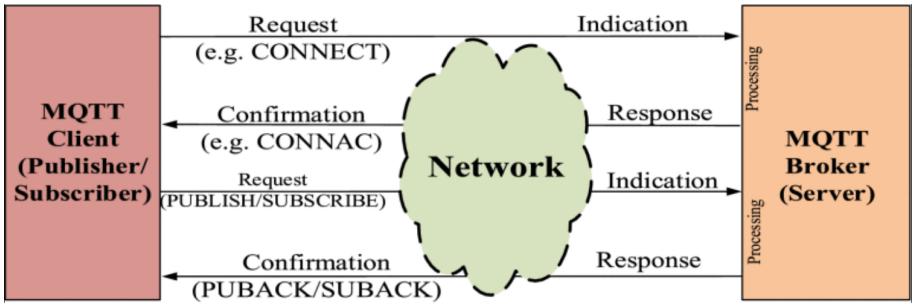

Figure 10: MQTT hand shaking. [25]

There are three key players in MQTT messaging: MQTT publisher, MQTT subscriber, and MQTT broker. MQTT publisher and subscriber are not connected to each other by IP address directly and they do not necessarily run at the same time. MQTT broker, which performs as a network hub and receives the messages from publishers, filters, prioritizes, and distributes them among up-to thousands of concurrently connected MQTT subscribers. MQTT broker is responsible for client authorization and a hand-shaking procedure for initialization of the communication. MQTT publishers use customizable topics for publishing data, which must be subscribed by the clients. MQTT protocol does not support labeling messages with Metadata. Hence, MQTT topic management in order to attach meaningful attributes to the topic could present metadata of the message payload and becomes substantial. MQTT topic is a string that has a hierarchical structure with multiple levels and attributes [26]. Each level is separated by a forward slash in the topic tree, for example: my_home/ground_floor/living_room. These topics could be modified to represent routing information. Figure 10 shows the initialization of the connection by exchanging control packets between clients and the broker. These control packets, such as CONNECT, CONNAC, PUBLISH, PUBACK, SUBSCRIBE, SUBACK, etc., contain details about the quality of service (QoS) level of transmission, topic, and payload [27].

2.5.2 Transport Layer Security (TLS)

TLS protocol is widely adopted in today's computer networks since it provides secure channels for end-to-end communications. According to a report from Fortinet networks [28]. HTTPS traffic accounts for 72.2% of all Internet traffic in the third quarter of 2018 and the proportion keeps growing. In 2018, IETF released TLS 1.3 [29], which requires one less round trip to establish a connection than TLS 1.2. Moreover, under session resumption, TLS 1.3 offers an even more efficient 0-RTT (Round Trip Time) mode which is a method of lowering the time to first byte on a TLS connection. TLS 1.3 only requires 1-RTT (a single round trip) of the protocol, where TLS 1.2 and below required two. This significantly improves the user experience over the previous version by speeding up the connection leading to smooth web experience.
TLS uses a combination of symmetric and asymmetric cryptography, as this provides a good compromise between performance and security when transmitting data securely. TLS uses asymmetric cryptography for securely generating and exchanging a session key. The session key is then used for encrypting the data transmitted by one party, and for decrypting the data received at the other end. Once the session is over, the session key is discarded. With TLS, a client connecting to a server is able to validate ownership of the server's public key. This is normally undertaken using an X.509 digital certificate issued by a trusted third party known as a Certificate Authority (CA) which asserts the authenticity of the public key. In some cases, a server may use a self-signed certificate which needs to be explicitly trusted by the client, but this may be acceptable in private networks and/or where secure certificate distribution is possible. It is highly recommended though, to use certificates issued by publicly trusted CAs. CA is an entity that issues digital certificates conforming to the ITU-T's X.509 standard for Public Key Infrastructures (PKIs). Digital certificates certify the public key of the owner of the certificate, and that the owner controls the domain being secured by the certificate. A CA therefore acts as a trusted third party that gives clients assurance they are connecting to a server operated by a validated entity. End entity certificates are themselves validated through a chain-of-trust originating from a root certificate, otherwise known as the trust anchor.

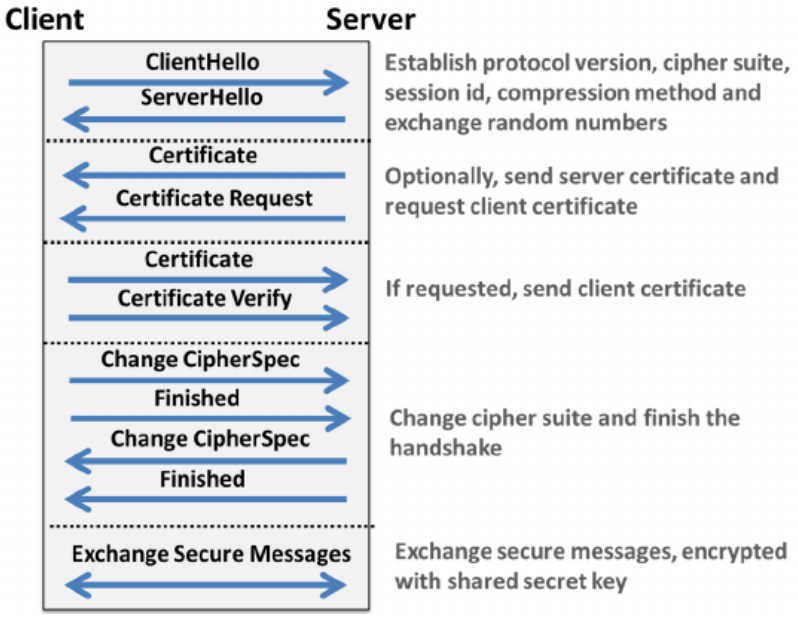

Figure 11: Overview of TLS Protocol. [30]

With asymmetric cryptography it is possible to use the private key of the root certificate to sign other certificates, which can then be validated using the public key of the root certificate and therefore inherit the trust of the issuing CA. In practice, end entity certificates are usually signed by one or more intermediate certificates as this protects the root certificate in the event that an end entity certificate is incorrectly issued or compromised. An overview of TLS is shown in Fig 11. TLS 1.3 also made several security improvements, including the removal of static RSA and Diffie–Hellman

(DH) cipher suites, encryption of handshake messages, etc. Many application-layer protocols for IoT also adopts TLS to establish underlying secure channels [31], including the Message Queuing Telemetry Transport (MQTT) protocol, the eXtensible Messaging and Presence Protocol (XMPP), the Advanced Message Queuing Protocol (AMQP) [32].

TLS encryption legitimates user privacy, it naturally disables middleboxes which are ubiquitous in computer networks. Particularly, middleboxes usually require access to application data to provide functionalities, including performance optimization devices, such as compression proxies and load balancers. It has been well known that middleboxes play a significant role in IoT systems where devices are usually resource constrained, e.g., with weak computation power and limited battery capacity. However, due to the encrypted traffic by TLS, attackers may exploit this fact and hide their attack traffic from security middleboxes [33]. Thus, it is practically infeasible to install endpoint-based programs such as virus scanners to protect IoT devices. Instead, in-network middleboxes should be deployed for IoT system protection. Deploying in-network middleboxes also brings additional benefits, including network-wide visibility and convenient management: a single middlebox is competent for handling all IoT devices in a local area network, while keeping the middlebox up-to-date is much easier than managing multiple scattered IoT devices.

2.5.3 Secure Sockets Layer (SSL)

Secure Sockets Layer (SSL) is a cryptographic protocol that is widely used in secure applications such as in web browsers and e-payments. The protocol is based on the principle of Public Key Infrastructure (PKI) [34] which is an algorithm used for authenticating users and devices within devices that are connected to a network. The main aim of PKI is to have one or more trusted parties digitally sign documents certifying that a particular cryptographic key belongs to a particular user or device. Because now many applications are based on the structure of Browser/Server, SSL is widely used in these applications, and provides security for them using HTTPS protocol [35]. The working of SSL is shown in Figure 12.

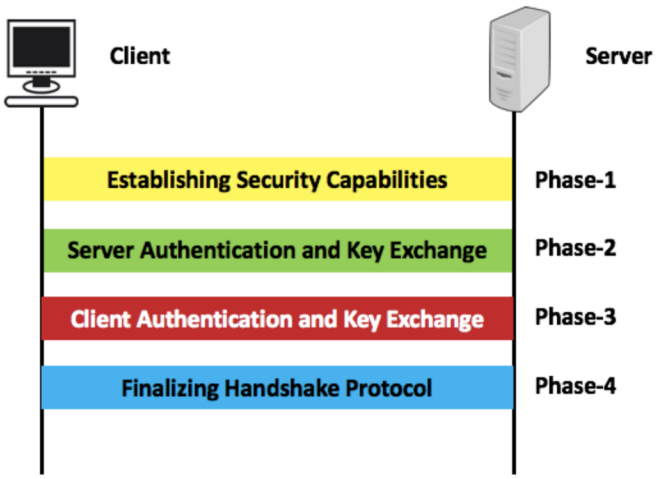

Figure 12: Phases of SSL Protocol. [11]

SSL consists of two sub-protocols. One is hand-shake protocol, the other is record protocol. Hand-shake protocol is the key of SSL, and it realizes certificates exchanging, key materials exchanging and identity authentication. The function of record protocol is using the session keys produced in hand-shake protocol to encapsulate the data to be exchanged so that the encapsulation can provide confidentiality and integrity for data [36].

In hand-shake protocol, client and server both will compute two session keys (client-write-key and server-write-key) with a key-block, client- and server-generated random numbers. However, due

[11] https://www.mysoftkey.com/security/4-phases-of-ssl-protocol/

to the limitation of exportation of USA, the length of key-block is only 42 bytes long, of which the 32nd to 36th bytes are used to generate client-write-key; the 37th to 41st bytes are used to generate server-write-key.

Client-write-key:

$$MD5(Key - Block_{32...36} \|randomc\| randoms) \tag{1}$$

Server-write-key:

$$MD5(Key - Block_{37...41} \|randomc\| randoms) \tag{2}$$

Here, MD5 is hash function producing a 128-bit hash value, randomc and randoms are client- and server-generated random numbers respectively. Since both random numbers are public in handshake protocol, only 5 bytes of key-block keep secret to the adversary. In other words, the valid lengths of both session keys are only 40 bits. In fact, a key of 40 bits is vulnerable to an exhaustive attack which is also known as Brute-force attack, wherein the attacker submits many passwords or keys with the hopes of eventually getting it right, these requests are sent multiple times over the course of a few minutes and have a 40-bit key leaves it vulnerable to this kind of attack.

SSL is used to authenticate endpoints and secure the contents of application-level communication. An SSL secured connection begins by establishing the identities of the peers and establishing an encryption method and key in a secure way. The communication for application-level can then begin.

2.5.4 Virtual Private Network (VPN)

VPN is a networking architecture that is implemented over public networks to support privacy in these networks, and it emerged as a cost-efficient and reliable solution in networking and telecommunication organizations. VPN saves the huge cost of infrastructure by using the public Internet to establish a highly secure communication medium from a corporate office to remote sites and remote users. VPN uses a tunnelling protocol to support its functionality. Tunnelling protocol provides a secure mode of transport for the network services which network does not support directly. A typical VPN scenario communication is show in Figure 13.

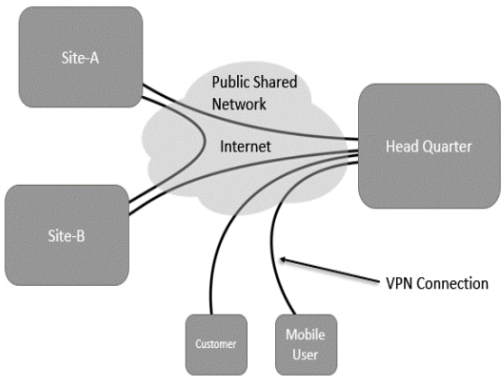

Figure 13: VPN Scenario. [37]

VPN services can be viewed from a different point of views within an organization, and it provides many benefits for individuals within multiple levels of a corporation [38]. VPN establishes a logical secure channel [39] for communication between two entities over the Internet by using the method of tunnelling, which encapsulates the IP datagram into a tunnelling protocol thus hiding the original data from outside view or to the intruders that are present in every network. It works by virtually establishing a point-to-point or multi-point link between the communicating parties in both the transmitting and receiving ends through the public or shared communication network. Traditional VPN uses Data Encryption Standard (DES), Advanced Encryption Standard (AES)

and Blowfish algorithm for encryption of user data. The link in which encrypted and encapsulated data is sent is known as VPN connection.

VPN enables any computer or device connected to a network to send and share data securely giving it a feeling of being connected to a private network, while at the same time providing the usability, functionality and speed of a public network [40]. User data may contain private and confidential information; therefore the security of user data needs to be ensured, the commonly implemented VPN is confined to DES encryption algorithm for encryption of user data inside an encapsulated tunnel packet.

To prevent any tampering of user data that is connected to a network, a sophisticated algorithm needs to be designed to prevent this. Multi-phase encryption algorithm provides this functionality by securing the data packet by performing encryption of different algorithms at different levels [41], which has also been proven as a very secure mode of encryption by using the standard encryption technique. In multi-phase encryption technique, even an outdated algorithm can be used to enhance the complexity of ciphertext and overall making a more secure packet.

2.5.5 Internet Key Exchange (IKE)

Internet Key Exchange (IKE) is a process that routers go through to create the logical tunnels between each other. The main purpose of IKE is to securely distribute keys between each other to encrypt data. There are two phases where two different tunnels are set up depending on which VPN is used.

In Phase 1, the ISAKMP (Internet Security Association and Key Management Protocol) is used, including authentication and encryption between the routers to determine if the other router is the one it claims to be. This is done using the Diffie-Hellman algorithm where asymmetric keys are used to create a common symmetric key. This symmetric key is used to allow both devices to encrypt and decrypt the data traffic sent between the devices. If the routers succeed in creating the phase 1 tunnel, which can also be called the management tunnel, they will switch to phase 2, and an IPSec tunnel will be created. The following

Hash	Hashing algorithms provides that the integrity is ensured. This is typically done with MD5 or Secure Hash Algorithms (SHA).
Authentication	ensures that the peer confirms and proves itself, this is done with a pre-shared key or digital certificates.
Group	Uses Diffie Hellman groups to determine the strength of the key that is used for the exchange process. The higher the group is, the more secure.
Lifetime	Specifies the lifetime of the transport or tunnel mode. Shorter deadlines provides more security.
Encryption	Ensures encryption throughout the transport or tunnel mode. For example DES, 3DES, or AES.

In phase 2, the same type of process is used except for authentication, since the routers have already authenticated each other. In phase 2, only hash and encryption must be specified; the rest is possible but not necessary as phase 2 can take this from phase 1. However, that another type of tunnel is created here, this is called IPSec. Here, traffic generated by end-users will be encrypted with the established rules specified in a so-called. *Transform Set*.

Authentication Header (AH)	AH is an IPSec protocol that ensures authentication only. AH provides data integrity and data origin authentication. In short, the data in the transmission is not tampered with.
Encapsulating Security Payload (ESP)	ESP protocol is another IPSec protocol that provides data confidentiality and and data authentication (integrity and origin authentication).

2.5.6 Internet Protocol Security (IPSec)

IPSec also is known as Internet Protocol Security provides security for services at the IP layer by enabling a system to select certain important security protocols, determining the algorithms that are needed to use the services, and by putting in place complex cryptographic keys to provide security for the requested services [42]. The parties that wish to create an IPSec tunnel must first negotiate on a standard way of communication since IPSec supports several modes of operation, both sides must decide on the security policy and mode to use. Encryption algorithms that are needed for the type of data that is being transmitted or received. In IPSec, all protocols that are on the network layer are encrypted between the two communicating parties. TCP, UDP, SNMP, HTTP, POP etc., are all encrypted regardless of their built-in (or lack of built-in) security and encryption.

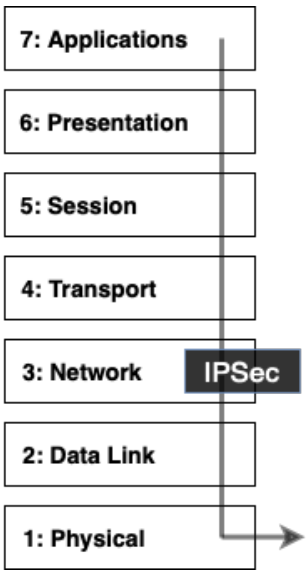

Figure 14: IPSec Layers.

When a VPN is used with IPSec, it is located on the Network layer, which is a Layer 3 protocol, as shown in Figure 14. Encapsulating Security Protocol (ESP) is used to provide packet-level data confidentiality and packet-by-packet host-level authentication, whereas an Authentication Header (AH) protocol is used to provide integrity checking. Both encryption and authentication/integrity check use private and public-key cryptography. IPSec also provides a session management protocol to manage the setting up of secure connections, authentication, and other policy-related things. In general, IPSec is created to work between any pair of networked computing devices (PCs, servers, routers, and others) while encrypting any IP traffic.

2.5.7 Symmetric and Asymmetric Encryption

Symmetric encryption is the most straightforward kind of encryption that involves only one secret key to cipher and decipher the information. It uses a secret key that is either a number, a word or a string of random letters, as shown in Figure 15. It is then blended with the plain text of a message in order to change the content in a particular way. Secret key needs to be known by the sender and receiver to encrypt and decrypt the messages [43]. Most widely used symmetric algorithms are AES-128, AES-192, and AES-256 [44]. The main advantage of symmetric encryption over asymmetric encryption is that it is fast and efficient for large amounts of data. The main drawback of symmetric key encryption is that the parties involved need to have the exchange key used to encrypt the data before one can decrypt it.

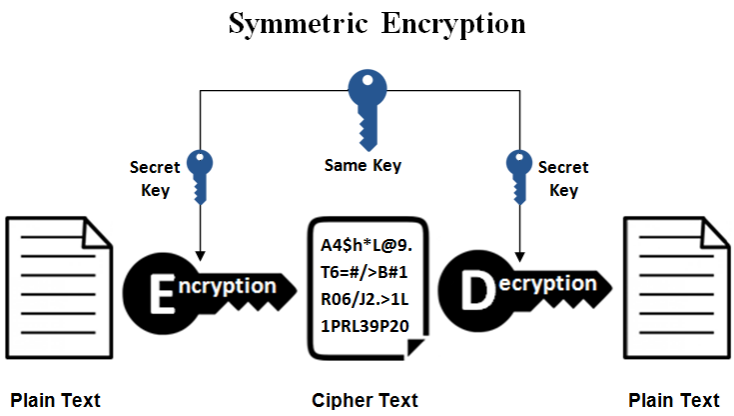

Figure 15: Symmetric Encryption. [12]

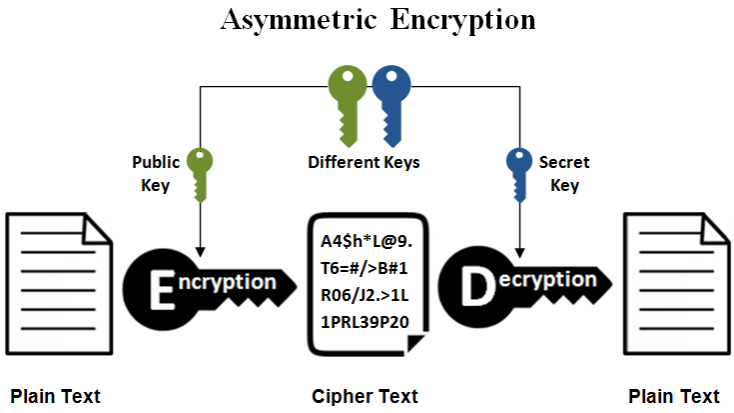

Figure 16: Asymmetric Encryption. [13]

Asymmetric encryption, also known as public-key cryptography is a relatively new method, compared to symmetric encryption. It uses public and private keys to encrypt a plain text as shown in Figure 16 [14]. The private and public keys are exchanged over the Internet or a large network. This ensures that malicious persons or intruders do not misuse the keys, and it is vital to remember that anyone with a secret key can decrypt the message. This is why asymmetrical encryption uses two related keys to boost security. A public key is freely made available to anyone who might want to send one a message. The private key is kept a secret so that only the desired person can

[12] https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences
[13] https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences
[14] https://www.rapidsslonline.com/blog/fundamental-differences-between-symmetric-and-asymmetric-encryption/

know. Any message that is encrypted by using a public key can be decrypted with a private and a message encrypted by using the private key can be decrypted with a public key [45]. Public key security is not essential as it can be passed over the Internet. The asymmetric key has far better power in ensuring the security of information transmitted during communication. One of the disadvantages of asymmetric encryption is that due to the presence of multiple parties and keys, the transmission is up-to three times slower than that of symmetric encryption. The main advantages of asymmetric encryption are that it allows the sending and receiving party to communicate the private key with one another, despite the communication being observed by a third party and since it contains multiple keys, the receiving party needs to keep only one private key to communicate with multiple senders.

2.6 Ethical Implications

eHealth monitoring systems provide many beneficial services for individuals and healthcare organizations and providers. With the help of these technologies and embedded sensor systems, the individual or the personnel can have continuous access to the health data. By using wireless communication to share this data, it leads to a better, reliable and cost-efficient healthcare system. On the other hand, this type of continuous access to data requires sensitive and confidential information such as name, location, underlying conditions, among many other things. Sharing such information raises important ethical challenges and problems regarding privacy and security [46]. If the means of communication and transferring of data is not secure then the individual's integrity, privacy and confidentiality are at risk and revealing of this data can lead to drastic consequences. An example of this kind of attack can be either snooping or spoofing, which creates a possibility for the third party to have access to this information [47]. Therefore, transmission and sharing of this confidential data should have the highest priority.

Regarding security in wireless communication, the Advanced Encryption Standard (AES) and Temporal Key Integrity Protocol (TKIP) are the two most commonly used standards. By using Dynamic Session Key and Extensible Authentication Protocol (EAP) for authentication, AES provides better security than TKIP or Rivest–Shamir–Adleman (RSA) algorithm while also providing confidentiality and integrity. Another mechanism such as using VPN and restricting user access can reduce the chances of threats and other vulnerabilities. Different standards have also been proposed to protect the integrity and privacy of users. One such regulation is the General Data Protection Regulation (GDPR) [15] which has been in place for almost two years and has been modernized the laws that protect the personal information of individuals. GDPR was designed to integrate data privacy laws across all the members' countries while also providing greater rights and protection to individuals. GDPR was also meant to alter how businesses and other organizations can handle the information of those that interact with them. In all the principles are similar to those that existed under the 1998 Data Protection Act.

[15] https://gdpr-info.eu/

3 Related Works

This section presents the related works on the topics and algorithms that are referred to and implemented throughout the thesis. Also, it discusses the main reasons behind selected works.. Different implementations on the software cycle development are done on the IEC standards 62304 and 82304-1 [48], thus it is important for one to understand how the software cycle needs to be maintained. When data is being sent through the cloud it is vital to lay down certain sets of rules and limitations to prevent confidential data from leaking out [49] [50]. Therefore, some revisions needs to be done to the pre-existing rules and observe the advantages the revisions bring to it. TLS is one of the cryptographic protocol that provides end-to-end communications security over networks that is investigated in this thesis. [51] [52], the method in which the types of messages are transmitted using this protocol varies depending on the scenarios, thus it is important to understand how certain type of files, for example: XML, CSV, .txt, etc. should be transmitted using the TLS protocol. IPSec is used to securely transmit the data over the Internet and different algorithms can be used alongside it while transmitting the data. For example: SSL, TLS, VPN and OpenVPN [53] [54]. A comparison is needed between these different algorithms to determine which is best suitable for transmitting the type of data that is required for this thesis.

Laukkarinen et al.[48], discussed the implementation of adopting DevOps methods in medical devices where software development is regulated. For this, two IEC standards, namely IEC 62304 and IEC 82304–1 were examined, to check for clauses that benefit or cause a conflict when working from the perspective of DevOps. The essential findings were that the standards required more focus and attention from continuous integration and also potentially prevent using continuous deployment after the protocol has been delivered to the customer. Thus, for using DevOps in a software regulated environment, new tools and methods should be developed specifically for this scenario.

Balboni and Iafelice [49] examined the privacy of sensitive Cloud Service Providers (CSPs) from Europe by observing and learning the rules designed for it. It also involved promoting robust network by investing a reliable and fail-safe mobile broadband network to move the data . It in concluded that the data protection roles set in the place of cloud environment are open to interpretation, making it difficult or impossible in establishing clear roles and obligations for controllers and processors and thus to absolutely allocate liabilities for protecting and processing the data. As Directive 95/46/EC, the protection of individuals with regard to the processing of personal data and on the free movement of such data, is a European Union directive adopted in 1995 which regulates the processing of personal data within the European Union (EU) is undergoing some changes and revision, it is important to either revise these rules and roles or to eliminate the needed dissimilarities and establish clear-cut criterion to allocate responsibilities and liability among the manufactures and developers.

Wolff et al. [50] reports on the on-going work of the key European research project MORE [16], which brings the embedded system world together with the Web Services world. MORE introduces a new connector concept for the handling of heterogeneous communication channels for reducing the message size for parsing XML messages. They conclude that the need for adapted Web Services for embedded systems is highlighted and motivated through performance measurements. A key characteristic of future work and asset of the project is the continuous involvement of application developers and end users. Once the validation is completed, it was planned to provide the middleware together with an SDK to the developer community.

Loreto et al. [51] proposes a method using which a middlebox performs TLS handshaking with client using its own certificate when it captures the client's ClientHello intended to a server. The middlebox's certificate can be distinguished from a server certificate by the client using the X.509 Extended Key Usage extension, so that the client learns that a middlebox is involved. The client can choose to accept the middlebox's certificate or not, depending on whether the middlebox should

[16]MORE: Network-centric Middleware for Group communication and Resource Sharing across Heterogeneous Embedded Systems

be introduced into the current session.

Lee et al. [55] proposed maTLS, where multiple middleboxes can be introduced into a session. Each two adjacent nodes establish a TLS connection, which is addressed in maTLS as a TLS segment. Each middlebox's certificate is sent to the client for explicit middlebox authentication. The security information of each TLS segment, including TLS version, cipher suite are also sent to the client for security parameter verification. To cope with the problems with the PKI system, maTLS introduces the middlebox transparency log servers for middlebox. The log servers generate signed certificate timestamps for middlebox certificates, which can be verified by the log servers' public keys. maTLS endpoints can learn which middleboxes modified traffic data, and perform path verification on the middleboxes, at the cost of more computation overheads. In maTLS, middleboxes join a session by modifying the ClientHello and ServerHello messages, but how a middlebox learns if it should join a session is not specified. In addition, there lacks a mechanism to prevent an undesired middlebox from joining a session except hanging up the connection.

Ahmad and Yacoob [52] discussed the incompatibilities issues between IPv4/IPv6 translation gateway and IPSec, since the presence of IPv4/IPv6 translation gateway provides transparent routing mechanism to IPv4-only nodes and IPv6-only nodes which trying to establish communication from disparate address realms. However, the mechanism breaks TCP/IP intrinsic functionalities that results in IPSec inability to be applied in this environment. The existing solutions to address the compatibility issues between translation gateway and IPSec are either to enhance the translation gateway operation or to modify IPSec architecture especially on IKE negotiation process. The solution proposed in the paper offers an effective and simple way to ensure end-to-end security using IPSec across the translation gateway. Due to the complexity of IPSec especially on IKE negotiation across the translation gateway, this paper proposed a new ABK mechanism to secure the communication channel between end nodes using their end IP addresses as public/secret keys for authentication. The proposed scheme achieves significant reduction in system complexity and the cost for establishing and managing the public key authentication such as PKI.

Kotuliak et al. [53] concentrated on VPN technologies which utilize SSL/TLS or IPsec protocols to create secure tunnel for data transmission, e.g. to interconnect two IP Multimedia Subsystem (IMS) networks which is an architectural framework for delivering IP multimedia services. A several tests have been performed to compare these technologies based on parameters such as throughput, response time and so on. It can be summarized by mentioning that it is difficult to choose between IPSec and TLS based on all views. Each user has different needs. They decided to choose OpenVPN [17], due to its simplicity and fast and straightforward implementation. On the other hand IPSec is somewhat faster and as it has been on the market much longer than SSL VPN solutions and it has far more support among hardware and software vendors.

Alshamsi and Saito [54] presented the differences between IPSec and SSL. The paper mentions that each of the protocols have their own unique properties. Choosing IPSec or SSL depends on the security needs. If some service is required and is supported by SSL it is better to select SSL, and if services or Gateway-to-Gateway communications are needed then IPSec is a good choice. Since IPSec uses a shorter form of Hash-based Message Authentication Code (HMAC), which is a mechanism for calculating a message authentication code involving a hash function in combination with a secret key than SSL, the data integrity is more secure and compatible with firewall. IPSec supports compression whereas SSL does not and IPSec is capable of protecting wireless networks in most cases. If the sender and receiver has different network vendors then IPSec is not suitable for data transmission. In this case using SSL is more suitable.

[17] https://openvpn.net/

4 Methodology

This Thesis aims to develop a communication middleware system that works efficiently for embedded sensor systems transmitting health parameters. Systems development research process presented by Nunamaker and Chen [56] is suited for this purpose. The research process for the method is separated into five major steps as shown in Figure 17.

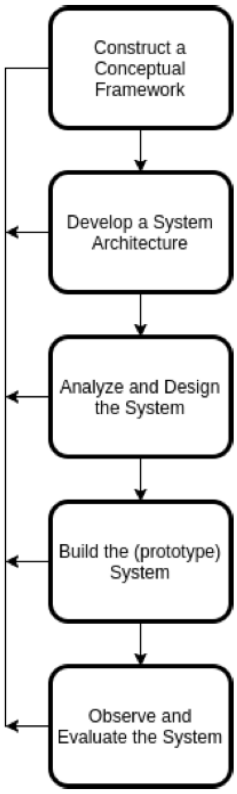

Figure 17: Systems Development Research Process. [56].

The first stage in the research process is the focus of forming a conceptual framework, i.e. constructing a meaningful research question, investigating system requirements, understanding building process, furthermore to do literature studies on relevant subject for ideas and approaches. Defining in the second stage, it focuses on developing a unique system architecture on the development and the function of the components and inter-relationship of the system. The third stage aims to design the system, functions and to analyze the system through different designing approaches to finalize a solution. The fourth stage focuses on the research process, gaining insight into the system and its concepts through the building process. This stage also gives valuable insight into the complexity of the system. The fifth and final stage focuses on the experimenting, observing and evaluating phase of the built system. Observation of the system is done by case studies during this stage.

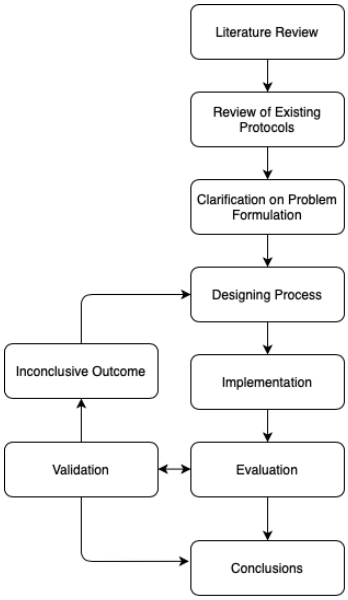

Figure 18: General Research Process.

The research is divided into eight stages, Literature Review, Review of Existing Protocols, Clarification on Problem Formulation, Designing Process, Implementation, Evaluation, and Conclusions. The thesis work could be summarized in the following steps in Figure 18. To understand and examine the research question presented in Section 1.1 a literature review is done into standards and technologies. After reviewing technologies and standards, a clarification is done to check if that is in-line with the problem formation. During the designing phase, a topology is designed with the problem formulation in mind. Using the designing phase, implementation was done as a proof of concept. From the implementation, controlled experiments were conducted to collect data on the performance of the system. As a side step to evaluation, there is a Validation process where the result is checked for satisfaction. If the outcome is inconclusive or not desirable, then a redesigning process is done and if the validation processes provide satisfactory results a conclusion is drawn from the results.

5 System Design

Initially, the goal was to implement the communication protocol by using the hardware components developed by Tjeders AB and also figuring out a way on how to get the operating table data out of the hospital network for Stille AB. After meeting with the companies, the communication protocols inside the home-care environment was supposed to be with loraWAN and the communication protocols outside the home-care environment with a secure communication profile like MQTT or IPSec while being in accordance with MDR. Due to the ongoing COVID-19 pandemic, there was a lot of restrictions from the clients side to get access to the hardware requirements and it restricted us to implement the entire client requirements mentioned in Section 1.2. To overcome this limitation, a different type of approach with change of network setup was planned to be implemented by using the Raspberry Pi 4 development board to act as a sender and receiver in the internal communication with the data to be transferred securely with Bluetooth 5.0 along with symmetric/asymmetric encryption. Whereas, the external communication done via IPSec with tunnel mode VPN was designed and simulated through a simulation software called GNS3 wherein the routers and switches were added and interfaced with the Raspberry Pi. Later, the client specified that the end to end communication needed to be done by using MQTT Broker along with TLS 1.3, and this setup was also simulated by using GNS3. The common outcome of this thesis is to construct a conceptual framework and develop a system architecture for the companies. The following sections will present the approach and scenarios implemented for the clients.

5.1 Use Case - Tjeders AB

During the discussions with Tjeders, a use case for the system was based on a home-care setup. For this setup, the internal communication i.e, within the home-care had to be done by using Bluetooth 5.0 secure profile along with either symmetric or asymmetric encryption algorithm. This encrypted data is then sent to the Raspberry Pi receiver along with the secret key that is required for decryption. The receiver then processes this data and publishes to the Internet by using MQTT broker topics. On the receiving side, Tjeders has a centralized front-end server that subscribes to the particular MQTT topic and the data is decrypted after the transfer is completed. The encryption of sampled data over the Internet is handled by IPSec that is using transport mode VPN, this achieves the end-to-end encryption of the sampled data from the internal home-care setup to the front-end. Figure 19 shows the network setup for our simulations.

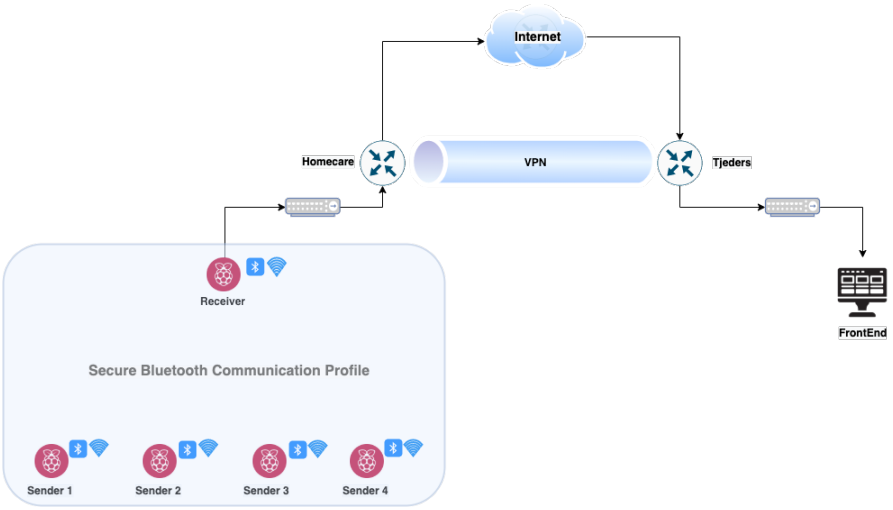

Figure 19: Tjeders Network Setup for Home-Care Use-Case.

The data being transferred from the sensors in the home-care setup is a periodical sampled data over time. If the Bluetooth communication breaks, the sender collects the sensor data that is being

sampled for the time frame and when the connection is re-established the collected data along with the newer sampled data will be transferred to the receiver.

5.2 Use Case - Stille AB

Stille's use case for the system is similar to that of Tjeders', except the data should be sent via the sender is a CSV (Comma Separated Value) file. During the operation procedure, an operating table equipped with multiple sensors goes online and starts collecting data while storing it locally. When the operating table is not in use, it goes offline and the data is sent to the receiver via a secure Bluetooth communication profile similar to Tjeders' scenario.

For the external communication, the requirement is to access the sensor data that is collected out of the hospital's network to Stille's centralized server. Due to the regulations that needs to be fulfilled for taking out any data from the hospital's network, this was not looked upon because Stille is still in negotiation with the hospital authority and the pandemic situation further delayed it. To test the proof of concept, a setup similar to Tjeders is simulated in GNS3 using IPSec. Figure 20 shows Stille's network setup.

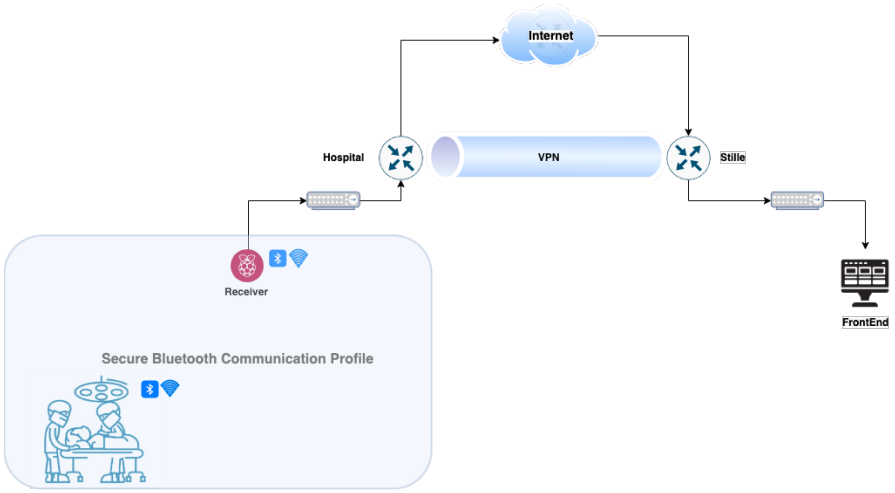

Figure 20: Stille Network Setup for Operating Table Use-Case.

6 Implementation

This section presents the need for the types of equipment and software technologies being used along with their advantages and disadvantages when comparing to their counterparts. The section will also, in brief, explain the impact of choosing these technologies to achieve our goals.

6.1 Raspberry Pi

Raspberry Pi 4 Model B is the latest version from the Raspberry Foundation. This device has a high-performance 64-bit quad-core processor and supports dual-display resolutions up-to 4K. It has RAM memory support up to 4 GB, dual-band 2,4/5 GHz wireless support and Bluetooth 5.0. A sample Raspberry Pi Gen 4 is shown in Figure 21.

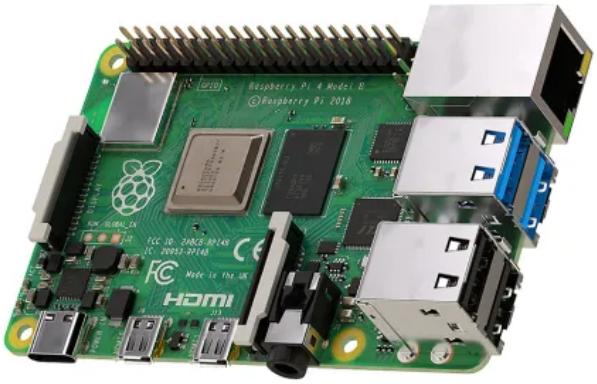

Figure 21: Raspberry Pi Gen 4. [18]

Raspberry Pi 4 was chosen over different development boards like BeagleBone Black [19] because of the different advantages that the Raspberry Pi offers. Such as, the Raspberry Pi model 4 version has 2.4GHz 802.11n Wi-Fi which works very well with the latest version of any default operating system, the chip enables 28 GPIO pins plus 12 power and ground pins to be used in a project. There are a few particular protocols which Raspberry supports. These include IIC (Inter-Integrated Circuit), SPI (Serial Peripheral Interface) and UART (Universal Asynchronous Receiver-Transmitter). This functionality enables Raspberry to connect with various devices which results in better functionalities. Raspberry provides better documentation Beagle enables a user to choose their own hardware.

6.2 MQTT Broker with TLS 1.3

MQTT Broker is used along with IPSec to send the data from the central hub to the server and to the front-end by using TLS version 1.3. This is chosen because considering the overhead that is created by using IPSec, along with the clients' requirement to send the data packets over the Internet using an MQTT broker. MQTT adds less overhead and takes into consideration such as limitations of CPU power and bandwidth. Since the users at the front-end, need to get information on a specific topic, this implementation, along with TLS 1.3 provides the most efficient solution. By providing a publish/subscribe model, MQTT provides a lightweight method of carrying out messaging. This makes it suitable for IoT messaging such as with low power sensors or mobile devices such as phones, embedded computers or micro-controllers. The Mosquitto project [20] also provides a C library for implementing MQTT clients, and the very popular mosquitto_pub and

[18] https://www.raspberrypi.org/products/raspberry-pi-4-model-b/
[19] https://beagleboard.org/bone
[20] https://mosquitto.org/

mosquitto_sub command line MQTT clients.

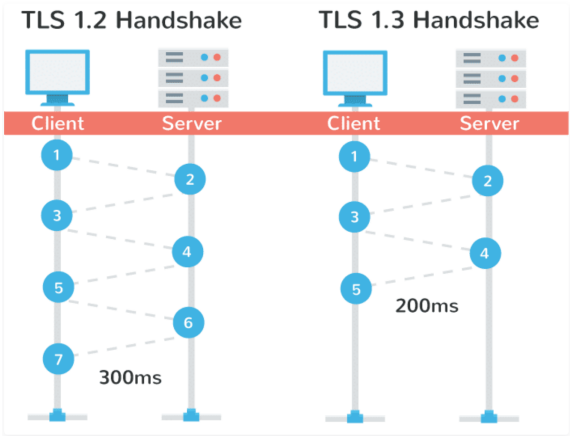

Figure 22: TLS 1.3 Handshake. [21]

TLS 1.3 has improved security and speed over the previous version. The list of symmetric algorithms supported has been sheared of all legacy algorithms. Static RSA and Diffie-Hellman cipher suites have been removed, and all public-key based fundamental exchange mechanisms now provide forward secrecy. The TLS handshake working is shown in Figure 22. With and without server-side state session resumption and the Pre-shared key (PSK)-based cipher suites of earlier versions of TLS have been replaced by a single new PSK exchange. There is always added a slight overhead when it comes to web performance with TLS and encrypted connections. HTTP/2 definitely helps with the problem, but TLS 1.3 speeds up encrypted connections with features such as TLS false start and 0-RTT. With TLS 1.2, two round-trips were required to complete the handshake. TLS 1.3 requires only one round-trip which cuts the encryption latency in half.

6.3 IPSec Transport Mode

Different types of IPSec are available for encapsulating the data and sending it over the Internet, for this implementation Tunnel Mode is used because the entire IP of the packet being sent is protected by IPSec, this means that the IPSec wraps the packet, encrypts it and adds a new IP address and sends it to the other side of the VPN tunnel. This is suitable as the implementation consists of an end-to-end gateway, and the IPSec acts as a proxy for the host. IPSec can be used to create VPN Tunnels to end-to-end IP Traffic (also called as IPSec Transport mode) or site-to-site IPSec Tunnels (between two VPN Gateways, also known as IPSec Tunnel mode). In IPSec Tunnel mode, the original IP packet (IP header and the Data payload) is encapsulated within another packet. In IPSec Tunnel mode the original IP datagram from is encapsulated with an AH (provides no confidentiality by encryption) or ESP (provides encryption) header and an additional IP header. The IP addresses of the newly added outer IP header are that of the VPN Gateways. The traffic between the two VPN Gateways appears to be from the two gateways, with the original IP datagram is encrypted (in case of ESP) within IPSec packet. IPSec Tunnel mode is most widely used to create site-to-site IPSec VPN. How different IPSec modes affect the packet is shown in Figure 23. In IPSec transport mode, only the data payload of the IP datagram is secured by IPSec. IP Header is the original IP header, and IPSec inserts its header between the IP header and the upper-level headers. IPSec transport mode can be used when encrypting traffic between two hosts or between a host and a VPN gateway [22].

[21] https://kinsta.com/blog/tls-1-3/
[22] https://heranonazure.wordpress.com/2019/10/11/ip-security-IPSec-basics/
[23] http://www.firewall.cx/networking-topics/protocols/870-ipsec-modes.html

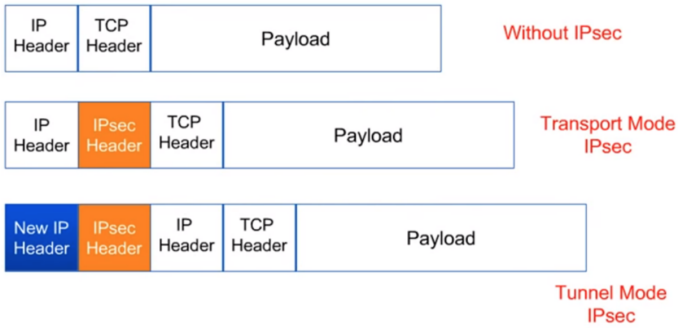

Figure 23: IPSec Tunnel vs Transport Mode. [23]

6.3.1 Advanced Encryption Standard (AES)

AES and RSA are the two most commonly used encryption algorithms when sending a packet over the Internet. For the implementation, AES is used because the algorithm uses 128 or 256-bit keys for both encryption and decryption making it extremely difficult to be hacked by an intruder. Also, AES maintains the confidentiality and higher throughput of the encrypted packet that is being sent. AES algorithm is based on a design principle known as a substitution-permutation network which is a combination of both substitution and permutation and is fast in both software and hardware [57]. It is based on a secret key cryptography algorithm in which identical keys are used for encryption and decryption. The steps of the AES algorithm through which a plain text is converted into the ciphertext is shown in Figure 24.

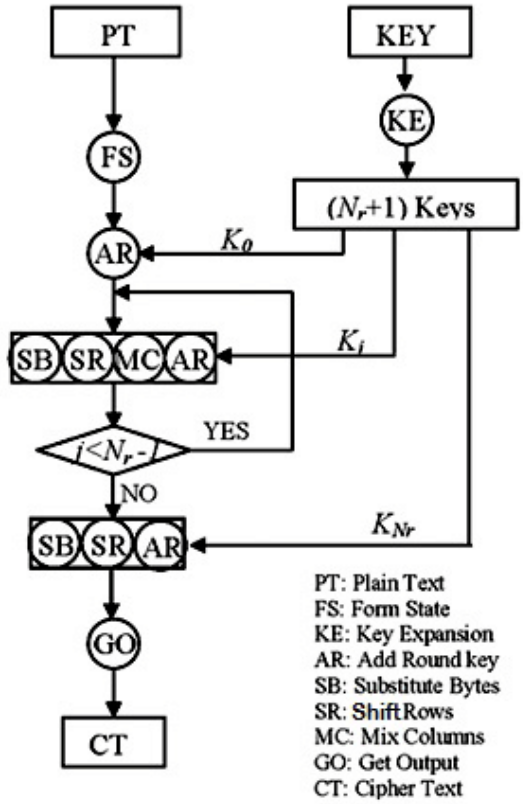

Figure 24: AES Flowchart. [58]

RSA algorithm is a public-key cryptography algorithm developed by Rivset et al. in 1977 [59]. It is used by today's modern computers to encrypt and decrypt the messages. It is an asymmetric key

cryptographic algorithm which is used for digital signature. The principle of the RSA algorithm is on the basis that it is easy to multiply prime numbers but hard to factor. Hence large prime numbers are used to generate public-key and private key respectively, as it usually takes a long time.

From the above analysis, it is clear that RSA solves the problem of key agreement and critical exchange generated in private-key cryptography algorithms, but still, there is a lack of confidentiality. Therefore, for enhancing the security, a comparative analysis along with various parameters for both the symmetric and asymmetric key encryption is presented in Table 1.

S. No	Features	AES	RSA
1	Type	Symmetric	Asymmetric
2	Key Used	Single Key	Different Keys
3	Throughput	Very High	Low
4	Confidentiality	High	Low

Table 1: AES and RSA Comparison.

AES is a private key based algorithm that suffers from crucial distribution and critical agreement problems, the key distribution and agreement problems are solved in RSA algorithm, but encryption and decryption take more time by using this. Therefore, both algorithms have their own merits and demerits.

6.4 Site-to-Site VPN

For the implementation, Site-Site VPN is used because of the architecture being gateway-gateway transfer and tunnel mode IPSec being used. One of the main advantages of this type of VPN is that it allows multiple users from fixed locations to establish secure connections and communicate with each other over a public network. When data needs to be transferred to a private network over the Internet, two VPN types are used to achieve this: Remote-access VPN connection allows an individual user to connect to a private network from a remote location over the Internet. Representation of the working of remote access VPN is shown in Figure 25. Two components are required in a remote-access VPN, the network access server (NAS), and a remote-access server (RAS). A NAS might either be a dedicated server, or one of the multiple software applications running on a shared server. Site-to-Site VPN communicates with the VPN gateway, which authenticates the individual as a remote user, and creates a secured "virtual" tunnel between the LAN and the gateway. Once the tunnel is created, any data that is sent from the device is encapsulated and encrypted by the remote-access VPN, and then sent to the VPN gateway that sits just outside the remote LAN.

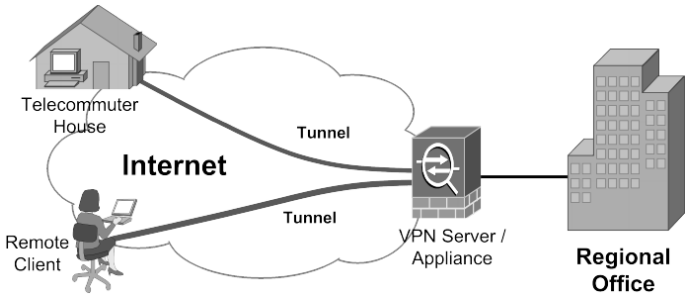

Figure 25: Remote-Access VPN. [60]

Remote-access VPNs securely connect individual devices to a remote LAN, site-to-site VPNs securely connect two or more LANs in different physical locations. Site-to-site VPNs use the public Internet to extend the network across multiple office locations. There are two common types of site-to-site VPNs: Intranet-based and Extranet-based. Intranet-based site-to-site VPNs are used to combine the LANs of multiple office locations into one single private network, which would then

be known as a WAN (Wide Area Network). Configuration of a typical site-site VPN is shown in Figure 26. Extranet-based site-to-site VPNs, on the other hand, allow one to use the public Internet to connect its LAN with other individuals. This allows sharing information with its partners, while still securing its LAN (intranet). With a site-to-site VPN, the VPN gateway of one remote LAN communicates with the gateway of another LAN to create a secure tunnel. Unlike remote-access VPNs, the remote devices do not need a VPN client, but rather send regular traffic through the VPN gateways. In the absence of VPN clients, the VPN gateways are in charge of authentication of the user and the network, encryption, and the integrity of the data. The gateway receives the encrypted data, decrypts it, and then sends the data to the target device in the network. The tunnel created by the site-to-site VPN, allows an individual to share their network and resources between its primary and remote branches regardless of the distance.

Figure 26: Site-Site VPN. [24]

6.5 Bluetooth 5.0

Bluetooth 5.0 achieves double speed, four times the transmission range, and eight times the advertising capacity than its previous version Bluetooth 4.0 according to Bluetooth SIG [61]. Apart from these, the new version also shows stronger robustness to interference compared to Bluetooth 4.2. For the newer version, plenty of new features are introduced in Bluetooth 5.0 [62]. Most of the innovations in this version are specifically proposed for BLE, and Bluetooth Basic Rate/ Enhanced Data Rate (BR/EDR) remains identical. Features that match the three most significant enhancements of Bluetooth 5.0:

- The newly added modulation scheme in the physical layer allows BLE to use 2MHz bandwidth to transmit data, which is double the speed.
- LE Long Range is for the quadrupled transmission range, which is done by a new coding scheme.
- LE Advertising Extensions contributes to the eight times advertising capacity. It utilizes another 37 channels that are not used for advertising in older versions.

In regards to Bluetooth, an entire period of a packet contains two segments of inter-frame space-time, one slot is for the received packet from the sender's peer device and the other slot for transmitting data, as shown in Figure 27. The packet transmitting time and response receiving time are halved, whereas the inter-frame space-time remains the same as previous versions. The specific time of the slot for transmitting data is reduced from 2,120 μs to 1,060 μs, and from 80 μs to 40 μs for receiving confirmation data. Inter-frame space-time remains the same at 150 μs.

Another new feature, high duty cycle non-connectable advertising, contributes to increased speed when considering certain advertising events. Different types of advertising events are defined in Bluetooth. In Bluetooth 5.0, the minimum advertising intervals of different advertising events are uniform at 20 ms. Theoretically, the advertising data rates of the advertising events can be four times larger.

[24] https://campus.barracuda.com/product/nextgenfirewallx/doc/14320455/example-configuring-a-site-to-site-ipsec-vpn-tunnel/

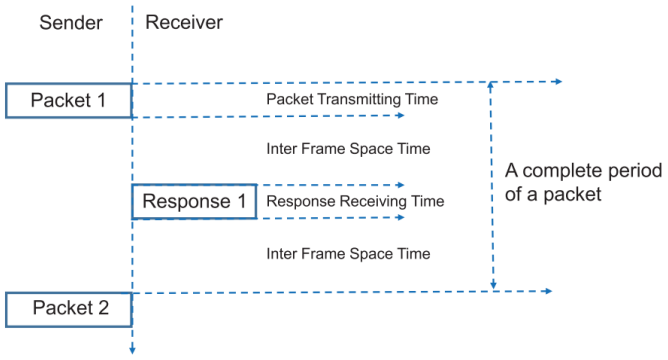

Figure 27: Parts in a complete period of a Bluetooth packet. [63]

6.6 Graphical Network Simulator 3 (GNS 3)

GNS 3 is a simulation software that provides an interface to emulation software such as Dynamips, VirtualBox [25], while also enabling the emulation and configuration of network systems with different devices and different operating systems. In this way, it is possible to test different physical hardware with GNS3. With Virtualbox, it is possible to add computers that emulate different operating systems to the virtual network system [26]. GNS3 can be installed on various operating systems.

The most significant difference of GNS 3 software from Cisco Packet Tracer [27] software is that GNS 3 is an emulator and Cisco Packet Tracer is a simulator. So while GNS 3 runs the operating system used on a real router, Packet Tracer uses a software-defined virtual operating system. While this prevents one from using all configuration commands in the Packet Tracer software, all commands valid for IOS used in GNS 3 can be used. Another critical difference is that switching devices (switch) are not emulated in GNS 3, while this is possible in packet tracer. In GNS 3 software, switching devices can be used only as unmanageable switches. Although the switching devices available by default in the GNS 3 software are unmanageable, this can be overcome by using routers as a switching device.

[25] https://www.virtualbox.org/
[26] https://docs.gns3.com/
[27] https://www.netacad.com/courses/packet-tracer

7 Experiments

In this section, we evaluate different security solutions for the considered use cases discussed in Section 5. We conducted two separate experiments for evaluating the Bluetooth connection with different encryption methods in 7.2 and 7.3. Next, we evaluate the entire system where the measured values are sent across the Internet to the server. Here we also considered two different setups where the measured value sent over Bluetooth will be forwarded to the server using IPSec and TLs 1.3 using MQTT broker. The packets sent using IPSec in Section 7.4 is sent by using tunnel mode with site-to-site VPN and the packets are encrypted with AES. In Section 7.5 the same packets are transmitted with TLS using MQTT broker. Here the packets are encrypted with both AES and RSA.
The performance that are measured through this experiments are overhead, round trip delay and throughput.

Overhead is the time it takes to transmit data on a network that transmits a packet-sized data. Each packet requires some extra information which is stored in the packet header as extra bytes which when combined with the creation and sending of packets reduces the overall transmission speed of the data. Round trip delay is the time it takes for the packet data to be transmitted again when the previous packet data that is transmitted over the network sends an acknowledgement to the transmitter after it has arrived at the destination which in this case is the front-end system. And, throughput is the amount of data that is transferred from the sender to the receiver successfully in a given time period. This is used to determine the total bandwidth of the network.

7.1 Experimental Setup

To evaluate and implement the proposed topologies in Section 5, a collection of protocols, hardware, and software were used. These are covered in Section 6. The outline for different experiments are covered in Section 7. The general hardware used for the thesis was the following:

- A PC for simulating the network (GNS3)
- Three Raspberry Pi Gen 4
- A 4 Ports Generic Switch

7.1.1 GNS 3 Simulation

For the implementation, to simulate the network segmentation of the experiments, GNS3 was used. The topology was set up with three Cisco router images and two generic switches. To connect the Raspberry Pis to the simulation, needed to bridge the laptop's network interface with the GNS3. The router image used for the experiments was Cisco IOSv 15.7(3)M3. These three routers represented each a site with a geographical location and were implemented accordingly to each scenario as mentioned in the Section 7.4 and 7.5.

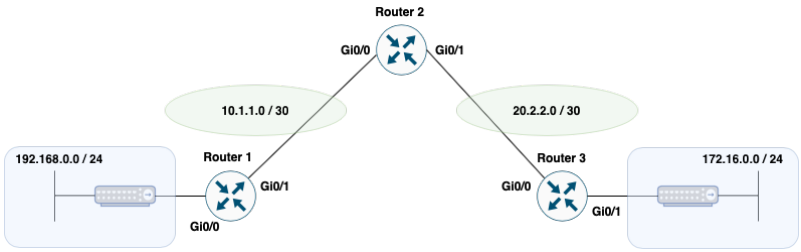

Figure 28: Basic GNS3 Setup for Experimental Purpose.

The routers were configured with a Dynamic Routing Protocol, OSPF (Open Shortest Path First) [64] to advertise routes between each router and to establish connection. This was done to create a more realistic experiment. A basic IP table was created for the experimental purpose and each interface was assigned an IP address with suitable sub-net. This is visualised in Figure 28.

7.2 Bluetooth Communication with Symmetric Encryption

In this scenario, a sampled data of size 5000 packets is encrypted by using the symmetric encryption algorithm. After the encryption, the packets are transmitted over Bluetooth to the receiver via the default communication profile. After all the packets are transmitted, the receiver contains the public key that was used in encrypting these packets. This public key is used to decrypt the packets and make them ready to be transmitted over the Internet. Different variations of the payload are tested by varying each packet size to 16 and 128 bytes. Visual representation of the experiment is shown in Figure 29. A transmission delay is also added to the packets that varies from 10 ms to 1000 ms. That is, each packet is transmitted with this delay after it receives an acknowledgement from the previous packet after it is decrypted. This is done to find the overhead and the round trip delay. Overhead, round trip delay and throughput are compared with packets that are transmitted with no encryption.

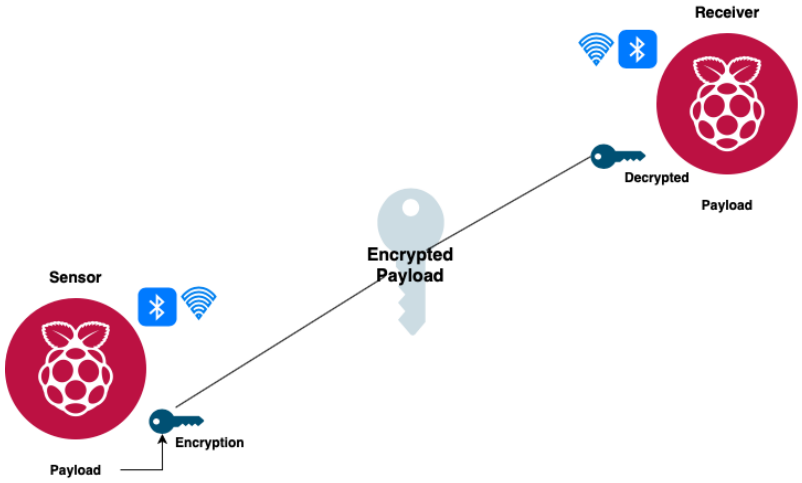

Figure 29: Symmetric Encryption with Bluetooth 5.0.

7.3 Bluetooth Communication with Asymmetric Encryption

In this scenario, a sampled data of size 5000 packets is encrypted by using the asymmetric encryption algorithm. After the encryption, the packets are transmitted over Bluetooth to the receiver via the default communication profile. After all the packets are transmitted, the receiver contains the private key along with the public key that was used in encrypting these packets. This public key is used to decrypt the packets and make them ready to be transmitted over the Internet. Different variations of the payload are tested by varying each packet size to 16 and 128 bytes. Visual representation of the experiment is shown in Figure 30. A transmission delay is also added to the packets that varies from 10 ms to 1000 ms. That is, each packet is transmitted with this delay after it receives an acknowledgement from the previous packet after it is decrypted. This is done to find the overhead and the round trip delay. Overhead, round trip delay and throughput are compared with packets that are transmitted with no encryption. This comparison is also done with the packets that are transmitted with symmetric encryption.

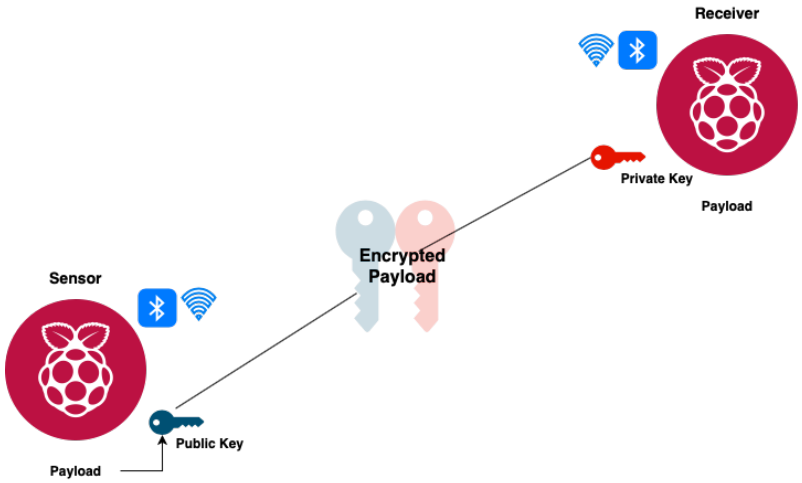

Figure 30: Asymmetric Encryption with Bluetooth 5.0.

7.4 Communication over IPSec

The packets that have arrived at the receiver in scenarios 7.2 and 7.3 must be further transmitted over the Internet to the central server and back to the front-end securely. For this, the simulation software GNS 3 is used to emulate the scenario where IPSec with transport mode VPN is used to transmit the packets. The encryption algorithm used here is AES with 256 bit key length. In this scenario, the packets that have arrived at the receiver are sent to the front-end system over the Internet by using IPSec. Similar to previous scenarios, payloads of packet size 16 and 128 bytes are transmitted over the Internet to the receiver. Visual representation of the experiment is shown in Figure 31. After this, overhead and throughput for the different payload sizes are calculated in order to determine the cost of the system for sending each packet by using the IPSec method with AES 256 bit key encryption algorithm. Similarly, a CSV format file is transmitted to calculate the throughput and overhead. Combining the overhead, time and throughput from the previous Bluetooth experiments, the overall cost to the system for transmitting a sampled packet is calculated.

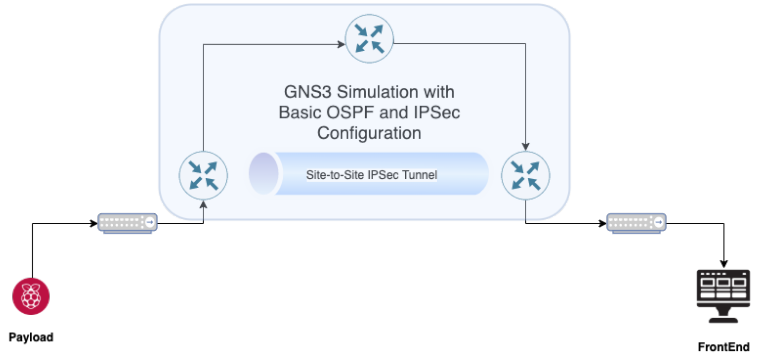

Figure 31: Site-to-Site IPSec VPN over GNS3 Simulated OSPF Network.

7.5 Communication over MQTT with TLS 1.3

The packets that have arrived at the receiver in scenarios 7.2 and 7.3 must be further transmitted over the Internet to the central server and back to the front-end securely. For this, the simulation software GNS 3 is used to emulate the scenario wherein a MQTT broker along with TLS 1.3 is used to transmit the packets. The receiver with the encrypted packets acts as a MQTT_publisher and the front-end systems act as MQTT_subscriber. Each packet here is encrypted with AES 256 bit algorithm along with 2048 key RSA. Visual representation of the experiment is shown in Figure 32. This experiment aims to focus only on MQTT with TLS 1.3 without IPSec layer. Combining the overhead, time and throughput from the previous Bluetooth experiments, the overall cost to the system for transmitting a sampled packet is calculated.

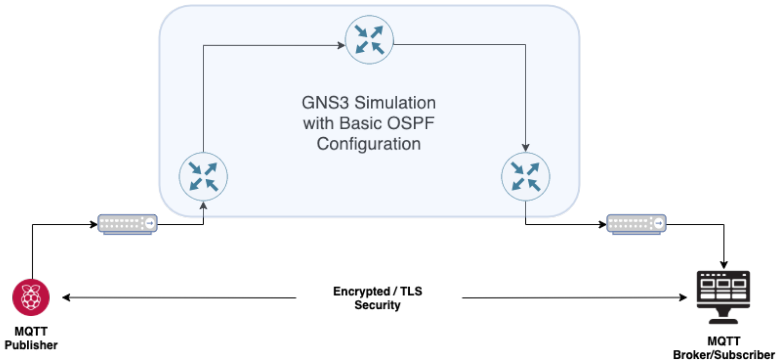

Figure 32: MQTT with TLS 1.3 over GNS3 Simulated OSPF Network.

8 Results and Discussion

This section discusses the different results obtained from the experiments conducted using Bluetooth communication with symmetric and asymmetric encryption, transmission of decrypted packets with IPSec and TLS 1.3 with MQTT while also explaining the significance of these results. Different scenarios that were carried out for the sections are observed and their findings are explained.

8.1 Bluetooth Communication

Results for the experiment using Bluetooth communication profile using symmetric and asymmetric encryption are shown from Figures 33 to 39 for the transmission delays of 10 ms, 100 ms and 1000 ms. The experiments are carried out for 5000 packets that are transmitted in three forms: raw packets with no encryption, packets with symmetric encryption and packets with asymmetric encryption. The raw and encrypted packets are sent over Bluetooth to the receiver where each packet that is received is encrypted with either the public or private key depending on the encryption algorithm used. The purpose of this experiment is to calculate the total round trip delay that each packet of size 16 and 128 bytes takes. This delay is the difference between the time taken by the packet to send an acknowledgement signal to the transmitter to send in the next packet and the time the previous packet had taken to reach the receiver. This delay will provide the total overhead and cost that is required for the system to send the packets using symmetric, asymmetric encryption and no encryption.

8.1.1 Raw Data

For raw data irrespective of the data the overhead doesn't exist as it is not encrypted with any algorithm. The round trip delay for this type of data will be least because just the total size of the packet without encryption will be transmitted to the receiver. The average mean delay differs by almost 10 μs when compared between 16 and 128 bytes. The results are observed in Figures 33, 34 and 35 for 10 ms, 100 ms and 1000 ms transmission times respectively. The peaks obtained are the highest for 1000 ms transmission time, this is because for each packet that sends an acknowledgement, 1000 ms is added as a delay for sending in the packet which adds to the total round trip delay in general.

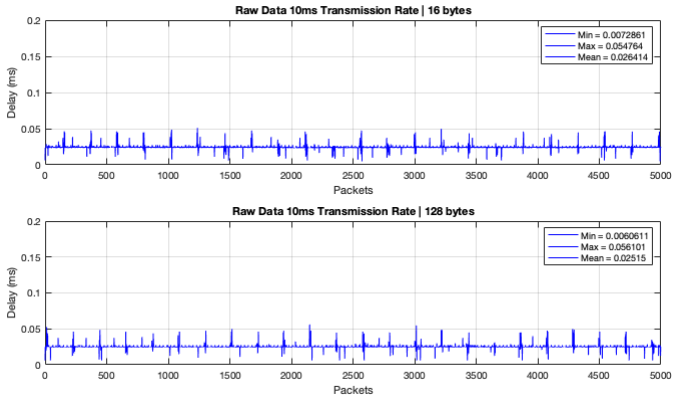

Figure 33: Delay for Raw Data over Bluetooth 5.0. Transmission Interval 10ms.

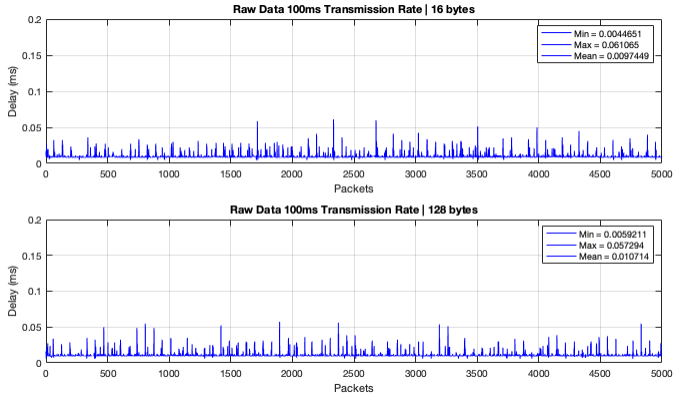

Figure 34: Delay for Raw Data over Bluetooth 5.0. Transmission Interval 100ms.

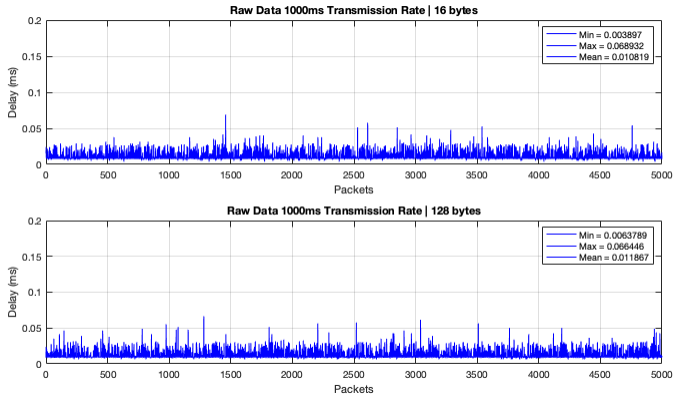

Figure 35: Delay for Raw Data over Bluetooth 5.0. Transmission Interval 1000ms.

The round trip delays for different transmission times using the raw packets is shown in Table 2.

Table 2: Round Trip Delays for Bluetooth Communication without Encryption.

		Round Trip Delay (ms)								
		Min			**Max**			**Mean**		
		Transmission Delay (ms)								
Packet Type	**Packet Size (bytes)**	**10**	**100**	**1000**	**10**	**100**	**1000**	**10**	**100**	**1000**
Raw	16	0.00728	0.00446	0.00389	0.05476	0.06106	0.06893	0.02641	0.00974	0.01081
	128	0.00606	0.00592	0.00637	0.05610	0.05729	0.06644	0.02515	0.01071	0.01186

8.1.2 Symmetric Encryption

For symmetric encryption each packet is encrypted with a key of 32 bytes and this adds to the total overhead. Combining with the different transmission delays and the overhead the total round trip time will increase when compared with raw data. The mean delay for 1000 ms is increased again by an order 2 μs when compared to raw data packets. The results are observed in Figures 36, 37 and 38 which represents the minimum, maximum and mean delay for the packets transmitted. Therefore, as the number of packets increases the round trip delay will also increase when combined with the average mean delay for each packet.

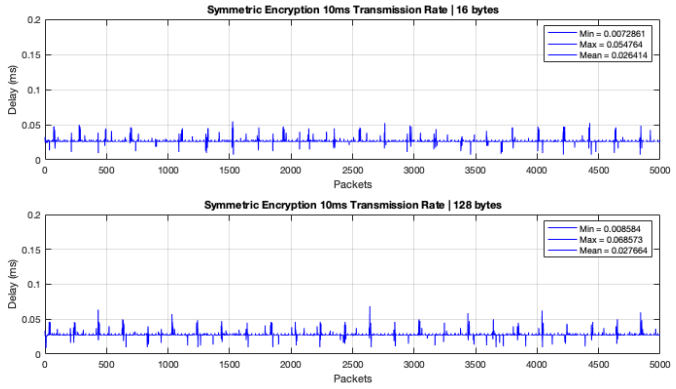

Figure 36: Delay for Symmetric Encryption over Bluetooth 5.0. Transmission Interval 10ms.

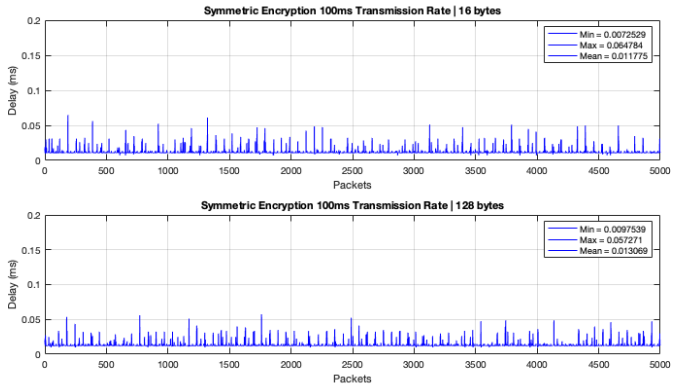

Figure 37: Delay for Symmetric Encryption over Bluetooth 5.0. Transmission Interval 100ms.

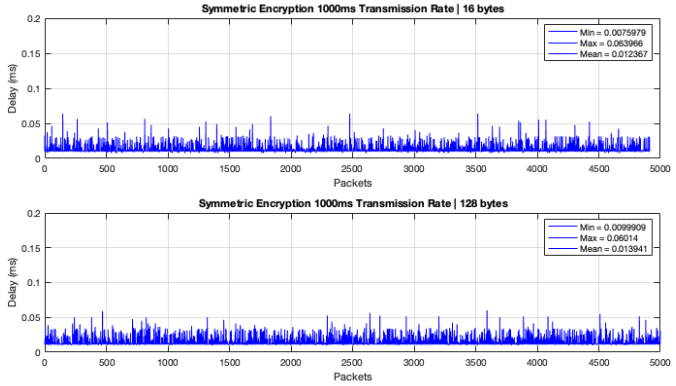

Figure 38: Delay for Symmetric Encryption over Bluetooth 5.0. Transmission Interval 1000ms.

The round trip delays for different transmission times using the packets with symmetric encryption is shown in Table 3.

Table 3: Round Trip Delays for Bluetooth Communication with Symmetric Encryption.

		Round Trip Delay (ms)								
		Min			**Max**			**Mean**		
		Transmission Delay (ms)								
Packet Type	**Packet Size (bytes)**	**10**	**100**	**1000**	**10**	**100**	**1000**	**10**	**100**	**1000**
Symmetric	16	0.00723	0.00725	0.00759	0.05476	0.06478	0.06396	0.02641	0.01177	0.01236
	128	0.00858	0.00975	0.00999	0.06857	0.05727	0.06014	0.02766	0.01306	0.01394

8.1.3 Asymmetric Encryption

For asymmetric encryption each packet is encrypted with a key of 32 bytes and this adds to the total overhead. Combining with the different transmission delays and the overhead the total round trip time will increase when compared with raw data. The mean delay for 1000 ms is increased again by an order 2 μs when compared to raw data packets. The results are observed in Figures 36, 37 and 38 which represents the minimum, maximum and mean delay for the packets transmitted. Therefore, as the number of packets increases the round trip delay will also increase when combined with the average mean delay for each packet.

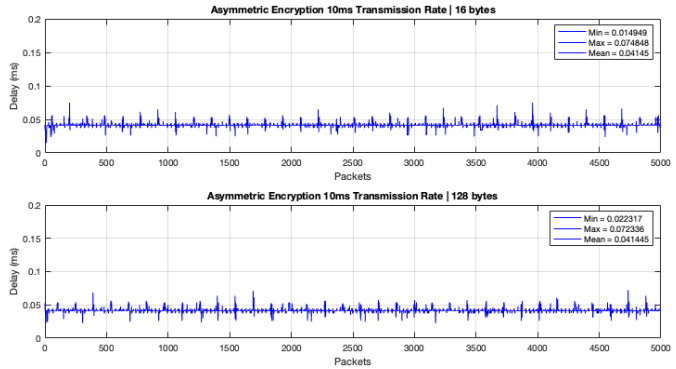

Figure 39: Delay for Asymmetric Encryption over Bluetooth 5.0. Transmission Interval 10ms.

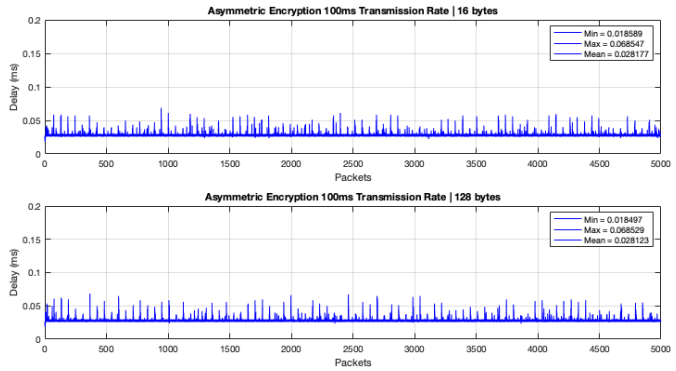

Figure 40: Delay for Asymmetric Encryption over Bluetooth 5.0. Transmission Interval 100ms.

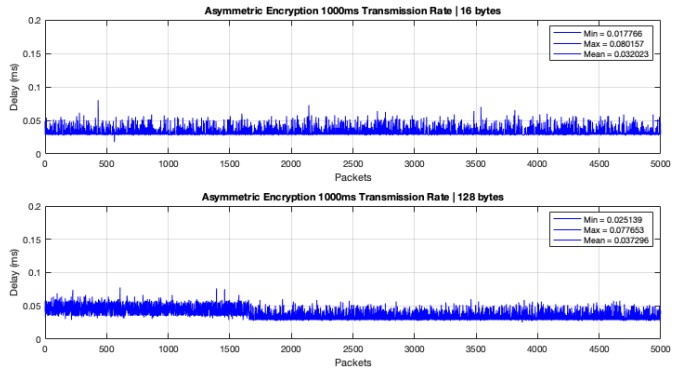

Figure 41: Delay for Asymmetric Encryption over Bluetooth 5.0. Transmission Interval 1000ms.

By analyzing the results obtained with different transmission intervals for transmitting the packets using Bluetooth and also by observing Figures 33 to 41, it shows that for both payload sizes and same transmission interval, the minimum and maximum delays are almost identical while differing in magnitudes of microseconds. Raw data has less delay as there is no overhead, thus the delay is much lower. Since asymmetric encryption uses a public and private key to encrypt the data, overhead for each packets is greater than that of symmetric encryption. The delay for both the encryption algorithm is not that significant as both the encryption is done by using a RSA 2048 key and this no major significance on the delay for the packets that are transmitted. The round trip delays for different transmission times using the packets with symmetric encryption is shown in Table 4.

Table 4: Round Trip Delays for Bluetooth Communication with Asymmetric Encryption.

		Round Trip Delay (ms)								
		Min			**Max**			**Mean**		
		Transmission Delay (ms)								
Packet Type	**Packet Size (bytes)**	**10**	**100**	**1000**	**10**	**100**	**1000**	**10**	**100**	**1000**
Asymmetric	16	0.01494	0.01858	0.01776	0.07484	0.06854	0.08015	0.04145	0.02812	0.03202
	128	0.02231	0.01849	0.02513	0.07233	0.06852	0.07765	0.04144	0.02812	0.03729

8.1.4 Overhead

By extracting the payload size for each method, symmetric and asymmetric, the differences in packet size when applying these methods is found. This could be considered as overhead. As visualized in Figure 42 the difference in packet size could be seen. The y-axis presents the size of the payload for a given size presented in x-axis.

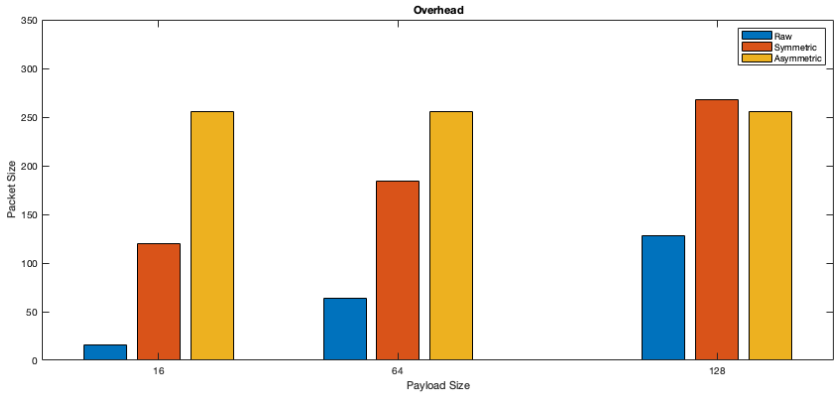

Figure 42: Overhead for the Bluetooth Communication.

In the presented packet overhead sizes could be seen in Table 5, it is clearly indicated that the asymmetrical method remains the same size independent of the payload size. This is clearly because of the RSA key size.

Table 5: Overhead increase for different packet types.

Payload (bytes)	Type	Packet size (bytes)	Overhead (bytes)	Overhead (n times)
16	**Raw**	16	0	0
	Symmetric	120	104	7.5
	Asymmetric	256	240	16
64	**Raw**	64	0	0
	Symmetric	184	120	2.9
	Asymmetric	256	192	4
128	**Raw**	128	0	0
	Symmetric	268	140	2.1
	Asymmetric	256	256	2

8.2 IPSec and TLS 1.3

Results for the experiment using IPSec are shown in Figures 43, 44 and 45 for the transmission delays of 10 ms, 100 ms and 1000 ms respectively. The experiments are carried out for 5000 decrypted packets which are sent over the Internet to the central server and then to the front-end machine using IPSec. The packets are encrypted with AES 256 bit key length and are transmitted with IPSec tunnel mode using site-to-site VPN. The purpose of this experiment is to calculate the total round trip delay that each packet of size 16 and 128 bytes takes. This delay is the difference between the time taken by the packet to send an acknowledgement signal to the transmitter to send in the next packet and the time the previous packet had taken to reach the receiver at the front-end. This delay will provide the total overhead and cost that is required for the system to send the packets using IPSec.

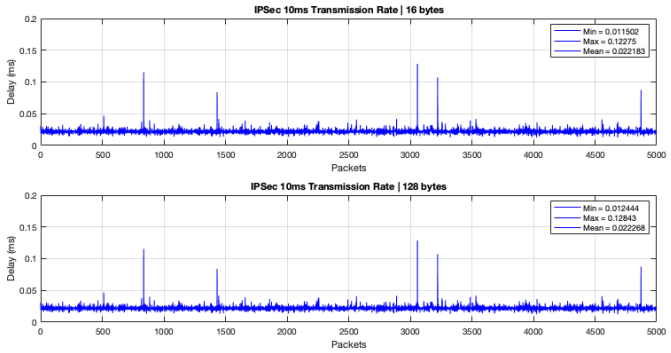

Figure 43: Delay for IPSec with 10ms Transmission Interval.

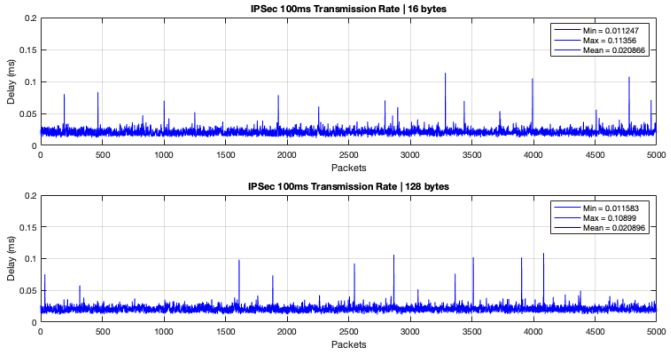

Figure 44: Delay for IPSec with 100ms Transmission Interval.

By analyzing the results obtained with different transmission intervals for transmitting the packets using IPSec and also by observing Figures 43, 44 and 45, it shows that for both payload sizes and same transmission interval, the minimum and maximum delays are almost identical while differing in magnitudes of microseconds. This is true for all the transmission intervals with different payload sizes. The mean delay is also differing in microseconds for all the scenarios. One main difference that can be observed from this results is that for payload size of 16 bytes, the number of peaks obtained are greater than that for 128 bytes. This is because in IPSec, AES 256 bit key length is used to encrypt each packet and using such a larger key length to encrypt a packet size of 16 bytes creates a lot of headroom. Thus creating delays at many round trips. This can be optimized by pre-determining the packet size and choosing the appropriate key length for encrypting the packets.

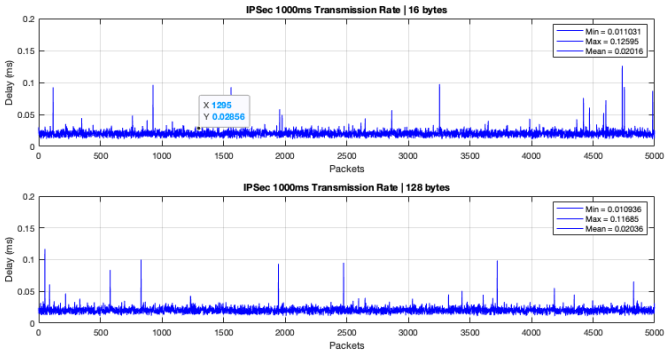

Figure 45: Delay for IPSec with 1000ms Transmission Interval.

Results for the experiment using TLS 1.3 with MQTT are shown in Figures 46, 47 and 48 for the transmission delays of 10 ms, 100 ms and 1000 ms respectively. The experiments are carried out for 5000 packets decrypted which are sent over the Internet to the central server and then to the front-end machine using TLS 1.3 and MQTT broker. The packets are encrypted with AES 256 bit key length along with RSA 2048 private key length. The purpose of this experiment is to calculate the total round trip delay that each packet of size 16 and 128 bytes takes. This delay is the difference between the time taken by the packet to send an acknowledgement signal to the transmitter to send in the next packet and the time the previous packet had taken to reach the receiver at the front-end. This delay will provide the total overhead and cost that is required for the system to send the packets using TLS 1.3 with MQTT.

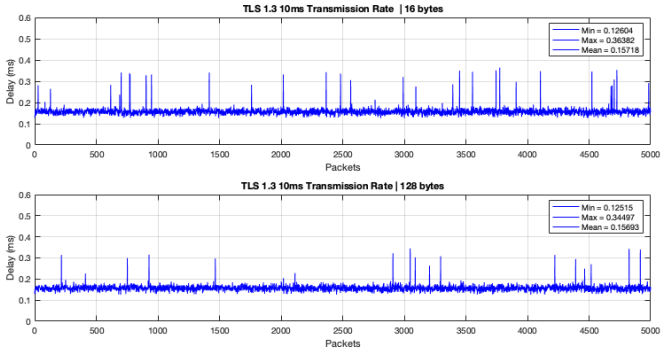

Figure 46: Delay for TLS 1.3 with 10ms Transmission Interval.

By analyzing the results obtained with different transmission intervals for transmitting the packets using IPSec and also by observing Figures 46, 47 and 48, it shows that for both payload sizes and same transmission interval, the minimum and maximum delays are almost identical while differing in magnitudes of microseconds. This is true for all the transmission intervals with different payload sizes. The mean delay is also differing in microseconds for all the scenarios. One main difference that can be observed from this results is that for payload size of 16 bytes, the number of peaks obtained are greater than that for 128 bytes. This is because in IPSec, AES 256 bit key length along with RSA 2048 key length is used to encrypt each packet and using such a larger key length to encrypt a packet size of 16 bytes creates a lot of headroom. Thus creating delays at many round trips. This can be optimized by pre-determining the packet size and choosing the appropriate private and RSA key length for encrypting the packets.

From the results obtained for packets transmitted using IPSec and TLS 1.3 with MQTT, the packets that are transmitted using TLS have a larger delay for all the different payload sizes, which in turn causes them to have lesser throughput. This is because each packet that is encrypted with AES and RSA that tends to increase the overhead for each packet. This result holds true even for

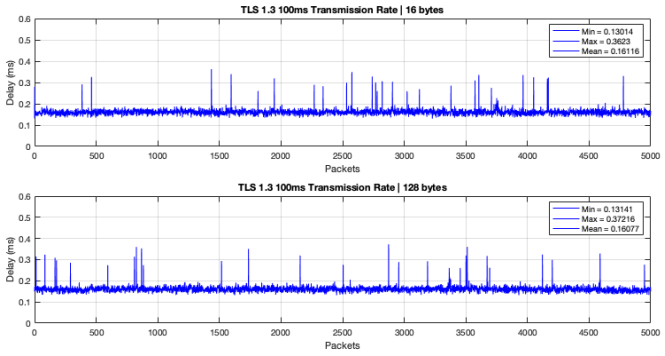

Figure 47: Delay for TLS 1.3 with 100ms Transmission Interval.

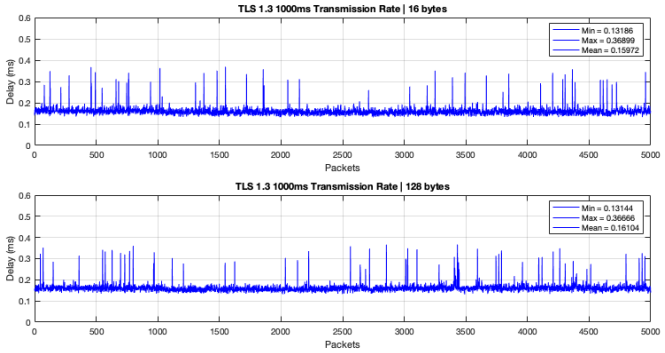

Figure 48: Delay for TLS 1.3 with 1000ms Transmission Interval.

Table 6: Round Trip Delays for Communication with IPSec and TLS.

			Round Trip Delay (ms)								
			Min			**Max**			**Mean**		
			Transmission Delay (ms)								
Connection	**Packet Type**	**Packet Size (bytes)**	**10**	**100**	**1000**	**10**	**100**	**1000**	**10**	**100**	**1000**
IPSec	**Raw**	16	0.01150	0.01124	0.01103	0.12275	0.11356	0.12595	0.02218	0.02028	0.02016
		128	0.02144	0.01158	0.01093	0.12843	0.10899	0.11685	0.02226	0.02089	0.02036
TLS	**Raw**	16	0.12604	0.13041	0.13186	0.36382	0.36235	0.36899	0.15718	0.16116	0.15972
		128	0.12515	0.13141	0.13144	0.34497	0.37216	0.36666	0.15693	0.16077	0.16104

all the scenarios. Therefore, combining the transmission interval delay required for sending each packet from the sender to the receiver at the front-end, the cost of system for transmitting each packet with different payloads can be calculated. The different round trip delays obtained for all the different scenario performed on different communication protocols with different transmission rates and data sizes is shown in Table 7.

Table 7: Round Trip Delays for Different Communication Protocols.

			Round Trip Delay (ms)								
			Min			Max			Mean		
			Transmission Delay (ms)								
Connection	**Packet Type**	**Packet Size (bytes)**	**10**	**100**	**1000**	**10**	**100**	**1000**	**10**	**100**	**1000**
Bluetooth	**Raw**	16	0.00728	0.00446	0.00389	0.05476	0.06106	0.06893	0.02641	0.00974	0.01081
		128	0.00606	0.00592	0.00637	0.05610	0.05729	0.06644	0.02515	0.01071	0.01186
	Symmetric	16	0.00723	0.00725	0.00759	0.05476	0.06478	0.06396	0.02641	0.01177	0.01236
		128	0.00858	0.00975	0.00999	0.06857	0.05727	0.06014	0.02766	0.01306	0.01394
	Asymmetric	16	0.01494	0.01858	0.01776	0.07484	0.06854	0.08015	0.04145	0.02812	0.03202
		128	0.02231	0.01849	0.02513	0.07233	0.06852	0.07765	0.04144	0.02812	0.03729
IPSec	**Raw**	16	0.01150	0.01124	0.01103	0.12275	0.11356	0.12595	0.02218	0.02028	0.02016
		128	0.02144	0.01158	0.01093	0.12843	0.10899	0.11685	0.02226	0.02089	0.02036
TLS	**Raw**	16	0.12604	0.13041	0.13186	0.36382	0.36235	0.36899	0.15718	0.16116	0.15972
		128	0.12515	0.13141	0.13144	0.34497	0.37216	0.36666	0.15693	0.16077	0.16104

Based on the presented research and experiments in this thesis, conclusions can be made in relation to the research questions.

RQ1 : What communication requirements must be fulfilled to achieve regulations for communication of medical devices transmitting health parameters?

In order to fulfill regulation regarding the communication of medical device, an internationally standardized Medical Device Communication profile available mentioned in Section 2.1.2 should be considered. For the implementation part, these requirements were not considered because the companies mentioned that adhering to the standard's requirements is not a high priority and the focus needs to be on finding the right communication profiles for safe and reliable transmission of the private and confidential information over the Internet to the front-end. The research has been done on the requirements and further work needs to be done in the future in the implementation part for the communication profile. Therefore, the consideration and implementation of the regulations for the medical devices is recommended as part of the future work.

RQ2 : Which protocols can transfer the data securely and reliably using a generic middleware for embedded sensor systems?

For this thesis, different communication protocols were considered for secure and reliable transmission of data. Two protocols were shortlisted in the end and research is done on these protocols. We focused on IPSec and TLS v1.3 protocols in this thesis work. After implementing both protocols, it is observed that both the protocols transfer the data securely and reliably while varying in encryption time, transmission time, delay and overhead. Also, these protocols ensure the CIA triad mentioned in 2.3 to fulfill the primary goal of MDR.

RQ3 : What changes should be made so that the requirements can be achieved with a combination of limited subset of the existing protocols?

Certain modifications and optimization of the protocols need to be considered before starting with the data transmission so that the results can be achieved with a combination of limited subset of the existing protocols. For IPSec, the routers between the different geographical sites need to be configured in order to establish a communication between them. Whereas, for implementing TLS, no hardware modifications needs to be done. Therefore, the requirements can be achieved with certain modifications to the existing protocols.

9 Conclusion

The goal for developing a secure communication profile for transmitting medical parameters reliably and securely is realized. Different communication profiles, protocols and algorithms are researched upon and few are used in fulfilling the given requirements. For securely transmitting the data from the home-care to the central hub, it is recommended to encrypt the sampled data packets with asymmetric encryption. Even though, the overhead, delay and transmission time is greater for asymmetric encryption when compared to symmetric encryption, it makes up in delivering improved security than its counterpart as asymmetric encryption uses a public and private key, whereas symmetric encryption uses just the public key to encrypt the samples. In case of an event where the public key is compromised, the confidentiality and security of the transmitted data is maintained because the private key is also needed along with the public key to decrypt the data.

For transmitting data from the hub to the front-head via a central server, two protocols namely IPSec and TLS 1.3 with MQTT is implemented. After analyzing the results, it is recommended to use IPSec with tunnel mode using a site-to-site VPN. The transmission rate delay and overhead for IPSec is lesser than that of TLS with MQTT and both the protocols use the same AES 256 bit encryption while TLS 1.3 with MQTT also uses a 2048 RSA key length. The use of the encryption and RSA key results in the overhead being larger for TLS 1.3 with MQTT. The transmission delay and overhead can be optimized by pre-determining the size of packets that is going to be sent. This is because in TLS 1.3 that uses RSA key, for a given key size there is a maximum file size limit attached to it. For example: a RSA key of length 1024 can send a maximum data size of 62 bytes whereas a key with length 2048 can send a maximum of 214 bytes of packet. In this thesis, experiments for 2048 RSA key length is used. Therefore, it is recommended to the companies to determine a fixed packet size so that the overhead and delay for transmitting is reduced and the system is optimized.

By combining the overhead, throughput and transmission delay for the Bluetooth communication from transmitter to hub and the transfer of this data from the hub to the front-end system through the central server. The total overhead and transmission delay of the system can be calculated. Since the data transfer from the hub to the front-end is done via a simulation software, there will be some variation in the results when the same setup is experimented in a real world scenario.

10 Future Work

If the payload size can be predetermined for the data that is being transmitted, then the option for optimizing the system would be to use RSA key of a shorter length than 2048. The length of the key determine maximum size of the packet. By optimizing a perfect length for the key, it will reduce a higher overhead. This length is suitable for packets with sizes up to 62 bytes. This also greatly decreases the overhead for each packet. The connection between the receiver at the home-care and the front-end receiver is through cabled network, for faster communication between them cellular networks like 4G and 5G can be used for also securing the availability of the data if other methods of communication methods fail. For scenarios containing multiple home-cares in different geographical location, using site-to-site VPN isn't a viable option as the VPN connection has to jump through many places to connect to the home-care. In this case dynamic multi-point VPN can be used to increase the reliability and availability for the different home-cares to transmit data to the front-end receiver.

Due to the recent pandemic situation, the transmission of decrypted packets to the front-end receiver is done by using a simulation software that emulates the routers and switches for different communication protocols. Performing this experiment in a real world scenario will give different results to the one obtained in this thesis in regards to the overhead and round trip delay.

References

[1] S. I. Shaheen, "*E-health in Egypt: challenges and opportunities*," in Proceedings. 2004 International Conference on Information and Communication Technologies: From Theory to Applications, 2004., April 2004, pp. 35–36.

[2] M. Lee, S. Yoon, S. Min, C. Park, H. Shin, H. Kang, H. Chang, and E. Lee, "*Perspectives on e-health industry in Korea*," in Proceedings of 7th International Workshop on Enterprise networking and Computing in Healthcare Industry, 2005. HEALTHCOM 2005., June 2005, pp. 106–112.

[3] L. Androuchko and I. Nakajima, "*Developing countries and e-health services*," in Proceedings. 6th International Workshop on Enterprise Networking and Computing in Healthcare Industry - Healthcom 2004 (IEEE Cat. No.04EX842), June 2004, pp. 211–214.

[4] M. Penhaker, M. Černý and J. Floder, *"Embedded Biotelemetry System for Home Care monitoring"*, in International Journal of Bioelectromagnetism, Vol. 9, No. 1, pp. 35-36, 2007.

[5] Nelwan S.P., Dam T.B., Klootwijk P., Meij S.H, *"Ubiquitous mobile access to real-time patient monitoring data"*, in Comput. Cardiology, vol. 29, sept. 2002.

[6] World Health Organization, *"Ageing and health"*, in World Health Organization, Washington DC, 2018.

[7] M. Cheng, *"Good Management Practice for Medical Equipment"*, in Handbook of Clinical Engineering (Editor: J. Dyro), Elsevier Science, 2004.

[8] M. Kasparick, S. Schlichting, F. Golatowski, D. Timmermann., *"New IEEE 11073 Standards for interoperable networked Point-of-Care Medical Devices"*, in Engineering in Medicine and Biology Society (EMBC) 37th Annual Int. Conference of the IEEE, 2015.

[9] M. Kasparick, S. Schlichting, F. Golatowski, and D, Timmermann, *"New IEEE 11073 Standards for interoperable, networked Point-of-CareMedical Devices"*, in Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference, 2015.

[10] A. L. de Oliveira, R. T. V. Braga, P. C. Masiero, Y. Papadopoulos, I. Habli and T. Kelly., *"Variability management in safety-critical software product line engineering"*, in New Opportunities for Software Reuse-17th International Conference ICSR, pp. 3-22, May 21-23, 2018.

[11] J. Hatcliff, E. Y. Vasserman, T. Carpenter, R. Whillock., *"Challenges of distributed risk management for medical application platforms"*, in E2018 IEEE Symposium on Product Compliance Engineering (ISPCE), pp. 1-14, May 2018.

[12] J. Hatcliff, A. King, I. Lee, A. Fernandez, A. McDonald, and E. Vasserman., *"Rationale and architecture principles for medical application platforms"*, in Proceedings of the 2012 International Conference on Cyberphysical Systems, 2012.

[13] W. Chen, I. J. Was Sell, *"Energy-efficient signal acquisition in wireless sensor networks: a compressive sensing framework"*, in Wireless Sensor Systems IET, vol. 2, no. 1, pp. 1-8, 2012.

[14] A. Koubaa, M. Alves, and E. Tovar, *"IEEE 802.15.4: a Federating Communication Protocol for Time-Sensitive Wireless Sensor Networks"*, in Sensor Networks and Configurations: Fundamentals,Techniques, Platforms 2nd Springer, 2007.

[15] C. M. Li, C. C. Nien, J. L. Liao, Y. C. Tseng, *"Development of wireless sensor module and network for temperature monitoring in cold chain logistics"*, in Wireless Information Technology and Systems (ICWITS) 2012 IEEE International Conference on, pp. 1-4, 2012, November.

[16] Y. Kitaura, Y. Minami, E. Kohno, Y. Kakuda, *"A self-organized approach for the communication method to adapt connectivity of terminals in Bluetooth MANETs"*, in Proc. 17th IEEE Symposium on OCS-oriented Real-time Distributed Computing(ISORC 2014) 5th IEEE Workshop on Self-Organized Real-Time Systems (SORT 2014), pp. 342-347, June 2014.

[17] K. Taketa, Y. Kakuda, *"A method for effective discovery and connection in bluetooth manet consisting of android terminals"*, in Proceedings of 8th International Conference on Broadband Communications and Biomedical Applications (IB2COM 2013), pp. 78-83, December 2013.

[18] Auletta, Vincenzo Blundo, Carlo De Cristofaro, Emiliano Raimato, Guerriero, *"Performance evaluation of web services invocation over Bluetooth"*, 1-8. 10.1145/1163653.1163655.

[19] V. Konstantakos, A. Chatzigeorgiou, S. Nikolaidis, T. Laopoulos, *"Energy consumption estimation in embedded systems"*, in Proceedings of the IEEE International Instrumentation and Measurement Technical Conference IMTC, 2006.

[20] C. Gomez, J. Paradells., *"Wireless Home Automation Networks: A survey of Architectures and Technologies"*, in IEEE Communications Magazine, vol. 48, pp. 92-101, 2010.

[21] R. S. H. Piggin and H. A. Boyes, "Safety and security — a story of interdependence," in *10th IET System Safety and Cyber-Security Conference 2015*, 2015, pp. 1–6.

[22] M. R. Rizqullah, A. R. Anom Besari, I. Kurnianto Wibowo, R. Setiawan, and D. Agata, "*Design and Implementation of Middleware System for IoT Devices based on Raspberry Pi*," in 2018 International Electronics Symposium on Knowledge Creation and Intelligent Computing (IES-KCIC), 2018, pp. 229–234.

[23] "MQTT," MQTT RSS. [Online]. Available: https://www.mqtt.org/. [Accessed: 01-May-2020].

[24] M. Singh, M. Rajan, V. Shivraj, P. Balamuralidhar, *"Secure MQTT for internet of things (IOT)"*, in Proc. 5th Int. Conf. Commun. Syst. Netw. Technol., pp. 746-751, 2015.

[25] K. Govindan and A. P. Azad., *"End-to-end service assurance in IoT MQTT-SN"*, in 2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC), Las Vegas, NV, 2015, pp. 290-296.

[26] N. Tantitharanukul, K. Osathanunkul, K. Hantrakul, P. Pramokchon, P. Khoenkaw, *"MQTT-topics management system for sharing of open data"*, in Proc. Int. Conf. Digit. Arts Media Technol., pp. 62-65, 2017.

[27] R. K. Kodali, S. Soratkal, *"MQTT based home automation system using ESP8266"*, in Proc. IEEE Region 10 Humanitarian Technol. Conf., pp. 1-5, 2016.

[28] Percentage of HTTPs (TLS) Encrypted Traffic on the Internet, Oct. 2019, [online] Available: https://etherealmind.com/percentage-of-https-tls-encrypted-traffic-on-the-internet/.

[29] E. Rescorla, *"The transport layer security (TLS) protocol version 1.3"*, 2018.

[30] Hamida ST, Hamida EB, and Ahmed B., *"A New mHealth Communication Framework for Use in Wearable WBANs and Mobile Technologies"*, in Sensors, 2015.

[31] V. Karagiannis, P. Chatzimisios, F. Vazquez-Gallego, J. Alonso-Zarate, *"A survey on application layer protocols for the Internet of Things"*, in Trans. IoT Cloud Comput., vol. 3, no. 1, pp. 11-17, 2015.

[32] S. Vinoski, *"Advanced message queuing protocol"*, in IEEE Internet Comput., vol. 10, no. 6, pp. 87-89, Nov./Dec. 2006.

[33] J. Jarmoc, D. Unit, *"SSL/TLS interception proxies and transitive trust"*, in Proc. Black Hat Europe, pp. 1-21, 2012.

[34] A. Nash, W. Duane, and C. Joseph, *PKI: Implementing and Managing E-Security.* USA: McGraw-Hill, Inc., 2001.

[35] Zhai Xuefeng, *"The security analysis of SSL and the research and realization of its being hijacked"*, in Sichuan University, 2004.

[36] G. Apostolopoulos, V. Peris, and D. Saha, "*Transport layer security: how much does it really cost?*" in IEEE INFOCOM '99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No.99CH36320), vol. 2, 1999, pp. 717–725 vol.2.

[37] K. Kuldeep, V. V. Singh, and H. Gupta., *"A New Approach for the Security of VPN"*, in Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies (ICTCS '16).

[38] OSI, "Security Audit Framework in Open Systems-Part 7", ISO/IEC CD 10181, 1993, 7.

[39] Zhiyong, et al., *"Research of A VPN secure networking model."*, in Proceedings of 2013 2nd International Conference on Measurement, Information and Control. 2013, 567-569.

[40] S. Mason and G. Andrew, *"Cisco Secure Virtual Private Network"*, in Cisco Press, January 4, 2002.

[41] H. Gupta and V. K. Sharma, *" Multiphase Encryption: A New Concept in Modern Cryptography"*, in International Journal of Computer Theory and Engineering vol. 5, no. 4, 2013, 638-640.

[42] R. Tao, X. Dong-Qing, *"Analysis and Research of Access Control Mechanism Based on SSL Protocol"*, in Microcomputer Information, pp. 2009-09.

[43] W. Stallings, Cryptography and Network Security 5/E., Pearson Education India, 2011.

[44] R. Andriani, S. Wijayanti and F. Wibowo., *"Comparision Of AES 128, 192 And 256 Bit Algorithm For Encryption And Description File"*, in 3rd International Conference on Information Technology, Information System and Electrical Engineering (ICITISEE), pp. 120-124, 2018.

[45] E. Surya, C. Diviya, *"A Survey on Symmetric Key Encryption Algorithms"*, in International Journal of Computer Science Communication Networks, vol. 2, no. 4, pp. 475-477, 2012.

[46] I. Todor, F. Marc and N. Viorel., *"Considerations towards security and privacy in Internet of Things based eHealth applications"*, in 2016 IEEE 14th International Symposium on Intelligent Systems and Informatics (SISY), Subotica, Serbia , 2016.

[47] O. Oladayo O., *"Lightweight Security and Privacy Scheme for Wireless Body Area Network in eHealth System"*, in International Journal of Information Security Science, vol. 6, no. 3, pp. 26-38, 2017.

[48] T. Laukkarinen, K. Kuusinen, T. Mikkonen., *"DevOps in regulated software development: case medical devices"*, in Proceedings of the 39th International Conference on Software Engineering: New Ideas and Emerging Results Track, pp. 15-18, 2017.

[49] P. Balboni and B. Iafelice., *"Mobile cloud for enabling the EU eHealth sector Regulatory issues and opportunities"*, 2011 Technical Symposium at ITU Telecom World (ITU WT)., pp. 51-56, Oct. 2011.

[50] Schall D. et al, *"Web Services on Embedded Devices"*, in International Journal of Web Information Systems (IJWIS), Troubador Publisher, February 2006.

[51] S. Loreto et al., Explicit trusted proxy in HTTP/2.0, Fremont, CA, USA, 2014.

[52] Nazrul M. Ahmad ; Asrul H. Yaacob, *"End to End Ipsec Support across Ipv4/Ipv6 Translation Gateway"*, in Second International Conference on Network Applications, Protocols and Services, Kedah, 2010, pp. 222-227.

[53] I. Kotuliak, P. Rybár and P. Trúchly, *"Performance Comparison of IPsec and TLS Based VPN Technologies"*, in 9th IEEE International Conference on Emerging eLearning Technologies and Applications, October 27-28, 2011.

[54] A. Alshamsi and T. Saito , *"A Technical Comparison of IPSec and SSL"*, 2018.

[55] H. Lee et al., *"maTLS: How to make TLS middlebox-aware?"*, in Proc. NDSS, pp. 1-15, 2019.

[56] J. F. Nunamaker and M. Chen, "*Systems development in information systems research*," in Twenty-Third Annual Hawaii International Conference on System Sciences, vol. 3, Jan 1990, pp. 631–640 vol.3.

[57] A. Nath, S. Ghosh and M. A. Mallik, *"Symmetric key Cryptography using Random Key Generator"*, in Proceedings of International conference on Security and Management, , Las Vegas ,USA, Vol. 2, pp.239-244, 12- 15 July, 2010.

[58] M. Alrammahi, and H. Kaur., *"Development of Advanced Encryption Standard (AES) Cryptography Algorithm for Wi-Fi Security Protocol"*, in Development of Advanced Encryption Standard (AES) for Wi-Fi Security Protocol, 2013.

[59] N. P. Dr. M. Kannan, "*Comparative Study of RSA and Probabilistic Encryption/Decryption Algorithms*," International Journal of Engineering and Computer Science, vol. 6, no. 1, Jan. 2017.

[60] Z. Yakova., *"A New Virtual Private Networks Access Model"*, in MIE 2013: Doctoral Conference in Mathematics, Informatics and Education, Sofia, Bulgaria, Volume: 1, 2016.

[61] J. Nieminen, C. Gomez, M. Isomaki, T. Savolainen, B. Patil, Z. Shelby, M. Xi, and J. Oller, "*Networking solutions for connecting bluetooth low energy enabled machines to the internet of things*," IEEE Network, vol. 28, no. 6, pp. 83–90, 2014.

[62] H. Kim, J. Lee, and J. W. Jang., *"BLEmesh: A wireless mesh network protocol for Bluetooth low energy devices."*, in Proceedings of the 2015 3rd International Conference on Future Internet of Things and Cloud. 558–563.

[63] J. Yin, Z. Yang, H. Cao, T. Liu, Z. Zhou, and C. Wu., *"A Survey on Bluetooth 5.0 and Mesh: New Milestones of IoT"*, in ACM Trans. Sen. Netw. 15, 3, Article 28, August 2019.

[64] A. A. Khan, M. Zafrullah, M. Hussain, and A. Ahmad, "Performance analysis of ospf and hybrid networks," in *2017 International Symposium on Wireless Systems and Networks (ISWSN)*, 2017, pp. 1–4.