UMinho | 2015

Carlos Manuel Dias Ferreira das Dores   Communications with QoS in IoT&Cloud Environment

Universidade do Minho
Escola de Engenharia

Carlos Manuel Dias Ferreira das Dores

Communications with QoS in IoT&Cloud Environment

dezembro de 2015

Universidade do Minho
Escola de Engenharia

Carlos Manuel Dias Ferreira das Dores

Communications with QoS in IoT&Cloud Environment

Dissertação de Mestrado
Ciclo de Estudos Integrados Conducentes ao Grau de
Mestre em Engenharia de Telecomunicações e Informática

Trabalho efectuado sob a orientação do
Professor Doutor Nuno Vasco Lopes
Professor Doutor Luís Paulo Reis

dezembro de 2015

DECLARAÇÃO

Nome: Carlos Manuel Dias Ferreira das Dores

Correio electrónico: dores.carlos@gmail.com

Tlm.: 919901778

Número do Bilhete de Identidade: 13050778

Título da dissertação:

Communications with QoS in IoT&Cloud Environment

Ano de conclusão: 2014/2015

Orientadores:

Professor Doutor Nuno Vasco Lopes

Professor Doutor Luís Paulo Reis

Designação do Mestrado: Ciclo de Estudos Integrados Conducentes ao Grau de Mestre em Engenharia de Telecomunicações e Informática

Escola: Engenharia

Departamento: Sistemas de Informação

Guimarães, 21/12/2015

Assinatura: ____

Acknowledgments

There were many contributions received from various people in order to make possible the development of this thesis. It is only fair to mention those people that were directly and indirectly involved in helping me complete this work.

As so first i would like to thank professors Nuno Vasco Lopes and Luis Paulo Reis for their guidance, support and availability for meetings and discussions while the work was being done. I also would like to thank them for their incentive and help in making a paper publication based on this thesis possible.

I like to thank my colleagues and professors of my Master course that accompany me on this academic journey, providing me with knowledge and new creative ideas. A special thanks to my friends Sergio Silva, Diogo Ferreira, Joana Silva e Hugo Pereira for their help and support and for keeping my motivation high.

Finally i like to thank my Mother, Father and Sister for their patience, understanding, support and big dose of motivation in all stages of my academic journey.

Abstract

With advances in communication technology the future internet presents numerous opportunities to develop new systems designed to make day to day life easier and to enhance and prolong the life of people with disabilities. Internet of Things (IoT) objectives are; enabling a more extensive interconnection of devices; delivering information perception; enabling development of more comprehensive intelligent services. Is this motivation that propels the development of new services that integrate the mobility of cloud systems and the diversity of IoT.

It will enable us to create new and more independent care systems for people with special needs, providing them a certain degree of independence from others, such are caregivers or their families. This can have a psychological and social impact due to the better quality of life that enables. Other motivation is, the versatility and mobility of services it can provide to everyone by making those services available anywhere through the internet.

In this thesis is explored and explained the different kinds of technologies that can be integrated in order to enable the creation of the future Internet platforms based on IoT and Cloud Computing. Relevant technologies for this thematic such as Next Generation Networks (NGNs), Body Sensor Networks (BSNs), cloud computing, IoT and IP multimedia Subsystem (IMS), are explained. Several Quality of Service (QoS) experiments were conducted in IoT&Cloud platform. The obtained results were analyzed and the main findings were used to recommend the main requirements for a platform that underlies in IoT and Cloud technologies. Furthermore, a system architecture fulfilling the requirements pointed out as demanded is proposed and its functionalities are exemplified in a specific scenario.

The proposed system is then implemented and tested in order to ascertain the feasibility of the implementation and the performance of communications. The thesis ends with the conclusion on the findings and future work that still needs to be done.

Resumo

Com os avanços nas tecnologias de comunicação a Internet do Futuro apresenta numerosas oportunidades para desenvolver novos sistemas projetados para fazerem a vida do dia a dia mais fácil e para melhorar e prolongar a vida de pessoas com carências especiais. Os objectivos da Internet of Things(IoT) são; permitir uma interconexão mais extensa dos dispositivos; fornecer perceção da informação; permitir o desenvolvimento de serviços inteligentes mais compreensivos. É esta motivação que impulsiona o desenvolvimento de novos serviços que integram a mobilidade dos sistemas cloud e a diversidade da IoT.

Isto vai nos permitir criar novos e mais sistemas de cuidados independentes para pessoas com carências especiais, permitindo-lhes um certo grau de independência de outros, tais como cuidadores ou as suas famílias. Isto pode ter um impacto psicológico e social devido a qualidade de vida que proporciona. Outra motivação é a versatilidade e mobilidade de serviços que pode fornecer a todos ao tornar esses disponíveis em qualquer lado através da internet.

Nesta tese são explorados e explicados os diferentes tipos de tecnologias que podem ser integradas de forma a permitir a criação de plataformas de internet do futuro baseadas em IoT e Cloud Computing. São explicadas tecnologias relevantes para esta temática tais como Next Generation Networks(NGNs), Body Sensor Networks(BSNs), cloud computing, IoT e IP multimedia Subsystem(IMS). Foram conduzidas várias experiências em uma plataforma IoT&Cloud. Os resultados obtidos foram analisados e as conclusões usadas para recomendar os requerimentos principais de uma plataforma que tem como base as tecnologias IoT e Cloud. Além disso, é proposta uma arquitectura de sistema que preenche os requisitos enunciados e as suas funcionalidades exemplificadas em um cenário especifico.

O sistema proposto é depois implementado e testado de modo a determinar a viabilidade da implementação e o desempenho das comunicações. A tese acaba com a conclusão dos resultados obtidos e trabalho futuro que ainda necessita ser feito.

Contents

List of Figures

Acronyms

3G	3rd Generation.
3GPP	3rd Generation Partnership Project.
4G	4th Generation.
6LoWPAN	IPv6 Low power Wireless Personal Area Networks.
AES	Advanced Encryption Standard.
BSN	Body Sensor Network.
CCM	Counter with CBC MAC.
CoAP	Constrained Applications Protocol.
CoRE	Constrained RESTful Environment.
DSCL	Device Service Capabilities Layer.
DTLS	Datagram Transport Layer Security.
ECG	electrocardiograph.
GSCL	Gateway Service Capabilities Layer.
GSM	Global System Mobile.
IaaS	Infrastructure as a Service.
IETF	Internet Engineering Task Force.
IMS	IP multimedia Subsystem.

IoT Internet of Things.

IP Internet Protocol.

IPHC IP Header Compression.

MQTT Message Queuing Telemetry Transport.

MTU Maximum Transmission Unit.

NGN Next Generation Network.

NHC Next Header Compression.

NSCL Network Service Capabilities Layer.

PaaS Platform as a Service.

PAN Personal Area Network.

QoS Quality of Service.

RTT Round Trip Time.

SaaS Software as a Service.

SSL Secure Sockets Layer.

TCP Transmission Control Protocol.

TLS Transport Layer Security.

UDP User Datagram Protocol.

WSN Wireless Sensor Network.

xDSL x (any type of) Digital Subscriber Line.

Chapter 1

Introduction

With the ever evolving technologies in computing and communication is important to explore new possibilities in integrating different areas of work with one another. This aspect of research enables new types of technologies to be created, developed and deployed.

Such is the case with wireless networks, cloud computing and internet of things technologies. By integrating these different areas of work we may achieve means to deploy systems that can facilitate services which enable or at least improve the quality of life of end users. In this thesis it will be explore the integration of these technologies and possible applications for that integration.

1.1 Focus and motivation

In recent years the themes that have raised most interest, in the scientific community, to be part of future internet are Internet of Things (IoT) and Cloud Computing. However there isn't any finished platform that powers and integrates these technologies.

In the telecommunications world, IMS was define by the 3rd Generation Partnership Project (3GPP) as the envisioned architecture for Next Generation Networks (NGNs), one that will permit the transport of multimedia services with quality of service.

Both IoT and Cloud Computing are future internet tendencies, however, IoT technology is based on diversity and not interoperability and on the other hand

Cloud Computing services are dependent on service providers so service mobility and interoperability between clouds on different providers is also an issue.

Considering IMS as a possible solution in the fulfillment of these requirements, there are still many challenges that need to be overcome so that we can integrate IoT and Cloud services. Although IMS it's one possible solution to the problem, that doesn't mean it's the better or the only one.

1.2 Objectives

The main objective of this thesis is to explore problems and solutions in using IMS as an platform architecture that integrates, IoT and Cloud Computing technologies. This platform should be able to fulfill interoperability and diversity requirements of both technologies.

Securing the communications of an application with the objective of health monitoring will then be implemented. IoT technology will be used on medical sensors for diagnose people (Blood glucose, electrocardiograph (ECG), Oximeter), while cloud computing will be useful on the management of information gathered by the sensors which in turn can later be analyzed by an attending physician. In order to fulfill this main objectives there are specific tasks that need to be done:

- Study relevant existent systems and techniques applied to similar situations and define one possible solution to this type of application.
- define necessary concepts and objectives required to realize real time monitor of people.
- Conduct a series of experiments in different scenarios in a way that enable to, test the developed solution, find limitations and parameters in which that solution best performs.

1.3 Summary and main contributions

This thesis presents a model for real-time monitoring of heart rate in Gymnasium. It also contributes for understanding the requirements and costs, of building a secure platform that interconnects IoT devices with Cloud capabilities.

The platform is constituted by an IoT application that includes, Constrained Applications Protocol (CoAP) with Datagram Transport Layer Security (DTLS) protocol, to secure and authenticate data exchanges over IPv6 Low power Wireless Personal Area Networks (6LoWPAN) stack. IoT devices are arduinos that send their sensing data to a Raspberry Pi (Proxy), which in turn sends data to a cloud server. In turn the cloud server sends data to mobile devices via a Web-app on Message Queuing Telemetry Transport (MQTT) protocol via Secure Sockets Layer (SSL) websocket.

Running security protocols on devices might have costs to their performance, such as memory usage, cpu activity, energy consumption and delay of information exchanges. Only by understanding this costs will we be able to determine if securing the platform is a feasible operation to real-time requirements and even to the limitations of the devices that form the platform.

The work done in this thesis contribute to elaborate a paper publication presented on CISTI 2014[1].

1.4 Thesis structure

In chapter 1 is made a brief introduction to the the integration of IoT and Cloud Computing thematic. On this chapter is also presented the motivation for the work and objectives to be achieved. Ending with the description of the thesis structure.

Chapter 2 describes concepts, protocols and mechanisms related to the development of cloud IoT technologies.

In chapter 3 is presented the state of the art of the technologies involved in making Future Internet a reality. Also some limitations and obstacles are discussed giving an idea of the challenges that need to be overcome. In this chapter is also presented IoT and Cloud Integration Pros and Cons, so it can be understood the benefits and drawbacks of using these technologies.

Chapter 4 presents some experiments on IoT and Cloud Integration to a platform in development for this area of work. Some preliminary tests are made, and results examined. After that some practical scenarios are described for which the platform parameters might be ideal. Ending with conclusions of the results obtained.

In Chapter 5 is presented a model proposal for integrating IoT and Cloud. The model requirements are contextualized followed by the description of the model proposed and its architecture. Ending with the explanation of protocols and mechanisms used.

Chapter 6 present the model implementation describing the details and challenges of the implementation. This chapter also presents results and analysis of the tests done, ending with conclusions of the results obtained.

Chapter 7 is presented the thesis conclusions. First is given a quick overview of all the work done, following with a conclusion of the results obtained. Ending with some future work objectives.

Chapter 2

Related Concepts

As this work has multiple technologies involved, it's important to give a general idea of what they do and how they work. In this chapter will be succinctly viewed and explained some technologies in this area of work, their purpose and limitations.

2.1 Next Generation Networks

Next Generation Networks (NGNs) are a concept that was introduced because of new situations and changes in the field of telecommunications. This concept aims to make network architectures more flexible and to help define and introduce new services with ease [2].

Being all IP packet based networks that transports different types of traffic (voice, video, data), they enable deployment of access independent services over converged fixed and mobile networks. They use IMS at its core which provides access independent platform for different kinds of access technologies such as Wi-Fi, Cable, xDSL, GSM, 3G, 4G and fiber. NGNs are characterized by a common network core, which is independent of physical layers and network access technologies. This enables a great adaptability integrating different or new technologies.

Integration of Body Sensor Networks (BSNs) and social networks through a NGN was analysed in the article of Mari Carmen Domingo, "A context-aware service architecture for the integration of body sensor networks and social networks through IP multimedia Subsystem". An architecture in which the authorized members of a social network can monitor real time data from other user's BSN was proposed [3].

IP multimedia Subsystem

With the huge success of packed-switched technology in the internet, multimedia services had a surge of growth. This resulted in a demand for higher bandwidths. Mobile networks are experiencing a similar phenomenon, in which, user's demand for higher bandwidths is a result of the need to use internet applications and multimedia services.[4]

Defined as a standard by 3GPP, the IP Multimedia Subsystem (IMS) architecture is the core of NGN services. IMS supply telecommunications operators with the ability to deploy an all IP based infrastructure, that enable the launch of new and richer multimedia communication services with ease, blending data services with telecommunication services [2].

It can offer cost reduction, by means of IP-boosted increase of capacity for classical services, and the enrichment of other service types, by using former communication services with new ones resulting in a great synergy of the system. There are many misconceptions on IMS such as, that will be utilized to provide all kinds of service of a provider or that constitutes an independent domain in a telecom infrastructure[5]. The services in the portfolio of a telecom provider are classified as:

- Broadband access to the internet and intranets;
- TV, media and information and entertainment services;
- Interpersonal communication services;

Although design principles are influenced by a provider will, IMS is mostly geared to enhance the interpersonal communication services. It cares for making a communication business sustainable and ready for the future beyond usual services of voice call and sms. It brings a new set of services such as multimedia telephony, instant sharing of multimedia content and live streams, between many other and possibly even new emerging technologies [6].

2.2 Internet of Things

Internet of Things (IoT) consists in interconnected devices ”things” and their addressable virtual representation using standard communication protocols. The heterogeneity of ”things” however, makes interoperability between them a problem that prevents adoption of generic solutions at a global level.[7]

Furthermore the volume, speed and volatile data from IoT, impose great challenges to existent information services. The will to extend the existent internet with objects, interconnected physical devices and their virtual representation has grown in the last years. This is going to enable the creation of a wide brand of services in many different domains, such as smart houses, e-health, transportation, logistics and environmental[8].

This technology is based on integrating cheap devices, equipped with sensors and wireless capabilities, with Internet Protocol (IP) enabling a wide range of possibilities for intelligent applications. Applications such as smart house management, E-Shopping and e-health monitoring of vital signs and intelligent detection of events or conditions that are a threat to patients health[9].

Wireless Sensor Networks

With the ever increasing usage of the internet in our everyday things and activities comes a need for interconnection of sensors in a network fashion. Wireless Sensor Networks (WSNs) to simply put are a large number of small sensing self-powered nodes which detect certain events, gather information and communicate wirelessly with the purpose of delivering their process data to a base station.

These types of networks provide endless opportunities, but at the same time many challenges need to be overcome, such as energy being a scarce a limited resource. Advances in low power VLSI, embedded computing, communication hardware and convergence of computing and communications are making possible the use of these technology [10].

The ever evolving effort in miniaturization and as of late research and development of nanotechnology, are pushing forward the concept of networked tiny distributed sensors and actuators [11].

Also current research in new and better ways of harvesting renewable energy from the environment make possible the long term usage of this technology. The sensor can harvest energy from the sun, from the movement of an individual, or induction in some cases.

Body Sensor Networks

Body Sensor Networks(BSNs) are defined as a set of sensing devices implanted, internally or externally on a person's body, linked in a network fashion. Sensors are linked in a manner that enables the exchange and process of information, so they can act upon recognized events or report them.

In the last decade there was a surge of growing interest in new devices for monitoring and sensing on the health-care area. For example in the treatment of patients with acute diabetes, the blood glucose level can be continuously monitored by a device that controls release of insulin through an implanted reservoir.

For epilepsy and other debilitating neurological disorders, there are already on market, implantable brain stimulator's that can save a patient's from surgical operations of removing brain tissue. For patients with cardio-vascular disorders, there is a growing acknowledgement of an implantable cardio-defibrillator for the prevention of sudden cardiac death [12].

With the advances in monitoring devices there are technologies that play a crucial part in development of medical preventive system, such as:

- Design and bio-sensors integration;
- Miniaturization of power source;
- Ultra low power RF data paths;
- Context awareness and multi-sensory data fusion;
- Automatic sensing, and secure, light-weight protocol;

BSNs can enable the development of new technologies in health-care as well in other areas[13]. However in order for this to happen some conception problems need to be solved, such as size, cost and compatibility. It is also required to demonstrate the advantages of being linked in a network so their value can be recognized in other areas [14].

2.3 Cloud Computing

Cloud computing is a technology that allows to rent infrastructure, run-time environments and services in a pay-per-use basis. This principle has many practical applications and can offer different types of solutions based on user's requirements, such as, scaling an enterprise infrastructure on demand and sizing it according to the business needs or in case of end users having their data available anytime anywhere from any device that has a connection to the internet.

To simply put, it doesn't matter to the user where the information and systems are, who maintains them, or where the applications are hosted. What is important is that the service and or data are always available from any device connected to the internet. This makes cloud computing an extremely appealing concept. Cloud computing turns IT services into utilities such as water and electricity [15].

Web 2.0 technologies, play a important role in making a cloud computing an appealing opportunity for building computing systems, as they have turned the internet into a rich application and service delivery platform [16].

Cloud Computing is an extremely flexible environment for building new systems and application and even integrating additional capacity or new features into existing systems. The most important advantages of cloud computing that made it a popular phenomenon is the ability of use provisioned IT resources dynamically instead of buying additional infrastructure of software [17].

Although cloud computing have evolved much its use is still limited to only one service, or to better say a set of services from a provider. The lack of standardization efforts in the past for this technology made difficult move services from one provider to another [18].

2.4 Communication protocols

Communication protocols are a set of rules that permit two entities or more to exchange information. Next are explained the protocols relevant to IoT and Cloud communication.

User Datagram Protocol

The User Datagram Protocol (UDP) enables datagram mode of packet-switched communication in interconnected computer networks. This protocol assumes that the Internet Protocol is used as the underlying protocol. It provides procedure for applications to send messages to other programs with minimum protocol mechanisms. Transmission, delivery and duplicate protection are not guaranteed [19].

The benefit of UDP are:

It's better for real-time streams, that don't need to ensure reliability of delivery (withstand some packet loss). This might be good for some emergency systems, patients vitals can be monitored by emergency systems on route, and in a near future might even be able to activate a fibrillation charge remotely.

This can be good also for video consultation with physicians, which can check patient vitals via their embedded devices and perform a diagnosis.

The disadvantages of UDP are:

It has no concept of connection, it needs to be coded by the developer. No flow control, it doesn't guarantee packet ordering; packets might be sent 1,2,3,4,5 and arrive 5,2,3,1,4 or received duplicated or even not arrive at all. Data needs to be manually broken into packets. Data cannot be sent too fast that internet connection cant handle. In case of packet lost mechanisms to detect and resend data again is necessary.

Transmission Control Protocol

The Transmission Control Protocol (TCP), is intended to be used as a highly reliable host-to-host protocol in packet-switched communications networks and interconnected systems [20].

The benefit of TCP are:

Better at ensure information delivering when real time is not needed. Since physical information should not have loss of data because that data might be critical to patient assessment, information transference need to be assured. For that reason the TCP was chose. It is connection based, ensure reliability and order in packet transmission, automatically breaks up data into packets, has flow control (make sure it doesn't send data too fast for internet connection to handle).

The disadvantages of TCP are:

Internal TCP code queues up the data it sends, then only when enough data is in the queue sends the packet to the other machine. This can be a problem if data amounts are very small, which is likely since embedded devices send essential information (very small amounts data) in order to be energy efficient. This mean data might be held if no more data need to be sent. There is an option that can be set that fix this behavior called TCP_NODELAY. This makes the protocol no to wait for enough data to be queued and send it immediately as it is written. Another problem is the control flow of packets, since it orders all packets that mean that if a packet fails to arrive it needs to wait for its arrival in order to deliver the rest of the data packets. This might cause a delay in receiving data.

Websockets

WebSockets is a technology that make possible open an interactive communication session between the user's browser and a server. A JavaScript interface, which defines a full-duplex single socket connection over which messages can be sent. It enables sending messages to a server and receive event-driven responses without having to request for a reply.

Also permits to secure communication using SSL, this protects vital information and access paramaters like passwords or other means of control to be secure from data captures.

Message Queuing Telemetry Transport

”Message Queue Telemetry Transport or MQTT for short is a publish/subscribe, extremely simple and lightweight messaging protocol. It is Designed for constrained devices and low-bandwidth, high-latency or unreliable networks. The design principles are to minimize network bandwidth and device resource requirements whilst also attempting to ensure reliability and some degree of assurance of delivery. These principles also turn out to make the protocol ideal of the emerging ”machine-to-machine” (M2M) or ”Internet of Things” world of connected devices, and for mobile applications where bandwidth and battery power are at a premium.”[21]

Constrained Applications Protocol

The Constrained Applications Protocol (CoAP) is a web transfer protocol used in constrained nodes (e.g. low-resource sensors or actuators) and constrained networks (e.g. low power, lossy). Its was designed for machine-to-machine(M2M) applications such as building automation, energy management and the internet of things[22].

The working group Constrained RESTful Environment (CoRE) at IETF, aims to provide a REST architecture that run in constrained networks (e.g. Low-Power wireless Personal Area Networks (LoWPANs)). Applications such as monitor simple sensors(e.g. temperature sensors, light switches, and power meters), control actuators (e.g. light switches, heating control, door and window locks), and to manage devices.

CoAP is designed for use between devices on the same constrained network, between devices and nodes on the internet, and between devices on different constrained networks connected via internet[23]. CoAP protocol has two layers, the top layer handles all mechanisms required to provide web services and the bottom layer handle the unreliability of the message exchange. Figure 2.1 presents CoAP protocol Layering [22].

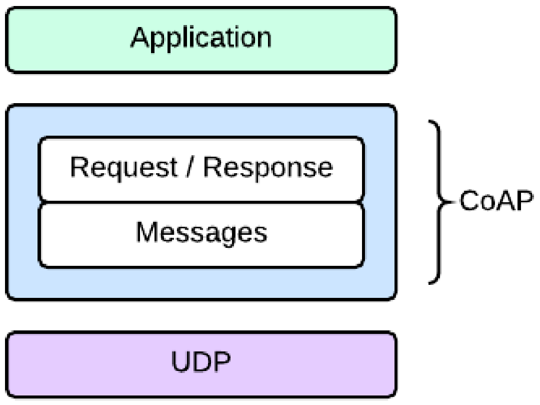


Figure 2.1: CoAP Abstract Layering.

Its application layer easily translates to HTTP making integration with exiting web seamless while meeting specific requirements like multicast support, low overhead, and machine-to-machine applications. It uses two types of messages requests and responses. COAP features [22]:

- Constrained web protocol satisfying M2M requirements.
- UDP binding with optional reliability supporting unicast and multicast requests.
- Asynchronous message exchanges.
- Low header overhead and parsing complexity.
- URI and Content-type support.
- Simple proxy and caching capabilities.
- A stateless HTTP mapping, allowing proxies to be built providing access to CoAP resources via HTTP in a uniform way or for HTTP simple interfaces to be realized alternatively over CoAP.
- Security binding to Datagram Transport Layer Security (DTLS).

Figure 2.2 presents the Constrained RESTful Environment arquitecture.

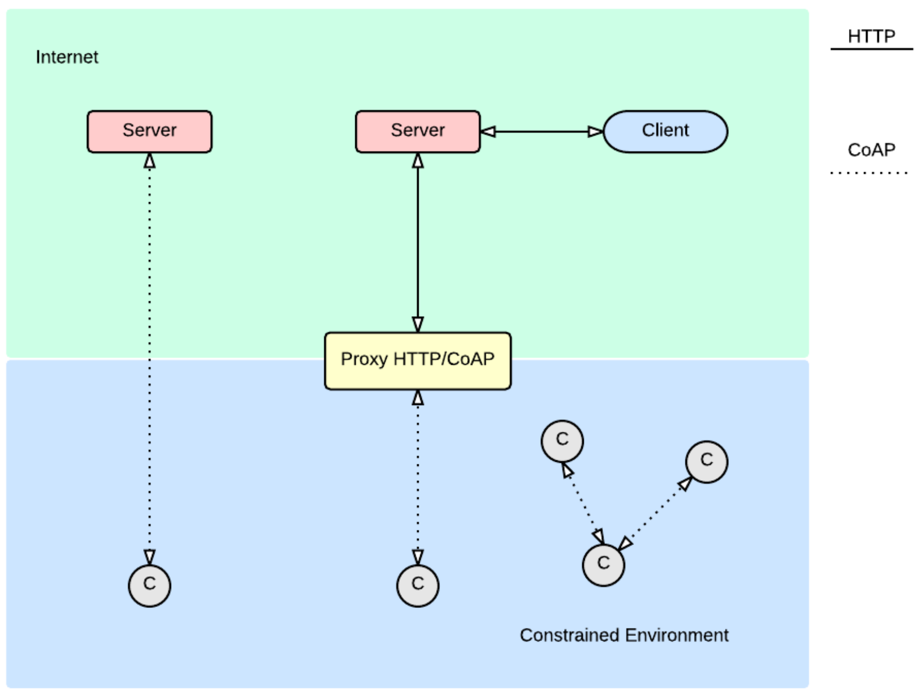


Figure 2.2: CoRE Architecture.

There are already a number of CoAP implementations developed, such as Eclipse Californium(Cf)[24], Copper (Cu)[25], libcoap[26], Erbium (Er)[27] and others.

2.5 Security

Datagram Transport Layer Security

The Datagram Transport Layer Security is a protocol used in securing network traffic specified bt IETF RFC 6347[28]. Unlike Transport Layer Security (TLS), it doesn't require reliable message transfer therefor it can be used in UDP. It is designed to mimic TLS as close as possible. So instead of being presented as a new protocol, it's presented as a series of deltas from TLS 1.2. The main changes have to do with the unreliable message transfer. Changes like handling packet loss, reordering messages, and message size. Figure 2.3 [28] demonstrates the mechanism for handling unreliable transmission.

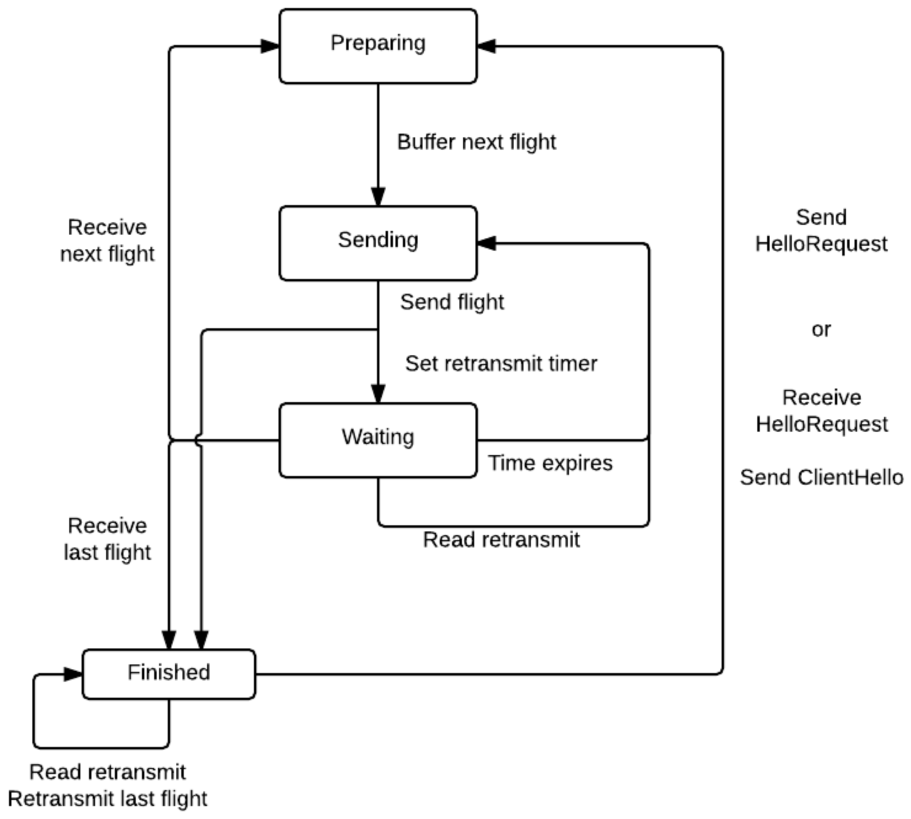


Figure 2.3: DTLS flow chart

DTLS Record Layer provide connection security with two features.

- The connection privacy by using symmetric encryption
- The connection reliability by doing message integrity check.

DTLS Handshake protocol provides connection with:

- Denial-of-Service Countermeasures
- Change Cipher Spec Protocol
- Certificate Verify and Finished Messages
- Message Fragmentation and Reassembly

Adding DTLS to CoAP means adding a layer to the CoAP protocol layer as represented in Figure 2.4 [28].

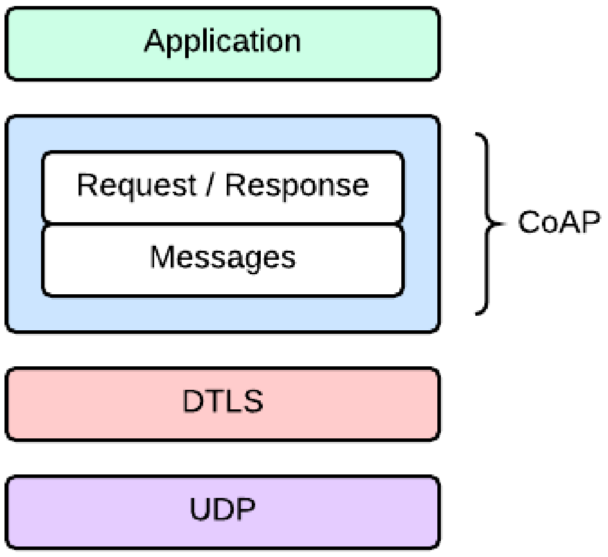


Figure 2.4: CoAP DTLS Abstract Layering

DTLS however is not optimal for resource-constrained networks as it was designed traditional computer networks. The need for large buffers to handle message loss by retransmitting the flight of the handshake protocol or to deal with fragmentation due to small Maximum Transmission Unit (MTU). Also the use of certificates to achieve mutual authentication. Certificates can grow to sizes of Kilobytes. To overcome that issue, the concept of raw public key introduced and declared mandatory in DTLS [29].

Conclusions

This chapter served to give a general understanding of the relevant concepts involved in IoT and cloud technologies. It can be assessed that these technologies provide numerous opportunities in developing new systems for e-health. There are however, some limitations that need to be overcome in order to create a platform that use all this technologies.

Chapter 3

State of the Art

In this chapter will be seen some work related to IoT Cloud technologies and communications. These works present some of the most relevant issues to be aware of when developing IoT and Cloud systems.

3.1 Research on IoT and Cloud Technologies

This section presents some related work about technologies in the area Cloud Computing and Internet of Things.

Internet of Things overview

The authors of *Internet of Things (IoT): A vision, architectural elements, and future directions*[30] advocated that, ubiquitous sensing enabled by Wireless Sensor Network technologies offer the ability to measure, process and understand environmental variables from nature and urban resources. Proliferation of these devices in a network creates the Internet of Things.

Nowadays we are living in a post-pc era, in which smart and hand-held devices are changing our way of living by making the environment more interactive and informative. Inter-networking these devices can potentially enable an endless amount of distributed computing resources and storage own by multiple owners. Miniature devices called nodes interconnected to form a WSN can be used in various applications such as environmental, infrastructure and traffic monitoring, retail, etc.

An efficient, secure, scalable and market oriented computing and storage resourcing is essential for the realization of a complete IoT vision. Moving from www (static web pages) to web2 (social network web) to web3 (ubiquitous computing web) increases the need for data-on-demand. Therefor deploying large scale platform-independent wireless sensor infrastructure that includes data management and processing, actuation and analytics is important. Cloud Computing is aimed to provide high reliability, scalability and autonomy to provide ubiquitous access, dynamic resource discovery required for next-gen IoT applications.

Several application domains will be affected by emerging Internet of Things. Application domains can classified based on the type of data. Personal and Home, Enterprise, Utilities and Transportation as seen in the next figure.

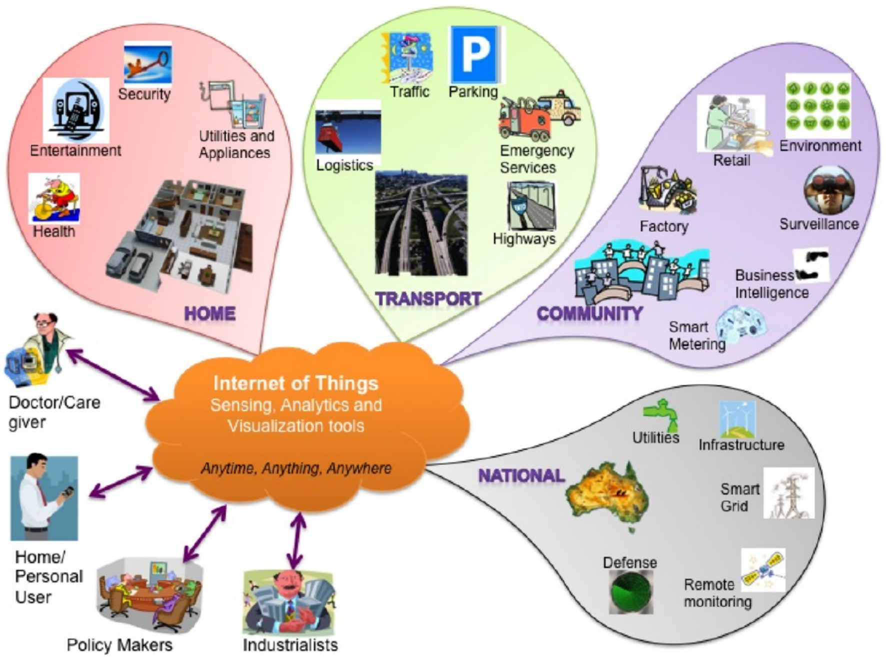


Figure 3.1: IoT domains[30]

"The internet enables sharing of data between different service providers in a seamless manner creating multiple business opportunities."

In Personal and Home domain, collected sensor information is usually only used by owners of the network. E-health has been envisioned for the last two decades and IoT gives the perfect platform to realize that vision using body area sensors and IoT backend to upload data to servers. Personal body area network enables home monitoring systems for elderly care. Control of home equipment's such as lights, kitchen appliances, and other electronic devices.

In Enterprise domain refers "Network of Things" in a work environment as an enterprise based application. Information collected is used only by the owners, data may be released selectively. Environmental monitoring is the more common application, to keep track of the number of occupants and manage utilities inside a building.

Information in the application domain of Utilities is usually for service optimization rather than consumer usage. Video based IoT, integrates image processing, computer vision and networking frameworks. Surveillance has the most widely used camera network application, helping track targets, identify suspicious activity, detect abandoned packages or luggage and monitor unauthorized access. Behavior analysis and event detection. Water network monitoring and quality assurance of drinking water, sensors measuring water parameters are installed at key locations in order to ensure high quality.

In Transportation domain, urban traffic is the main cause of noise pollution and also air quality degradation due to gas emissions. Traffic congestion imposes costs on economic and social activities in most cities due to stalling supply chain efficiency and productivity causing freight delays and delivery schedules failures. Dynamic traffic information can help to better plan and improve scheduling. Transport IoT process traffic information and allow better route choice and planning. Key developments in IoT are shown in the next figure.

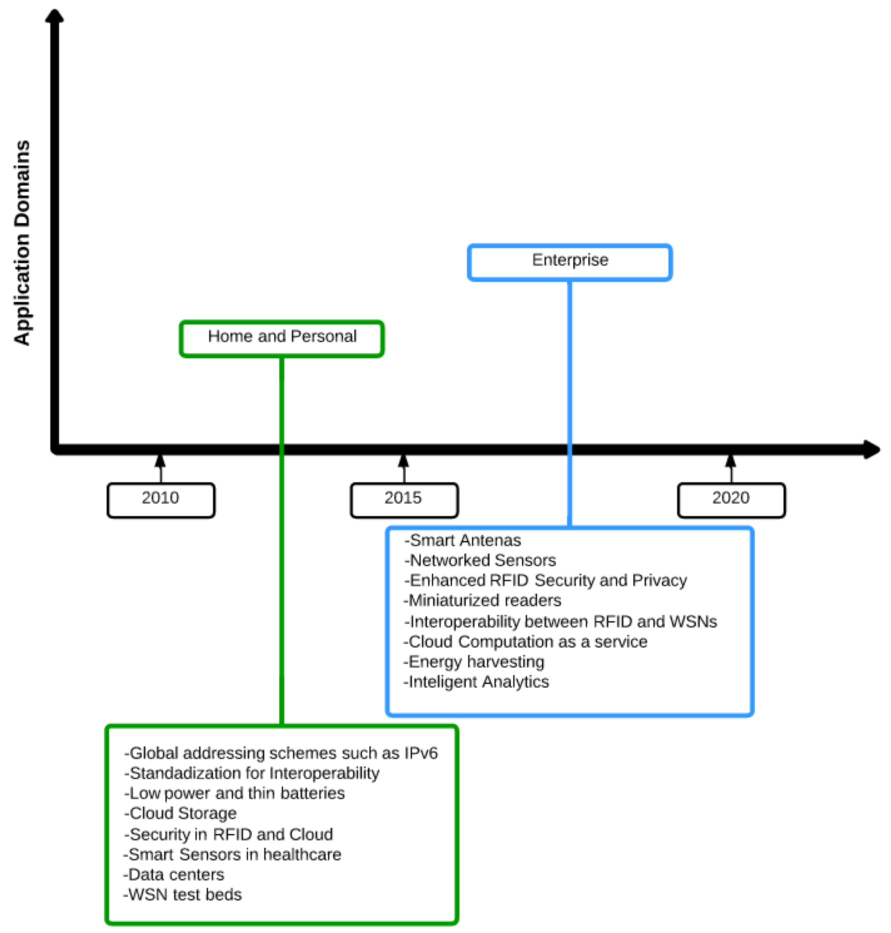

Application Domains
Home and Personal
Enterprise
2010
2015
2020
-Smart Antenas
-Networked Sensors
-Enhanced RFID Security and Privacy
-Miniaturized readers
-Interoperability between RFID and WSNs
-Cloud Computation as a service
-Energy harvesting
-Inteligent Analytics
-Global addressing schemes such as IPv6
-Standadization for Interoperability
-Low power and thin batteries
-Cloud Storage
-Security in RFID and Cloud
-Smart Sensors in healthcare
-Data centers
-WSN test beds

Figure 3.2: IoT future directions

Challenges include IoT specific challenges such as privacy, participatory sensing, data analytics apart from the standard WSN challenges including architecture, energy efficiency, security, protocols and Quality of Service[30]. This paper was useful to comprehend the required components needed in order to realize the envisioned IoT, also some challenges to the process as well some future directions that IoT might take .

Security challenges in Cloud Computing

The authors of *Beyond lightning: A survey on security challenges in cloud computing*[31] presented a general review of cloud computing, and examine some of the related security challenges. Cloud computing can be classified as follows:

- Private Cloud is owned or rented by an organization for private use of the cloud resources. Resources are used to serve the organization business critical applications.
- A Public Cloud is owned by a service provider and resources are sold to end-users. Resources can be partially rented to consumers as their needs for resources scale up or down.
- In Community Cloud resources are shared exclusively by members of a community with similar goals.
- A Hybrid Cloud combines two or more cloud infrastructures; can be private, public or community clouds. Its purpose is to provide extra resources in case of high demand.

Cloud service models are divided in three classes Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS).

- SaaS is the applications running on a cloud infrastructure to provide services to users. One example of this is Google Apps, users don't control the infrastructure that provides the service.
- PaaS is the use of resources and tools running on a cloud infrastructure to provide services to end-users. Examples of this are Windows Azure and Google App Engine. Consumers control the deployment of applications but don't control the infrastructure or operating system.
- IaaS is the use of fundamental computing resources like storage, networks and servers to users. Users can then run 3rd party software like applications or operative systems. Amazon EC2 is one example of this.

The primary benefits of using cloud is fast and easy deployment, the pay-per-use model, and reduction in costs of IT equipment. However security is an important issue to be addressed in order to make Cloud Computing prevalent. Providers need to solve typical security issues of traditional communication systems. Security challenges can be divided in two categories, traditional and new cloud computing security problems.

- Traditional security problems - Authentication and authorization mechanisms for enterprise environments may need to be change in order to use cloud environments. Cloud service reliability is also a concern, if a cloud service is disrupted affects more users than traditional models. Amazon cloud disruption some time ago is a good example of this, the disruption took down a number of websites (Reddit, Foursquare, Quora).

- Cloud Security problems - The most serious security consideration is about confidentiality and privacy of users data.

  - Users use services provided by the cloud provider without knowing for sure where the resources for that service are located, this might pose a problem if they are in other legislative domains (some countries might not have legislation that protects users data).

  - The Multi-tenancy issue, difficult the protection of users data against unauthorized access from other users running processes on the same servers.

  - The Authentication and integrity issue, data may be altered without authorization from owner therefor authenticity of data and integrity is very important, however standards to ensure this do not exist.

  - Cloud Standards are needed across different cloud developing organizations in order to achieve cloud interoperability, stability and security.

Cloud Computing is a technology that helps reduce operating costs while increasing efficiency. Even though Cloud Computing is been around for a few years security is still a big concern issue and needs further research [31].

3.2 Research on IoT and Cloud Communications

This section presents some related work about communication in the area Cloud Computing and Internet of Things.

Security and Authentication in Cloud communications

The lack of good security measures in the cloud is slowing down the widespread of cloud computing. The authors of *Securing Authentication and Trusted Migration of Weblets in the Cloud with Reduced Traffic*[32] presented the security threats faced by weblets during migration between the cloud and mobile devices.

Since Cloud Computing relies on internet for communication, security threats of the Internet are inherited by Cloud Computing. Threats like DDoS attacks which, that make cloud resources unavailable to its users.

Virtual machines are the communication interface for cloud and they vulnerable to DDoS attacks, their failure means the interruption of communication between cloud and mobile devices.

Also threats like man-in-the-middle attacks and packet injection are an issue if proper authentication is not ensured. In Man-in-the-middle attacks it's a third party that inserts itself in the middle of the communication link redirecting traffic from and to both ends of the link.

Increasing the link traffic can result in increases of delay times, which means that Quality of Service (QoS) will be affected also. As Cloud Computing is an open source architecture and cloud resources are open to all users this means they are also open to malicious attackers.

The proposed model for solving authentication and traffic issues is composed of three components that are established on communication channels. Establishing Channel, Building Authentication and Reducing traffic.

Communication channel is established by using ssh (secured shell) protocol tunnel between the mobile device and the cloud. SSH protocol uses public/private key authentication technique to verify end nodes. The advantage of using the SSH tunnel is that the tunnel is encrypted. The SSH Tunneling communication is shown in the next figure.

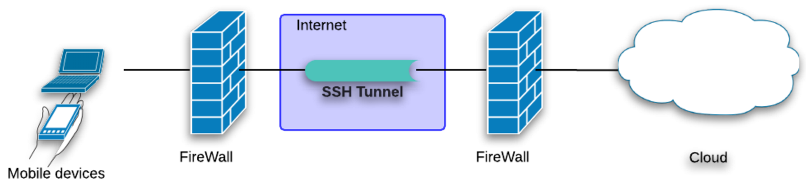


Figure 3.3: SSH tunnel communication Link between the mobile device and the cloud

Cryptography is an effective technique for transmitting secure messages in the internet. To ensure protection of data an extra layer of security was added. Files were subjected to the secure file transfer protocol (SFTP). SFTP is a file transfer protocol that encrypts files for secure transmission.

Being a key feature of SFTP, data integrity ensures that content of communication messages is not modified. So SSH protocol and SFTP protocol working together ensure a secure and authenticated communications between mobile devices and the cloud.

Network Intrusion Detection System (NIDS) is a passive monitoring technique used to monitor the channel. It checks channel traffic and monitors for network attacks. If traffic is too high the channel waits for traffic to get cleared.

In this paper is proposed a secure communication channel between mobile and the cloud. Some techniques for traffic management and security and authentication are also discussed and explained. This gives a better idea of what can be done to improve certain draw backs in using cloud computing technologies [32].

Datagram Transport Layer Security on Android Operating System

The author of *Implementation and Evaluation of Datagram Transport Layer Security (DTLS) for the Android Operating System*[33] presented this thesis that describe design and evaluation of CoAP for android smart-phones. Implementing all the cryptographic cipher suits proposed in the CoAP protocol, and pre-shared key and certificate-based authentications using the Elliptic Curve Cryptography and Advanced Encryption Standard (AES) algorithm in the Counter with CBC MAC (CCM) mode.

The implementation is evaluated on Nexus phone, which takes the handshake time to exchange parameters to secure the connection to about five seconds, and increases the DTLS re-transmission timer from one to three seconds.

Apart from the initial delays, the performance using secure CoAP is comparable to the performance obtained using the same protocol without security. This work also shows that is possible to secure the UDP transport thanks to the DTLS implementations allowing any potential application to secure data exchanges and have mutual authentication.

The Android Software Development Kit(SDK) was used to develop an application. It provide Android APIs and is easily installed on Eclipse IDE. Also Android Developer Tools (ADT) for Eclipse IDE was also used in order to provide a graphical interface for users.

The first objective was to allow CoAP enable smartphone to communicate in a secure fashion. That was achieved with the use of DTLS protocol. The main purpose of the implementation was to provide an API so any CoAP implementation would be able to transmit and receive secure messages.

Firstly the implementation of DTLS protocol was developed, once the DTLS handshake and secure transmissions were possible it was integrated in CoAP so it could send secure CoAP messages. The implementation configured CoAP to communicate in two fashions, unsecure CoAP on UDP port 5683 and secure CoAP on UDP port 4433.

After the implementation some tests were done, the test environment was composed of a smartphone (Google Nexus S) and a laptop. The smartphone communicates with the laptop using CoAP or secure CoAP protocol using wireless(802.11 N). Performance was evaluated using Perf4j [34] that allows to calculate and display performance statistics for java code.

Conclusions made, point to the fact that objectives were accomplished through the implementation of DTLS API for android. The implementation allowed an application to send and receive encrypted data using UDP datagrams as a transport[33].

CoAP over Datagram Transport Layer Security

Since CoAP works on UDP, standard security implementations like TLS are not applicable. For that reason a different security implementation that work on UDP needed to be found. This was the case with Datagram Transport Layer Security (DTLS), this protocol was developed in order to provide security and authentication to UDP. It functions very much like the TLS providing security and authentication only it works on UDP.

The author of *CoAP over DTLS TinyOS Implementation and Performance Analysis*[35] presents an application called Blink ToSCoAP, that is developed by integration of three libraries implementing lightweight versions of DTLS and CoAP protocols and the IPv6/6LoWPAN stack.

In this work is also presented experiments that evaluates the performance of the DTLS security operations. The experiments analyze the BlinkToSCoAP's communications exchanged between two Zolertia Z1 devices. The following figure presents the Zolertia Z1 module.

Figure 3.4: Zolertia Z1 module[35].

This experiments will allow evaluations of memory footprint, energy consumption, latency and packet overhead.

The programming language used to develop the implementations of the protocols mentioned was nesC, a iteration of C programming language created for TinyOS operating system.

The Z1 module is a general purpose development platform for wireless sensor networks that does not require any external hardware to be programmed. The device was composed by following hardware:

- 2nd generation MSP430F2617 low power micro-controller
- 16-bit RISC CPU, 16MHz clock speed, built-in clock factory calibration
- 8KB RAM
- 92KB flash memory
- CC2420 transceiver, IEEE 802.15.4 compliant, at 2.4GHz with a data rate of 250 Kbps
- digital programmable accelerometer (ADXL345)
- digital temperature sensor (TMP102)
- CP2102 USB-to-serial chip from SiLabs
- user and reset buttons
- three RGB LEDs.

The board can be powerd by a battery pack, a coin cell, a usb cable or directly connected to a power source. The system set-up was a composed by a 32-bit Linux Ubuntu 12.04 LTS, withe kernel 3.8.0, virtualized on VMware Fusion software running on a 2012 Macbook air with 1GB of Ram and 2 virtual cores of 1.8 GHz Intel Core i5. TyneOS version installed on the linux distribution was 2.1.2 with the GCC MSP430 compiler version 4.6.3. Some practical issues were encountered while implementing the solution:

- Compiler dependent implementation - failure of the DTLS decryption process

- Data structure Incompatibilities - caused by usage of different libraries to assemble the application

- Ram usage - one of the most relevant issues since in constrained devices resources are very limited. Compilation procedure was failing due to a Ram over flow of 500 bytes. Configuration parameters were reduced in order to solve this issue.

- UDP ports - CoAP library implements request/response mechanism based on different adjacent UDP ports. SiGLoWPAN force the use of a predetermined number of UDP ports.

After results analysis that are somewhat extensive to go through, the conclusion remarks are the following. Despite unoptimized protocol implementation the amount of RAM available on Zolertia Z1 was sufficient to store all the required data. DTLS implementation however, introduced a considerable amount of energy consumption and delay fore every handshake. Security operations introduced a energy consumption increase factor of about 3.2 and affected the responsiveness of nodes which work for an additional 8 ms per transmission [35].

Secure CoAP for Internet of Things

The authors of *Lithe: Lightweight Secure CoAP for the Internet of Things*[36] explained that to communicate at the application layer, resource-constrained devices should employ the constrain application protocol (CoAP) which is currently being standardized by the Internet Engineering Task Force (IETF). In order to secure transmissions with sensitive data, secure COAP makes use of datagram transport layer security (DTLS) as the security protocol for authenticated and confidential communications.

DTLS was designed for powerful devices that are interconnected on reliable, high-bandwidth links. This paper presented Lithe an integration of CoAP and DTLS for the IoT, also it presented a new DTLS header compression scheme that aims to reduce energy consumption.

The heterogeneity in IoT, makes connecting resource-constrained devices in a secure and reliable manner a considerable challenge. Protocols as CoAP, 6LoWPAN, IPv6 are being standardized by IETF to enable IoT.

Running over unreliable UDP protocol CoAP is a web protocol designed for IoT. CoAP is a variant of the most used synchronous web protocol, HTTP, envisioned for constrained devices and machine-to-machine communication.

CoAP provides a REST interface similar to HTTP, however it focus on being more lightweight-and cost effective than its variant for Internet. In order to secure transmissions, datagram TLS (DTLS) has been proposed as the primary security protocol.

This protocol however makes handshake a rather complex process increasing dramatically the number of message exchanges back and forward in an asynchronous manner as seen in the figure bellow.

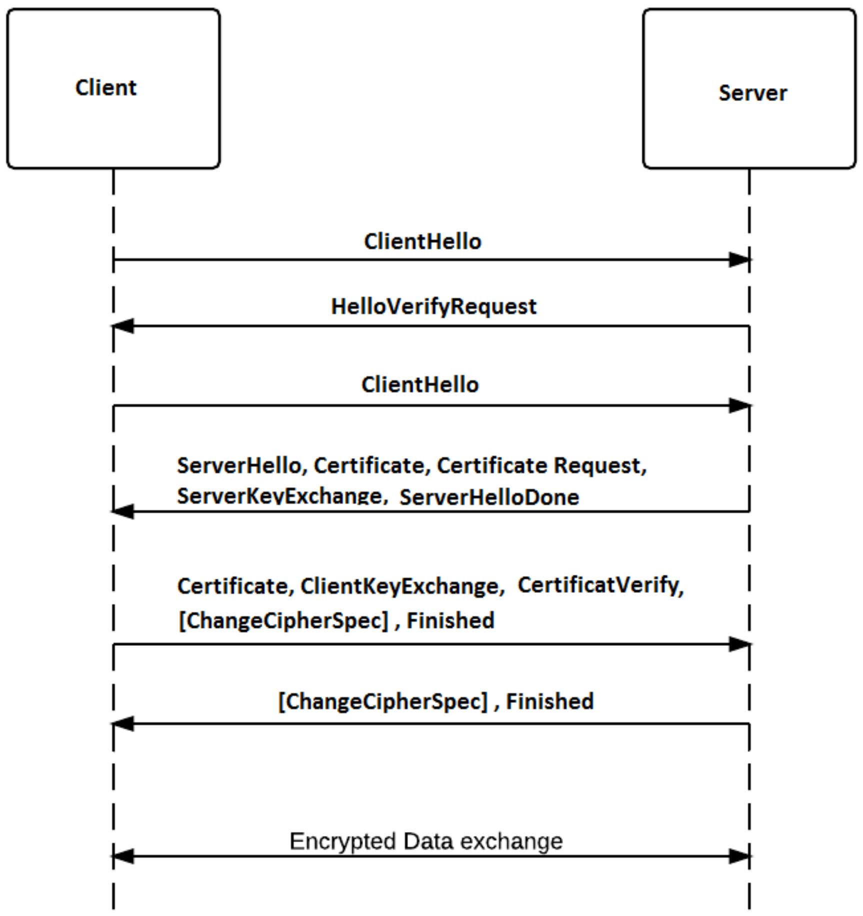

Client
Server
ClientHello
HelloVerifyRequest
ClientHello
ServerHello, Certificate, Certificate Request,
ServerKeyExchange, ServerHelloDone
Certificate, ClientKeyExchange, CertificatVerify,
[ChangeCipherSpec] , Finished
[ChangeCipherSpec] , Finished
Encrypted Data exchange

Figure 3.5: DTLS Handshake process[36]

This paper is focused on header compression only and not on the cryptographic process. 6LoWPAN standard defines header compression and fragmentation mechanisms of IPv6 datagrams within 6loWPAN networks. The compression mechanism described consists of IP Header Compression (IPHC) and Next Header Compression (NHC).

DTLS header compression, like IPHC, is applied within 6LoWPAN networks because DTLS headers are part of UDP payload and information required is already extracted at the IP layer. The following figures shows comparisons between compressed and non-compressed transmissions on energy and round-trip-time costs.

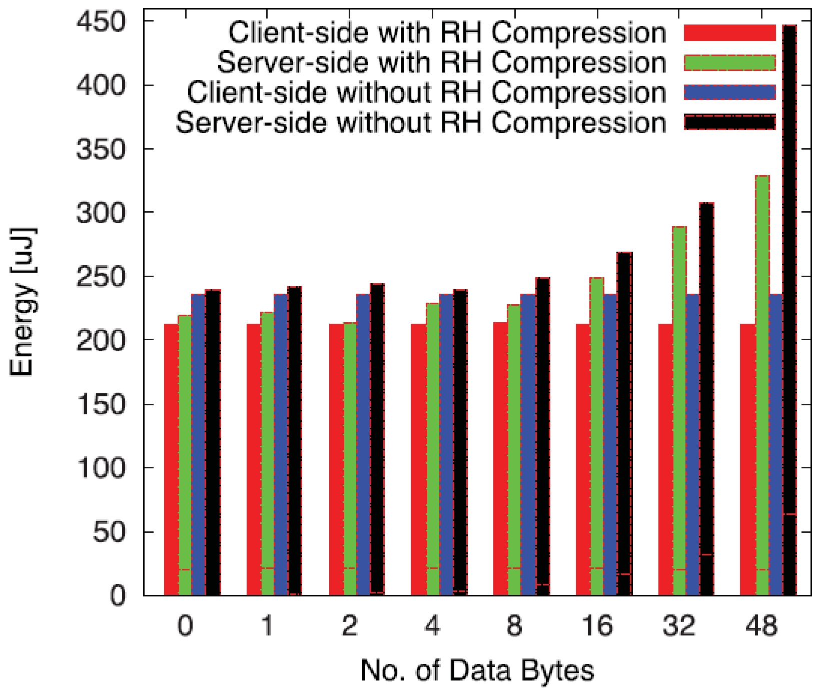


Figure 3.6: Lithe Energy consumption comparison [36]

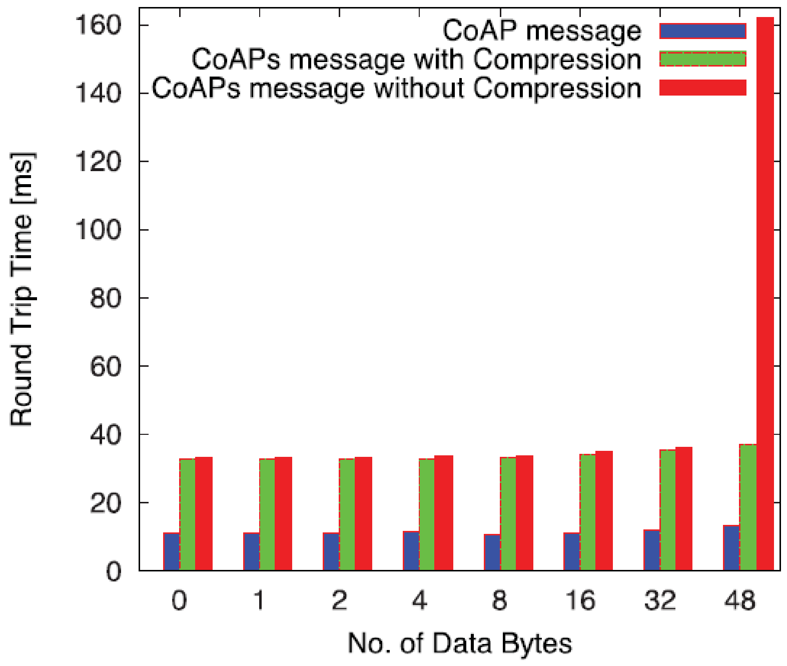


Figure 3.7: Lithe RTT comparison [36]

Lithe is implemented with Contiki, an IoT open source operating system, but can also be implemented with any OS that support 6LoWPAN. Lithe implementation consists of 4 components, DTLS, CoAP, CoAP-DTLS integration and DTLS header compression.

In this paper is shown how constrained-devices are interconnected to form an IoT, problems like the increase in message exchange during handshakes in order to secure connections. Also a solution is presented, in this case the compression of headers significantly increases performance [36].

3.3 IoT and Cloud Integration - Cons and Pros

After having viewed some aspects of the technologies involved on making IoT cloud integration a reality, problems and limitations need to be identified. And after that decisions on how to approach the problems need to be made.

3.3.1 IoT and Cloud Integration Cons

Concerning IMS

IMS is a technology originally designed in order to provision multimedia services on 3G wireless networks[2]. Since it was developed for commercial telecommunications, it's aimed at existent networks infrastructures.

That means the implementation is complex, and difficult on an independent approach because access to commercial telecommunications infrastructure is an issue. Also since it was developed for telephone infrastructures and not internet of things technology, lightweight protocols and energy efficiency are not a design principle.

With that in mind a different method implementation needed to be found.

Security

In order to accurately diagnose a patient physiological data needs to be collected and sent automatically to a data base (or immediately to emergency systems) and later be examined by his attending physician.

That information is highly privacy sensitive data, so strict policies need to be placed in order to protect data transmissions on the network. Most governments already have strict policy's to regulate medical data transmissions on networks, for example the HIIPA law in US. Communication security should be considered by system security, including authentication and encryption.

Security in cloud computing is an issue, most clouds don't insure security of data.

QoS

With current advances in communication technologies and modern day trending, networks experience a surge requests for multimedia data. That translate that flood of requests are common, and because of that resource allocation and differentiated services need to be made in order to assure QoS to services that require it.

Interaction between patients and physicians has become versatile, with richer services like multimedia consultation and physiological data transmission. But as patients are spread around a wide area, bandwidth and Round Trip Time (RTT) delay may differ. Even for same user this might be true depending on the time of day.

Because certain types of services require QoS, such as emergency services or teleconference, some type of QoS will need to be implemented.

3.3.2 IoT and Cloud Integration Pros

Interoperability

In telemedicine and assisted living there is a need for multiple medical devices like blood glucose, ECG, EGG, pulse oximeters. There are multiple companies developing their own systems in e-health, without a standard for building this type of devices mean that they diverse in numerous aspects.

That makes gathering health data from different devices an issue because they may differ in means of communications protocols and data structures. That results in integrating data from different sources and devices, a complex and difficult task.

This is a field in which IoT technologies combined with Cloud can help, since the principle of IoT is the interconnection of different devices in a networked fashion.

Storage

Telemedicine and Assisted living systems are always connected gathering and sending data that is archived for authorized personal to be able to access it at any time. Continuously collecting and storing data from patients and multimedia sources, increases the need for storage resources.

Because of that systems infrastructures need to be suited to supply the resources needed. It is in this area that cloud computing can help, dynamic resource allocation makes it the ideal technology to resolve this issue.

Chapter 4

Experiments on IoT and Cloud Integration

In this section will be tested and evaluated an open source IoT Cloud platform. This will provide knowledge of the requirements this kind of platform should have and how they integrate components of various technologies.

4.1 Scenario Description

Although cloud and IoT technologies integrating platforms are scarce to say the least, after some search a free and open source platform was found. Although still in its developing stages of conception, it already took the first steps for being one viable tool in future internet. Its focus it's centered in machine-to-machine (M2M) instant messaging communication.

Being an open communication network and API for the internet of things, "Skynet is a cloud-based MQTT (MQ Telemetry Transport) powered network that scales to meet any needs, whether the nodes are smart home devices, sensors, cloud resources, drones, Arduinos, Raspberry Pis or other"[37].

"Powered by Node.JS and based entirely on its integrated REST API and real-time websocket API with extensible JSON device and messaging this platform is able to register and networks devices, giving the ability to store, update and exchange information"[37]].

"It also provides a queriable device directory API for registering and discovering nodes on the network and maintains presence for each device making easy to know which ones are online and offline" [37].

Skynet also permits connections from a web browser or mobile device making it in fact ideal in day to day usage. It also allows sending messages to one or all devices and subscriptions to messages being sent to or from devices and their sensor activities.

"Skynet offer a realtime websocket API as well as a Node.JS NPM module to make event-driven IoT development fast and easy" [37].

When nodes or devices register, they are assigned a unique id along with a security token which can be provided or assigned by skynet, this token is required as a safety feature to authenticate a device for making changes that are allowed only to him. "Websocket API commands include: status, register, unregister, update, whoami, device, subscribe, unsubscribe, authenticate and message.

The Node.JS NPM module for making event-driven IoT development, allows specifying messaging protocol as MQTT or Websocket. In case of MQTT protocol is selected, QoS level needs to be specified on the configuration as well in each message and subscription" [37]. As for the architecture, it can be represented as the figure below, the Skynet server on the web, which is connected to a cloud database, in this case a mongoDB (database for clouds).

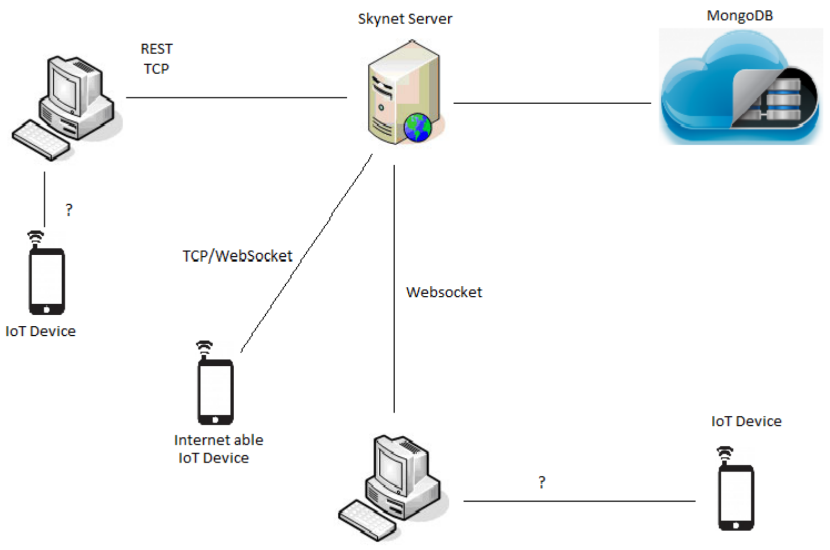


Figure 4.1: Skynet Network Arquitecture

Communications with the Skynet Server are typically done via tcp connection via html requests, such as Get, Put, Post or Delete. The protocol of communications between IoT devices that do not have internet able capabilities and an access point such as a computer, are left to the developer criteria.

4.2 Tests and Results

Skynet truly seams a great tool for developing new systems that use IoT and cloud technology. But before we can say that with certainty, some aspects need to be explored. Aspects such as how well it performs, if it works, security, authentication, reliability, quality of service (QoS).

To answer these questions some tests need to be preformed and after that an evaluation of what it's acceptable based on the results. Tests were conducted on a computer with a 2.5Ghz processor running Ubuntu 32-bits, the only task being executed was the test itself. Because the server didn't allow ICMP messages, pings couldn't be made to measure the delay. By doing this the server is somewhat protected to DDoS attacks, so it can be seen as a measure of security. That also mean another way of measure the delay needed to be found. Using a tcp connection simulating pings, delay times were obtained. The following table of results was calculated based on ten samples made periodically hour to hour in a period of 24 hours from 4PM of one day to the 4PM of the next day.

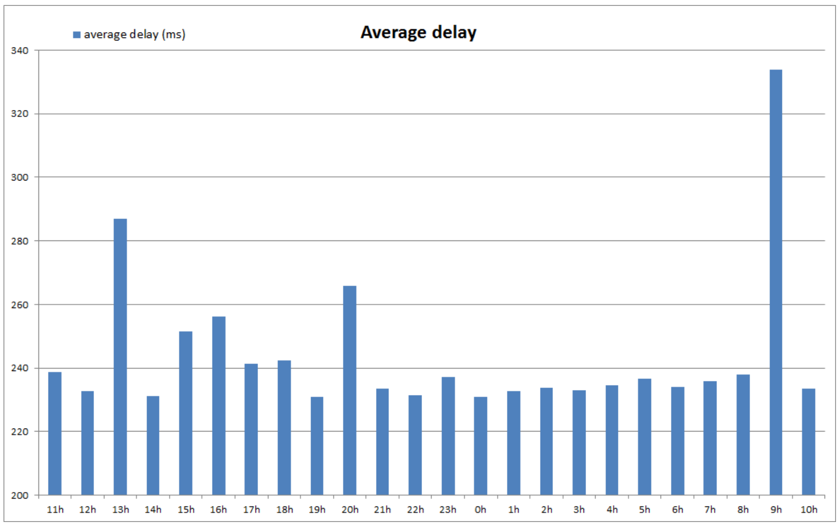


Figure 4.2: Average Delay times

The delay is quite high because the Skynet server is located on the West North American continent, so about 200 ms are of traveling time (RTT).

The Jitter was obtained by calculating the differences in delayed times using the formula:

$$Jitter = \frac{\sum_{N=1}^{N} |T(N-1) - T(N)|}{N-1}$$

where N = Number of samples;

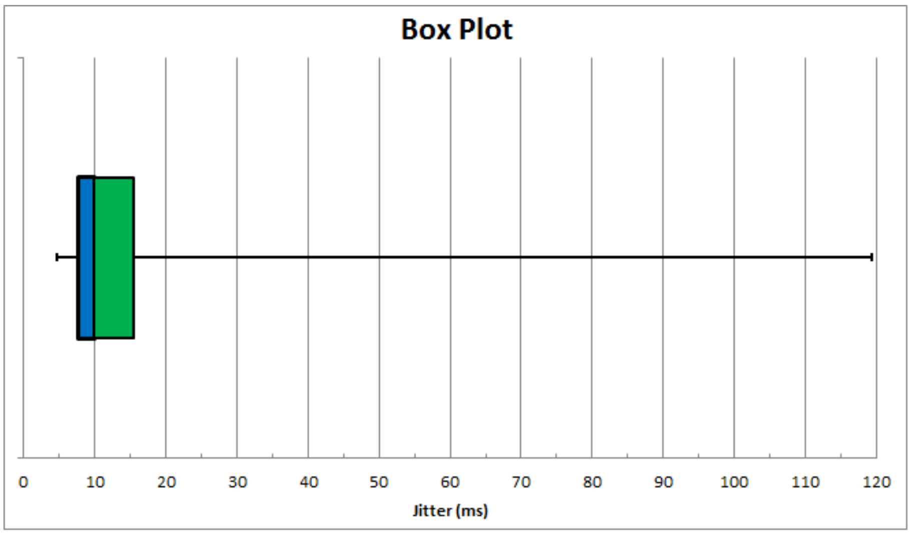
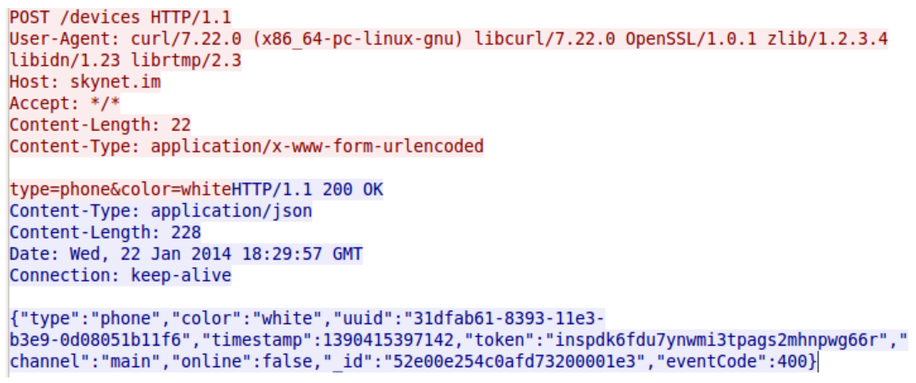


Figure 4.3: Jitter Boxplot

Values registered were never lower than 4,76ms and never higher than 119,37ms. Higher values were result of intense activity on the network. As we can see in the plot above, 25% of values of jitter were lower than 9ms, 50% were between 9ms and 15ms and 25% of values were higher than 15ms. With high congestion of the network jitter can be a severe problem, resulting in poor QoS potentially resulting in inoperability of a real-time system.

Moving to security testes, the security that Skynet provide is, a secret token associated to a unique uuid. The purpose of this token is to ensure that only the device or the authorized personal is able to change critical information associated to that same device. This information might be essential to the running operations in devices, and as such need to be protected via an authentication process such as the secret token. Whenever the changes to critical information need to be done, the token must be provided.

As this is an open communication system the information is not ciphered and senders and receivers are not authenticated via secured connections. This mean users are susceptible to man in the middle attacks and information privacy is not ensured, as seen in the following figure.



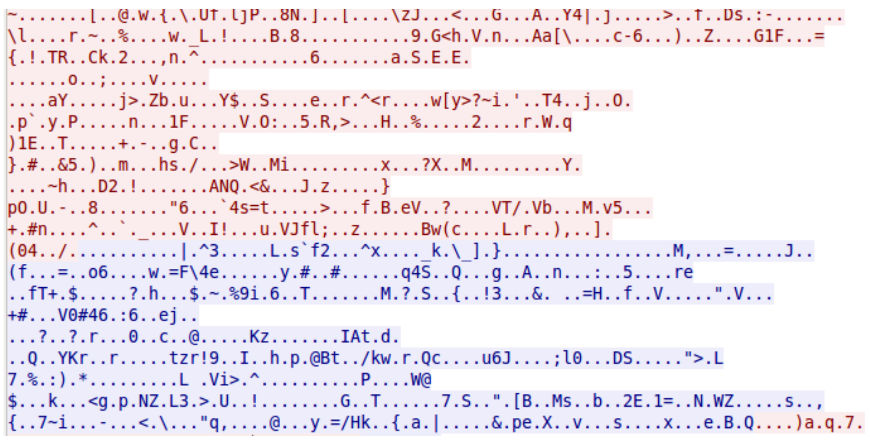
```
POST /devices HTTP/1.1
User-Agent: curl/7.22.0 (x86_64-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4
libidn/1.23 librtmp/2.3
Host: skynet.im
Accept: */*
Content-Length: 22
Content-Type: application/x-www-form-urlencoded

type=phone&color=whiteHTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 228
Date: Wed, 22 Jan 2014 18:29:57 GMT
Connection: keep-alive

{"type":"phone","color":"white","uuid":"31dfab61-8393-11e3-
b3e9-0d08051b11f6","timestamp":1390415397142,"token":"inspdk6fdu7ynwmi3tpags2mhnpwg66r","
channel":"main","online":false,"_id":"52e00e254c0afd73200001e3","eventCode":400}
```

Figure 4.4: Data capture

As we can see the information is completely visible, even the token that is secret information that only the owner should know can be seen. As far as it was discovered this was the only security measure in place, and in some scenarios that might be enough. On the other hand this level of security, is not enough in scenarios that unauthorized access to the devices can have catastrophic results or/and loss of human lives.

One of such cases is for ex. the use of emergency button on the elderly. In case the secret token was intercepted, other entities can in theory use it to send a emergency signal that is authenticated as a user in distress. This failure in privacy can trigger a DDoS attack flooding the system with emergency requests, leaving emergency services without means to respond to all requests and unable to tell which ones are real and which ones are not.

This mean communications like this need to be protected by authenticating and ciphering means such as certificates and private and public keys. If this is done, the information exchange should be secure as seen in the figure bellow.



```
~.......[..@.w.{.\.Uf.ljP..8N.]..[....\zJ...<...G...A..Y4|.j.....>..f..Ds.:-.......
\l....r.~..%....w._L.!....B.8...........9.G<h.V.n...Aa[\....c-6...)..Z....G1F...=
{.!.TR..Ck.2...,n.^...........6.......a.S.E.E.
......o..;....v.....
....aY.....j>.Zb.u...Y$..S....e..r.^<r....w[y>?~i.'..T4..j..O.
.p`.y.P.....n...1F.....V.O:..5.R,>...H..%.....2....r.W.q
)1E..T.....+.-..g.C..
}.#..&5.)..m...hs./...>W..Mi.........x...?X..M.........Y.
....~h...D2.!.......ANQ.<&...J.z.....}
pO.U.-..8......."6...`4s=t.....>...f.B.eV..?....VT/.Vb...M.v5...
+.#n....^..`._...V..I!...u.VJfl;..z......Bw(c....L.r..),..].
(04../...........|.^3.....L.s`f2...^x...._k.\_].}.................M,...=.....J..
(f...=..o6....w.=F\4e......y.#..#......q4S..Q...g..A..n...:..5....re
..fT+.$.....?.h...$.~.%9i.6..T.......M.?.S..{..!3...&. ..=H..f..V.....".V...
+#...V0#46.:6..ej..
...?..?.r...0..c..@.....Kz.......IAt.d.
..Q..YKr..r.....tzr!9..I..h.p.@Bt../kw.r.Qc....u6J....;l0...DS.....">.L
7.%.:).*.........L .Vi>.^..........P....W@
$...k...<g.p.NZ.L3.>.U..!........G..T......7.S..".[B..Ms..b..2E.1=..N.WZ.....s..,
{..7~i...-...<.\..."q,....@...y.=/Hk..{.a.|.....&.pe.X..v...s....x...e.B.Q....)a.q.7.
```

Figure 4.5: Secure data capture

This is an example of a secure communication using an ssl web-socket, as we can see information is ciphered and as such only the right receiver should be able to read the data. Also using this method, receiver and sender are authenticated to one another.

This however signifies that more processing power and more energy is going to be required, and that might be a problem due to the limited processing power and energy reserves of an IoT device. That mean that more research needs to be done before we can say for sure which way is best for secure IoT communications.

4.3 Practical Implications

As the Skynet platform is an open communications system, information privacy is not a main concern. This mean that it can only be used in scenarios that don't require that feature, such as a device that information can be seen and altered by everybody, like an air-conditioner thermostat.

In scenarios that require that feature, this is not a system recommended, because it can result in loss of human life. For example, if the levels of insulin administered by a device to a patient that suffers from diabetes, were to be altered by other than the authorized personal, this could result in death of the patient.

There are many scenarios that security is a required feature, and nowadays with the ever growing threat of cyber attacks and information espionage, a good level of security is recommended to be used in all systems. As for QoS the platform wouldn't be able to provide a service with real time requirements to user further away, since the server is localized in west United States, the round trip time is to long so the responsiveness of devices is affected.

4.4 Conclusions

With these tests we can say that IoT and cloud platforms will be a certainty in the near future, however more research and tests need to be done. Especially on the privacy and authentication process's that secure communications between IoT devices and their platform. Also performance tests to measure the impact of using these security technologies need to be done, because energy consumption is an issue. On the aspect of QoS, service also needs to be improved. Since the distance to users that use the service might be long, delay could be a problem.

The ITU G.114 specification recommends less than 150 millisecond (ms) one-way end-to-end delay for high-quality real-time traffic such as voice, so we need to fulfill that requirement in order to provide a service with QoS. This could be achieved by installing servers closer to users, reducing round-trip time. The servers need to be synchronized, and although the delay remains between them, the end-users will be better served.

Chapter 5

Model Proposal for Integrating IoT and Cloud

5.1 Contextualize the Model Requirements

In the development of this thesis a scenario was thought, a platform that could integrate all the technologies. The scenario is the following:

In a gymnasium user's do activities in order to improve their health or to simply stay fit. This implies that a workout program is fulfilled and actual physical effort is put into the exercises. To measure this, the gymnasium has user's using a arm band that measures physical effort through integrated sensors on it. Sensors like accelerometers and heart rate sensors are put into an arm band measuring heart rate and motion of the body.

This information is transmitted in real time to the cloud and stored allowing the creation of statistics such as, how well an exercise was done, how effective it was to the user health, creation of a record of exercises done, calories spent in each exercise, time spent in each exercise and the progression of user's health like strength, resistance and energy. Users can then share these statistics on social networks at their will.

Also real time information of users is available to the personal trainer of the gymnasium so he can accurately assess the physical effort being put into the exercises. This allows to better decide and/or adjust the workout plan to the progress of user's health.

Sensors in the arm bands can detect emergency events as well, such as heart attack and arrhythmia. The personal trainer is immediately notified and means of response activated. This permits a fast response time of assistance to users that are suffering life threatening events, which often are time critical.

5.2 Model Description

Having as basis the requirements of the scenario and ETSI M2M recommendation for IoT networks, a system architecture was proposed. The following figure represents the components that constitute the system architecture and the connections between them.

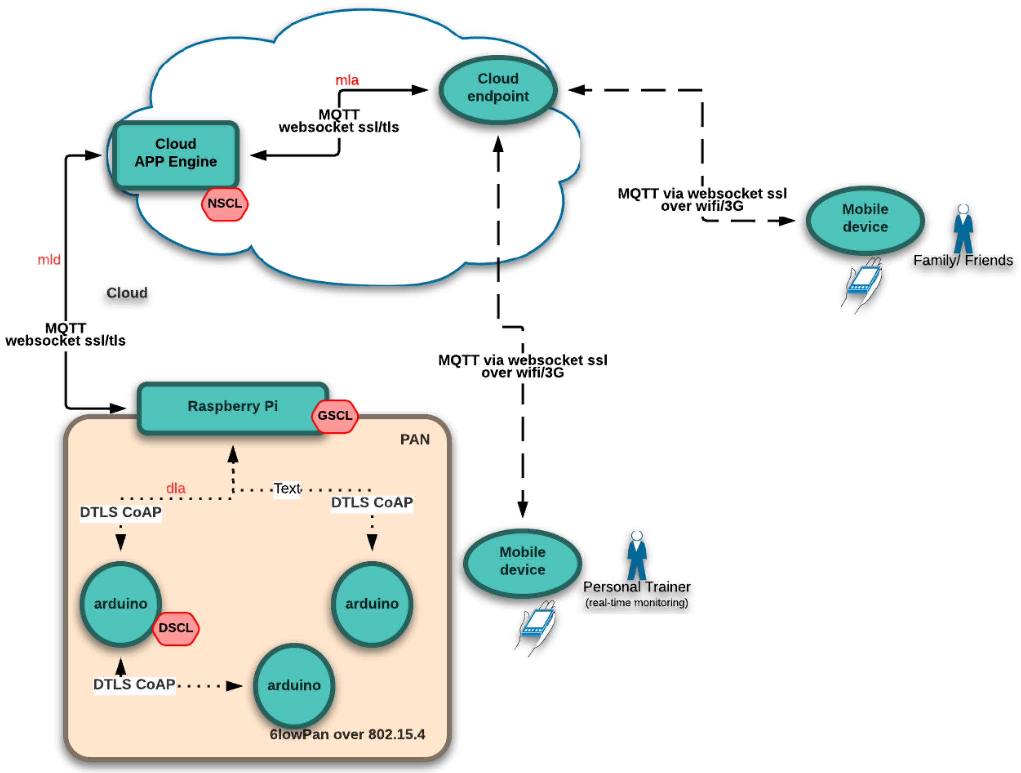


Figure 5.1: System Architecture

The previous image describes the architecture of the system which is composed by two areas. The Cloud area where the Application that manages all system is running. And the Personal Area Network (PAN), which houses the nodes(arduinos) to make readings on user's vitals and the gateway(Raspeberry Pi) for the nodes that manages information gathered. Nodes or devices are fitted with a Device Service Capabilities Layer (DSCL) that announces capabilities to the Gateway through the dIa reference point. Gateway Service Capabilities Layer (GSCL) is located on the Gateway and announces capabilities to the cloud through the mId reference point. The Network Service Capabilities Layer (NSCL) is located on the cloud, it announces capabilities to cloud endpoints and ultimately mobile devices of users via the mIa reference point.

The proposed system is constituted by two areas of network:

- The PAN which houses the sensor nodes, and the Gateway MQTT/CoAP Broker.
- The Cloud area in which the server, service layer, MQTT Broker and cloud web App are located.

Located out of these areas there is the Mobile App for Personal trainers and the Mobile App for users.

Personal Area Network - This area is mainly composed by sensor nodes that are sensing biological parameters of patients/users. These parameters may be heart rate, blood glucose, blood pressure and/or other types of parameters relevant in the assessment of physical condition of users.

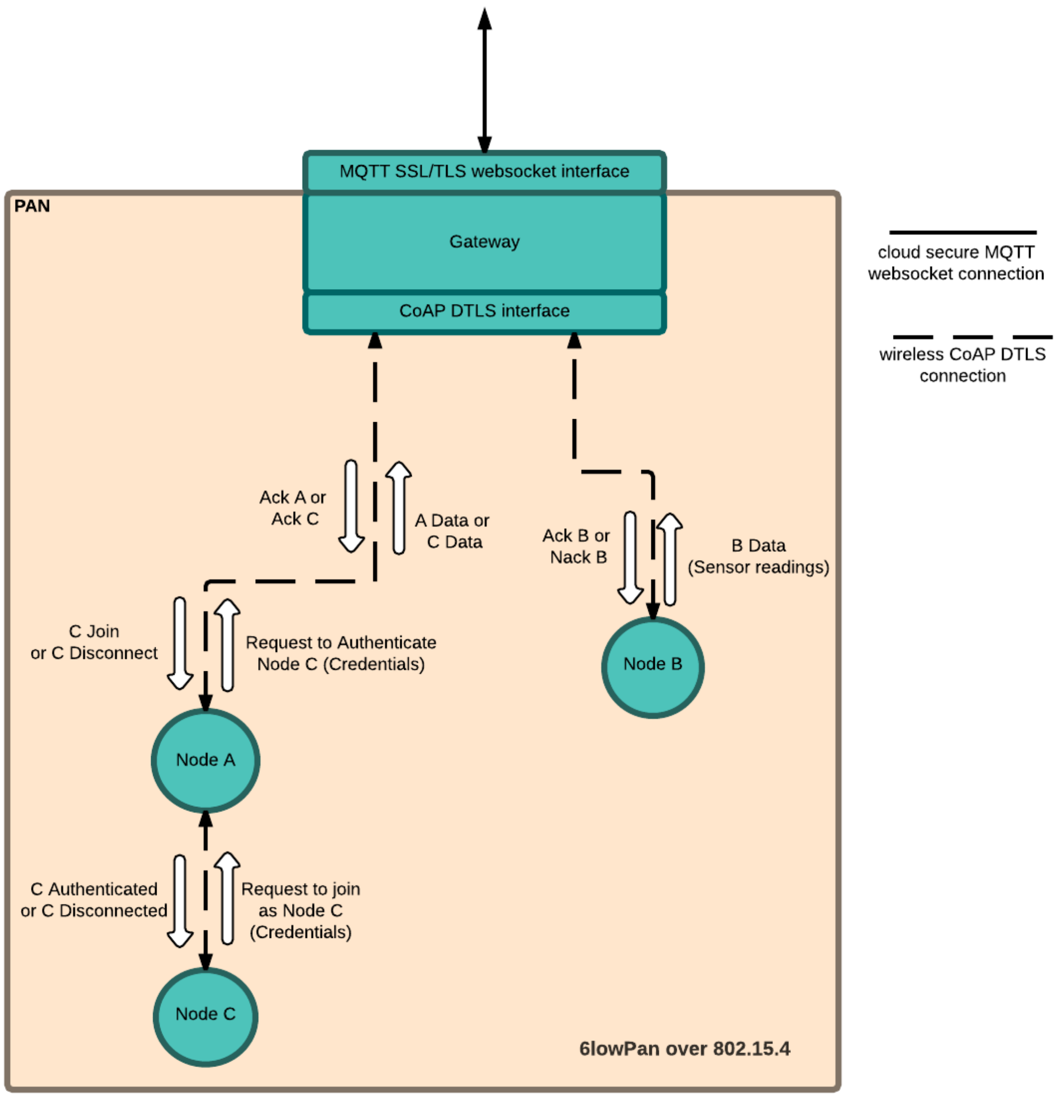
After the parameters are read, they will be sent to the Gateway Broker either directly or through another node. Communications between nodes and broker, are done using CoAP DTLS Protocol, which provides authentication and secrecy to data. DTLS provides the same assurances as TLS in TCP but for UDP transmissions.

As for CoAP, it is a web transfer protocol that unlike http is design for the needs of constrained devices. Client and Servers communicate through connectionless datagrams. CoAP allows UDP broadcast and multicast to be used. It follows the client/server model in which clients make requests to servers and servers send back responses. Clients may GET, PUT, POST and DELETE resources. Doing this nodes are able to announce events, or changes in patient's conditions on the fly.

On the level of QoS the application can provide messages requests and responses as "confirmable" or "nonconfirmable". Confirmable messages must be acknowledge by the receiver with and ack packet and nonconfirmable messages follow the ”fire and forget” approach.

The Gateway MQTT/CoAP Broker serves as an adaptation layer for communications between the Cloud server and the nodes in the PAN, converting MQTT messages into CoAP messages and vice-versa. The gateway is also responsible for establishing secure connections between CoAP devices and the cloud.

Personal trainers that follow user condition and workout plan are connected via a Mobile APP on a wifi/3G connection. They receive information on MQTT protocol, from the Cloud server through an ssl websocket connection. In case of an emergency event such as heart attack, or other kind of symptom that indicates distress of users, the personal trainer receives an alert on his device in order to render assistance immediately. This event can also alert family members of the patient that suffered an emergency event.



MQTT SSL/TLS websocket interface
PAN
Gateway
CoAP DTLS interface
cloud secure MQTT websocket connection
wireless CoAP DTLS connection
Ack A or Ack C
A Data or C Data
Ack B or Nack B
B Data (Sensor readings)
C Join or C Disconnect
Request to Authenticate Node C (Credentials)
Node B
Node A
C Authenticated or C Disconnected
Request to join as Node C (Credentials)
Node C
6lowPan over 802.15.4

Figure 5.2: Personal Area Network

Since Nodes are constrained devices, resource consumption such as memory and energy is a main concern. This means that devices protocol implementations and data that it has to process should be light. For that reason nodes only store three keys, their private key, their public key and the public key of the gateway.

Nodes cannot authenticate other nodes, they can only act as data relay devices, authentication is made in the gateway and only this device can validate a Node to be accepted in the network. If one Node tries to connect through another Node and authentication fails, the gateway sends a message to the relay Node to drop the connection to the first Node trying to be accepted in the network. If the Node is authenticated the gateway sends a message that the Node is valid and to keep relaying its data.

This saves energy since the power requirements for transmitting data over short ranges is lower. On the other hand it might require more energy from relay nodes, to process and transmitting data, and might induce errors and delays in transmissions. The last part can compromise quality of service to the system, so testing is needed in order to accurately decide the best strategy to be applied. With that in mind it was decided to adopt a strategy of directly connect Nodes to the gateway without Nodes acting as relays, until testing can be done.

To secure data transmissions communications need to be protected through the DTLS protocol. In order to do that the handshake procedure needs to be done at the beginning of communications. The next figure shows how the procedure takes place.

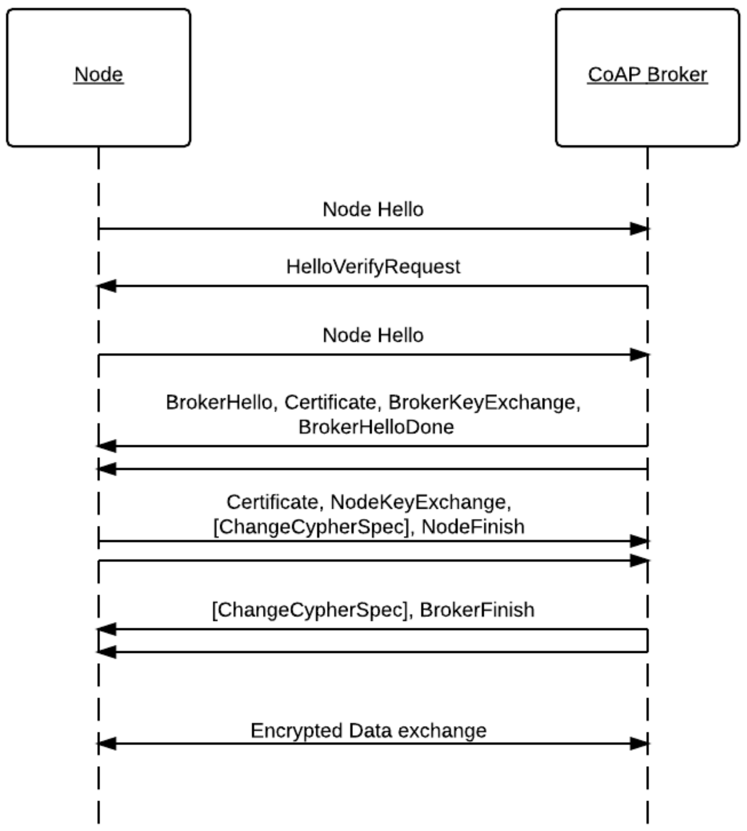

Node
CoAP Broker
Node Hello
HelloVerifyRequest
Node Hello
BrokerHello, Certificate, BrokerKeyExchange, BrokerHelloDone
Certificate, NodeKeyExchange, [ChangeCypherSpec], NodeFinish
[ChangeCypherSpec], BrokerFinish
Encrypted Data exchange

Figure 5.3: Node - Broker Handshake

Cloud - This area contains the Server in which all information will be stored and redistributed. It possesses and MQTT Broker to interpret messages from the cloud APP and the information from nodes sent by the Gateway in the PAN. The Cloud APP is the application that will serve end users, treating connection requests, session management and relay data. Communications to and from this area are made using a MQTT websocket SSL/TLS protocol to provide authentication and security of data.

As for quality of service, the connections are evaluated to access if QoS requirements can be met. Requirements such as delay, jitter and bandwidth need be in compliance with the standards for providing QoS in real-time systems.

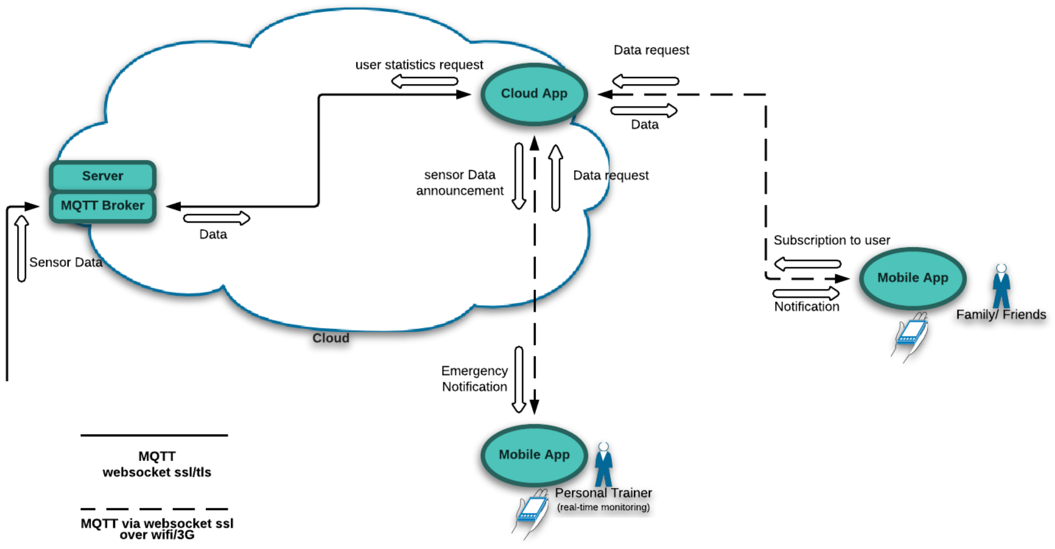


Figure 5.4: Cloud

Besides the establishing the secure connection to the cloud, the PAN gateway sends sensor data from users that are doing exercise with the arm band. This data is then interpreted in the cloud broker and processed by the server.

The Cloud App is responsible managing Mobile App connections and data requests to the server. Mobile Apps can subscribe to a user, which will enable notification of user's activities. There are two types of user in mobile App, the normal user accounts and Personal trainer accounts the latter gives access to physical data of users that are being monitored in real time.

Chapter 6

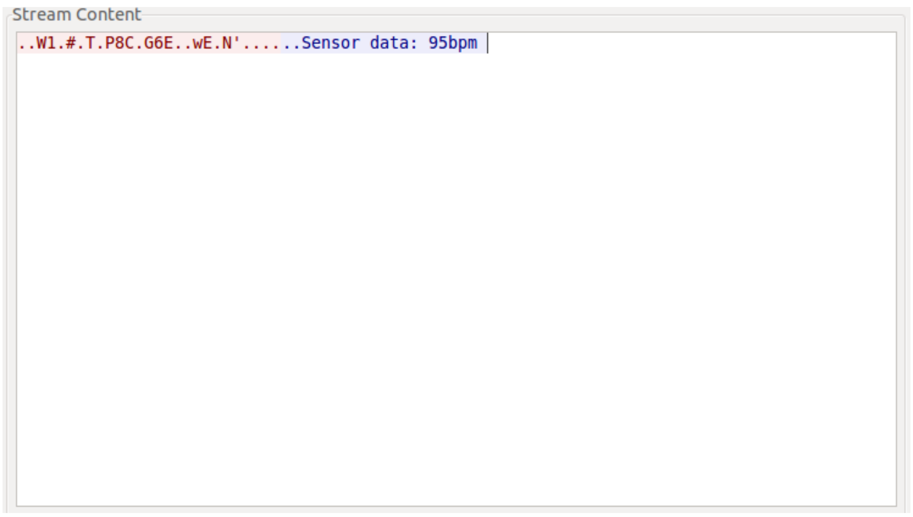
Model Implementation and Tests

This chapter explains implementation and some decisions made as well implementations problems faced. After that, some tests and results are presented followed by a critical review.

6.1 Implementations Details

Since this work is composed of many different technologies and requires hardware and extensive time to develop a full working platform, the implementation will focus on the communication mechanisms of the platform.

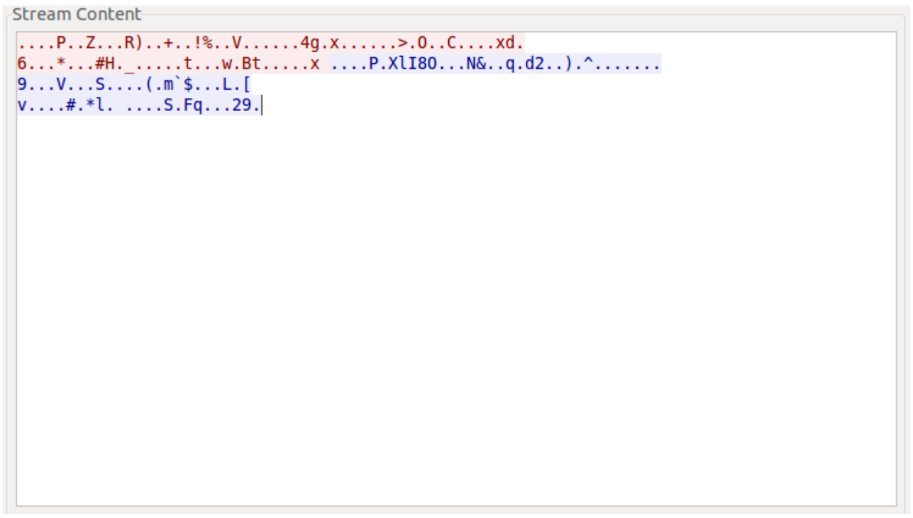
One issue of the platform communication is the connection to the cloud web services that house the web server for information exchange. To secure communication to and from the cloud was decided to use ssl web-sockets, this will cipher data exchanges and at the same time authenticate both points of the communication. The following figures show same message transmitted in websockets and SSLwebsockets captured on wireshark.



```
..W1.#.T.P8C.G6E..wE.N'......Sensor data: 95bpm
```

Figure 6.1: Open websocket

In this figure it can be seen a message captured on wireshark, since the transmission as no security protocols its content is visible.



```
....P..Z...R)..+..!%..V......4g.x......>.O..C....xd.
6...*...#H._.....t...w.Bt.....x ....P.XlI8O...N&..q.d2..).^.......
9...V...S....(.m`$...L.[
v....#.*l. ....S.Fq...29.
```

Figure 6.2: Secure websocket

In this figure it can be seen a message captured on wireshark, since the transmission as security protocols its content is not visible(understandable).

While planning implementation of the platform, there where some decisions to consider:

- How to implement CoAP.
- How to implement security and authentication.

- What system to use.

Deciding on how to implement CoAP, whether to implement from scratch or to use existing libraries, it was decided to make use of existing libraries. This decision was made having in mind time constrains and problem complexity. As mentioned in chapter 2, there are already some implementations of CoAP. One of those is Eclipse Californium(Cf), this project provides libraries for easy to use implementation of CoAP platforms. This project also has plug-tests and example sub-projects which makes it ideal for testing.

While planning implementation of security of CoAP we were faced with some decisions:

- How to implement SCoAP (Secure CoAP)

- what is the most feasible approach.

Since CoAP works on UDP (user datagram protocol) standard TLS (Transport layer security) cant be used in order to secure communications. In order to secure UDP, DTLS needs to be used. DTLS is protocol specified by IETF in RFC 6347 [28].

The CoAP specification defines two ways of providing security and authentication DTLS or IPsec. IPsec Encapsulating Security Payload (ESP) is a protocol used in securing IP communications by encrypting and authenticating each IP packet [28]. It operates at the network layer of the internet protocol meaning it is transparent to the application so remains indifferent to a CoAP implementation. But there is also a downside to IPsec, it provides confidentiality and authentication at a cost of about 10 bytes per packet [22].

Eclipse Californium(Cf) also provides a DTLS implementation sub-project named Scandium(Sc). The sub-project provides security for Californium implementing DTLS 1.2 to secure the application through ECC with pre-shared keys, certificates, or raw public keys. Eclipse Californium uses endpoint connectors in order to establish information exchange between peers. The endpoint connectors need therefor to be modified to add the security and authentication protocol. This was done using the Scandium sub-project libraries. The following figure shows how endpoints are constructed in Eclipse Californium.

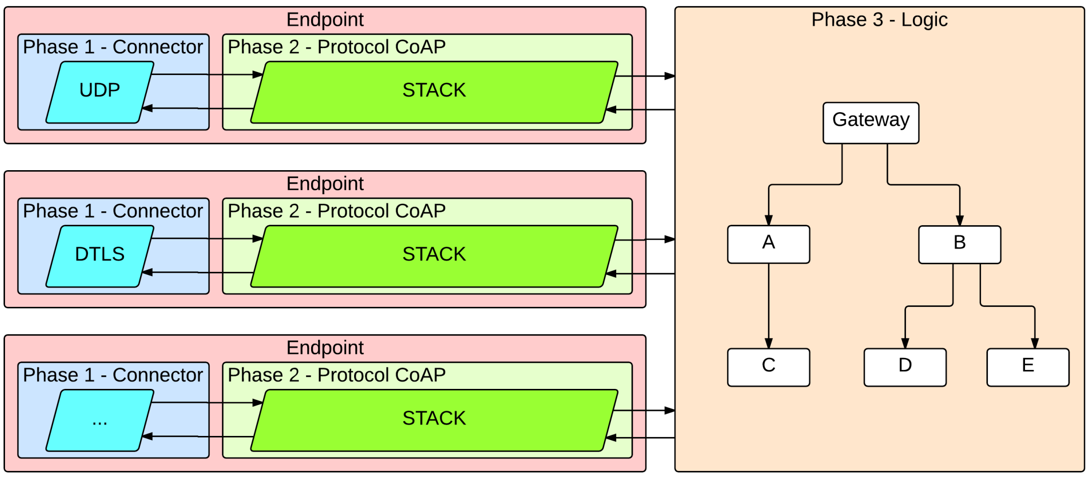
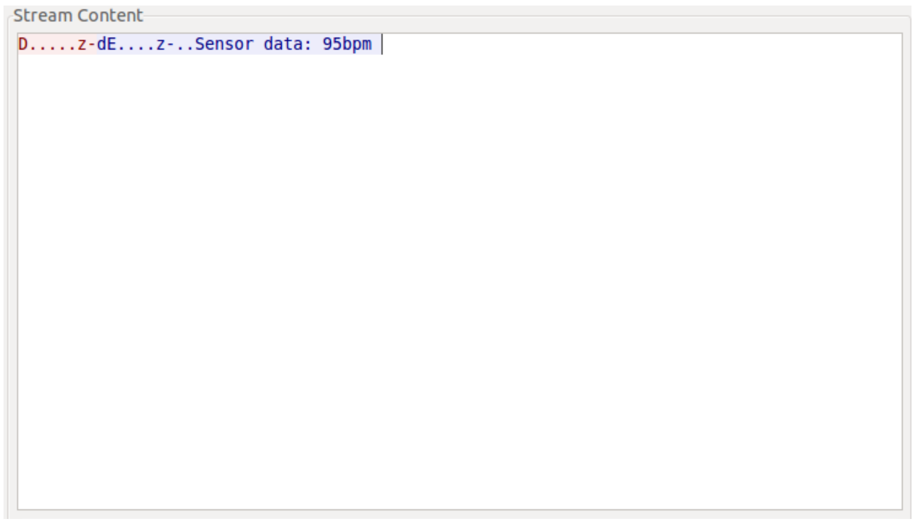


Figure 6.3: Endpoint Phases

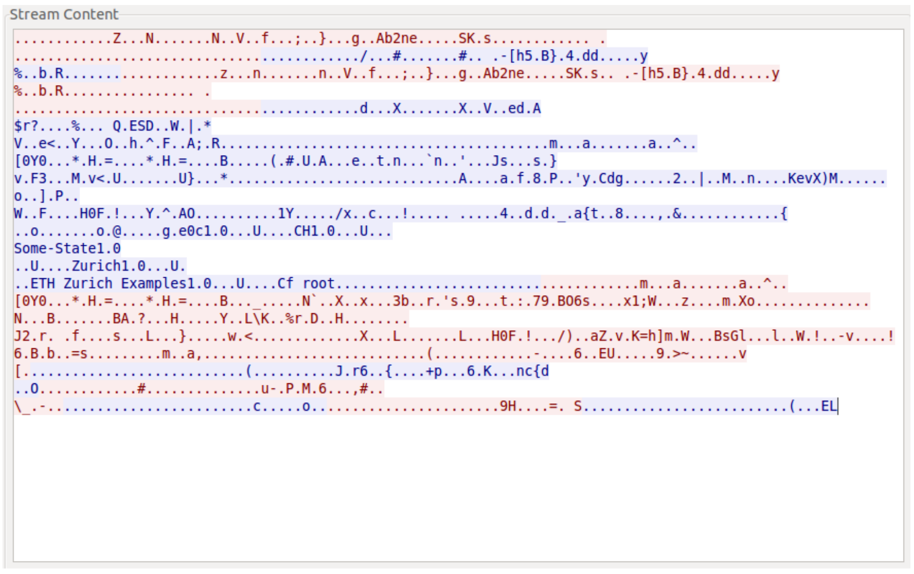
The first phase of endpoint creation is constructing the connector associating it to the protocol that is going to be used, such as UDP, DTLS or other kind of iteration. The second phase passes by implementing the CoAP on top of that connector for usage on phase 3.
The following figures show the same CoAP message transmitted in Open CoAP and Secure CoAP(DTLS) captured using wireshark.



```
D.....z-dE....z-..Sensor data: 95bpm
```

Figure 6.4: CoAP Message capture

This figure shows a captured Open CoAP message, as it can be seen the message content is perfectly visible(understandable) by someone who can intercept it.


Stream Content

```
...........Z...N.......N..V..f...;..}...g..Ab2ne.....SK.s............ .
........................................../...#.......#.. .-[h5.B}.4.dd.....y
%..b.R.................z...n.......n..V..f...;..}...g..Ab2ne.....SK.s.. .-[h5.B}.4.dd.....y
%..b.R............... .
..........................................d...X.......X..V..ed.A
$r?....%... Q.ESD..W.|.*
V..e<..Y...O..h.^.F..A;.R.......................................m...a.......a..^..
[0Y0...*.H.=....*.H.=....B.....(.#.U.A...e..t.n...`n..'...Js...s.}
v.F3...M.v<.U.......U}...*...........................A....a.f.8.P..'y.Cdg......2..|..M..n....KevX)M......
o..].P..
W..F....H0F.!...Y.^.AO.........1Y...../x..c...!..... .....4..d.d._.a{t..8....,.&...........{
..o.......o.@.....g.e0c1.0...U....CH1.0...U...
Some-State1.0
..U....Zurich1.0...U.
..ETH Zurich Examples1.0...U....Cf root.....................................m...a.......a..^..
[0Y0...*.H.=....*.H.=....B..._.....N`..X..x...3b..r.'s.9...t.:.79.BO6s....x1;W...z....m.Xo.............
N...B.......BA.?...H.....Y..L\K..%r.D..H.......
J2.r. .f....s...L...}.....w.<............X...L.......L...H0F.!.../)..aZ.v.K=h]m.W...BsGl...l..W.!..-v....!
6.B.b..=s.........m..a,..........................(...........-....6..EU.....9.>~......v
[..........................(.........J.r6..{....+p...6.K...nc{d
..O...........#.............u-.P.M.6...,#..
\_.-.........................c.....o.......................9H....=. S........................(...EL
```

Figure 6.5: Secure CoAP Message capture

This figure shows a captured Secure CoAP message, as it can be seen the message is not readable by someone who can intercept it.

Since Scandium(Sc) is an eclipse library built specifically for Californium(Cf), and DTLS is transparent to higher levels protocols and implementation has less implementation problems than IPsec, this library seamed the most feasible approach for implementation and for that reason was chosen as the security and authentication module.

6.2 Implementations Challenges

In implementation phase it was used Oracle Virtual Machine to simulate the work of an Raspberry Pi. However during the installation of required software some difficulties were encountered. Difficulties like some software requiring an arm processor, and since the Raspberry Pi simulation was running on x64 processor some installations failed.

Ideally for testing purposes the Raspberry Pi would be better, since it produce a very close result off the real hardware. But as problems continue to emerge this approach seam less and less a viable testing tool. Since Raspberry Pi use a debian-based OS type (Raspbian), it was chosen to test implementations on Ubuntu since it is also a debian-based OS.

For developing and implementing libraries it was chosen Eclipse IDE. This environment presented the best compatibility for testing since it's the same developer as Californium(Cf) and Scandium(Sc). Also tutorials for testing referred eclipse IDE as the tool used for tests. For building the libraries of both Californium(Cf) and Scandium(Sc) it was used the maven from Apache. In order to create keys and certificates for securing communications it was used OpenSSL.

6.3 Tests Results and Analysis

In order to evaluate the performance of the two modes of transmission, Open and Secure, several tests have been made. Tests were made using two computers, a intel core 2 Quad 2,4GHz running ubuntu x32 and a intel core 2 Duo 2.5GHz running ubuntu x32. Since this is to simulate smart devices performances, whose processing power and energy are limited, attention will be focused on the time it takes to complete the exchange of a message.

Also since that smart devices are embedded with sensors and routines that can use the CPU's processing power, tests were made with cpu on three conditions:

- CPU load 0-30
- CPU load 30-60
- CPU load 60-90

In order to simulate these work loads the computer acting as node was running some multimedia applications, such as audio and video.

The purpose off doing this is to evaluate the impact suffered by Open and Secure transmission times when there are variations in processing load, and how this will affect the system.

Will transmissions be more affected when they are secure or will the impact be relatively equal to that suffered in the Open transmissions. Next will be present the results from calculations based on 40 tests for each condition and different mode Open or Secure. The calculations made were used to draw boxplot graphs, that allows us to do a quick analysis of the results in each set of tests.

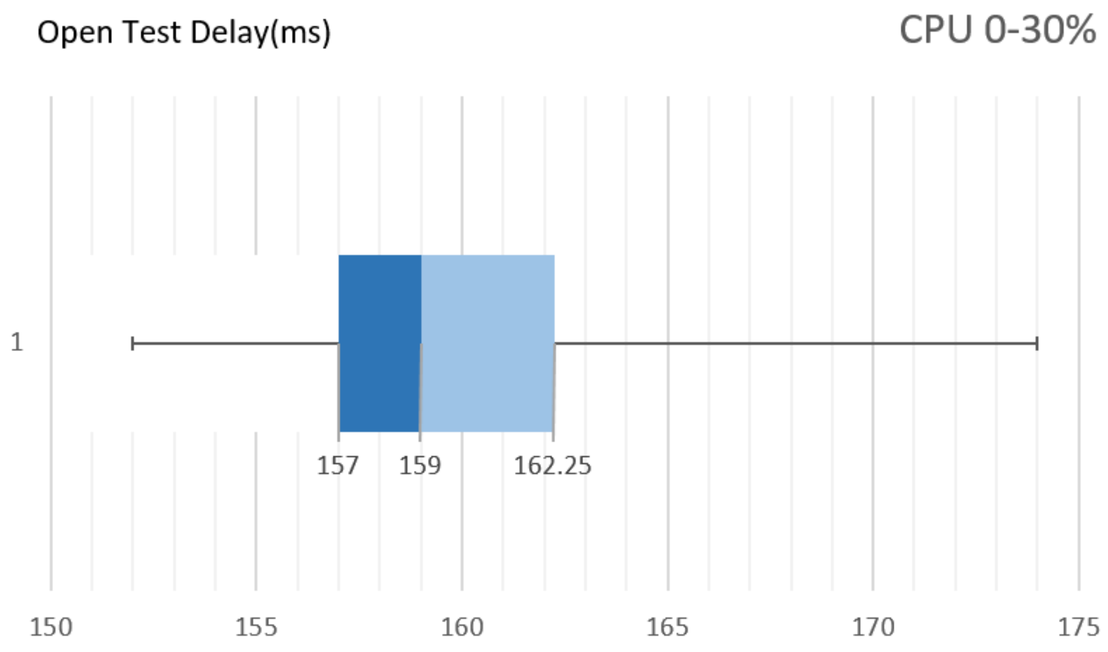


Figure 6.6: Open Test Boxplot graph (cpu load 0-30%)

In this graph we may observe the time it takes to complete an exchange of information in Open mode, between a device and the Coap Server, when the CPU load is from 0% to 30%. The following graph was drawn with values obtained in calculations where, the minimum value was 152 ms the maximum value was 174 ms and the median of the values obtained was 159 ms. Moreover it can be observed that 25% of the values were below 157 ms, 50% are between 157 ms to 162.25 ms and 25% are greater than 162.25 ms.

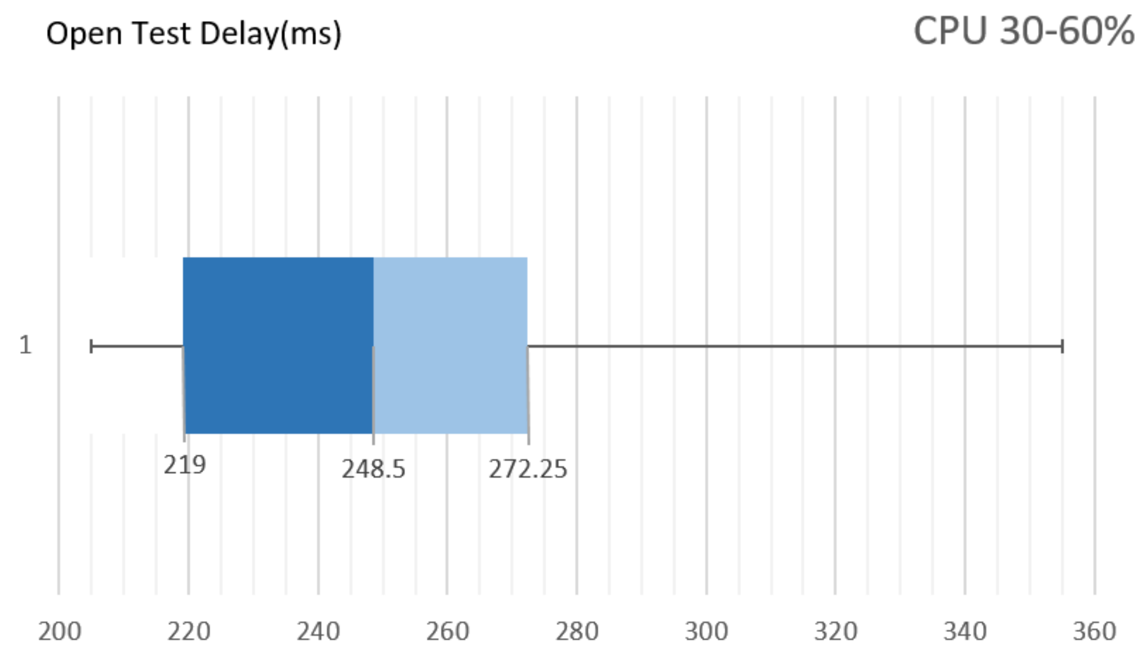


Figure 6.7: Open Test Boxplot graph (cpu load 30-60%)

In this graph we may observe the time it takes to complete an exchange of information in Open mode, between a device and the Coap Server, when the load of the CPU is from 30% to 60%. The following graph was drawn with values obtained in calculations where, the minimum value was 205 ms the maximum value was 355 ms and the median of the values obtained was 248.5 ms. Moreover it can be observed that 25% of the values were below 219 ms, 50% are between 219 ms to 272.25 ms and 25% are greater than 272.25 ms.

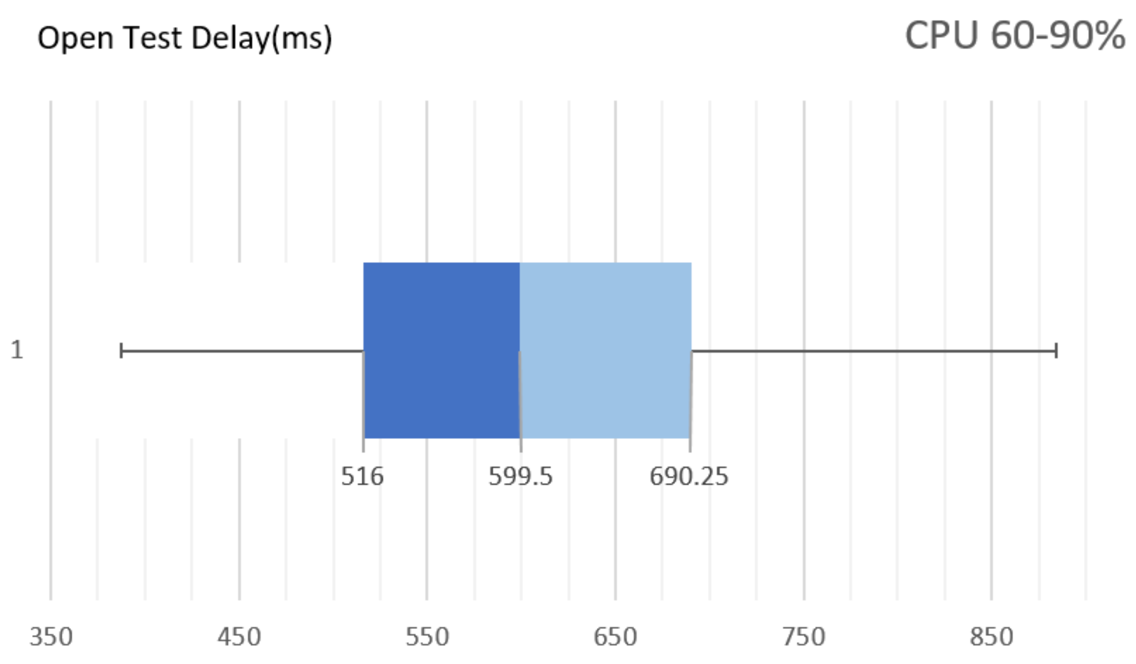


Figure 6.8: Open Test Boxplot graph (cpu load 60-90%)

In this graph we may observe the time it takes to complete an exchange of information in Open mode, between a device and the Coap Server, when the load of the CPU is from 60% to 90%. The following graph was drawn with values obtained in calculations where, the minimum value was 387 ms the maximum value was 884 ms and the median of the values obtained was 599.5 ms. Moreover it can be observed that 25% of the values were below 516 ms, 50% are between 516 ms to 690.25 ms and 25% are greater than 690.25 ms.

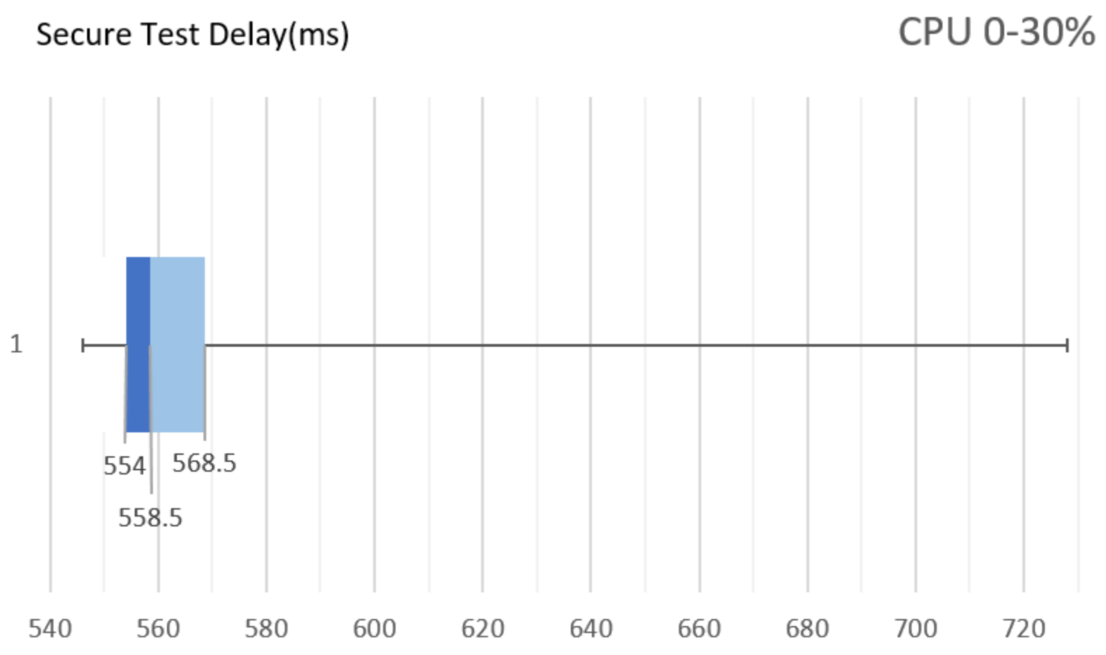


Figure 6.9: Secure Test Boxplot graph (cpu load 0-30%)

In this graph we may observe the time it takes to complete an exchange of information in Secure mode, between a device and the Coap Server, when the CPU load is from 0% to 30%. The following graph was drawn with values obtained in calculations where, the minimum value was 546 ms the maximum value was 728 ms and the median of the values obtained was 558.5 ms. Moreover it can be observed that 25% of the values were below 554 ms, 50% are between 554 ms to 568.5 ms and 25% are greater than 568.5 ms.

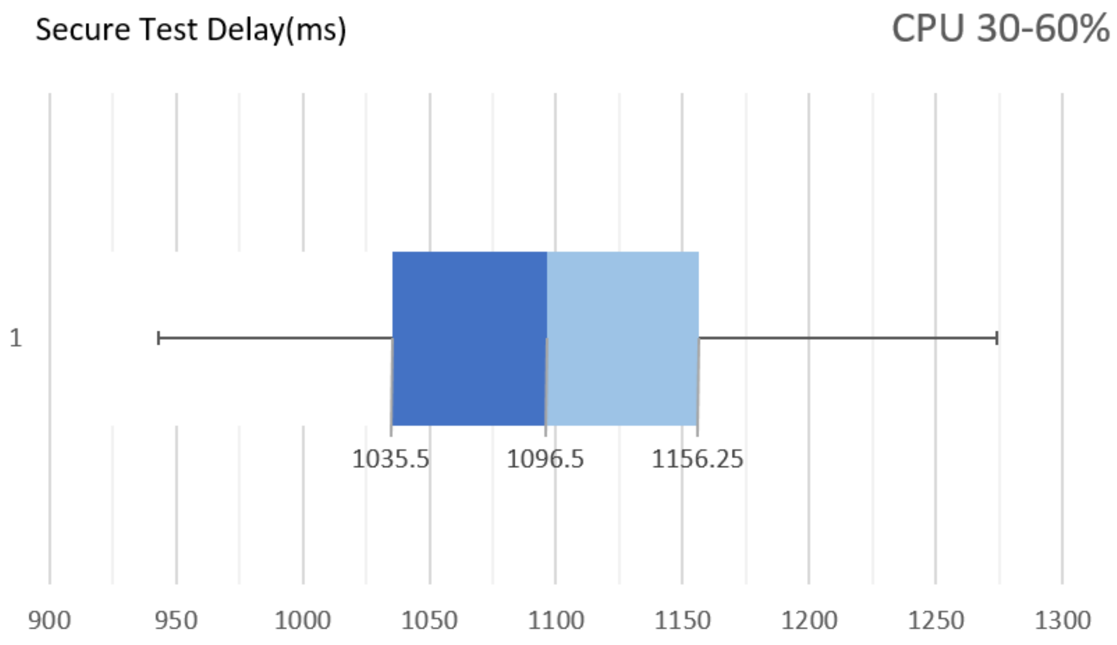


Figure 6.10: Secure Test Boxplot graph (cpu load 30-60%)

In this graph we may observe the time it takes to complete an exchange of information in Secure mode, between a device and the Coap Server, when the load of the CPU is from 30% to 60%. The following graph was drawn with values obtained in calculations where, the minimum value was 943 ms the maximum value was 1274 ms and the median of the values obtained was 1096.5 ms. Moreover it can be observed that 25% of the values were below 1035.5 ms, 50% are between 1035.5 and 1156.25 ms ms and 25% are above 1156.25 ms.

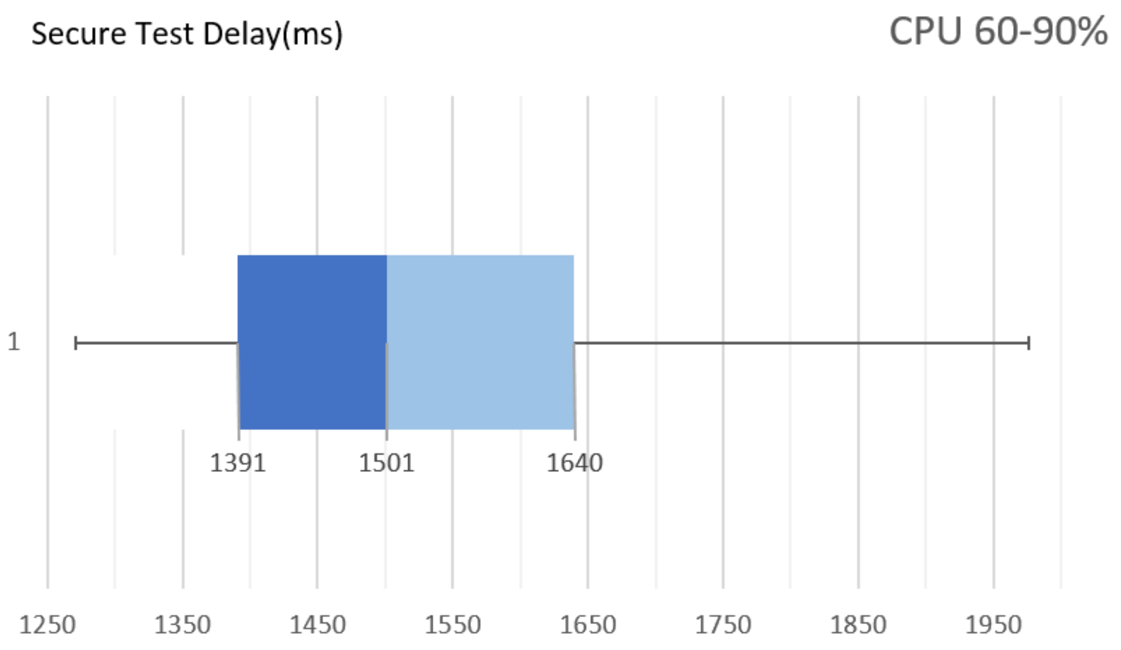


Figure 6.11: Secure Test Boxplot graph (cpu load 60-90%)

In this graph we may observe the time it takes to complete an exchange of information in Secure mode, between a device and the Coap Server when the load of the CPU is from 60% to 90%. The following graph was drawn with values obtained in calculations where, the minimum value was 1271 ms the maximum value was 1976 ms and the median of the values obtained was 1501 ms. Moreover it can be observed that 25% of the values were below 1391 ms, 50% are between 1391 ms and 1640 ms and 25% are above 1640 ms.

6.4 Conclusions

After having analysed test results it is ascertained that in open communication mode with typical cpu loads (0 to 30%), information exchanges take a median of 159 ms to be completed.

The results obtained from exchanging information in secure mode with typical loads of cpu (0 to 30%) were of a median of 558.5 ms. As expected the time it takes to complete the communication process in secure mode is higher than in open mode, about 3.5 times higher. As cpu load increase, the time it takes to complete the exchanges of information also increase. Those values can be about three times higher than typical load values and that increase can be verified in both open and secure mode.

Despite the fact that IoT device performance is being simulated, results can be expected to be in about the same in proportion for real scenarios. It was ascertained that securing a connection for transmitting low bursts of data has high costs in system performance. That is due on the overhead of the information exchange, that is required in order to secure a connection, being higher than the data to be transmitted itself.

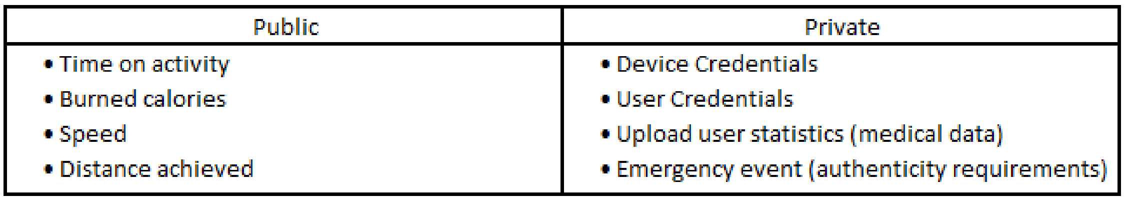
IoT devices should be programmed with power saving requirements in mind since they are energy constrained devices. Assuming that requirement is met and devices have no high activity due to, being used as a node for communication between the server and other devices or a routine that take priority over energy saving requirements being active(information required in real time), that will be the usual state of an IoT device.

Higher cpu usage translate into a higher delay of time in exchanging messages, that means more energy spent since the wireless capabilities of a device drain more energy for longer periods. That delay of time also translate into lower responsiveness of the system affecting Quality of Service. Although exchanging information with low cpu load might be ideal for energy saving purposes there are situations in which that is not possible.

Securing communications for continuous or big streams of data might be suitable since the overhead of data needed for securing a connection, is minimized by the size of the information that is going to be exchanged.

The ideal Communication method that answers the requirements of QoS, Security and Authentication, might be a mix mode that combines both modes Open and Secure. This mode would permit exchange of information that require security and authentication.At the same time it would be able to maintain the performance when transmitting data without security requirements. That data would be transmitted in Open mode.

The following figure displays a table that exemplifies some data that can be transmitted in open mode (public) and data that requires security to be transmitted (private).



Public	Private
• Time on activity • Burned calories • Speed • Distance achieved	• Device Credentials • User Credentials • Upload user statistics (medical data) • Emergency event (authenticity requirements)

Figure 6.12: Open data vs Private data

It's hard to specify for all systems which information to be public or private since it varies from one scenario to another, but information with authentication or private concerns should be transmitted in a secure fashion.

Specifying a mode that combines both open and secure communication, would be achieved a balance between QoS, security and authentication improving the overall system functionality.

Chapter 7

Thesis Conclusions

The main objective of this thesis was the propose, implementation and analysis of a secure IoT Cloud environment. This work starts by giving an overview in the various technologies involved in making IoT and Cloud computing environments a possibility, as well some of their limitations and/or problems. Adding to that it was also presented some work already done in those areas.

After that introduction some preliminary tests were done in a platform named Skynet. This platform uses a machine-to-machine mqtt protocol. Those tests served to demonstrate that at certain hours of the day there is an increase in activity on the web, that increase in activity affect the QoS. Distance to the server that provides the required services also proved to be an issue that affected QoS. On security tests it was ascertained that mechanisms needed to be implemented in order to protect crucial data, as an example it was shown that tokens transmitted to authenticate devices can be captured and its content read.

After the preliminary tests were done, the scenario for implementing these technologies was described. Moving on to the proposed system the architecture was presented and explained. On implementations and decisions, it was decided to use CoAP as the protocol for the IoT 6LoWPAN network. Eclipse Californium(Cf) was chosen as the CoAP implementation to be used and Eclipse Scandium(Sc) was chosen as the security module for implementing DTLS protocol.

Implementation testing compare the delay of completing an information exchange in Open CoAP versus the Secure approach. Those tests served to show how the system is affected by using security in information exchanges. Tests were done on different levels of activity of the cpu, which was done in order to ascertain how much impact that would have on the delay of communications.

Gathered data from the testing phase was used in order to draw some box-plot graphs that permit an easier results analysis. It was concluded that securing communications has high impact on system performance, also the variation of cpu activity heavily affect the delay of exchanging information.

Securing the the entirety of the system might not be a good approach when designing the platform since it has high impact on performance and energy consumption. The ideal method of applying security and authentication to these kind of systems might be implement a mix mode of communication. One that comprehend open and secure communications between peers.

Information with little to none security requirements would be transmitted in open mode, since capture does not compromises security or privacy, and a secure mode for exchanging sensitive information with security requirements.

These modifications would potentially optimize system performance and energy consumption.

After reaching conclusions on the benefits and disadvantages of each method of securing an IoT Cloud platform some objectives for future work were defined.

As future work its proposed the following. Implementation and analysis of a mix mode of communication that permit open and secure transmissions by doing this it can be assessed if using the two modes will have a beneficial impact.

Hardware installation and testing and testing performance when using nodes as relayers this will permit to assess the limitations of using nodes relayers. And adding header compression to the DTLS technology in order to minimize the overhead of transmissions.

Bibliography

[1] Carlos Dores, Luis Paulo Reis, Nuno Vasco Lopes, *Internet of Things and Cloud Computing*, 9th Iberian Conference on Information Systems and Technologies, Barcelona, June 2014, 18-21.

[2] Syed A. Ahson and Mohammad Ilyas, *IP multimedia subsystem (IMS) handbook*, 2nd edition, ISBN 9781420064599, Taylor & Francis Group, 2009.

[3] M. C. Domingo, *A context-aware service architecture for the integration of body sensor networks and social networks through IP multimedia Subsystem*, IEEE Communication Magazine, Jan. 2011.

[4] Khalid Al-Begain, Chitra Balakrishna, Luis Angel Galindo and David Moro, *IMS: A Development and Deployment Perspective*, ISBN 9780470740347, John Wiley & Sons Ltd, 2009.

[5] G. Camarillo and M. A. García-Martín, *The 3G IP Multimedia Subsystem (IMS): Merging the Internet and the Cellular Worlds*, 2nd edition, ISBN 9780470018187, John Wiley & Sons Ltd, 2006.

[6] 3GPP Tech. Spec., *IP Multimedia Subsystem (IMS) (Release 7)*, 2006; `http://www.3gpp.org/ftp/Specs/html-info/23228.htm`.

[7] Payam Barnaghi, Wei Wang, Cory Henson and Kerry Taylor, *Semantics for the Internet of Things: early progress and back to the future*, International Journal on Semantic Web and Information Systems (IJSWIS), IGI Global, 2012; `http://personal.ee.surrey.ac.uk/Personal/P.Barnaghi/doc/IJSWIS_SemIoT_CR_2.pdf`.

[8] Rajkumar Buyyab, Jayavardhana Gubbia, Slaven Marusica and Marimuthu Palaniswamia, *Internet of Things (IoT): A vision, architectural elements,*

and future directions, Elsevier Inc, February 2013; `http://www.buyya.com/papers/Internet-of-Things-Vision-Future2013.pdf`.

[9] Mari Carmen Domingo, *An overview of the Internet of Things for people with disabilities*, in Journal of Network and Computer Applications vol. 35, Elsevier Ltd, 2012, 584-596.

[10] Martin Haenggi and Daniele Puccinelli, *Wireless Sensor Networks: Applications and Challenges of Ubiquitous Sensing*, IEEE Circuits and Systems Magazine, Third Quarter 2005.

[11] P. Fergus *A Framework for Physical Health Improvement using Wireless Sensor Networks and Gaming*, Pervasive Computing Technologies for Healthcare 3rd International Conference, IEEE, London, April 2009, pp. 1-4.

[12] Benny P L Lo, Guang-Zhong Yang, *Key Technical challenges and current implementations of Body sensor networks* 2nd International Workshop on Body Sensor Networks, London, UK, April 2005; `http://vip.doc.ic.ac.uk/bsn/public/UbiMonPapers/Key_Technical_Challenges_and_Current_Implementations_of_Body_Sensor_Networks.pdf`.

[13] M. Li, W. Lou, and K. Ren, *Data Security and Privacy in Wireless Body Area Networks*, Wireless Communications, IEEE vol. 17 Issue: 1, February 2010, pp. 51-58.

[14] M. Patel and J. Wang, *Applications, Challenges, and Prospective in Emerging Body Area Networking Technologies*, Wireless Communications, IEEE vol. 17 Issue:1, February 2010, pp. 80-88.

[15] Dan C. Marinescu, *Cloud Computing Theory and Practice*, ISBN 9780124046276, Elsevier Inc., 2013.

[16] Rajkumar Buyya, Christian Vecchiola and S. Thamarai Selvi, *Mastering Cloud Computing Foundations and Applications Programming*, ISBN 9780124114548, Elsevier Inc.,2013.

[17] Ileana Castrillo, Derrick Rountree and Hai Jiang as Technical Editor, *The Basics of Cloud Computing: Understanding the Fundamentals of Cloud Computing in Theory and Practice*, ISBN 9780124059320, Elsevier Inc., 2013.

[18] Randee Adams and Eric Bauer, *Reliability and availability of cloud computing*, ISBN 9781118177013, IEEE, John Wiley & Sons Ltd, 2012.

[19] IETF RFC 768 `http://tools.ietf.org/html/rfc768` [accessed on Jan. 2015].

[20] IETF RFC 793 `http://tools.ietf.org/html/rfc793`[accessed on Jan. 2015].

[21] `http://mqtt.org/faq` [accessed on Jan. 2015].

[22] Z. Shelby, K. Hartke, C. Bormann, and B. Frank. *Constrained Application Protocol (CoAP)*, draft-ietf-core-coap-18, Jun. 28, 2012.

[23] https://datatracker.ietf.org/wg/core/charter/ [accessed on Aug. 2015].

[24] `http://www.eclipse.org/californium/`[accessed on Aug. 2015].

[25] `http://people.inf.ethz.ch/mkovatsc/copper.php`[accessed on Aug. 2015].

[26] `https://libcoap.net/`[accessed on Aug. 2015].

[27] `http://people.inf.ethz.ch/mkovatsc/erbium.php`[accessed on Aug. 2015].

[28] E. Rescorla and N. Modadugu. *Datagram Transport Layer Security Version 1.2. RFC 6347 (Proposed Standard)*, IETF, Jan. 2012.

[29] P. Wouters, J. Gilmore, S. Weiler, T. Kivinen, and H. Tschofenig. *Out-of-Band Public Key Validation for Transport Layer Security*, draft-ietf-tls-oob-pubkey-04, IETF, July 2012.

[30] Jayavardhana Gubbia, Rajkumar Buyyab, Slaven Marusic, Marimuthu Palaniswami, *Internet of Things (IoT): A vision, architectural elements, and*

future directions, Future Generation Computer Systems vol. 29, Elsevier Inc., 2013, pp. 1645–1660.

[31] Chunming Rong, Son T. Nguyen, Martin Gilje Jaatun, *Beyond lightning: A survey on security challenges in cloud computing*, Computers and Electrical Engineering Vol. 39, Elsevier Ltd, 2013, pp. 47-54.

[32] John Panneerselvam, Stelios Sotiriadis, Nik Bessis, Nick Antonopoulos, *Securing Authentication and Trusted Migration of Weblets in the Cloud with Reduced Traffic*, ISBN 9781467319867 , Emerging Intelligent Data and Web Technologies 3rd International Conference, IEEE, 2012, pp. 316 - 319.

[33] Daniele Trabalza, *Implementation and Evaluation of Datagram Transport Layer Security (DTLS) for the Android Operating System*, Master's Degree Project, Stockholm, Sweden, June 2012.

[34] `http://www.infoq.com/articles/perf4j`[accessed on Oct. 2015].

[35] Giulio Peretti, *CoAP over DTLS TinyOS Implementation and Performance Analysis*, Tesi di laurea magistrale, University of Padova, December 10, 2013.

[36] Shahid Raza, Hossein Shafagh, Kasun Hewage, Ren Hummen, and Thiemo Voigt, *Lithe: Lightweight Secure CoAP for the Internet of Things*,IEEE Sensors Journal Vol.13 NO. 10, October 2013.

[37] `http://skynet.im` [accessed on Fev. 2014].